

# digiKam Developer Documentation

Generated on Tue Jan 7 2025 06:32:21 for digiKam Developer Documentation by Doxygen  
1.9.8

Tue Jan 7 2025 06:32:21



---

<b>1 digiKam project API reference.</b>	<b>1</b>
1.1 Source Code Directories	1
1.2 External Dependencies	6
1.2.1 Dependencies To Checkout All Source Code	6
1.2.2 Dependencies To Process Translations Files (optional)	6
1.2.3 Dependencies To Compile And Link Source Code	6
1.3 Get Source Code	11
1.3.1 Software Components	11
1.4 Development Environment	11
1.5 Cmake Configuration Options	12
1.5.1 Top Level Configuration	12
1.5.2 Core Configuration	12
1.6 Setup Local Compilation and Run-Time	13
1.7 Debug Traces At Run-Time	13
1.7.1 Logging Using an Environment Variable	13
1.7.2 Logging Categories in digiKam	14
1.7.3 Further Reading	15
1.8 Cmake compilation rules	15
1.8.1 Introduction	15
1.8.2 CMake Implementation Details	16
1.8.2.1 Include Directories	16
1.8.2.2 Shared Libraries	16
1.8.2.3 Static Libraries	16
1.8.2.4 Object Libraries	17
1.9 Contribute To The Code	17
1.9.1 Starting With Open-Source	17
1.9.2 Source Code Formatting	17
1.9.2.1 Indentation length	17
1.9.2.2 Tabs vs Spaces	18
1.9.2.3 Line length	18
1.9.2.4 Bracketing	18
1.9.2.5 Positioning of Access modifiers	18
1.9.3 Class, file and Variable names	18
1.9.3.1 Class and filenames	18
1.9.3.2 Protected Member variables	19
1.9.3.3 Non-Member variables	19
1.9.3.4 Private Member variables	19
1.9.4 Comments and Whitespace	19
1.9.5 Header Files	19
1.9.6 Automatic source code formatting	20
1.9.7 General recommendations	20
1.9.8 GDB Backtrace	21

---

1.9.9 Memory Leak	21
1.9.10 Profiling With Cachegrind	21
1.9.11 Unit Testing / Automated Testing	21
1.9.12 Checking For Corrupt Qt Signal Slot Connection	22
1.9.13 Finding Duplicated Code	22
1.9.14 API Documentation Validation, User Documentation Validation, Source Code Checking	22
1.9.15 Usability Issues	22
1.9.16 Generate API Documentation	22
1.9.17 Speed Up The Code-Compile-Test Cycle	23
1.9.18 Working With Branches From Git Repository	23
1.9.19 Sync a Branch With Master From Git Repository	23
<b>2 CODE_OF_CONDUCT</b>	<b>25</b>
<b>3 COMMIT POLICY</b>	<b>27</b>
3.1 CODING STYLE	27
3.2 DOCUMENTATION	27
3.3 LICENSING	27
<b>4 README</b>	<b>29</b>
4.1 About	29
4.2 Authors	30
4.3 Related URLs	30
4.4 Contact	30
4.5 Bug reports	31
4.6 Compilation and Installation	31
4.7 Donate Money	31
<b>5 Namespace Index</b>	<b>33</b>
5.1 Namespace List	33
<b>6 Hierarchical Index</b>	<b>35</b>
6.1 Class Hierarchy	35
<b>7 Class Index</b>	<b>65</b>
7.1 Class List	65
<b>8 Namespace Documentation</b>	<b>95</b>
8.1 Digikam Namespace Reference	95
8.1.1 Detailed Description	133
8.1.2 Typedef Documentation	134
8.1.2.1 ActionJobCollection	134
8.1.2.2 DItemsListsLessThanHandler	134
8.1.3 Enumeration Type Documentation	134
8.1.3.1 DetectorNNModel	134



8.1.3.2 FullScreenOptions	134
8.1.3.3 GeoGroupStateEnum	135
8.1.3.4 HistogramRenderingType	135
8.1.3.5 HistogramScale	135
8.1.3.6 HudSide	135
8.1.3.7 MeaningOfDirection	135
8.1.3.8 OperationType	136
8.1.3.9 YoloVersions	136
8.1.4 Function Documentation	136
8.1.4.1 adjustedEnvironmentForApplImage()	136
8.1.4.2 coordinatesToClipboard()	136
8.1.4.3 DNotificationWrapper()	137
8.1.4.4 fastNumberToString()	137
8.1.4.5 GeofaceHelperParseLatLonString()	137
8.1.4.6 image2Mat()	137
8.1.4.7 image2Mat_shared()	138
8.1.4.8 openOnlineDocumentation()	138
8.1.4.9 operator<<()	139
8.1.4.10 QPointSquareDistance()	139
8.1.4.11 s_inlineTranslateString()	139
8.1.4.12 s_rawFileExtensionsdWithDesc()	139
8.1.4.13 s_rawFileExtensionsVersion()	140
8.1.4.14 s_setXmpTagStringFromEntry()	140
8.1.4.15 setExifXmpTagDataVariant()	140
8.1.4.16 supportedImageMimeTypes()	141
8.1.5 Variable Documentation	141
8.1.5.1 accessCol	141
8.1.5.2 accessRow	141
8.1.5.3 CR_basis	141
8.1.5.4 ExifHumanList	142
8.1.5.5 FACE_TEMPLATE	142
8.1.5.6 faceenum2size	142
8.1.5.7 GeofaceMinMarkerGroupingRadius	142
8.1.5.8 lptcHumanList	143
8.1.5.9 namespaceTitleDefinitions	143
8.1.5.10 spectral_chromaticity	143
8.1.5.11 videoStrip16	144
8.1.5.12 videoStrip4	144
8.1.5.13 videoStrip8	144
8.1.5.14 XmpHumanList	145
8.2 Digikam::Matrix Namespace Reference	145
8.2.1 Detailed Description	145

---

<b>9 Class Documentation</b>	<b>147</b>
9.1 CoreDbWatchAdaptor Class Reference	147
9.2 Digikam::AbstractAlbumModel Class Reference	148
9.2.1 Member Enumeration Documentation	150
9.2.1.1 AlbumDataRole	150
9.2.1.2 RootAlbumBehavior	151
9.2.2 Constructor & Destructor Documentation	151
9.2.2.1 AbstractAlbumModel()	151
9.2.3 Member Function Documentation	151
9.2.3.1 albumCleared()	151
9.2.3.2 albumData()	151
9.2.3.3 allAlbumsCleared()	152
9.2.3.4 columnHeader()	152
9.2.3.5 decorationRoleData()	152
9.2.3.6 filterAlbum()	152
9.2.3.7 fontRoleData()	152
9.2.3.8 retrieveAlbum()	152
9.2.3.9 rootAlbumAvailable	152
9.2.3.10 rootAlbumIndex()	153
9.2.3.11 setEnableDrag()	153
9.2.3.12 sortRoleData()	153
9.3 Digikam::AbstractAlbumTreeView Class Reference	153
9.3.1 Detailed Description	158
9.3.2 Member Enumeration Documentation	158
9.3.2.1 Flag	158
9.3.3 Constructor & Destructor Documentation	158
9.3.3.1 AbstractAlbumTreeView()	158
9.3.4 Member Function Documentation	158
9.3.4.1 addCustomContextMenuActions()	158
9.3.4.2 contextMenuIcon()	159
9.3.4.3 contextMenuTitle()	159
9.3.4.4 doLoadState()	159
9.3.4.5 doSaveState()	159
9.3.4.6 expandEverything	159
9.3.4.7 expandMatches()	160
9.3.4.8 handleCustomContextMenuAction()	160
9.3.4.9 indexVisuallyAt()	160
9.3.4.10 pixmapForDrag()	160
9.3.4.11 selectedAlbumsChanged	161
9.3.4.12 setAlbumManagerCurrentAlbum()	161
9.3.4.13 setContextMenuIcon()	161
9.3.4.14 setCurrentAlbums	161

---

9.3.4.15	setEnabledContextMenu()	161
9.3.4.16	setSelectAlbumOnClick()	162
9.3.4.17	setSelectOnContextMenu()	162
9.3.4.18	showContextMenuAt()	162
9.4	Digikam::AbstractAlbumTreeView::ContextMenuElement Class Reference	162
9.4.1	Detailed Description	163
9.4.2	Member Function Documentation	163
9.4.2.1	addActions()	163
9.5	Digikam::AbstractAlbumTreeViewSelectComboBox Class Reference	164
9.5.1	Constructor & Destructor Documentation	167
9.5.1.1	AbstractAlbumTreeViewSelectComboBox()	167
9.5.2	Member Function Documentation	167
9.5.2.1	addCheckUncheckContextMenuActions()	167
9.5.2.2	installView()	167
9.5.2.3	sendViewportEventToView()	168
9.5.2.4	setTreeView()	168
9.6	Digikam::AbstractCheckableAlbumModel Class Reference	169
9.6.1	Constructor & Destructor Documentation	174
9.6.1.1	AbstractCheckableAlbumModel()	174
9.6.2	Member Function Documentation	174
9.6.2.1	albumCleared()	174
9.6.2.2	albumData()	175
9.6.2.3	allAlbumsCleared()	175
9.6.2.4	checkStateChanged	175
9.6.2.5	setData()	175
9.6.2.6	setRootCheckable()	175
9.6.2.7	setTristate()	176
9.7	Digikam::AbstractCheckableAlbumTreeView Class Reference	177
9.7.1	Constructor & Destructor Documentation	182
9.7.1.1	AbstractCheckableAlbumTreeView()	182
9.7.2	Member Function Documentation	182
9.7.2.1	doLoadState()	182
9.7.2.2	doSaveState()	182
9.7.2.3	isRestoreCheckState()	182
9.7.2.4	middleButtonPressed()	182
9.7.2.5	setRestoreCheckState()	182
9.8	Digikam::AbstractCountingAlbumModel Class Reference	183
9.8.1	Member Function Documentation	187
9.8.1.1	albumCleared()	187
9.8.1.2	albumCount()	187
9.8.1.3	albumData()	187
9.8.1.4	albumForId()	187

---

9.8.1.5 albumName()	188
9.8.1.6 allAlbumsCleared()	188
9.8.1.7 excludeChildrenCount	188
9.8.1.8 includeChildrenCount	188
9.8.1.9 setCountHash	188
9.9 Digikam::AbstractCountingAlbumTreeView Class Reference	189
9.10 Digikam::AbstractDetector Class Reference	193
9.11 Digikam::AbstractItemDragDropHandler Class Reference	194
9.11.1 Member Function Documentation	195
9.11.1.1 accepts()	195
9.11.1.2 acceptsMimeData()	195
9.11.1.3 createMimeData()	195
9.11.1.4 dropEvent()	195
9.11.1.5 mimeTypes()	195
9.12 Digikam::AbstractMarkerTiler Class Reference	196
9.12.1 Member Function Documentation	197
9.12.1.1 bestRepresentativeIndexFromList()	197
9.12.1.2 getTile()	198
9.12.1.3 getTileGroupState()	198
9.12.1.4 getTileRepresentativeMarker()	198
9.12.1.5 indicesEqual()	198
9.12.1.6 onIndicesClicked()	198
9.12.1.7 pixmapFromRepresentativeIndex()	198
9.12.1.8 prepareTiles()	199
9.12.1.9 setActive()	199
9.12.1.10 tilerFlags()	199
9.13 Digikam::AbstractMarkerTiler::ClickInfo Class Reference	199
9.14 Digikam::AbstractMarkerTiler::NonEmptyIterator Class Reference	199
9.15 Digikam::AbstractMarkerTiler::Tile Class Reference	200
9.16 Digikam::AbstractSearchGroupContainer Class Reference	201
9.16.1 Member Function Documentation	202
9.16.1.1 addGroupToLayout()	202
9.16.1.2 createSearchGroup()	202
9.17 Digikam::AbstractSpecificAlbumModel Class Reference	203
9.17.1 Member Function Documentation	206
9.17.1.1 columnHeader()	206
9.18 Digikam::AbstractWidgetDelegateOverlay Class Reference	206
9.18.1 Constructor & Destructor Documentation	208
9.18.1.1 AbstractWidgetDelegateOverlay()	208
9.18.2 Member Function Documentation	208
9.18.2.1 checkIndex()	208
9.18.2.2 createWidget()	209

9.18.2.3 hide()	209
9.18.2.4 setActive()	209
9.18.2.5 slotEntered	209
9.18.2.6 slotReset	209
9.18.2.7 viewportLeaveEvent()	210
9.18.2.8 widgetEnterEvent()	210
9.19 Digikam::ActionCategorizedView Class Reference	211
9.20 Digikam::ActionData Class Reference	213
9.21 Digikam::ActionItemModel Class Reference	214
9.21.1 Member Enumeration Documentation	215
9.21.1.1 MenuCategoryFlag	215
9.21.2 Constructor & Destructor Documentation	216
9.21.2.1 ActionItemModel()	216
9.21.3 Member Function Documentation	216
9.21.3.1 actionForIndex()	216
9.21.3.2 createFilterModel()	216
9.21.3.3 hover	216
9.21.3.4 itemForAction()	216
9.22 Digikam::ActionJob Class Reference	217
9.22.1 Constructor & Destructor Documentation	218
9.22.1.1 ~ActionJob()	218
9.23 Digikam::ActionSortFilterProxyModel Class Reference	218
9.24 Digikam::ActionTask Class Reference	220
9.25 Digikam::ActionThread Class Reference	222
9.26 Digikam::ActionThreadBase Class Reference	224
9.26.1 Member Function Documentation	225
9.26.1.1 appendJobs()	225
9.26.1.2 setDefaultMaximumNumberOfThreads()	225
9.27 Digikam::ActionVersionsOverlay Class Reference	226
9.27.1 Member Function Documentation	229
9.27.1.1 checkIndex()	229
9.27.1.2 createButton()	229
9.27.1.3 setActive()	230
9.27.1.4 updateButton()	230
9.28 Digikam::AddBookmarkDialog Class Reference	230
9.29 Digikam::AddBookmarkProxyModel Class Reference	231
9.29.1 Detailed Description	231
9.30 Digikam::AddTagsComboBox Class Reference	232
9.30.1 Member Function Documentation	236
9.30.1.1 currentTaggingAction()	236
9.30.1.2 setAlbumModels()	236
9.30.1.3 taggingActionSelected	236

---

9.31 Digikam::AddTagsLineEdit Class Reference	237
9.31.1 Member Function Documentation	238
9.31.1.1 setFilterModel()	238
9.31.1.2 setParentTag	238
9.31.1.3 taggingActionSelected	238
9.32 Digikam::AdvancedMetadataTab Class Reference	239
9.32.1 Constructor & Destructor Documentation	239
9.32.1.1 AdvancedMetadataTab()	239
9.33 Digikam::AdvancedRenameDialog Class Reference	240
9.34 Digikam::AdvancedRenameInput Class Reference	241
9.35 Digikam::AdvancedRenameLineEdit Class Reference	242
9.36 Digikam::AdvancedRenameListItem Class Reference	243
9.37 Digikam::AdvancedRenameManager Class Reference	244
9.38 Digikam::AdvancedRenameProcessDialog Class Reference	246
9.39 Digikam::AdvancedRenameWidget Class Reference	248
9.39.1 Member Function Documentation	249
9.39.1.1 parse()	249
9.39.1.2 setControlWidgets()	249
9.39.1.3 setLayoutStyle()	250
9.39.1.4 setParser()	250
9.39.1.5 setParseString()	250
9.40 Digikam::AdvancedSettings Class Reference	251
9.41 Digikam::AestheticDetector Class Reference	252
9.41.1 Member Function Documentation	253
9.41.1.1 detect()	253
9.42 Digikam::Akonadiface Class Reference	253
9.43 Digikam::Album Class Reference	254
9.43.1 Detailed Description	256
9.43.2 Member Enumeration Documentation	256
9.43.2.1 Type	256
9.43.3 Constructor & Destructor Documentation	257
9.43.3.1 ~Album()	257
9.43.4 Member Function Documentation	257
9.43.4.1 childAlbumIds()	257
9.43.4.2 childAlbums()	257
9.43.4.3 childAtRow()	257
9.43.4.4 childCount()	257
9.43.4.5 databaseUrl()	257
9.43.4.6 extraData()	257
9.43.4.7 firstChild()	258
9.43.4.8 globalID() [1/2]	258
9.43.4.9 globalID() [2/2]	258

9.43.4.10 id()	259
9.43.4.11 isAncestorOf()	259
9.43.4.12 isRoot()	259
9.43.4.13 isTrashAlbum()	259
9.43.4.14 isUsedByLabelsTree()	260
9.43.4.15 lastChild()	260
9.43.4.16 next()	260
9.43.4.17 parent()	260
9.43.4.18 prev()	260
9.43.4.19 removeExtraData()	260
9.43.4.20 rowFromAlbum()	261
9.43.4.21 setExtraData()	261
9.43.4.22 setUsedByLabelsTree()	261
9.43.4.23 title()	262
9.43.4.24 type()	262
9.44 Digikam::AlbumChangeset Class Reference	262
9.45 Digikam::AlbumCopyMoveHint Class Reference	263
9.46 Digikam::AlbumCustomizer Class Reference	264
9.47 Digikam::AlbumDragDropHandler Class Reference	265
9.47.1 Member Function Documentation	266
9.47.1.1 accepts()	266
9.47.1.2 createMimeData()	266
9.47.1.3 dropEvent()	266
9.47.1.4 mimeTypes()	267
9.48 Digikam::AlbumFilterModel Class Reference	268
9.48.1 Member Enumeration Documentation	270
9.48.1.1 FilterBehavior	270
9.48.1.2 MatchResult	270
9.48.2 Member Function Documentation	271
9.48.2.1 hasSearchResult	271
9.48.2.2 isFiltering()	271
9.48.2.3 lessThan()	271
9.48.2.4 matches()	271
9.48.2.5 matchResult() [1/2]	271
9.48.2.6 matchResult() [2/2]	272
9.48.2.7 searchTextSettings()	272
9.48.2.8 searchTextSettingsAboutToChange	272
9.48.2.9 searchTextSettingsChanged	272
9.48.2.10 setFilterBehavior()	272
9.48.2.11 setSearchTextSettings	273
9.48.2.12 setSourceAlbumModel()	273
9.48.2.13 setSourceFilterModel()	273

---

9.48.2.14 setSourceModel()	273
9.49 Digikam::AlbumFolderViewSideBarWidget Class Reference	274
9.49.1 Member Function Documentation	276
9.49.1.1 applySettings()	276
9.49.1.2 changeAlbumFromHistory()	276
9.49.1.3 doLoadState()	277
9.49.1.4 doSaveState()	277
9.49.1.5 getCaption()	277
9.49.1.6 getIcon()	277
9.49.1.7 setActive()	277
9.50 Digikam::AlbumHistory Class Reference	278
9.50.1 Detailed Description	279
9.50.2 Member Function Documentation	279
9.50.2.1 addAlbums()	279
9.51 Digikam::AlbumInfo Class Reference	280
9.52 Digikam::AlbumIterator Class Reference	280
9.52.1 Detailed Description	280
9.53 Digikam::AlbumLabelsSearchHandler Class Reference	281
9.53.1 Member Function Documentation	281
9.53.1.1 albumForSelectedItems()	281
9.53.1.2 generatedName()	282
9.53.1.3 imageUrls()	282
9.53.1.4 isRestoringSelectionFromHistory()	282
9.53.1.5 restoreSelectionFromHistory()	282
9.54 Digikam::AlbumManager Class Reference	282
9.54.1 Detailed Description	287
9.54.2 Member Function Documentation	287
9.54.2.1 albumTitles()	287
9.54.2.2 allDAAlbums()	287
9.54.2.3 allPAAlbums()	287
9.54.2.4 allSAAlbums()	288
9.54.2.5 allTAAlbums()	288
9.54.2.6 changeDatabase()	288
9.54.2.7 createPAAlbum() [1/3]	288
9.54.2.8 createPAAlbum() [2/3]	288
9.54.2.9 createPAAlbum() [3/3]	289
9.54.2.10 createSAAlbum()	289
9.54.2.11 createTAAlbum()	290
9.54.2.12 currentAlbums()	290
9.54.2.13 currentPAAlbum()	291
9.54.2.14 currentTAAlbums()	291
9.54.2.15 deleteSAAlbum()	291



9.54.2.16 deleteTAlbum()	291
9.54.2.17 findAlbum() [1/2]	292
9.54.2.18 findAlbum() [2/2]	292
9.54.2.19 findDAlbum()	292
9.54.2.20 findOrCreateTAlbums()	293
9.54.2.21 findPAlbum() [1/2]	293
9.54.2.22 findPAlbum() [2/2]	293
9.54.2.23 findSAlbum() [1/2]	294
9.54.2.24 findSAlbum() [2/2]	294
9.54.2.25 findSAlbumsBySearchType()	294
9.54.2.26 findTAlbum() [1/2]	295
9.54.2.27 findTAlbum() [2/2]	295
9.54.2.28 getDAlbumsCount()	295
9.54.2.29 getFaceCount()	295
9.54.2.30 getItemFromAlbum()	295
9.54.2.31 getPAlbumsCount()	296
9.54.2.32 getRecentlyAssignedTags()	296
9.54.2.33 getTAlbumsCount()	296
9.54.2.34 getUnconfirmedFaceCount()	296
9.54.2.35 isMovingAlbum()	297
9.54.2.36 mergeTAlbum()	297
9.54.2.37 moveTAlbum()	297
9.54.2.38 refresh()	298
9.54.2.39 renamePAlbum()	298
9.54.2.40 renameTAlbum()	298
9.54.2.41 setCurrentAlbums()	299
9.54.2.42 setDatabase()	299
9.54.2.43 signalAlbumAboutToBeMoved	299
9.54.2.44 signalAlbumHasBeenDeleted	299
9.54.2.45 signalAlbumMoved	299
9.54.2.46 signalShowOnlyAvailableAlbumsChanged	299
9.54.2.47 startScan()	300
9.54.2.48 tagNames() [1/2]	300
9.54.2.49 tagNames() [2/2]	300
9.54.2.50 tagPaths() [1/2]	300
9.54.2.51 tagPaths() [2/2]	301
9.54.2.52 updatePAlbumIcon()	301
9.54.2.53 updateSAlbum()	301
9.54.2.54 updateTAlbumIcon()	302
9.55 Digikam::AlbumModel Class Reference	303
9.55.1 Member Function Documentation	309
9.55.1.1 albumData()	309

---

9.55.1.2 albumForId()	309
9.55.1.3 decorationRoleData()	309
9.56 Digikam::AlbumModelDragDropHandler Class Reference	310
9.56.1 Member Function Documentation	311
9.56.1.1 accepts()	311
9.56.1.2 acceptsMimeData()	311
9.56.1.3 createMimeData()	311
9.56.1.4 dropEvent()	311
9.56.1.5 mimeTypes()	311
9.57 Digikam::AlbumModificationHelper Class Reference	312
9.57.1 Detailed Description	313
9.57.2 Constructor & Destructor Documentation	313
9.57.2.1 AlbumModificationHelper()	313
9.57.3 Member Function Documentation	313
9.57.3.1 bindAlbum()	313
9.57.3.2 boundAlbum()	314
9.57.3.3 slotAlbumDelete	314
9.57.3.4 slotAlbumEdit	314
9.57.3.5 slotAlbumNew	314
9.57.3.6 slotAlbumRename	314
9.58 Digikam::AlbumParser Class Reference	315
9.59 Digikam::AlbumPointer< T > Class Template Reference	317
9.59.1 Detailed Description	317
9.60 Digikam::AlbumPointerList< T > Class Template Reference	318
9.61 Digikam::AlbumPropsEdit Class Reference	319
9.62 Digikam::AlbumRootChangeset Class Reference	320
9.63 Digikam::AlbumRootInfo Class Reference	320
9.64 Digikam::AlbumsDBJobInfo Class Reference	321
9.65 Digikam::AlbumsDBJobsThread Class Reference	322
9.65.1 Member Function Documentation	324
9.65.1.1 albumsListing()	324
9.66 Digikam::AlbumSelectComboBox Class Reference	325
9.66.1 Member Function Documentation	328
9.66.1.1 installView()	328
9.66.1.2 model()	328
9.66.1.3 setCheckable()	328
9.66.1.4 setCloseOnActivate()	328
9.66.1.5 setDefaultAlbumModel()	328
9.66.1.6 setNoSelectionText()	328
9.66.1.7 setShowCheckStateSummary()	329
9.66.1.8 updateText	329
9.67 Digikam::AlbumSelectDialog Class Reference	329

---

9.68 Digikam::AlbumSelectionTreeView Class Reference	330
9.68.1 Detailed Description	335
9.68.2 Member Function Documentation	335
9.68.2.1 signalFindDuplicates	335
9.69 Digikam::AlbumSelectors Class Reference	336
9.69.1 Constructor & Destructor Documentation	337
9.69.1.1 AlbumSelectors()	337
9.69.2 Member Function Documentation	338
9.69.2.1 loadState	338
9.69.2.2 saveState	338
9.69.2.3 setAlbumSelected()	338
9.69.2.4 setTagSelected()	338
9.70 Digikam::AlbumSelectTabs Class Reference	339
9.71 Digikam::AlbumSelectTreeView Class Reference	339
9.71.1 Detailed Description	345
9.71.2 Constructor & Destructor Documentation	345
9.71.2.1 AlbumSelectTreeView()	345
9.71.3 Member Function Documentation	345
9.71.3.1 addCustomContextMenuActions()	345
9.71.3.2 handleCustomContextMenuAction()	346
9.72 Digikam::AlbumSelectWidget Class Reference	346
9.73 Digikam::AlbumShortInfo Class Reference	347
9.74 Digikam::AlbumSimplified Class Reference	347
9.74.1 Detailed Description	347
9.75 Digikam::AlbumsJob Class Reference	348
9.76 Digikam::AlbumThumbnailLoader Class Reference	350
9.76.1 Member Enumeration Documentation	352
9.76.1.1 RelativeSize	352
9.76.2 Member Function Documentation	352
9.76.2.1 getAlbumThumbnail()	352
9.76.2.2 getAlbumThumbnailDirectly()	352
9.76.2.3 getStandardTagIcon()	352
9.76.2.4 getTagThumbnail()	352
9.76.2.5 getTagThumbnailDirectly()	353
9.76.2.6 instance()	353
9.76.2.7 setThumbnailSize()	353
9.76.2.8 signalFailed	353
9.76.2.9 signalReloadThumbnails	353
9.76.2.10 signalThumbnail	353
9.77 Digikam::AlbumTreeView Class Reference	354
9.78 Digikam::AlbumTreeViewSelectComboBox Class Reference	359
9.79 Digikam::AlbumWatch Class Reference	362

---

9.80 Digikam::AltLangStrEdit Class Reference	364
9.80.1 Constructor & Destructor Documentation	366
9.80.1.1 AltLangStrEdit()	366
9.80.2 Member Function Documentation	366
9.80.2.1 addCurrent()	366
9.80.2.2 setLinesVisible()	366
9.80.2.3 setTitle()	366
9.80.2.4 setTitleWidget()	366
9.80.2.5 slotEnabledInternalWidgets	367
9.80.2.6 titleWidget()	367
9.81 Digikam::AnimatedClearButton Class Reference	368
9.81.1 Member Function Documentation	369
9.81.1.1 setShallBeShown()	369
9.81.1.2 stayVisibleWhenAnimatedOut()	369
9.82 Digikam::AnimatedVisibility Class Reference	370
9.82.1 Constructor & Destructor Documentation	371
9.82.1.1 AnimatedVisibility()	371
9.83 Digikam::AntiVignettingContainer Class Reference	371
9.84 Digikam::AntiVignettingFilter Class Reference	372
9.84.1 Member Function Documentation	376
9.84.1.1 filterAction()	376
9.84.1.2 filterIdentifier()	376
9.84.1.3 readParameters()	376
9.85 Digikam::AntiVignettingSettings Class Reference	376
9.86 Digikam::ApplicationSettings Class Reference	377
9.86.1 Member Enumeration Documentation	383
9.86.1.1 StringComparisonType	383
9.86.2 Member Function Documentation	383
9.86.2.1 askGroupingOperateOnAll()	383
9.86.2.2 getGroupingOperateOnAll()	383
9.86.2.3 getStringComparisonType()	384
9.86.2.4 operationTypeExplanation()	384
9.86.2.5 operationTypeTitle()	384
9.86.2.6 readMsgBoxShouldBeShown()	385
9.86.2.7 saveMsgBoxShouldBeShown()	385
9.86.2.8 setGroupingOperateOnAll()	385
9.86.2.9 setStringComparisonType()	385
9.87 Digikam::AssignedBatchTools Class Reference	386
9.87.1 Detailed Description	386
9.88 Digikam::AssignedListView Class Reference	387
9.89 Digikam::AssignedListViewItem Class Reference	388
9.90 Digikam::AssignNameOverlay Class Reference	390

---

9.90.1 Member Function Documentation	393
9.90.1.1 checkIndex()	393
9.90.1.2 createWidget()	393
9.90.1.3 setActive()	394
9.90.1.4 setFocusOnWidget()	394
9.90.1.5 showOnIndex()	394
9.90.1.6 updateFace()	394
9.90.1.7 viewportLeaveEvent()	394
9.90.1.8 visualChange()	394
9.90.1.9 widgetEnterEvent()	395
9.90.1.10 widgetLeaveEvent()	395
9.91 Digikam::AssignNameWidget Class Reference	396
9.91.1 Member Function Documentation	398
9.91.1.1 assigned	398
9.91.1.2 rejected	398
9.91.1.3 selected	398
9.91.1.4 setMode()	399
9.91.1.5 setUserData	399
9.92 Digikam::AssignNameWidgetStates Class Reference	400
9.93 Digikam::AudPlayerWdg Class Reference	403
9.94 Digikam::AutoCrop Class Reference	404
9.94.1 Member Function Documentation	408
9.94.1.1 startAnalyse()	408
9.95 Digikam::AutoExpoFilter Class Reference	409
9.95.1 Member Function Documentation	413
9.95.1.1 filterAction()	413
9.95.1.2 filterIdentifier()	414
9.95.1.3 readParameters()	414
9.96 Digikam::AutoLevelsFilter Class Reference	415
9.96.1 Member Function Documentation	419
9.96.1.1 filterAction()	419
9.96.1.2 filterIdentifier()	419
9.96.1.3 readParameters()	419
9.97 Digikam::AutoTagsAssign Class Reference	419
9.97.1 Member Function Documentation	419
9.97.1.1 generateTagsList()	419
9.98 Digikam::AutotagsAssignment Class Reference	420
9.98.1 Constructor & Destructor Documentation	422
9.98.1.1 AutotagsAssignment()	422
9.98.2 Member Function Documentation	423
9.98.2.1 setUseMultiCoreCPU()	423
9.99 Digikam::AutotagsAssignmentTask Class Reference	423

---

9.100 Digikam::AutoTagsScanSettings Class Reference	425
9.100.1 Member Enumeration Documentation	425
9.100.1.1 DetectorModel	425
9.100.1.2 ScanMode	425
9.101 Digikam::AutoTagsScanWidget Class Reference	426
9.101.1 Member Function Documentation	427
9.101.1.1 doLoadState()	427
9.101.1.2 doSaveState()	427
9.102 Digikam::BackendGeonamesRG Class Reference	428
9.102.1 Constructor & Destructor Documentation	429
9.102.1.1 BackendGeonamesRG()	429
9.102.2 Member Function Documentation	429
9.102.2.1 backendName()	429
9.102.2.2 callRGBBackend()	429
9.102.2.3 cancelRequests()	430
9.102.2.4 getErrorMessage()	430
9.102.2.5 makeQMapFromXML()	430
9.103 Digikam::BackendGeonamesUSRG Class Reference	430
9.103.1 Constructor & Destructor Documentation	432
9.103.1.1 BackendGeonamesUSRG()	432
9.103.2 Member Function Documentation	432
9.103.2.1 backendName()	432
9.103.2.2 callRGBBackend()	432
9.103.2.3 cancelRequests()	433
9.103.2.4 getErrorMessage()	433
9.103.2.5 makeQMapFromXML()	433
9.104 Digikam::BackendGoogleMaps Class Reference	434
9.104.1 Constructor & Destructor Documentation	436
9.104.1.1 ~BackendGoogleMaps()	436
9.104.2 Member Function Documentation	436
9.104.2.1 addActionToConfigurationMenu()	436
9.104.2.2 backendHumanName()	437
9.104.2.3 backendName()	437
9.104.2.4 centerOn()	437
9.104.2.5 geoCoordinates()	437
9.104.2.6 getCenter()	437
9.104.2.7 getMarkerModelLevel()	437
9.104.2.8 getNormalizedBounds()	437
9.104.2.9 getZoom()	438
9.104.2.10 isReady()	438
9.104.2.11 mapSize()	438
9.104.2.12 mapWidget()	438

9.104.2.13 mapWidgetDocked()	438
9.104.2.14 mouseModeChanged()	438
9.104.2.15 readSettingsFromGroup()	438
9.104.2.16 regionSelectionChanged()	439
9.104.2.17 releaseWidget()	439
9.104.2.18 reload()	439
9.104.2.19 saveSettingsToGroup()	439
9.104.2.20 screenCoordinates()	439
9.104.2.21 setActive()	439
9.104.2.22 setCenter()	439
9.104.2.23 setMarkerPixmap()	440
9.104.2.24 setZoom()	440
9.104.2.25 updateActionAvailability()	440
9.104.2.26 updateClusters()	440
9.104.2.27 updateMarkers()	440
9.104.2.28 zoomIn()	440
9.104.2.29 zoomOut()	440
9.105 Digikam::BackendMarble Class Reference	441
9.105.1 Constructor & Destructor Documentation	444
9.105.1.1 ~BackendMarble()	444
9.105.2 Member Function Documentation	444
9.105.2.1 addActionToConfigurationMenu()	444
9.105.2.2 applyCacheToWidget()	444
9.105.2.3 backendHumanName()	444
9.105.2.4 backendName()	444
9.105.2.5 centerOn()	444
9.105.2.6 eventFilter()	444
9.105.2.7 geoCoordinates()	445
9.105.2.8 GeoPainter_drawPixmapAtCoordinates()	445
9.105.2.9 getCenter()	445
9.105.2.10 getMarkerModelLevel()	445
9.105.2.11 getNormalizedBounds()	445
9.105.2.12 getProjection()	445
9.105.2.13 getZoom()	446
9.105.2.14 isReady()	446
9.105.2.15 mapSize()	446
9.105.2.16 mapWidget()	446
9.105.2.17 mapWidgetDocked()	446
9.105.2.18 marbleCustomPaint()	446
9.105.2.19 mouseModeChanged()	446
9.105.2.20 readSettingsFromGroup()	446
9.105.2.21 regionSelectionChanged()	447

---

9.105.2.22	releaseWidget()	447
9.105.2.23	reload()	447
9.105.2.24	saveSettingsToGroup()	447
9.105.2.25	screenCoordinates()	447
9.105.2.26	setActive()	447
9.105.2.27	setCenter()	447
9.105.2.28	setZoom()	448
9.105.2.29	slotScheduleUpdate	448
9.105.2.30	updateActionAvailability()	448
9.105.2.31	updateClusters()	448
9.105.2.32	updateMarkers()	448
9.105.2.33	zoomIn()	448
9.105.2.34	zoomOut()	448
9.106	Digikam::BackendMarbleLayer Class Reference	449
9.107	Digikam::BackendOsmRG Class Reference	449
9.107.1	Constructor & Destructor Documentation	451
9.107.1.1	BackendOsmRG()	451
9.107.2	Member Function Documentation	451
9.107.2.1	backendName()	451
9.107.2.2	callRGBBackend()	451
9.107.2.3	cancelRequests()	452
9.107.2.4	getErrorMessage()	452
9.107.2.5	makeQMapFromXML()	452
9.108	Digikam::BalooInfo Class Reference	452
9.109	Digikam::BalooWrap Class Reference	452
9.109.1	Detailed Description	454
9.109.2	Member Function Documentation	454
9.109.2.1	getSemanticInfo()	454
9.109.2.2	setSemanticInfo()	454
9.110	Digikam::BasicDImgFilterGenerator< T > Class Template Reference	455
9.110.1	Constructor & Destructor Documentation	456
9.110.1.1	BasicDImgFilterGenerator()	456
9.110.2	Member Function Documentation	456
9.110.2.1	createFilter()	456
9.110.2.2	displayableName()	456
9.110.2.3	supportedFilters()	456
9.110.2.4	supportedVersions()	456
9.111	Digikam::BatchTool Class Reference	457
9.111.1	Member Enumeration Documentation	460
9.111.1.1	BatchToolGroup	460
9.111.2	Constructor & Destructor Documentation	461
9.111.2.1	BatchTool()	461



9.111.3 Member Function Documentation	461
9.111.3.1 apply()	461
9.111.3.2 applyFilter()	461
9.111.3.3 cancel()	461
9.111.3.4 clone()	461
9.111.3.5 isCancelled()	461
9.111.3.6 outputSuffix()	462
9.111.3.7 registerSettingsWidget()	462
9.111.3.8 savefromDlmg()	462
9.111.3.9 setOutputUrlFromInputUrl()	462
9.111.3.10 setSettings()	462
9.111.3.11 settingsWidget()	462
9.111.3.12 signalAssignSettings2Widget	462
9.111.3.13 slotAssignSettings2Widget	463
9.111.3.14 toolGroup()	463
9.111.3.15 toolOperations()	463
9.111.3.16 toolVersion()	463
9.112 Digikam::BatchToolSet Class Reference	463
9.112.1 Member Function Documentation	464
9.112.1.1 operator==()	464
9.113 Digikam::BatchToolsFactory Class Reference	464
9.114 Digikam::BCGContainer Class Reference	465
9.115 Digikam::BCGFilter Class Reference	466
9.115.1 Member Function Documentation	470
9.115.1.1 filterAction()	470
9.115.1.2 filterIdentifier()	470
9.115.1.3 readParameters()	470
9.116 Digikam::BCGSettings Class Reference	470
9.117 Digikam::BdEngineBackend Class Reference	471
9.117.1 Member Enumeration Documentation	474
9.117.1.1 QueryStateEnum	474
9.117.1.2 Status	475
9.117.2 Constructor & Destructor Documentation	475
9.117.2.1 BdEngineBackend()	475
9.117.3 Member Function Documentation	475
9.117.3.1 asDBDateTime()	475
9.117.3.2 checkOrSetWALMode()	475
9.117.3.3 close()	476
9.117.3.4 connectionErrorHandling()	476
9.117.3.5 execDBAction() [1/2]	476
9.117.3.6 execDBAction() [2/2]	476
9.117.3.7 execDBActionQuery()	476

---

9.117.3.8	execDirectSql()	476
9.117.3.9	execDirectSqlWithResult()	477
9.117.3.10	execQuery()	477
9.117.3.11	execSql() [1/2]	477
9.117.3.12	execSql() [2/2]	477
9.117.3.13	execUpsertDBAction()	477
9.117.3.14	handleQueryResult()	478
9.117.3.15	isInTransaction()	478
9.117.3.16	lastError()	478
9.117.3.17	lastSQLException()	478
9.117.3.18	maximumBoundValues()	478
9.117.3.19	open()	478
9.117.3.20	queryErrorHandling()	479
9.117.3.21	readToList()	479
9.117.3.22	setDbEngineErrorHandler()	479
9.117.3.23	setForeignKeyChecks()	479
9.118	Digikam::BdEngineBackend::QueryState Class Reference	479
9.119	Digikam::BlackFrameListView Class Reference	480
9.120	Digikam::BlackFrameListViewItem Class Reference	481
9.121	Digikam::BlackFrameParser Class Reference	482
9.122	Digikam::BlackFrameToolTip Class Reference	483
9.122.1	Member Function Documentation	484
9.122.1.1	repositionRect()	484
9.122.1.2	tipContents()	484
9.123	Digikam::BlurDetector Class Reference	485
9.123.1	Member Function Documentation	486
9.123.1.1	detect()	486
9.124	Digikam::BlurFilter Class Reference	487
9.124.1	Member Function Documentation	491
9.124.1.1	filterAction()	491
9.124.1.2	filterIdentifier()	491
9.124.1.3	readParameters()	491
9.125	Digikam::BlurFXFilter Class Reference	492
9.125.1	Member Function Documentation	496
9.125.1.1	filterAction()	496
9.125.1.2	filterIdentifier()	496
9.125.1.3	readParameters()	496
9.126	Digikam::BookmarkNode Class Reference	497
9.127	Digikam::BookmarksDialog Class Reference	498
9.128	Digikam::BookmarksManager Class Reference	499
9.129	Digikam::BookmarksMenu Class Reference	500
9.129.1	Member Function Documentation	502

---

9.129.1.1 prePopulated()	502
9.130 Digikam::BookmarksModel Class Reference	503
9.131 Digikam::BorderContainer Class Reference	504
9.132 Digikam::BorderFilter Class Reference	505
9.132.1 Member Function Documentation	509
9.132.1.1 filterAction()	509
9.132.1.2 filterIdentifier()	509
9.132.1.3 readParameters()	509
9.133 Digikam::BorderSettings Class Reference	509
9.134 Digikam::BqmInfolface Class Reference	511
9.135 Digikam::BuildTrashCountersJob Class Reference	514
9.136 Digikam::BWSepiaContainer Class Reference	515
9.136.1 Member Enumeration Documentation	516
9.136.1.1 BlackWhiteConversionType	516
9.137 Digikam::BWSepiaFilter Class Reference	517
9.137.1 Member Function Documentation	521
9.137.1.1 filterAction()	521
9.137.1.2 filterIdentifier()	521
9.137.1.3 readParameters()	521
9.138 Digikam::BWSepiaSettings Class Reference	521
9.139 Digikam::CameraAutoDetectThread Class Reference	522
9.140 Digikam::CameraController Class Reference	524
9.141 Digikam::CameraFolderDialog Class Reference	526
9.142 Digikam::CameraFolderItem Class Reference	527
9.143 Digikam::CameraFolderView Class Reference	528
9.144 Digikam::CameraHistoryUpdater Class Reference	529
9.145 Digikam::CameraInfoDialog Class Reference	530
9.146 Digikam::CameraItem Class Reference	531
9.147 Digikam::CameraItemList Class Reference	532
9.148 Digikam::CameraList Class Reference	533
9.149 Digikam::CameraMessageBox Class Reference	534
9.149.1 Member Function Documentation	534
9.149.1.1 warningContinueCancelList()	534
9.150 Digikam::CameraNameHelper Class Reference	534
9.151 Digikam::CameraNameOption Class Reference	535
9.151.1 Member Function Documentation	537
9.151.1.1 parseOperation()	537
9.152 Digikam::CameraSelection Class Reference	538
9.153 Digikam::CameraThumbsCtrl Class Reference	539
9.153.1 Member Function Documentation	539
9.153.1.1 getThumbInfo()	539
9.154 Digikam::CameraType Class Reference	540

---

9.155 Digikam::CamItemInfo Class Reference	540
9.155.1 Member Enumeration Documentation	541
9.155.1.1 DownloadStatus	541
9.155.2 Member Data Documentation	542
9.155.2.1 downloaded	542
9.155.2.2 size	542
9.156 Digikam::CamItemSortSettings Class Reference	542
9.156.1 Member Enumeration Documentation	543
9.156.1.1 SortOrder	543
9.156.2 Member Function Documentation	543
9.156.2.1 compare()	543
9.156.2.2 compareCategories()	544
9.156.2.3 lessThan() [1/2]	544
9.156.2.4 lessThan() [2/2]	544
9.157 Digikam::Canvas Class Reference	545
9.158 Digikam::CaptionEdit Class Reference	549
9.159 Digikam::CaptionsMap Class Reference	550
9.159.1 Member Function Documentation	551
9.159.1.1 setAuthorsList()	551
9.160 Digikam::CaptionValues Class Reference	552
9.161 Digikam::CaptureDlg Class Reference	552
9.162 Digikam::CaptureWidget Class Reference	553
9.163 Digikam::CaseModifier Class Reference	554
9.163.1 Member Function Documentation	556
9.163.1.1 parseOperation()	556
9.164 Digikam::CategorizedItemModel Class Reference	557
9.164.1 Member Enumeration Documentation	558
9.164.1.1 ExtraRoles	558
9.165 Digikam::CBContainer Class Reference	558
9.166 Digikam::CBFilter Class Reference	559
9.166.1 Member Function Documentation	563
9.166.1.1 filterAction()	563
9.166.1.2 filterIdentifier()	563
9.166.1.3 readParameters()	563
9.167 Digikam::CBSettings Class Reference	563
9.168 Digikam::ChangeBookmarkCommand Class Reference	564
9.169 Digikam::ChangeFaceRecognitionModelDlg Class Reference	565
9.170 Digikam::CharcoalFilter Class Reference	566
9.170.1 Member Function Documentation	570
9.170.1.1 filterAction()	570
9.170.1.2 filterIdentifier()	570
9.170.1.3 readParameters()	570

---

9.171 Digikam::CheckableAlbumFilterModel Class Reference . . . . .	570
9.171.1 Member Function Documentation . . . . .	574
9.171.1.1 isFiltering() . . . . .	574
9.171.1.2 matches() . . . . .	574
9.172 Digikam::ChoiceSearchComboBox Class Reference . . . . .	575
9.172.1 Constructor & Destructor Documentation . . . . .	577
9.172.1.1 ChoiceSearchComboBox() . . . . .	577
9.172.2 Member Function Documentation . . . . .	577
9.172.2.1 installView() . . . . .	577
9.172.2.2 setSearchModel() . . . . .	577
9.173 Digikam::ChoiceSearchModel Class Reference . . . . .	578
9.173.1 Member Function Documentation . . . . .	579
9.173.1.1 checkedKeys() . . . . .	579
9.173.1.2 setChecked() . . . . .	580
9.173.1.3 setChoice() . . . . .	580
9.174 Digikam::ChoiceSearchModel::Entry Class Reference . . . . .	580
9.174.1 Member Function Documentation . . . . .	580
9.174.1.1 operator==( ) . . . . .	580
9.175 Digikam::CIETongueWidget Class Reference . . . . .	581
9.176 Digikam::ClickDragReleaseItem Class Reference . . . . .	582
9.176.1 Member Function Documentation . . . . .	583
9.176.1.1 started . . . . .	583
9.177 Digikam::ClockPhotoDialog Class Reference . . . . .	583
9.177.1 Member Function Documentation . . . . .	584
9.177.1.1 setImage() . . . . .	584
9.178 Digikam::CMat Struct Reference . . . . .	584
9.178.1 Detailed Description . . . . .	584
9.179 Digikam::CollectionImageChangeset Class Reference . . . . .	584
9.179.1 Member Enumeration Documentation . . . . .	585
9.179.1.1 Operation . . . . .	585
9.179.2 Constructor & Destructor Documentation . . . . .	585
9.179.2.1 CollectionImageChangeset() . . . . .	585
9.179.3 Member Function Documentation . . . . .	586
9.179.3.1 ids() . . . . .	586
9.179.3.2 operator<<() . . . . .	586
9.180 Digikam::CollectionLocation Class Reference . . . . .	586
9.180.1 Member Enumeration Documentation . . . . .	587
9.180.1.1 CaseSensitivity . . . . .	587
9.180.1.2 Status . . . . .	587
9.180.1.3 Type . . . . .	587
9.180.2 Member Function Documentation . . . . .	588
9.180.2.1 albumRootPath() . . . . .	588

---

9.180.2.2 asQtCaseSensitivity()	588
9.180.2.3 caseSensitivity()	588
9.180.2.4 status()	588
9.180.2.5 type()	588
9.181 Digikam::CollectionManager Class Reference	589
9.181.1 Member Enumeration Documentation	591
9.181.1.1 LocationCheckResult	591
9.181.2 Member Function Documentation	592
9.181.2.1 addLocation()	592
9.181.2.2 album()	592
9.181.2.3 albumRoot()	592
9.181.2.4 albumRootLabel()	592
9.181.2.5 albumRootPath()	593
9.181.2.6 checkHardWiredLocations()	593
9.181.2.7 checkLocation()	593
9.181.2.8 isAlbumRoot() [1/2]	593
9.181.2.9 isAlbumRoot() [2/2]	593
9.181.2.10 locationForAlbumRoot()	593
9.181.2.11 locationForUrl()	594
9.181.2.12 locationStatusChanged	594
9.181.2.13 migrateToVolume()	594
9.181.2.14 oneAlbumRoot()	594
9.181.2.15 refresh()	594
9.181.2.16 removeLocation()	594
9.181.2.17 setWatchDisabled()	595
9.182 Digikam::CollectionPage Class Reference	595
9.183 Digikam::CollectionScanner Class Reference	597
9.183.1 Member Enumeration Documentation	599
9.183.1.1 FileScanMode	599
9.183.2 Member Function Documentation	600
9.183.2.1 completeScan()	600
9.183.2.2 createHintContainer()	600
9.183.2.3 databaseInitialScanDone()	600
9.183.2.4 finishCompleteScan()	600
9.183.2.5 finishedScanningAlbumRoot	600
9.183.2.6 partialScan()	601
9.183.2.7 scanFile() [1/2]	601
9.183.2.8 scanFile() [2/2]	601
9.183.2.9 scannedFiles	601
9.183.2.10 setNeedFileCount()	601
9.183.2.11 setPerformFastScan()	601
9.183.2.12 setSignalsEnabled()	602

9.183.2.13 totalFilesToScan . . . . .	602
9.184 Digikam::CollectionScannerHintContainer Class Reference . . . . .	602
9.185 Digikam::CollectionScannerObserver Class Reference . . . . .	603
9.186 Digikam::ColorCorrectionDlg Class Reference . . . . .	604
9.187 Digikam::ColorFXContainer Class Reference . . . . .	604
9.188 Digikam::ColorFXFilter Class Reference . . . . .	605
9.188.1 Member Function Documentation . . . . .	609
9.188.1.1 filterAction() . . . . .	609
9.188.1.2 filterIdentifier() . . . . .	609
9.188.1.3 readParameters() . . . . .	609
9.189 Digikam::ColorFXSettings Class Reference . . . . .	609
9.190 Digikam::ColorGradientWidget Class Reference . . . . .	610
9.191 Digikam::ColorLabelFilter Class Reference . . . . .	611
9.192 Digikam::ColorLabelMenuAction Class Reference . . . . .	613
9.193 Digikam::ColorLabelSelector Class Reference . . . . .	614
9.194 Digikam::ColorLabelWidget Class Reference . . . . .	615
9.194.1 Member Function Documentation . . . . .	616
9.194.1.1 setButtonsExclusive() . . . . .	616
9.194.1.2 setColorLabels() . . . . .	617
9.195 Digikam::ComboBoxDelegate Class Reference . . . . .	617
9.195.1 Member Function Documentation . . . . .	618
9.195.1.1 startEditing() . . . . .	618
9.196 Digikam::CommentInfo Class Reference . . . . .	618
9.197 Digikam::CommonKeys Class Reference . . . . .	619
9.197.1 Member Function Documentation . . . . .	620
9.197.1.1 getDbValue() . . . . .	620
9.198 Digikam::CompressionDetector Class Reference . . . . .	621
9.198.1 Member Function Documentation . . . . .	622
9.198.1.1 detect() . . . . .	622
9.199 Digikam::ContentAwareContainer Class Reference . . . . .	622
9.200 Digikam::ContentAwareFilter Class Reference . . . . .	623
9.200.1 Member Function Documentation . . . . .	627
9.200.1.1 filterAction() . . . . .	627
9.200.1.2 filterIdentifier() . . . . .	627
9.200.1.3 readParameters() . . . . .	627
9.201 Digikam::ContextMenuHelper Class Reference . . . . .	627
9.201.1 Detailed Description . . . . .	630
9.201.2 Constructor & Destructor Documentation . . . . .	630
9.201.2.1 ContextMenuHelper() . . . . .	630
9.201.3 Member Function Documentation . . . . .	630
9.201.3.1 addAction() [1/3] . . . . .	630
9.201.3.2 addAction() [2/3] . . . . .	631

---

9.201.3.3 addAction() [3/3]	631
9.201.3.4 addActionNewAlbum()	631
9.201.3.5 addActionNewTag()	631
9.201.3.6 addAlbumCheckUncheckActions()	632
9.201.3.7 addAssignTagsMenu()	632
9.201.3.8 addGotoMenu()	632
9.201.3.9 addGroupMenu()	633
9.201.3.10 addIQSAction()	633
9.201.3.11 addLabelsAction()	633
9.201.3.12 addOpenAndNavigateActions()	633
9.201.3.13 addRemoveAllTags()	634
9.201.3.14 addRemoveTagsMenu()	634
9.201.3.15 addServicesMenu()	634
9.201.3.16 addStandardActionCopy()	635
9.201.3.17 addStandardActionCut()	635
9.201.3.18 addStandardActionItemDelete()	635
9.201.3.19 addStandardActionLightTable()	635
9.201.3.20 addStandardActionPaste()	635
9.201.3.21 addStandardActionThumbnail()	636
9.201.3.22 addSubMenu()	636
9.201.3.23 exec()	636
9.201.3.24 setAlbumModel()	637
9.201.3.25 setItemFilterModel()	637
9.202 Digikam::CoordinatesOverlayWidget Class Reference	637
9.203 Digikam::CopyOrMoveJob Class Reference	638
9.204 Digikam::CopyrightInfo Class Reference	639
9.205 Digikam::CoreDB Class Reference	640
9.205.1 Member Function Documentation	649
9.205.1.1 addAlbum()	649
9.205.1.2 addAlbumRoot()	649
9.205.1.3 addImageMetadata()	650
9.205.1.4 addImageTagProperty()	650
9.205.1.5 addItem()	650
9.205.1.6 addItemInformation()	651
9.205.1.7 addItemPosition()	651
9.205.1.8 addItemTag() [1/2]	651
9.205.1.9 addItemTag() [2/2]	652
9.205.1.10 addSearch()	652
9.205.1.11 addTag()	652
9.205.1.12 addTagProperty()	653
9.205.1.13 addToDownloadHistory()	653
9.205.1.14 addVideoMetadata()	653



---

9.205.1.15 changeImageComment()	653
9.205.1.16 changeImageMetadata()	654
9.205.1.17 changeItemInformation()	654
9.205.1.18 changeItemPosition()	654
9.205.1.19 changeVideoMetadata()	654
9.205.1.20 copyAlbumProperties()	654
9.205.1.21 copyItem()	655
9.205.1.22 databaseUuid()	655
9.205.1.23 deleteAlbum()	655
9.205.1.24 deleteAlbumRoot()	655
9.205.1.25 deleteItem() [1/2]	656
9.205.1.26 deleteItem() [2/2]	656
9.205.1.27 deleteObsoleteItem()	656
9.205.1.28 deleteRemovedItems()	656
9.205.1.29 deleteSearch()	656
9.205.1.30 deleteTag()	657
9.205.1.31 findImageId()	657
9.205.1.32 findInDownloadHistory()	657
9.205.1.33 getAlbumAndSubalbumsForPath()	658
9.205.1.34 getAlbumAverageDate()	658
9.205.1.35 getAlbumForPath()	658
9.205.1.36 getAlbumHighestDate()	659
9.205.1.37 getAlbumLowestDate()	659
9.205.1.38 getAlbumModificationDate()	659
9.205.1.39 getAlbumModificationMap()	659
9.205.1.40 getAlbumRelativePath()	660
9.205.1.41 getAlbumRootId()	660
9.205.1.42 getAlbumRoots()	660
9.205.1.43 getAlbumsOnAlbumRoot()	660
9.205.1.44 getAllItemsWithAlbum()	660
9.205.1.45 getDatabaseEncoding()	661
9.205.1.46 getDirtyOrMissingFacelImageUrls()	661
9.205.1.47 getFilterSettings()	661
9.205.1.48 getIdenticalFiles()	661
9.205.1.49 getImageId()	661
9.205.1.50 getImageIds() [1/4]	661
9.205.1.51 getImageIds() [2/4]	662
9.205.1.52 getImageIds() [3/4]	662
9.205.1.53 getImageIds() [4/4]	662
9.205.1.54 getImageMetadata()	663
9.205.1.55 getImagesFields()	663
9.205.1.56 getImagesRelatedFrom()	663

---

9.205.1.57	getImageRelatingTo()	663
9.205.1.58	getImageTagProperties()	663
9.205.1.59	getItemAlbum()	663
9.205.1.60	getItemCommonTagIDs()	664
9.205.1.61	getItemCopyright()	664
9.205.1.62	getItemFromAlbum()	664
9.205.1.63	getItemIDsAndURLsInAlbum()	664
9.205.1.64	getItemIDsInAlbum()	665
9.205.1.65	getItemIDsInTag()	665
9.205.1.66	getItemInformation()	665
9.205.1.67	getItemName()	666
9.205.1.68	getItemNamesInAlbum()	666
9.205.1.69	getItemPosition()	666
9.205.1.70	getItemTagIDs()	666
9.205.1.71	getItemTagIDs()	667
9.205.1.72	getItemTagNames()	667
9.205.1.73	getItemURLsInAlbum()	667
9.205.1.74	getItemURLsInTag()	668
9.205.1.75	getNumberOfAllItemsAndAlbums()	668
9.205.1.76	getNumberOfItemsInAlbum()	668
9.205.1.77	getOneRelatedImageEach()	668
9.205.1.78	getRecentlyAssignedTags()	669
9.205.1.79	getRelationCloud()	669
9.205.1.80	getSetting()	669
9.205.1.81	getTagsWithProperty()	669
9.205.1.82	getUniqueHashVersion()	669
9.205.1.83	getUserFilterSettings()	669
9.205.1.84	getVideoMetadata()	670
9.205.1.85	hasTags()	670
9.205.1.86	makeStaleAlbum()	670
9.205.1.87	migrateAlbumRoot()	670
9.205.1.88	moveItem()	670
9.205.1.89	removeImageRelation()	671
9.205.1.90	removeImageTagProperties()	671
9.205.1.91	removeItemAllTags()	671
9.205.1.92	removeItemCopyrightProperties()	671
9.205.1.93	removeItems()	672
9.205.1.94	removeItemsFromAlbum()	672
9.205.1.95	removeItemsPermanently()	672
9.205.1.96	removeItemTag()	673
9.205.1.97	removeTagProperties()	673
9.205.1.98	renameItem()	673

---

9.205.1.99 scanAlbums()	673
9.205.1.100 scanSearches()	673
9.205.1.101 scanTags()	674
9.205.1.102 setAlbumCaption()	674
9.205.1.103 setAlbumCategory()	674
9.205.1.104 setAlbumDate()	674
9.205.1.105 setAlbumIcon()	674
9.205.1.106 setAlbumModificationDate()	675
9.205.1.107 setAlbumRootLabel()	675
9.205.1.108 setAlbumRootPath()	675
9.205.1.109 setFilterSettings()	675
9.205.1.110 setImageComment()	676
9.205.1.111 setItemAlbum()	676
9.205.1.112 setItemStatus()	676
9.205.1.113 setSetting()	676
9.205.1.114 setTagIcon()	677
9.205.1.115 setTagName()	677
9.205.1.116 setTagParentID()	677
9.205.1.117 setUserFilterSettings()	677
9.205.1.118 updateItem()	678
9.205.1.119 updateSearch()	678
9.206 Digikam::CoreDbAccess Class Reference	678
9.206.1 Detailed Description	679
9.206.2 Constructor & Destructor Documentation	680
9.206.2.1 CoreDbAccess()	680
9.206.3 Member Function Documentation	680
9.206.3.1 checkReadyForUse()	680
9.206.3.2 cleanUpDatabase()	680
9.206.3.3 setLastError()	680
9.206.3.4 setParameters()	680
9.207 Digikam::CoreDbAccessUnlock Class Reference	681
9.207.1 Constructor & Destructor Documentation	681
9.207.1.1 CoreDbAccessUnlock()	681
9.208 Digikam::CoreDbBackend Class Reference	682
9.208.1 Member Function Documentation	686
9.208.1.1 initSchema()	686
9.209 Digikam::CoreDbCopyManager Class Reference	686
9.210 Digikam::CoreDbDownloadHistory Class Reference	687
9.210.1 Member Function Documentation	687
9.210.1.1 status()	687
9.211 Digikam::CoreDbNameFilter Class Reference	688
9.211.1 Constructor & Destructor Documentation	688

---

9.211.1.1 CoreDbNameFilter() . . . . .	688
9.212 Digikam::CoreDbOperationGroup Class Reference . . . . .	688
9.212.1 Detailed Description . . . . .	688
9.212.2 Member Function Documentation . . . . .	689
9.212.2.1 allowLift() . . . . .	689
9.212.2.2 lift() . . . . .	689
9.213 Digikam::CoreDbPrivilegesChecker Class Reference . . . . .	689
9.214 Digikam::CoreDbSchemaUpdater Class Reference . . . . .	689
9.215 Digikam::CoreDbTransaction Class Reference . . . . .	689
9.215.1 Detailed Description . . . . .	690
9.216 Digikam::CoreDbUrl Class Reference . . . . .	691
9.216.1 Member Function Documentation . . . . .	693
9.216.1.1 album() . . . . .	693
9.216.1.2 albumRoot() . . . . .	693
9.216.1.3 areaCoordinates() . . . . .	693
9.216.1.4 fromAlbumAndName() . . . . .	694
9.216.1.5 fromDateForMonth() . . . . .	694
9.216.1.6 fromDateForYear() . . . . .	694
9.216.1.7 fromDateRange() . . . . .	694
9.216.1.8 fromFileUrl() . . . . .	694
9.216.1.9 fromTagIds() . . . . .	695
9.216.1.10 isAlbumUrl() . . . . .	695
9.216.1.11 name() . . . . .	695
9.216.1.12 parameters() . . . . .	695
9.216.1.13 searchId() . . . . .	695
9.216.1.14 startDate() . . . . .	695
9.216.1.15 tagId() . . . . .	695
9.217 Digikam::CoreDbWatch Class Reference . . . . .	696
9.217.1 Member Function Documentation . . . . .	698
9.217.1.1 databaseChanged . . . . .	698
9.217.1.2 imageChange . . . . .	698
9.218 Digikam::CountrySelector Class Reference . . . . .	698
9.219 Digikam::CurvesBox Class Reference . . . . .	699
9.220 Digikam::CurvesContainer Class Reference . . . . .	700
9.220.1 Constructor & Destructor Documentation . . . . .	701
9.220.1.1 CurvesContainer() . . . . .	701
9.220.2 Member Function Documentation . . . . .	701
9.220.2.1 isEmpty() . . . . .	701
9.220.2.2 isStoredLosslessly() . . . . .	701
9.220.3 Member Data Documentation . . . . .	701
9.220.3.1 curvesType . . . . .	701
9.221 Digikam::CurvesFilter Class Reference . . . . .	702

9.221.1 Member Function Documentation	706
9.221.1.1 filterAction()	706
9.221.1.2 filterIdentifier()	706
9.221.1.3 readParameters()	706
9.222 Digikam::CurvesSettings Class Reference	707
9.223 Digikam::CurvesWidget Class Reference	709
9.223.1 Member Function Documentation	710
9.223.1.1 restoreCurve()	710
9.223.1.2 saveCurve()	711
9.223.1.3 updateData()	711
9.224 Digikam::CustomStepsDoubleSpinBox Class Reference	712
9.224.1 Member Function Documentation	713
9.224.1.1 setSuggestedValues()	713
9.225 Digikam::CustomStepsIntSpinBox Class Reference	713
9.225.1 Member Function Documentation	714
9.225.1.1 setSuggestedValues()	714
9.226 Digikam::DAboutData Class Reference	715
9.227 Digikam::DAbstractSliderSpinBox Class Reference	716
9.227.1 Member Function Documentation	717
9.227.1.1 setBlockUpdateSignalOnDrag()	717
9.227.1.2 setInternalValue()	718
9.228 Digikam::DActiveLabel Class Reference	718
9.229 Digikam::DAdjustableLabel Class Reference	719
9.230 Digikam::DAlbum Class Reference	720
9.230.1 Member Function Documentation	722
9.230.1.1 databaseUrl()	722
9.231 Digikam::DAlbumDrag Class Reference	722
9.231.1 Detailed Description	723
9.232 Digikam::DAlbumInfo Class Reference	723
9.233 Digikam::DArrowClickLabel Class Reference	724
9.234 Digikam::DatabaseCopyThread Class Reference	725
9.235 Digikam::DatabaseFields::DatabaseFieldsEnumIterator< FieldName > Class Template Reference	725
9.235.1 Detailed Description	726
9.236 Digikam::DatabaseFields::DatabaseFieldsEnumIteratorSetOnly< FieldName > Class Template Reference	726
9.237 Digikam::DatabaseFields::FieldMetaInfo< FieldName > Class Template Reference	726
9.238 Digikam::DatabaseFields::Hash< T > Class Template Reference	726
9.238.1 Detailed Description	728
9.239 Digikam::DatabaseFields::Set Class Reference	728
9.240 Digikam::DatabaseLoadSaveFileInfoProvider Class Reference	729
9.240.1 Member Function Documentation	729
9.240.1.1 dimensionsHint()	729

---

9.240.1.2 orientationHint() . . . . .	730
9.241 Digikam::DatabaseMigrationDialog Class Reference . . . . .	730
9.242 Digikam::DatabaseOption Class Reference . . . . .	731
9.242.1 Member Function Documentation . . . . .	733
9.242.1.1 parseOperation() . . . . .	733
9.243 Digikam::DatabaseOptionDialog Class Reference . . . . .	734
9.244 Digikam::DatabasePage Class Reference . . . . .	735
9.245 Digikam::DatabaseServer Class Reference . . . . .	736
9.246 Digikam::DatabaseServerError Class Reference . . . . .	737
9.246.1 Member Enumeration Documentation . . . . .	737
9.246.1.1 DatabaseServerErrorEnum . . . . .	737
9.247 Digikam::DatabaseServerStarter Class Reference . . . . .	738
9.247.1 Member Function Documentation . . . . .	738
9.247.1.1 instance() . . . . .	738
9.248 Digikam::DatabaseSettingsWidget Class Reference . . . . .	739
9.248.1 Member Function Documentation . . . . .	740
9.248.1.1 checkDatabaseSettings() . . . . .	740
9.249 Digikam::DatabaseTask Class Reference . . . . .	740
9.250 Digikam::DatabaseWorkerInterface Class Reference . . . . .	742
9.251 Digikam::DatabaseWriter Class Reference . . . . .	745
9.252 Digikam::DateAlbumModel Class Reference . . . . .	747
9.252.1 Constructor & Destructor Documentation . . . . .	752
9.252.1.1 DateAlbumModel() . . . . .	752
9.252.2 Member Function Documentation . . . . .	753
9.252.2.1 albumForId() . . . . .	753
9.252.2.2 albumName() . . . . .	753
9.252.2.3 decorationRoleData() . . . . .	753
9.252.2.4 monthIndexForDate() . . . . .	753
9.252.2.5 sortRoleData() . . . . .	753
9.253 Digikam::DateFolderView Class Reference . . . . .	754
9.253.1 Member Function Documentation . . . . .	756
9.253.1.1 doLoadState() . . . . .	756
9.253.1.2 doSaveState() . . . . .	756
9.253.1.3 setConfigGroup() . . . . .	756
9.254 Digikam::DateFolderViewSideBarWidget Class Reference . . . . .	757
9.254.1 Member Function Documentation . . . . .	759
9.254.1.1 applySettings() . . . . .	759
9.254.1.2 changeAlbumFromHistory() . . . . .	759
9.254.1.3 doLoadState() . . . . .	759
9.254.1.4 doSaveState() . . . . .	759
9.254.1.5 getCaption() . . . . .	760
9.254.1.6 getIcon() . . . . .	760

9.254.1.7 setActive()	760
9.255 Digikam::DateFormat Class Reference	760
9.256 Digikam::DateOption Class Reference	761
9.256.1 Member Function Documentation	763
9.256.1.1 parseOperation()	763
9.257 Digikam::DateOptionDialog Class Reference	764
9.258 Digikam::DatesDBJobInfo Class Reference	765
9.259 Digikam::DatesDBJobsThread Class Reference	766
9.259.1 Member Function Documentation	768
9.259.1.1 datesListing()	768
9.260 Digikam::DatesJob Class Reference	769
9.261 Digikam::DateTreeView Class Reference	771
9.262 Digikam::DbCleaner Class Reference	776
9.262.1 Member Function Documentation	778
9.262.1.1 setUseMultiCoreCPU()	778
9.263 Digikam::DbEngineAccess Class Reference	779
9.263.1 Member Function Documentation	779
9.263.1.1 checkReadyForUse()	779
9.264 Digikam::DbEngineAction Class Reference	779
9.265 Digikam::DbEngineActionElement Class Reference	779
9.266 Digikam::DbEngineActionType Class Reference	779
9.266.1 Member Function Documentation	780
9.266.1.1 isValue()	780
9.266.1.2 setValue()	780
9.267 Digikam::DbEngineConfig Class Reference	780
9.268 Digikam::DbEngineConfigSettings Class Reference	780
9.269 Digikam::DbEngineConfigSettingsLoader Class Reference	781
9.270 Digikam::DbEngineConnectionChecker Class Reference	781
9.271 Digikam::DbEngineErrorAnswer Class Reference	782
9.272 Digikam::DbEngineErrorHandler Class Reference	783
9.272.1 Member Function Documentation	783
9.272.1.1 connectionError	783
9.272.1.2 consultUserForError	784
9.273 Digikam::DbEngineGuiErrorHandler Class Reference	784
9.274 Digikam::DbEngineLocking Class Reference	785
9.275 Digikam::DbEngineParameters Class Reference	785
9.275.1 Detailed Description	787
9.275.2 Member Function Documentation	787
9.275.2.1 defaultParameters()	787
9.275.2.2 getCoreDatabaseNameOrDir()	788
9.275.2.3 readFromConfig()	788
9.275.2.4 SQLiteDatabaseType()	788

---

9.276 Digikam::DbEngineSqlQuery Class Reference	788
9.277 Digikam::DbHeaderListItem Class Reference	789
9.278 Digikam::DBinaryIface Class Reference	790
9.279 Digikam::DBinarySearch Class Reference	792
9.280 Digikam::DBInfolface Class Reference	793
9.280.1 Member Function Documentation	795
9.280.1.1 albumChooser()	795
9.280.1.2 albumChooserItems()	795
9.280.1.3 albumInfo()	795
9.280.1.4 albumItems()	795
9.280.1.5 albumsItems()	796
9.280.1.6 allAlbumItems()	796
9.280.1.7 currentAlbumItems()	796
9.280.1.8 currentGPSItems()	796
9.280.1.9 currentSelectedItems()	796
9.280.1.10 defaultUploadUrl()	796
9.280.1.11 deleteImage()	796
9.280.1.12 itemInfo()	797
9.280.1.13 openSetupPage()	797
9.280.1.14 parseAlbumItemsRecursive()	797
9.280.1.15 passShortcutActionsToWidget()	797
9.280.1.16 setItemInfo()	797
9.280.1.17 supportAlbums()	797
9.280.1.18 tagFilterModel()	797
9.280.1.19 uploadUrl()	798
9.280.1.20 uploadWidget()	798
9.281 Digikam::DBJob Class Reference	798
9.282 Digikam::DBJobInfo Class Reference	800
9.283 Digikam::DBJobsManager Class Reference	801
9.283.1 Member Function Documentation	802
9.283.1.1 instance()	802
9.283.1.2 startAlbumsJobThread()	802
9.283.1.3 startDatesJobThread()	802
9.283.1.4 startGPSJobThread()	802
9.283.1.5 startSearchesJobThread()	803
9.283.1.6 startTagsJobThread()	803
9.284 Digikam::DBJobsThread Class Reference	804
9.284.1 Member Function Documentation	805
9.284.1.1 connectFinishAndErrorSignals()	805
9.284.1.2 error	806
9.284.1.3 errorsList()	806
9.284.1.4 hasErrors()	806



9.285 Digikam::DbKeysCollection Class Reference . . . . .	806
9.285.1 Detailed Description . . . . .	807
9.285.2 Constructor & Destructor Documentation . . . . .	807
9.285.2.1 DbKeysCollection() . . . . .	807
9.285.3 Member Function Documentation . . . . .	807
9.285.3.1 addId() . . . . .	807
9.285.3.2 collectionName() . . . . .	808
9.285.3.3 getDbValue() . . . . .	808
9.285.3.4 getValue() . . . . .	808
9.285.3.5 ids() . . . . .	809
9.286 Digikam::DbKeySelector Class Reference . . . . .	809
9.287 Digikam::DbKeySelectorItem Class Reference . . . . .	810
9.288 Digikam::DbKeySelectorView Class Reference . . . . .	811
9.289 Digikam::DbShrinkDialog Class Reference . . . . .	812
9.290 Digikam::DBStatDlg Class Reference . . . . .	813
9.291 Digikam::DBusSignalListenerThread Class Reference . . . . .	814
9.292 Digikam::DBusyDlg Class Reference . . . . .	815
9.293 Digikam::DBusyThread Class Reference . . . . .	816
9.294 Digikam::DCameraDragObject Class Reference . . . . .	817
9.294.1 Detailed Description . . . . .	817
9.295 Digikam::DCameraItemDrag Class Reference . . . . .	818
9.295.1 Detailed Description . . . . .	818
9.296 Digikam::DCategorizedSortFilterProxyModel Class Reference . . . . .	819
9.296.1 Detailed Description . . . . .	820
9.296.2 Member Enumeration Documentation . . . . .	820
9.296.2.1 AdditionalRoles . . . . .	820
9.296.3 Member Function Documentation . . . . .	821
9.296.3.1 compareCategories() . . . . .	821
9.296.3.2 isCategorizedModel() . . . . .	821
9.296.3.3 lessThan() . . . . .	822
9.296.3.4 setCategorizedModel() . . . . .	822
9.296.3.5 setSortCategoriesByNaturalComparison() . . . . .	822
9.296.3.6 sort() . . . . .	822
9.296.3.7 sortCategoriesByNaturalComparison() . . . . .	823
9.296.3.8 sortColumn() . . . . .	823
9.296.3.9 sortOrder() . . . . .	823
9.296.3.10 subSortLessThan() . . . . .	823
9.297 Digikam::DCategorizedView Class Reference . . . . .	824
9.297.1 Detailed Description . . . . .	825
9.297.2 Member Function Documentation . . . . .	825
9.297.2.1 categorizedIndexesIn() . . . . .	825
9.297.2.2 categoryAt() . . . . .	826

---

9.297.2.3 categoryRange()	826
9.297.2.4 categoryVisualRect()	826
9.297.2.5 setDrawDraggedItems()	826
9.298 Digikam::DCategoryDrawer Class Reference	827
9.298.1 Detailed Description	828
9.298.2 Member Function Documentation	828
9.298.2.1 actionRequested	828
9.298.2.2 categoryHeight()	829
9.298.2.3 drawCategory()	829
9.298.2.4 leftMargin()	829
9.298.2.5 mouseButtonDoubleClicked()	829
9.298.2.6 mouseButtonPressed()	830
9.298.2.7 mouseButtonReleased()	830
9.298.2.8 mouseLeft()	831
9.298.2.9 mouseMoved()	831
9.298.2.10 rightMargin()	831
9.298.2.11 view()	831
9.299 Digikam::DClickLabel Class Reference	832
9.300 Digikam::DColor Class Reference	833
9.300.1 Constructor & Destructor Documentation	834
9.300.1.1 DColor()	834
9.300.2 Member Function Documentation	834
9.300.2.1 blendZero()	834
9.300.2.2 getHSL()	834
9.300.2.3 getYCbCr()	834
9.300.2.4 premultiply()	834
9.300.2.5 setColor()	835
9.300.2.6 setHSL()	835
9.300.2.7 setPixel()	835
9.300.2.8 setYCbCr()	835
9.301 Digikam::DColorComposer Class Reference	835
9.301.1 Member Enumeration Documentation	836
9.301.1.1 CompositingOperation	836
9.301.2 Member Function Documentation	836
9.301.2.1 compose() [1/2]	836
9.301.2.2 compose() [2/2]	837
9.301.2.3 getComposer()	837
9.302 Digikam::DColorSelector Class Reference	837
9.303 Digikam::DColorValueSelector Class Reference	839
9.303.1 Member Function Documentation	841
9.303.1.1 chooserMode()	841
9.303.1.2 colorValue()	841

---

9.303.1.3 drawContents()	841
9.303.1.4 hue()	842
9.303.1.5 saturation()	842
9.303.1.6 setChooserMode()	842
9.303.1.7 setColorValue()	842
9.303.1.8 setHue()	842
9.303.1.9 setSaturation()	843
9.304 Digikam::DComboBox Class Reference	843
9.305 Digikam::DConfigDlg Class Reference	844
9.305.1 Detailed Description	847
9.305.2 Member Enumeration Documentation	847
9.305.2.1 FaceType	847
9.305.3 Constructor & Destructor Documentation	848
9.305.3.1 DConfigDlg()	848
9.305.4 Member Function Documentation	848
9.305.4.1 addPage() [1/2]	848
9.305.4.2 addPage() [2/2]	848
9.305.4.3 addSubPage() [1/2]	849
9.305.4.4 addSubPage() [2/2]	849
9.305.4.5 currentPage()	850
9.305.4.6 currentPageChanged	850
9.305.4.7 insertPage() [1/2]	850
9.305.4.8 insertPage() [2/2]	851
9.305.4.9 pageRemoved	851
9.305.4.10 removePage()	851
9.305.4.11 setButtonBox()	852
9.305.4.12 setCurrentPage()	852
9.305.4.13 setPageWidget()	852
9.306 Digikam::DConfigDlgMgr Class Reference	852
9.306.1 Detailed Description	855
9.306.2 Constructor & Destructor Documentation	855
9.306.2.1 DConfigDlgMgr()	855
9.306.3 Member Function Documentation	855
9.306.3.1 addWidget()	855
9.306.3.2 getCustomProperty()	855
9.306.3.3 getCustomPropertyChangedSignal()	856
9.306.3.4 init()	856
9.306.3.5 parseChildren()	856
9.306.3.6 settingsChanged [1/2]	856
9.306.3.7 settingsChanged [2/2]	856
9.306.3.8 updateSettings	857
9.306.3.9 updateWidgets	857

---

9.306.3.10 updateWidgetsDefault	857
9.306.3.11 widgetModified	857
9.307 Digikam::DConfigDlgModel Class Reference	858
9.307.1 Detailed Description	859
9.307.2 Member Enumeration Documentation	859
9.307.2.1 Role	859
9.308 Digikam::DConfigDlgTitle Class Reference	859
9.308.1 Detailed Description	861
9.308.2 Member Enumeration Documentation	861
9.308.2.1 ImageAlignment	861
9.308.2.2 MessageType	862
9.308.3 Constructor & Destructor Documentation	862
9.308.3.1 DConfigDlgTitle()	862
9.308.4 Member Function Documentation	862
9.308.4.1 autoHideTimeout()	862
9.308.4.2 comment()	862
9.308.4.3 pixmap()	863
9.308.4.4 setAutoHideTimeout	863
9.308.4.5 setBuddy()	863
9.308.4.6 setComment	863
9.308.4.7 setPixmap [1/4]	864
9.308.4.8 setPixmap [2/4]	864
9.308.4.9 setPixmap [3/4]	864
9.308.4.10 setPixmap [4/4]	865
9.308.4.11 setText [1/2]	865
9.308.4.12 setText [2/2]	865
9.308.4.13 setWidget()	866
9.308.4.14 text()	866
9.309 Digikam::DConfigDlgView Class Reference	866
9.309.1 Detailed Description	869
9.309.2 Member Enumeration Documentation	869
9.309.2.1 FaceType	869
9.309.3 Member Function Documentation	869
9.309.3.1 createView()	869
9.309.3.2 currentPageChanged	869
9.309.3.3 setCurrentPage()	869
9.309.3.4 setItemDelegate()	870
9.309.3.5 setModel()	870
9.309.3.6 showPageHeader()	870
9.309.3.7 viewPosition()	870
9.310 Digikam::DConfigDlgWdg Class Reference	871
9.310.1 Detailed Description	873

---

9.310.2 Constructor & Destructor Documentation	873
9.310.2.1 DConfigDlgWdg()	873
9.310.3 Member Function Documentation	874
9.310.3.1 addPage() [1/2]	874
9.310.3.2 addPage() [2/2]	874
9.310.3.3 addSubPage() [1/2]	874
9.310.3.4 addSubPage() [2/2]	875
9.310.3.5 currentPage()	875
9.310.3.6 currentPageChanged	876
9.310.3.7 insertPage() [1/2]	876
9.310.3.8 insertPage() [2/2]	876
9.310.3.9 pageRemoved	877
9.310.3.10 pageToggled	877
9.310.3.11 removePage()	877
9.310.3.12 setCurrentPage()	878
9.311 Digikam::DConfigDlgWdgItem Class Reference	878
9.311.1 Constructor & Destructor Documentation	880
9.311.1.1 DConfigDlgWdgItem() [1/2]	880
9.311.1.2 DConfigDlgWdgItem() [2/2]	880
9.311.2 Member Function Documentation	880
9.311.2.1 setCheckable()	880
9.311.2.2 setHeader()	881
9.311.2.3 setIcon()	881
9.311.2.4 toggled	881
9.311.3 Property Documentation	881
9.311.3.1 enabled	881
9.312 Digikam::DConfigDlgWdgModel Class Reference	881
9.312.1 Detailed Description	884
9.312.2 Constructor & Destructor Documentation	884
9.312.2.1 DConfigDlgWdgModel()	884
9.312.3 Member Function Documentation	884
9.312.3.1 addPage() [1/2]	884
9.312.3.2 addPage() [2/2]	884
9.312.3.3 addSubPage() [1/2]	885
9.312.3.4 addSubPage() [2/2]	885
9.312.3.5 index()	886
9.312.3.6 insertPage() [1/2]	886
9.312.3.7 insertPage() [2/2]	886
9.312.3.8 item()	887
9.312.3.9 removePage()	887
9.312.3.10 toggled	887
9.313 Digikam::DCursorTracker Class Reference	888

---

9.313.1 Detailed Description	889
9.314 Digikam::DDateEdit Class Reference	889
9.314.1 Detailed Description	890
9.314.2 Member Function Documentation	890
9.314.2.1 assignDate()	890
9.314.2.2 date()	891
9.314.2.3 dateChanged	891
9.314.2.4 isReadOnly()	891
9.314.2.5 setDate	891
9.314.2.6 setReadOnly()	891
9.314.2.7 setupKeywords()	892
9.315 Digikam::DDatePicker Class Reference	892
9.315.1 Constructor & Destructor Documentation	895
9.315.1.1 DDatePicker() [1/2]	895
9.315.1.2 DDatePicker() [2/2]	895
9.315.2 Member Function Documentation	895
9.315.2.1 date()	895
9.315.2.2 dateChanged	895
9.315.2.3 dateEntered	896
9.315.2.4 dateSelected	896
9.315.2.5 dateTable()	896
9.315.2.6 hasCloseButton()	896
9.315.2.7 setCloseButton()	896
9.315.2.8 setDate()	897
9.315.2.9 sizeHint()	897
9.316 Digikam::DDatePickerPopup Class Reference	897
9.316.1 Detailed Description	899
9.316.2 Constructor & Destructor Documentation	899
9.316.2.1 DDatePickerPopup()	899
9.316.3 Member Function Documentation	899
9.316.3.1 datePicker()	899
9.316.3.2 items()	900
9.317 Digikam::DDateTable Class Reference	900
9.317.1 Detailed Description	903
9.317.2 Member Function Documentation	903
9.317.2.1 aboutToShowContextMenu	903
9.317.2.2 date()	903
9.317.2.3 dateChanged	903
9.317.2.4 dateFromPos()	903
9.317.2.5 posFromDate()	904
9.317.2.6 setPopupMenuEnabled()	904
9.317.2.7 sizeHint()	904

---

9.318 Digikam::DDateTimeEdit Class Reference . . . . .	904
9.318.1 Constructor & Destructor Documentation . . . . .	906
9.318.1.1 DDateTimeEdit() . . . . .	906
9.318.2 Member Function Documentation . . . . .	906
9.318.2.1 dateTime() . . . . .	906
9.318.2.2 dateTimeChanged . . . . .	907
9.319 Digikam::DDoubleNumInput Class Reference . . . . .	907
9.320 Digikam::DDoubleSliderSpinBox Class Reference . . . . .	909
9.320.1 Member Function Documentation . . . . .	911
9.320.1.1 setInternalValue() . . . . .	911
9.320.1.2 valueString() . . . . .	911
9.321 Digikam::DefaultRenameParser Class Reference . . . . .	912
9.322 Digikam::DefaultValueDialog Class Reference . . . . .	913
9.323 Digikam::DefaultValueModifier Class Reference . . . . .	914
9.323.1 Member Function Documentation . . . . .	916
9.323.1.1 parseOperation() . . . . .	916
9.324 Digikam::DefaultVersionNamingScheme Class Reference . . . . .	917
9.324.1 Member Function Documentation . . . . .	918
9.324.1.1 baseName() . . . . .	918
9.324.1.2 directory() . . . . .	918
9.324.1.3 incrementedCounter() . . . . .	918
9.324.1.4 initialCounter() . . . . .	919
9.324.1.5 intermediateDirectory() . . . . .	919
9.324.1.6 intermediateFileName() . . . . .	919
9.324.1.7 versionFileName() . . . . .	919
9.325 Digikam::DeleteDialog Class Reference . . . . .	920
9.326 Digikam::DeleteItem Class Reference . . . . .	921
9.327 Digikam::DeleteItemList Class Reference . . . . .	922
9.328 Digikam::DeleteJob Class Reference . . . . .	923
9.329 Digikam::DeleteWidget Class Reference . . . . .	925
9.330 Digikam::DeltaTime Class Reference . . . . .	925
9.331 Digikam::DetByClockPhotoButton Class Reference . . . . .	926
9.332 Digikam::DetectionBenchmarker Class Reference . . . . .	927
9.332.1 Member Function Documentation . . . . .	929
9.332.1.1 result() . . . . .	929
9.333 Digikam::DetectionWorker Class Reference . . . . .	930
9.334 Digikam::DExpanderBox Class Reference . . . . .	932
9.334.1 Member Function Documentation . . . . .	933
9.334.1.1 addItem() . . . . .	933
9.334.1.2 insertItem() . . . . .	934
9.335 Digikam::DExpanderBoxExclusive Class Reference . . . . .	935
9.336 Digikam::DFileDialog Class Reference . . . . .	937

---

9.337 Digikam::DFileOperations Class Reference	938
9.337.1 Member Function Documentation	939
9.337.1.1 findExecutable()	939
9.337.1.2 localFileRename()	939
9.337.1.3 removeAndCopyFile()	939
9.337.1.4 setModificationTime()	939
9.338 Digikam::DFileSelector Class Reference	939
9.338.1 Detailed Description	941
9.339 Digikam::DFontProperties Class Reference	942
9.339.1 Member Enumeration Documentation	944
9.339.1.1 DisplayFlag	944
9.339.1.2 FontColumn	944
9.339.1.3 FontDiff	944
9.339.1.4 FontListCriteria	944
9.339.2 Constructor & Destructor Documentation	944
9.339.2.1 DFontProperties()	944
9.339.3 Member Function Documentation	945
9.339.3.1 backgroundColor()	945
9.339.3.2 color()	945
9.339.3.3 enableColumn()	945
9.339.3.4 font()	946
9.339.3.5 fontDiffFlags()	946
9.339.3.6 getFontList()	946
9.339.3.7 makeColumnVisible()	946
9.339.3.8 sampleText()	947
9.339.3.9 setFont()	947
9.339.3.10 setSampleBoxVisible()	947
9.339.3.11 setSampleText()	947
9.339.3.12 setSizelsRelative()	948
9.339.3.13 sizelsRelative()	948
9.340 Digikam::DFontSelect Class Reference	949
9.341 Digikam::DGradientSlider Class Reference	951
9.342 Digikam::DHBox Class Reference	952
9.343 Digikam::DHistoryView Class Reference	953
9.344 Digikam::DHueSaturationSelector Class Reference	954
9.344.1 Member Function Documentation	956
9.344.1.1 chooserMode()	956
9.344.1.2 colorValue()	956
9.344.1.3 drawContents()	957
9.344.1.4 hue()	957
9.344.1.5 saturation()	957
9.344.1.6 setChooserMode()	957



9.344.1.7 setColorValue()	957
9.344.1.8 setHue()	958
9.344.1.9 setSaturation()	958
9.345 Digikam::DigikamApp Class Reference	959
9.345.1 Member Function Documentation	962
9.345.1.1 infolface()	962
9.346 Digikam::DigikamItemDelegate Class Reference	963
9.346.1 Member Function Documentation	967
9.346.1.1 updateRects()	967
9.347 Digikam::DigikamItemView Class Reference	968
9.347.1 Member Function Documentation	975
9.347.1.1 activated()	975
9.347.1.2 confirmFaces	975
9.347.1.3 hasHiddenGroupedImages()	975
9.347.1.4 rejectFaces	976
9.347.1.5 removeFaces	976
9.347.1.6 setThumbnailSize()	976
9.347.1.7 showContextMenu()	976
9.347.1.8 showContextMenuOnInfo()	976
9.347.1.9 slotSetupChanged()	976
9.348 Digikam::DImageHistory Class Reference	976
9.348.1 Member Function Documentation	978
9.348.1.1 clearReferredImages()	978
9.348.1.2 entries()	978
9.348.1.3 hasActions()	978
9.348.1.4 operator<<()	978
9.348.1.5 purgePathFromReferredImages()	978
9.348.1.6 toXml()	979
9.349 Digikam::DImageHistory::Entry Class Reference	979
9.349.1 Member Data Documentation	979
9.349.1.1 action	979
9.350 Digikam::DImg Class Reference	979
9.350.1 Member Enumeration Documentation	984
9.350.1.1 FORMAT	984
9.350.1.2 PrepareMetadataFlag	984
9.350.2 Constructor & Destructor Documentation	986
9.350.2.1 DImg() [1/4]	986
9.350.2.2 DImg() [2/4]	987
9.350.2.3 DImg() [3/4]	987
9.350.2.4 DImg() [4/4]	987
9.350.3 Member Function Documentation	987
9.350.3.1 addAsReferredImage()	987

---

9.350.3.2 addCurrentUniqueImageId()	987
9.350.3.3 bitBlendImage()	988
9.350.3.4 bitBlendImageOnColor()	988
9.350.3.5 bitBltImage()	988
9.350.3.6 convertDepth()	988
9.350.3.7 copyMetaData()	989
9.350.3.8 createImageUniqueId()	989
9.350.3.9 detach()	989
9.350.3.10 detectedFormat()	989
9.350.3.11 fileOriginData()	989
9.350.3.12 fill()	990
9.350.3.13 format()	990
9.350.3.14 getPixelColor()	990
9.350.3.15 getUniqueHash()	990
9.350.3.16 getUniqueHashVersion()	991
9.350.3.17 hasTransparentPixels()	991
9.350.3.18 imageSavedAs()	991
9.350.3.19 isReadOnly()	991
9.350.3.20 lastSavedFilePath()	991
9.350.3.21 loadItemInfo()	992
9.350.3.22 operator==(())	992
9.350.3.23 originalBitDepth()	992
9.350.3.24 originalColorModel()	992
9.350.3.25 originalFilePath()	992
9.350.3.26 prepareMetadataToSave()	992
9.350.3.27 pureColorMask()	993
9.350.3.28 putImageData() [1/2]	993
9.350.3.29 putImageData() [2/2]	993
9.350.3.30 rawDecodingSettings()	993
9.350.3.31 removeAlphaChannel()	993
9.350.3.32 rotateAndFlip()	993
9.350.3.33 savedFormat()	994
9.350.3.34 setHistoryBranchAfter()	994
9.350.3.35 smoothScale()	994
9.350.3.36 smoothScaleClipped()	994
9.350.3.37 stripImageData()	994
9.350.3.38 transform()	995
9.350.3.39 wasExifRotated()	995
9.351 Digikam::DImgBuiltinFilter Class Reference	995
9.351.1 Member Enumeration Documentation	996
9.351.1.1 Type	996
9.351.2 Constructor & Destructor Documentation	996

9.351.2.1 DImgBuiltinFilter() [1/2]	996
9.351.2.2 DImgBuiltinFilter() [2/2]	996
9.351.3 Member Function Documentation	997
9.351.3.1 filterAction()	997
9.351.3.2 reverseFilter()	997
9.352 Digikam::DImgChildItem Class Reference	998
9.352.1 Constructor & Destructor Documentation	1000
9.352.1.1 DImgChildItem()	1000
9.352.2 Member Function Documentation	1000
9.352.2.1 boundingRect()	1000
9.352.2.2 originalRect()	1000
9.352.2.3 positionChanged	1000
9.352.2.4 positionOnImageChanged	1000
9.352.2.5 rect()	1001
9.352.2.6 relativeRect()	1001
9.352.2.7 setOriginalPos()	1001
9.352.2.8 setPos()	1001
9.352.2.9 setRelativePos()	1001
9.353 Digikam::DImgFilterGenerator Class Reference	1002
9.353.1 Member Function Documentation	1003
9.353.1.1 createFilter()	1003
9.353.1.2 displayableName()	1003
9.353.1.3 isSupported()	1003
9.353.1.4 supportedFilters()	1003
9.353.1.5 supportedVersions()	1003
9.354 Digikam::DImgFilterManager Class Reference	1004
9.354.1 Member Function Documentation	1005
9.354.1.1 createFilter()	1005
9.354.1.2 displayableName()	1005
9.354.1.3 filterIcon()	1006
9.354.1.4 isSupported() [1/2]	1006
9.354.1.5 isSupported() [2/2]	1006
9.354.1.6 supportedFilters()	1006
9.354.1.7 supportedVersions()	1006
9.355 Digikam::DImgLoader Class Reference	1006
9.355.1 Member Enumeration Documentation	1008
9.355.1.1 LoadFlag	1008
9.356 Digikam::DImgLoaderObserver Class Reference	1009
9.356.1 Member Function Documentation	1010
9.356.1.1 granularity()	1010
9.356.1.2 progressInfo()	1010
9.357 Digikam::DImgLoaderSettings Class Reference	1010

---

9.358 Digikam::DImgPreviewItem Class Reference	1012
9.358.1 Member Function Documentation	1014
9.358.1.1 userLoadingHint()	1014
9.359 Digikam::DImgThreadedAnalyser Class Reference	1015
9.359.1 Constructor & Destructor Documentation	1018
9.359.1.1 DImgThreadedAnalyser() [1/2]	1018
9.359.1.2 DImgThreadedAnalyser() [2/2]	1019
9.359.2 Member Function Documentation	1019
9.359.2.1 startAnalyse()	1019
9.360 Digikam::DImgThreadedFilter Class Reference	1019
9.360.1 Constructor & Destructor Documentation	1022
9.360.1.1 DImgThreadedFilter() [1/3]	1022
9.360.1.2 DImgThreadedFilter() [2/3]	1022
9.360.1.3 DImgThreadedFilter() [3/3]	1023
9.360.2 Member Function Documentation	1023
9.360.2.1 cancelFilter()	1023
9.360.2.2 cleanupFilter()	1023
9.360.2.3 filterAction()	1023
9.360.2.4 filterIdentifier()	1024
9.360.2.5 filterImage()	1024
9.360.2.6 finished	1024
9.360.2.7 initFilter()	1024
9.360.2.8 initSlave()	1024
9.360.2.9 modulateProgress()	1025
9.360.2.10 multithreadedSteps()	1025
9.360.2.11 parametersSuccessfullyRead()	1025
9.360.2.12 run()	1025
9.360.2.13 setFilterVersion()	1025
9.360.2.14 setSlave()	1025
9.360.2.15 setupFilter()	1026
9.360.3 Member Data Documentation	1026
9.360.3.1 m_master	1026
9.360.3.2 m_slave	1026
9.361 Digikam::DImgThreadedFilter::DefaultFilterAction< Filter > Class Template Reference	1026
9.362 Digikam::DInfoInterface Class Reference	1030
9.362.1 Member Function Documentation	1032
9.362.1.1 albumChooser()	1032
9.362.1.2 currentSelectedItems()	1032
9.362.1.3 defaultUploadUrl()	1032
9.362.1.4 deleteImage()	1032
9.362.1.5 openSetupPage()	1032
9.362.1.6 passShortcutActionsToWidget()	1032

9.362.1.7 slotDateTimeForUrl()	1033
9.362.1.8 slotMetadataChangedForUrl()	1033
9.362.1.9 tagFilterModel()	1033
9.362.1.10 uploadWidget()	1033
9.363 Digikam::DIntNumInput Class Reference	1034
9.364 Digikam::DIntRangeBox Class Reference	1035
9.364.1 Member Function Documentation	1036
9.364.1.1 maxValue()	1036
9.364.1.2 minValue()	1036
9.364.1.3 setEnabled()	1036
9.364.1.4 setInterval()	1036
9.364.1.5 setRange()	1036
9.364.1.6 setSuffix()	1037
9.365 Digikam::DIO Class Reference	1037
9.365.1 Member Function Documentation	1038
9.365.1.1 copy()	1038
9.366 Digikam::DirectoryNameOption Class Reference	1039
9.366.1 Member Function Documentation	1041
9.366.1.1 parseOperation()	1041
9.367 Digikam::DisjointMetadata Class Reference	1042
9.367.1 Member Enumeration Documentation	1044
9.367.1.1 WriteMode	1044
9.367.2 Member Function Documentation	1044
9.367.2.1 changedFlags()	1044
9.367.2.2 colorLabel()	1044
9.367.2.3 colorLabelInterval()	1045
9.367.2.4 comments()	1045
9.367.2.5 dateTime()	1045
9.367.2.6 dateTimeInterval()	1045
9.367.2.7 keywords()	1045
9.367.2.8 metadataTemplate()	1045
9.367.2.9 pickLabel()	1046
9.367.2.10 pickLabelInterval()	1046
9.367.2.11 rating()	1046
9.367.2.12 ratingInterval()	1046
9.367.2.13 replaceColorLabel()	1046
9.367.2.14 tags()	1046
9.367.2.15 titles()	1047
9.367.2.16 write()	1047
9.368 Digikam::DisjointMetadataDataFields Class Reference	1047
9.368.1 Member Enumeration Documentation	1048
9.368.1.1 Status	1048

---

9.369 Digikam::DistortionFXFilter Class Reference . . . . .	1049
9.369.1 Member Function Documentation . . . . .	1053
9.369.1.1 filterAction() . . . . .	1053
9.369.1.2 filterIdentifier() . . . . .	1053
9.369.1.3 readParameters() . . . . .	1053
9.370 Digikam::DItemDelegate Class Reference . . . . .	1054
9.370.1 Member Function Documentation . . . . .	1055
9.370.1.1 acceptsToolTip() . . . . .	1055
9.370.1.2 gridSize() . . . . .	1055
9.370.1.3 mouseMoved() . . . . .	1056
9.370.1.4 setDefaultViewOptions() . . . . .	1056
9.370.1.5 setThumbnailSize() . . . . .	1056
9.371 Digikam::DItemDrag Class Reference . . . . .	1056
9.371.1 Detailed Description . . . . .	1057
9.372 Digikam::DItemInfo Class Reference . . . . .	1057
9.372.1 Detailed Description . . . . .	1058
9.373 Digikam::DItemsList Class Reference . . . . .	1059
9.373.1 Member Function Documentation . . . . .	1061
9.373.1.1 appendControlButtonsWidget() . . . . .	1061
9.373.1.2 setControlButtonsPlacement() . . . . .	1061
9.373.1.3 setIsLessThanHandler() . . . . .	1061
9.374 Digikam::DItemsListView Class Reference . . . . .	1062
9.375 Digikam::DItemsListViewItem Class Reference . . . . .	1063
9.375.1 Member Function Documentation . . . . .	1064
9.375.1.1 updateItemWidgets() . . . . .	1064
9.376 Digikam::DItemToolTip Class Reference . . . . .	1065
9.376.1 Member Function Documentation . . . . .	1065
9.376.1.1 tipContents() . . . . .	1065
9.377 Digikam::DKCamera Class Reference . . . . .	1066
9.377.1 Member Function Documentation . . . . .	1068
9.377.1.1 capture() . . . . .	1068
9.377.1.2 getFreeSpace() . . . . .	1068
9.377.1.3 getItemsInfoList() . . . . .	1068
9.377.1.4 getPreview() . . . . .	1068
9.378 Digikam::DLabelExpander Class Reference . . . . .	1069
9.379 Digikam::DLineWidget Class Reference . . . . .	1070
9.380 Digikam::DLogoAction Class Reference . . . . .	1071
9.381 Digikam::DMessageBox Class Reference . . . . .	1071
9.381.1 Member Function Documentation . . . . .	1072
9.381.1.1 readMsgBoxShouldBeShown() . . . . .	1072
9.381.1.2 saveMsgBoxShouldBeShown() . . . . .	1072
9.381.1.3 showContinueCancel() . . . . .	1073

9.381.1.4 showContinueCancelList()	1073
9.381.1.5 showContinueCancelWidget()	1073
9.381.1.6 showYesNo()	1073
9.381.1.7 showYesNoList()	1073
9.381.1.8 showYesNoWidget()	1074
9.382 Digikam::DMetadata Class Reference	1075
9.382.1 Member Enumeration Documentation	1090
9.382.1.1 VIDEOCOLOURMODEL	1090
9.382.2 Member Function Documentation	1090
9.382.2.1 addToXmpTagStringBag()	1090
9.382.2.2 getCopyrightInformation()	1090
9.382.2.3 getIccProfile()	1091
9.382.2.4 getItemFacesMap()	1091
9.382.2.5 getLensDescription()	1091
9.382.2.6 getMetadataField()	1092
9.382.2.7 getXmpKeywords()	1092
9.382.2.8 getXmpSubCategories()	1092
9.382.2.9 getXmpSubjects()	1092
9.382.2.10 load()	1092
9.382.2.11 mSecTimeStamp()	1092
9.382.2.12 possibleValuesForEnumField()	1093
9.382.2.13 removeFromXmpTagStringBag()	1093
9.382.2.14 removeXmpKeywords()	1093
9.382.2.15 removeXmpSubCategories()	1093
9.382.2.16 removeXmpSubjects()	1093
9.382.2.17 setItemFacesMap()	1093
9.382.2.18 setXmpKeywords()	1094
9.382.2.19 setXmpSubCategories()	1094
9.382.2.20 setXmpSubjects()	1094
9.382.2.21 valueToString()	1094
9.383 Digikam::DMetadataSettings Class Reference	1095
9.383.1 Member Function Documentation	1096
9.383.1.1 instance()	1096
9.384 Digikam::DMetadataSettingsContainer Class Reference	1096
9.385 Digikam::DMetaInfolface Class Reference	1097
9.385.1 Member Function Documentation	1099
9.385.1.1 allAlbumItems()	1099
9.385.1.2 currentActiveItem()	1099
9.385.1.3 currentAlbumItems()	1099
9.385.1.4 currentGPSItems()	1099
9.385.1.5 currentSelectedItems()	1099
9.385.1.6 defaultUploadUrl()	1100

---

9.385.1.7 deleteImage()	1100
9.385.1.8 itemInfo()	1100
9.385.1.9 parseAlbumItemsRecursive()	1100
9.385.1.10 setItemInfo()	1100
9.385.1.11 slotDateTimeForUrl()	1100
9.385.1.12 slotMetadataChangedForUrl()	1100
9.385.1.13 supportAlbums()	1101
9.385.1.14 uploadUrl()	1101
9.385.1.15 uploadWidget()	1101
9.386 Digikam::DModelFactory Class Reference	1101
9.386.1 Detailed Description	1102
9.387 Digikam::DMultiTabBar Class Reference	1102
9.387.1 Member Enumeration Documentation	1104
9.387.1.1 TextStyle	1104
9.387.2 Member Function Documentation	1105
9.387.2.1 appendButton()	1105
9.387.2.2 appendTab()	1105
9.387.2.3 position()	1105
9.387.2.4 setPosition()	1105
9.387.2.5 setTab()	1106
9.387.2.6 tabStyle()	1106
9.388 Digikam::DMultiTabBarButton Class Reference	1107
9.388.1 Member Function Documentation	1108
9.388.1.1 signalClicked	1108
9.389 Digikam::DMultiTabBarFrame Class Reference	1109
9.390 Digikam::DMultiTabBarTab Class Reference	1110
9.390.1 Member Function Documentation	1112
9.390.1.1 setPosition	1112
9.390.1.2 setState	1112
9.390.1.3 setStyle	1112
9.391 Digikam::DNGConvertSettings Class Reference	1113
9.392 Digikam::DNGSettings Class Reference	1114
9.393 Digikam::DNGWriter Class Reference	1115
9.393.1 Member Enumeration Documentation	1115
9.393.1.1 ConvertError	1115
9.393.1.2 JPEGPreview	1116
9.394 Digikam::DNGWriterHost Class Reference	1116
9.395 Digikam::DNNBaseDetectorModel Class Reference	1117
9.395.1 Member Data Documentation	1118
9.395.1.1 uiConfidenceThreshold	1118
9.396 Digikam::DNNFaceDetectorBase Class Reference	1119
9.396.1 Member Function Documentation	1120



9.396.1.1 selectBbox() . . . . .	1120
9.397 Digikam::DNNFaceDetectorSSD Class Reference . . . . .	1121
9.397.1 Member Function Documentation . . . . .	1122
9.397.1.1 detectFaces() . . . . .	1122
9.398 Digikam::DNNFaceDetectorYOLO Class Reference . . . . .	1123
9.398.1 Member Function Documentation . . . . .	1124
9.398.1.1 detectFaces() . . . . .	1124
9.399 Digikam::DNNFaceDetectorYuNet Class Reference . . . . .	1125
9.399.1 Member Function Documentation . . . . .	1126
9.399.1.1 detectFaces() . . . . .	1126
9.399.1.2 setFaceDetectionSize() . . . . .	1127
9.400 Digikam::DNNFaceExtractorBase Class Reference . . . . .	1127
9.400.1 Member Function Documentation . . . . .	1128
9.400.1.1 getThreshold() . . . . .	1128
9.400.1.2 loadModels() . . . . .	1128
9.401 Digikam::DNNModelBase Class Reference . . . . .	1129
9.401.1 Member Function Documentation . . . . .	1130
9.401.1.1 getThreshold() . . . . .	1130
9.402 Digikam::DNNModelConfig Class Reference . . . . .	1130
9.403 Digikam::DNNModelInfoContainer Class Reference . . . . .	1131
9.404 Digikam::DNNModelManager Class Reference . . . . .	1133
9.404.1 Member Function Documentation . . . . .	1133
9.404.1.1 getModel() . . . . .	1133
9.404.1.2 instance() . . . . .	1134
9.405 Digikam::DNNModelNet Class Reference . . . . .	1134
9.406 Digikam::DNNModelSFace Class Reference . . . . .	1135
9.407 Digikam::DNNModelYuNet Class Reference . . . . .	1137
9.408 Digikam::DNNOpenFaceExtractor Class Reference . . . . .	1139
9.408.1 Member Function Documentation . . . . .	1140
9.408.1.1 alignFace() [1/2] . . . . .	1140
9.408.1.2 alignFace() [2/2] . . . . .	1140
9.408.1.3 getFaceEmbedding() [1/2] . . . . .	1140
9.408.1.4 getFaceEmbedding() [2/2] . . . . .	1140
9.408.1.5 getThreshold() . . . . .	1141
9.408.1.6 loadModels() . . . . .	1141
9.409 Digikam::DNNResnetDetector Class Reference . . . . .	1142
9.409.1 Member Function Documentation . . . . .	1143
9.409.1.1 loadModels() . . . . .	1143
9.410 Digikam::DNNSFaceExtractor Class Reference . . . . .	1144
9.410.1 Member Function Documentation . . . . .	1145
9.410.1.1 alignFace() [1/2] . . . . .	1145
9.410.1.2 alignFace() [2/2] . . . . .	1145

9.410.1.3	getFaceEmbedding() [1/2]	1145
9.410.1.4	getFaceEmbedding() [2/2]	1145
9.410.1.5	getThreshold()	1146
9.410.1.6	loadModels()	1146
9.411	Digikam::DNNYoloDetector Class Reference	1147
9.411.1	Member Function Documentation	1149
9.411.1.1	loadModels()	1149
9.412	Digikam::DNotificationPopup Class Reference	1149
9.412.1	Detailed Description	1153
9.412.2	Member Enumeration Documentation	1153
9.412.2.1	PopupStyle	1153
9.412.3	Member Function Documentation	1153
9.412.3.1	autoDelete()	1153
9.412.3.2	defaultLocation()	1154
9.412.3.3	message() [1/14]	1154
9.412.3.4	message() [2/14]	1154
9.412.3.5	message() [3/14]	1154
9.412.3.6	message() [4/14]	1155
9.412.3.7	message() [5/14]	1155
9.412.3.8	message() [6/14]	1155
9.412.3.9	message() [7/14]	1155
9.412.3.10	message() [8/14]	1156
9.412.3.11	message() [9/14]	1156
9.412.3.12	message() [10/14]	1156
9.412.3.13	message() [11/14]	1157
9.412.3.14	message() [12/14]	1157
9.412.3.15	message() [13/14]	1157
9.412.3.16	message() [14/14]	1157
9.412.3.17	moveNear()	1158
9.412.3.18	positionSelf()	1158
9.412.3.19	setAnchor()	1158
9.412.3.20	setAutoDelete()	1158
9.412.3.21	setPopupStyle	1158
9.412.3.22	setTimeout	1159
9.412.3.23	standardView()	1159
9.413	Digikam::DNotificationWidget Class Reference	1159
9.413.1	Member Enumeration Documentation	1162
9.413.1.1	MessageType	1162
9.413.2	Member Function Documentation	1162
9.413.2.1	addAction()	1162
9.413.2.2	animatedShowTemporized()	1163
9.413.2.3	clearAllActions()	1163

9.413.2.4 heightForWidth()	1163
9.413.2.5 hideAnimationFinished	1163
9.413.2.6 icon()	1164
9.413.2.7 isCloseButtonVisible()	1164
9.413.2.8 isHideAnimationRunning()	1164
9.413.2.9 isShowAnimationRunning()	1164
9.413.2.10 linkActivated	1164
9.413.2.11 linkHovered	1165
9.413.2.12 messageType()	1165
9.413.2.13 removeAction()	1165
9.413.2.14 setCloseButtonVisible	1165
9.413.2.15 setMessageType	1166
9.413.2.16 setText	1166
9.413.2.17 setWordWrap	1166
9.413.2.18 showAnimationFinished	1167
9.413.2.19 text()	1167
9.413.2.20 wordWrap()	1167
9.414 Digikam::DOnlineTranslator Class Reference	1167
9.414.1 Member Enumeration Documentation	1171
9.414.1.1 TranslationError	1171
9.414.2 Constructor & Destructor Documentation	1171
9.414.2.1 DOnlineTranslator()	1171
9.414.3 Member Function Documentation	1172
9.414.3.1 detectLanguage()	1172
9.414.3.2 error()	1172
9.414.3.3 errorString()	1172
9.414.3.4 isRunning()	1172
9.414.3.5 isSourceTranscriptionEnabled()	1173
9.414.3.6 isSourceTranslitEnabled()	1173
9.414.3.7 isSupportTranslation()	1173
9.414.3.8 isTranslationOptionsEnabled()	1173
9.414.3.9 isTranslationTranslitEnabled()	1174
9.414.3.10 language() [1/2]	1174
9.414.3.11 language() [2/2]	1174
9.414.3.12 languageCode()	1174
9.414.3.13 languageName()	1175
9.414.3.14 setEngineApiKey()	1175
9.414.3.15 setEngineUrl()	1175
9.414.3.16 setSourceTranscriptionEnabled()	1175
9.414.3.17 setSourceTranslitEnabled()	1176
9.414.3.18 setTranslationOptionsEnabled()	1176
9.414.3.19 setTranslationTranslitEnabled()	1176

---

9.414.3.20 signalFinished	1176
9.414.3.21 source()	1177
9.414.3.22 sourceLanguage()	1177
9.414.3.23 sourceLanguageName()	1177
9.414.3.24 sourceTranscription()	1177
9.414.3.25 sourceTranslit()	1177
9.414.3.26 toJson()	1177
9.414.3.27 translate()	1177
9.414.3.28 translation()	1178
9.414.3.29 translationLanguage()	1178
9.414.3.30 translationLanguageName()	1178
9.414.3.31 translationOptions()	1178
9.414.3.32 translationTranslit()	1179
9.415 Digikam::DOnlineTranslatorOption Struct Reference	1179
9.415.1 Detailed Description	1179
9.415.2 Member Function Documentation	1180
9.415.2.1 toJson()	1180
9.416 Digikam::DOnlineTts Class Reference	1180
9.416.1 Detailed Description	1181
9.416.2 Member Enumeration Documentation	1181
9.416.2.1 Emotion	1181
9.416.2.2 TtsError	1181
9.416.2.3 Voice	1182
9.416.3 Constructor & Destructor Documentation	1182
9.416.3.1 DOnlineTts()	1182
9.416.4 Member Function Documentation	1182
9.416.4.1 emotion()	1182
9.416.4.2 emotionCode()	1183
9.416.4.3 error()	1183
9.416.4.4 errorString()	1183
9.416.4.5 generateUrls()	1183
9.416.4.6 media()	1184
9.416.4.7 voice()	1184
9.416.4.8 voiceCode()	1184
9.417 Digikam::DownloadInfo Class Reference	1185
9.418 Digikam::DownloadSettings Class Reference	1185
9.419 Digikam::DPixelsAliasFilter Class Reference	1186
9.419.1 Member Function Documentation	1186
9.419.1.1 pixelAntiAliasing()	1186
9.419.1.2 pixelAntiAliasing16()	1186
9.420 Digikam::DPlainTextEdit Class Reference	1187
9.420.1 Detailed Description	1188

9.420.2 Constructor & Destructor Documentation	1188
9.420.2.1 DPlainTextEdit()	1188
9.420.3 Member Function Documentation	1189
9.420.3.1 acceptedCharacters()	1189
9.420.3.2 ignoredCharacters()	1189
9.420.3.3 isClearButtonEnabled()	1189
9.420.3.4 returnPressed	1189
9.420.3.5 setCurrentLanguage()	1189
9.420.3.6 setLinesVisible()	1189
9.420.3.7 setMaxLength()	1190
9.420.3.8 spellCheckSettings()	1190
9.420.3.9 text()	1190
9.421 Digikam::DPlugin Class Reference	1190
9.421.1 Member Function Documentation	1192
9.421.1.1 categories()	1192
9.421.1.2 cleanUp()	1192
9.421.1.3 count()	1192
9.421.1.4 extraAboutData()	1192
9.421.1.5 extraAboutDataRowTitles()	1192
9.421.1.6 extraAboutDataTitle()	1192
9.421.1.7 handbookChapter()	1193
9.421.1.8 handbookReference()	1193
9.421.1.9 handbookSection()	1193
9.421.1.10 hasVisibilityProperty()	1193
9.421.1.11 icon()	1193
9.421.1.12 ifacelid()	1193
9.421.1.13 iid()	1194
9.421.1.14 libraryFileName()	1194
9.421.1.15 name()	1194
9.421.1.16 setLibraryFileName()	1194
9.421.1.17 setShouldLoaded()	1194
9.421.1.18 setVisible()	1194
9.421.1.19 shouldLoaded()	1194
9.421.1.20 version()	1195
9.422 Digikam::DPluginAboutDlg Class Reference	1195
9.423 Digikam::DPluginAction Class Reference	1196
9.423.1 Member Enumeration Documentation	1197
9.423.1.1 ActionCategory	1197
9.423.1.2 ActionType	1197
9.423.2 Member Function Documentation	1198
9.423.2.1 toString()	1198
9.424 Digikam::DPluginAuthor Class Reference	1198

---

9.424.1 Member Function Documentation	1198
9.424.1.1 toString()	1198
9.425 Digikam::DPluginBqm Class Reference	1199
9.425.1 Member Function Documentation	1201
9.425.1.1 categories()	1201
9.425.1.2 count()	1201
9.425.1.3 hasVisibilityProperty()	1201
9.425.1.4 ifacelid()	1202
9.425.1.5 setVisible()	1202
9.426 Digikam::DPluginConfView Class Reference	1202
9.426.1 Member Function Documentation	1203
9.426.1.1 setFilter()	1203
9.426.1.2 signalSearchResult	1203
9.427 Digikam::DPluginConfViewBqm Class Reference	1204
9.427.1 Member Function Documentation	1205
9.427.1.1 loadPlugins()	1205
9.428 Digikam::DPluginConfViewDImg Class Reference	1206
9.428.1 Member Function Documentation	1207
9.428.1.1 loadPlugins()	1207
9.429 Digikam::DPluginConfViewEditor Class Reference	1208
9.429.1 Member Function Documentation	1209
9.429.1.1 loadPlugins()	1209
9.430 Digikam::DPluginConfViewGeneric Class Reference	1210
9.430.1 Member Function Documentation	1211
9.430.1.1 loadPlugins()	1211
9.431 Digikam::DPluginDialog Class Reference	1212
9.432 Digikam::DPluginDImg Class Reference	1213
9.432.1 Member Function Documentation	1215
9.432.1.1 canRead()	1215
9.432.1.2 canWrite()	1215
9.432.1.3 categories()	1215
9.432.1.4 count()	1216
9.432.1.5 exportWidget()	1216
9.432.1.6 extraAboutData()	1216
9.432.1.7 extraAboutDataRowTitles()	1216
9.432.1.8 extraAboutDataTitle()	1216
9.432.1.9 hasVisibilityProperty()	1216
9.432.1.10 ifacelid()	1216
9.432.1.11 loaderName()	1217
9.432.1.12 setVisible()	1217
9.432.1.13 typeMimes()	1217
9.433 Digikam::DPluginEditor Class Reference	1218

9.433.1 Member Function Documentation	1220
9.433.1.1 categories()	1220
9.433.1.2 count()	1220
9.433.1.3 ifacelid()	1220
9.433.1.4 setVisible()	1220
9.434 Digikam::DPluginGeneric Class Reference	1221
9.434.1 Member Function Documentation	1223
9.434.1.1 categories()	1223
9.434.1.2 count()	1223
9.434.1.3 ifacelid()	1223
9.434.1.4 setVisible()	1223
9.435 Digikam::DPluginLoader Class Reference	1224
9.435.1 Detailed Description	1225
9.435.2 Member Function Documentation	1225
9.435.2.1 appendPluginToBlackList()	1225
9.435.2.2 appendPluginToWhiteList()	1226
9.435.2.3 cleanUp()	1226
9.435.2.4 exportWidget()	1226
9.435.2.5 init()	1226
9.435.2.6 instance()	1226
9.435.2.7 pluginAction()	1227
9.435.2.8 pluginActions()	1227
9.435.2.9 pluginsActions() [1/2]	1227
9.435.2.10 pluginsActions() [2/2]	1227
9.436 Digikam::DPluginRawImport Class Reference	1228
9.436.1 Member Function Documentation	1230
9.436.1.1 categories()	1230
9.436.1.2 count()	1230
9.436.1.3 ifacelid()	1230
9.436.1.4 setVisible()	1230
9.437 Digikam::DPluginSetup Class Reference	1231
9.438 Digikam::DPointSelect Class Reference	1232
9.438.1 Member Function Documentation	1233
9.438.1.1 contentsRect()	1233
9.438.1.2 drawContents()	1234
9.438.1.3 setMarkerColor()	1234
9.438.1.4 setValues()	1234
9.438.1.5 setXValue()	1234
9.438.1.6 setYValue()	1234
9.438.1.7 valueChanged	1235
9.438.1.8 xValue()	1235
9.438.1.9 yValue()	1235

---

9.439 Digikam::DPopupFrame Class Reference	1236
9.439.1 Constructor & Destructor Documentation	1237
9.439.1.1 DPopupFrame()	1237
9.439.2 Member Function Documentation	1237
9.439.2.1 close	1237
9.439.2.2 resizeEvent()	1238
9.439.2.3 setMainWidget()	1238
9.440 Digikam::DPreviewImage Class Reference	1239
9.440.1 Member Function Documentation	1240
9.440.1.1 setSelectionArea()	1240
9.440.1.2 slotSetHighlightArea	1241
9.440.1.3 slotSetHighlightShown	1241
9.440.1.4 slotSetSelection	1241
9.441 Digikam::DPreviewManager Class Reference	1242
9.441.1 Member Function Documentation	1243
9.441.1.1 setSelectionArea()	1243
9.442 Digikam::DProgressDlg Class Reference	1244
9.443 Digikam::DProgressWdg Class Reference	1245
9.443.1 Member Function Documentation	1246
9.443.1.1 progressScheduled()	1246
9.444 Digikam::DragDropModelImplementation Class Reference	1247
9.444.1 Constructor & Destructor Documentation	1248
9.444.1.1 DragDropModelImplementation()	1248
9.444.2 Member Function Documentation	1248
9.444.2.1 dragDropFlags()	1248
9.444.2.2 dragDropFlagsV2()	1248
9.444.2.3 supportedDropActions()	1249
9.445 Digikam::DragDropViewImplementation Class Reference	1249
9.445.1 Member Function Documentation	1250
9.445.1.1 dragDropHandler()	1250
9.445.1.2 mapIndexForDragDrop()	1250
9.445.1.3 pixmapForDrag()	1250
9.446 Digikam::DragHandle Class Reference	1251
9.447 Digikam::DRawDecoder Class Reference	1252
9.447.1 Member Function Documentation	1254
9.447.1.1 checkToCancelWaitingData()	1254
9.447.1.2 decodeHalfRAWImage()	1254
9.447.1.3 decodeRAWImage()	1255
9.447.1.4 extractRAWData()	1255
9.447.1.5 librawUseGomp()	1255
9.447.1.6 loadEmbeddedPreview() [1/3]	1255
9.447.1.7 loadEmbeddedPreview() [2/3]	1256



9.447.1.8 loadEmbeddedPreview() [3/3]	1256
9.447.1.9 loadFullImage()	1256
9.447.1.10 loadHalfPreview() [1/3]	1256
9.447.1.11 loadHalfPreview() [2/3]	1256
9.447.1.12 loadHalfPreview() [3/3]	1256
9.447.1.13 loadRawPreview() [1/3]	1257
9.447.1.14 loadRawPreview() [2/3]	1257
9.447.1.15 loadRawPreview() [3/3]	1257
9.447.1.16 rawFileIdentify()	1257
9.447.1.17 rawFilesVersion()	1257
9.447.1.18 setWaitingDataProgress()	1257
9.447.2 Member Data Documentation	1258
9.447.2.1 m_cancel	1258
9.447.2.2 m_decoderSettings	1258
9.448 Digikam::DRawDecoderSettings Class Reference	1258
9.448.1 Member Enumeration Documentation	1260
9.448.1.1 DecodingQuality	1260
9.448.1.2 InputColorSpace	1260
9.448.1.3 NoiseReduction	1260
9.448.1.4 OutputColorSpace	1261
9.448.1.5 WhiteBalance	1261
9.448.2 Member Data Documentation	1261
9.448.2.1 dcbliterations	1261
9.448.2.2 DontStretchPixels	1261
9.448.2.3 expoCorrectionHighlight	1261
9.448.2.4 expoCorrectionShift	1261
9.448.2.5 halfSizeColorImage	1261
9.448.2.6 inputColorSpace	1262
9.448.2.7 NRThreshold	1262
9.448.2.8 outputColorSpace	1262
9.448.2.9 RAWQuality	1262
9.448.2.10 unclipColors	1262
9.448.2.11 whiteBalance	1262
9.449 Digikam::DRawDecoderWidget Class Reference	1263
9.449.1 Constructor & Destructor Documentation	1265
9.449.1.1 DRawDecoderWidget()	1265
9.449.2 Member Function Documentation	1265
9.449.2.1 readSettings()	1265
9.449.2.2 writeSettings()	1265
9.450 Digikam::DRawDecoding Class Reference	1266
9.450.1 Constructor & Destructor Documentation	1266
9.450.1.1 DRawDecoding()	1266

---

9.450.2 Member Data Documentation	1267
9.450.2.1 bcg	1267
9.451 Digikam::DRawInfo Class Reference	1267
9.451.1 Constructor & Destructor Documentation	1270
9.451.1.1 DRawInfo()	1270
9.451.2 Member Data Documentation	1270
9.451.2.1 ambientAcceleration	1270
9.451.2.2 ambientElevationAngle	1270
9.451.2.3 ambientHumidity	1270
9.451.2.4 ambientPressure	1270
9.451.2.5 ambientTemperature	1270
9.451.2.6 ambientWaterDepth	1270
9.451.2.7 baselineExposure	1271
9.451.2.8 DNGVersion	1271
9.451.2.9 exposureIndex	1271
9.451.2.10 exposureProgram	1271
9.451.2.11 flashUsed	1271
9.451.2.12 meteringMode	1271
9.451.2.13 pixelAspectRatio	1271
9.452 Digikam::DSaveSettingsWidget Class Reference	1272
9.453 Digikam::DSelectedItem Class Reference	1273
9.454 Digikam::DSelector Class Reference	1274
9.454.1 Detailed Description	1276
9.454.2 Member Function Documentation	1276
9.454.2.1 arrowDirection()	1276
9.454.2.2 contentsRect()	1277
9.454.2.3 drawContents()	1277
9.454.2.4 indent()	1277
9.454.2.5 setIndent()	1277
9.455 Digikam::DServiceInfo Class Reference	1277
9.456 Digikam::DServiceMenu Class Reference	1278
9.457 Digikam::DSliderSpinBox Class Reference	1279
9.457.1 Member Function Documentation	1281
9.457.1.1 setInternalValue()	1281
9.457.1.2 valueString()	1282
9.458 Digikam::DSplashScreen Class Reference	1282
9.459 Digikam::DSqueezedClickLabel Class Reference	1283
9.460 Digikam::DTagListDrag Class Reference	1284
9.460.1 Detailed Description	1285
9.461 Digikam::DTextBrowser Class Reference	1285
9.462 Digikam::DTextEdit Class Reference	1286
9.462.1 Detailed Description	1287

---

9.462.2 Constructor & Destructor Documentation	1288
9.462.2.1 DTextEdit()	1288
9.462.3 Member Function Documentation	1288
9.462.3.1 acceptedCharacters()	1288
9.462.3.2 ignoredCharacters()	1288
9.462.3.3 isClearButtonEnabled()	1288
9.462.3.4 returnPressed	1288
9.462.3.5 setCurrentLanguage()	1288
9.462.3.6 setLinesVisible()	1289
9.462.3.7 setMaxLength()	1289
9.462.3.8 spellCheckSettings()	1289
9.462.3.9 text()	1289
9.463 Digikam::DTextLabelName Class Reference	1290
9.464 Digikam::DTextLabelValue Class Reference	1291
9.465 Digikam::DTextList Class Reference	1292
9.466 Digikam::DToolTipStyleSheet Class Reference	1292
9.467 Digikam::DTrash Class Reference	1293
9.467.1 Member Function Documentation	1293
9.467.1.1 deleteDirRecursively()	1293
9.467.1.2 deleteImage()	1294
9.467.1.3 extractJsonForItem()	1294
9.468 Digikam::DTrashItemInfo Class Reference	1294
9.469 Digikam::DTrashItemModel Class Reference	1295
9.469.1 Member Function Documentation	1296
9.469.1.1 append	1296
9.469.1.2 changeThumbSize()	1297
9.469.1.3 isEmpty()	1297
9.469.1.4 loadItemsForCollection()	1297
9.469.1.5 pixmapForItem()	1297
9.469.1.6 refreshThumbnails	1297
9.469.1.7 removeItems	1298
9.470 Digikam::DTrashItemsListingJob Class Reference	1299
9.471 Digikam::DuplicatesFinder Class Reference	1301
9.472 Digikam::DuplicatesProgressObserver Class Reference	1304
9.472.1 Member Function Documentation	1304
9.472.1.1 imageProcessed()	1304
9.472.1.2 isCanceled()	1305
9.473 Digikam::DVBox Class Reference	1305
9.474 Digikam::DWItemDelegate Class Reference	1306
9.474.1 Detailed Description	1308
9.474.2 Constructor & Destructor Documentation	1308
9.474.2.1 DWItemDelegate()	1308

---

9.474.3 Member Function Documentation	1308
9.474.3.1 blockedEventTypes()	1308
9.474.3.2 createItemWidgets()	1309
9.474.3.3 focusedIndex()	1309
9.474.3.4 itemView()	1309
9.474.3.5 setBlockedEventTypes()	1309
9.474.3.6 updateItemWidgets()	1310
9.475 Digikam::DWItemDelegatePool Class Reference	1310
9.475.1 Constructor & Destructor Documentation	1311
9.475.1.1 DWItemDelegatePool()	1311
9.475.2 Member Function Documentation	1311
9.475.2.1 findWidgets()	1311
9.476 Digikam::DWItemDelegatePoolPrivate Class Reference	1311
9.477 Digikam::DWizardDlg Class Reference	1312
9.478 Digikam::DWizardPage Class Reference	1312
9.479 Digikam::DWorkingPixmap Class Reference	1313
9.480 Digikam::DXmlGuiWindow Class Reference	1314
9.480.1 Member Function Documentation	1316
9.480.1.1 createFullScreenAction()	1316
9.480.1.2 customizedFullScreenMode()	1316
9.480.1.3 editKeyboardShortcuts()	1316
9.480.1.4 infoface()	1317
9.480.1.5 registerPluginsActions()	1317
9.480.1.6 showSideBars()	1317
9.480.1.7 showThumbBar()	1317
9.480.1.8 thumbbarVisibility()	1317
9.481 Digikam::DynamicLayout Class Reference	1318
9.482 Digikam::DynamicThread Class Reference	1319
9.482.1 Constructor & Destructor Documentation	1320
9.482.1.1 DynamicThread()	1320
9.482.2 Member Function Documentation	1320
9.482.2.1 run()	1320
9.482.2.2 setPriority()	1320
9.482.2.3 shutDown()	1321
9.482.2.4 start()	1321
9.482.2.5 threadMutex()	1321
9.482.2.6 wait	1321
9.483 Digikam::DZoomBar Class Reference	1322
9.483.1 Member Enumeration Documentation	1323
9.483.1.1 BarMode	1323
9.484 Digikam::EditableSearchTreeView Class Reference	1324
9.484.1 Detailed Description	1330

---

9.484.2 Constructor & Destructor Documentation	1330
9.484.2.1 EditableSearchTreeView()	1330
9.484.3 Member Function Documentation	1330
9.484.3.1 addCustomContextMenuActions()	1330
9.484.3.2 contextMenuTitle()	1330
9.484.3.3 handleCustomContextMenuAction()	1331
9.485 Digikam::EditorCore Class Reference	1331
9.486 Digikam::EditorStackView Class Reference	1334
9.487 Digikam::EditorTool Class Reference	1336
9.488 Digikam::EditorTooliface Class Reference	1338
9.489 Digikam::EditorToolSettings Class Reference	1339
9.490 Digikam::EditorToolThreaded Class Reference	1341
9.490.1 Member Function Documentation	1344
9.490.1.1 deleteFilterInstance()	1344
9.490.1.2 setProgressMessage()	1344
9.491 Digikam::EditorWindow Class Reference	1345
9.491.1 Member Function Documentation	1350
9.491.1.1 registerExtraPluginsActions()	1350
9.491.1.2 saveDestinationUrl()	1351
9.491.1.3 toggleZoomActions()	1351
9.491.2 Member Data Documentation	1351
9.491.2.1 m_transformQue	1351
9.492 Digikam::EffectMngr Class Reference	1351
9.492.1 Member Enumeration Documentation	1351
9.492.1.1 EffectType	1351
9.493 Digikam::EffectPreview Class Reference	1352
9.494 Digikam::Ellipsoid Class Reference	1352
9.494.1 Detailed Description	1354
9.494.2 Constructor & Destructor Documentation	1354
9.494.2.1 Ellipsoid()	1354
9.494.3 Member Function Documentation	1355
9.494.3.1 createEllipsoid()	1355
9.494.3.2 createFlattenedSphere()	1355
9.494.3.3 eccentricity()	1355
9.494.3.4 inverseFlattening()	1355
9.494.3.5 isIvfDefinitive()	1356
9.494.3.6 isSphere()	1356
9.494.3.7 orthodromicDistance()	1356
9.494.3.8 radiusOfCurvature()	1356
9.494.3.9 semiMajorAxis()	1357
9.494.3.10 semiMinorAxis()	1357
9.494.3.11 SPHERE()	1357

---

9.494.3.12 WGS84()	1357
9.494.4 Member Data Documentation	1357
9.494.4.1 m_inverseFlattening	1357
9.494.4.2 m_ivfDefinitive	1358
9.494.4.3 m_semiMajorAxis	1358
9.494.4.4 m_semiMinorAxis	1358
9.495 Digikam::EmbossFilter Class Reference	1359
9.495.1 Member Function Documentation	1363
9.495.1.1 filterAction()	1363
9.495.1.2 filterIdentifier()	1363
9.495.1.3 readParameters()	1363
9.496 Digikam::EmptyDTrashItemsJob Class Reference	1364
9.497 Digikam::EmptyImageListProvider Class Reference	1366
9.497.1 Member Function Documentation	1367
9.497.1.1 atEnd()	1367
9.497.1.2 image()	1367
9.497.1.3 images()	1367
9.497.1.4 proceed()	1367
9.497.1.5 setImages()	1367
9.497.1.6 setUnpairedImages()	1367
9.497.1.7 size()	1367
9.498 Digikam::EqualizeFilter Class Reference	1368
9.498.1 Member Function Documentation	1372
9.498.1.1 filterAction()	1372
9.498.1.2 filterIdentifier()	1372
9.498.1.3 readParameters()	1372
9.499 Digikam::ExifMetaEngineMergeHelper Class Reference	1372
9.500 Digikam::ExifToolBinary Class Reference	1374
9.501 Digikam::ExifToolConfPanel Class Reference	1376
9.502 Digikam::ExifToolErrorView Class Reference	1377
9.503 Digikam::ExifToolListView Class Reference	1378
9.503.1 Member Function Documentation	1379
9.503.1.1 setGroupList()	1379
9.504 Digikam::ExifToolListViewGroup Class Reference	1379
9.505 Digikam::ExifToolListViewItem Class Reference	1380
9.506 Digikam::ExifToolLoadingView Class Reference	1381
9.507 Digikam::ExifToolParser Class Reference	1382
9.507.1 Member Typedef Documentation	1384
9.507.1.1 ExifToolData	1384
9.507.2 Member Function Documentation	1384
9.507.2.1 applyChanges() [1/2]	1384
9.507.2.2 applyChanges() [2/2]	1384

---

9.507.2.3 applyMetadataFile()	1385
9.507.2.4 changeTimestamps()	1385
9.507.2.5 copyTags()	1385
9.507.2.6 load()	1386
9.507.2.7 loadChunk()	1386
9.507.2.8 readableFormats()	1386
9.507.2.9 tagsDatabase()	1386
9.507.2.10 tagsDbToOrderedMap()	1386
9.507.2.11 translateTags()	1386
9.507.2.12 translationsList()	1387
9.507.2.13 version()	1387
9.507.2.14 writableFormats()	1387
9.508 Digikam::ExifToolProcess Class Reference	1388
9.508.1 Member Enumeration Documentation	1390
9.508.1.1 Action	1390
9.508.1.2 CopyTagsSource	1390
9.508.1.3 TranslateTagsOps	1390
9.508.1.4 WritingTagsMode	1391
9.508.2 Constructor & Destructor Documentation	1391
9.508.2.1 ~ExifToolProcess()	1391
9.508.3 Member Function Documentation	1391
9.508.3.1 command()	1391
9.508.3.2 initExifTool()	1391
9.508.3.3 setExifToolProgram()	1391
9.508.3.4 shutDownExifTool()	1391
9.508.3.5 waitForExifToolResult()	1392
9.509 Digikam::ExifToolProcess::Result Class Reference	1392
9.510 Digikam::ExifToolThread Class Reference	1392
9.511 Digikam::ExifToolWidget Class Reference	1393
9.512 Digikam::ExifWidget Class Reference	1395
9.512.1 Member Function Documentation	1397
9.512.1.1 getMetadataTitle()	1397
9.512.1.2 getTagDescription()	1397
9.512.1.3 getTagTitle()	1397
9.512.1.4 loadFromURL()	1397
9.513 Digikam::ExposureDetector Class Reference	1398
9.513.1 Member Function Documentation	1399
9.513.1.1 detect()	1399
9.514 Digikam::ExposureSettingsContainer Class Reference	1399
9.514.1 Member Data Documentation	1399
9.514.1.1 exposureIndicatorMode	1399
9.515 Digikam::FaceClassifier Class Reference	1400

---

9.515.1 Member Function Documentation	1401
9.515.1.1 loadTrainingData()	1401
9.515.1.2 predict() [1/2]	1401
9.515.1.3 predict() [2/2]	1401
9.515.1.4 retrain()	1402
9.516 Digikam::FaceClassifierBase Class Reference	1402
9.517 Digikam::FaceDb Class Reference	1403
9.517.1 Member Function Documentation	1404
9.517.1.1 clearDNNTraining()	1404
9.517.1.2 insertFaceVector()	1404
9.517.1.3 removeFaceVector() [1/2]	1404
9.517.1.4 removeFaceVector() [2/2]	1404
9.517.1.5 trainData()	1405
9.518 Digikam::FaceDbAccess Class Reference	1405
9.518.1 Constructor & Destructor Documentation	1405
9.518.1.1 FaceDbAccess()	1405
9.518.2 Member Function Documentation	1406
9.518.2.1 setLastError()	1406
9.519 Digikam::FaceDbAccessUnlock Class Reference	1406
9.519.1 Constructor & Destructor Documentation	1406
9.519.1.1 FaceDbAccessUnlock()	1406
9.520 Digikam::FaceDbBackend Class Reference	1407
9.520.1 Member Function Documentation	1411
9.520.1.1 initSchema()	1411
9.521 Digikam::FaceDbOperationGroup Class Reference	1411
9.521.1 Detailed Description	1411
9.521.2 Member Function Documentation	1411
9.521.2.1 allowLift()	1411
9.521.2.2 lift()	1411
9.522 Digikam::FaceDbSchemaUpdater Class Reference	1412
9.523 Digikam::FaceDetector Class Reference	1412
9.523.1 Constructor & Destructor Documentation	1412
9.523.1.1 FaceDetector()	1412
9.523.2 Member Function Documentation	1413
9.523.2.1 detectFaces() [1/2]	1413
9.523.2.2 detectFaces() [2/2]	1413
9.523.2.3 recommendedImageSize()	1413
9.523.2.4 setParameter()	1413
9.524 Digikam::FaceGroup Class Reference	1414
9.524.1 Member Function Documentation	1416
9.524.1.1 aboutToSetInfo	1416
9.524.1.2 closestItem()	1416



9.524.1.3 setAutoSuggest()	1416
9.525 Digikam::Faceltem Class Reference	1417
9.526 Digikam::FaceltemRetriever Class Reference	1420
9.527 Digikam::FacePipeline Class Reference	1421
9.527.1 Member Enumeration Documentation	1423
9.527.1.1 FilterMode	1423
9.527.1.2 WriteMode	1423
9.527.2 Member Function Documentation	1424
9.527.2.1 confirm	1424
9.527.2.2 editRegion	1424
9.527.2.3 editTag	1424
9.527.2.4 plugDatabaseFilter()	1425
9.527.2.5 process [1/2]	1425
9.527.2.6 process [2/2]	1425
9.527.2.7 setPriority()	1425
9.528 Digikam::FacePipelineBase Class Reference	1426
9.528.1 Member Enumeration Documentation	1428
9.528.1.1 FilterMode	1428
9.528.1.2 WriteMode	1429
9.528.2 Member Function Documentation	1429
9.528.2.1 enqueue()	1429
9.529 Digikam::FacePipelineDetect Class Reference	1430
9.529.1 Member Function Documentation	1433
9.529.1.1 addMoreWorkers()	1433
9.529.1.2 classifier()	1433
9.529.1.3 extractor()	1433
9.529.1.4 finder()	1433
9.529.1.5 loader()	1433
9.529.1.6 start()	1433
9.529.1.7 trainer()	1433
9.529.1.8 writer()	1434
9.530 Digikam::FacePipelineDetectRecognize Class Reference	1434
9.530.1 Member Function Documentation	1437
9.530.1.1 addMoreWorkers()	1437
9.530.1.2 classifier()	1437
9.530.1.3 extractor()	1437
9.530.1.4 finder()	1437
9.530.1.5 loader()	1437
9.530.1.6 start()	1437
9.530.1.7 trainer()	1437
9.530.1.8 writer()	1438
9.531 Digikam::FacePipelineEdit Class Reference	1438

9.531.1 Member Function Documentation	1441
9.531.1.1 addMoreWorkers()	1441
9.531.1.2 classifier()	1441
9.531.1.3 extractor()	1441
9.531.1.4 finder()	1442
9.531.1.5 loader()	1442
9.531.1.6 start()	1442
9.531.1.7 trainer()	1442
9.531.1.8 writer()	1442
9.532 Digikam::FacePipelineExtendedPackage Class Reference	1443
9.533 Digikam::FacePipelineFaceTagsIface Class Reference	1445
9.533.1 Member Enumeration Documentation	1447
9.533.1.1 Role	1447
9.534 Digikam::FacePipelineFaceTagsIfaceList Class Reference	1448
9.535 Digikam::FacePipelinePackage Class Reference	1449
9.536 Digikam::FacePipelinePackageBase Class Reference	1450
9.537 Digikam::FacePipelineRecognize Class Reference	1452
9.537.1 Member Function Documentation	1455
9.537.1.1 addMoreWorkers()	1455
9.537.1.2 classifier()	1455
9.537.1.3 extractor()	1455
9.537.1.4 finder()	1455
9.537.1.5 loader()	1455
9.537.1.6 start()	1455
9.537.1.7 trainer()	1455
9.537.1.8 writer()	1456
9.538 Digikam::FacePipelineReset Class Reference	1456
9.538.1 Member Function Documentation	1459
9.538.1.1 addMoreWorkers()	1459
9.538.1.2 classifier()	1459
9.538.1.3 extractor()	1459
9.538.1.4 finder()	1459
9.538.1.5 loader()	1459
9.538.1.6 start()	1459
9.538.1.7 trainer()	1459
9.538.1.8 writer()	1460
9.539 Digikam::FacePipelineRetrain Class Reference	1460
9.539.1 Member Function Documentation	1463
9.539.1.1 addMoreWorkers()	1463
9.539.1.2 classifier()	1463
9.539.1.3 extractor()	1463
9.539.1.4 finder()	1463

---

9.539.1.5 loader()	1463
9.539.1.6 start()	1463
9.539.1.7 trainer()	1463
9.539.1.8 writer()	1464
9.540 Digikam::FacePreprocessor Class Reference	1464
9.541 Digikam::FacePreviewLoader Class Reference	1465
9.542 Digikam::FaceRejectionOverlay Class Reference	1471
9.542.1 Member Function Documentation	1474
9.542.1.1 checkIndex()	1474
9.542.1.2 createButton()	1474
9.542.1.3 setActive()	1474
9.542.1.4 updateButton()	1475
9.542.1.5 widgetEnterEvent()	1475
9.542.1.6 widgetLeaveEvent()	1475
9.543 Digikam::FaceRejectionOverlayButton Class Reference	1476
9.543.1 Member Function Documentation	1477
9.543.1.1 icon()	1477
9.543.1.2 sizeHint()	1478
9.543.1.3 updateToolTip()	1478
9.544 Digikam::FaceScanSettings Class Reference	1478
9.544.1 Member Enumeration Documentation	1479
9.544.1.1 AlreadyScannedHandling	1479
9.544.1.2 FaceDetectionModel	1479
9.544.1.3 FaceRecognitionModel	1479
9.544.1.4 ScanTask	1479
9.544.2 Member Data Documentation	1480
9.544.2.1 detectAccuracy	1480
9.544.2.2 recognizeAccuracy	1480
9.545 Digikam::FaceScanWidget Class Reference	1480
9.545.1 Member Function Documentation	1482
9.545.1.1 doLoadState()	1482
9.545.1.2 doSaveState()	1482
9.546 Digikam::FacesDetector Class Reference	1483
9.547 Digikam::FacesEngine Class Reference	1486
9.548 Digikam::FaceTags Class Reference	1489
9.548.1 Member Function Documentation	1489
9.548.1.1 applyTagIdentityMapping()	1489
9.548.1.2 ensureIsPerson()	1490
9.548.1.3 getOrCreateTagForPerson()	1490
9.548.1.4 tagForPerson()	1490
9.549 Digikam::FaceTagsEditor Class Reference	1491
9.549.1 Member Function Documentation	1493

---

9.549.1.1	add()	1493
9.549.1.2	addNormalTag()	1493
9.549.1.3	changeRegion()	1493
9.549.1.4	changeTag()	1493
9.549.1.5	confirmName()	1494
9.549.1.6	getSuggestedNames()	1494
9.549.1.7	getTagRects()	1494
9.549.1.8	removeFace()	1494
9.549.1.9	removeNormalTag()	1494
9.549.1.10	unconfirmedEntry()	1495
9.549.1.11	unconfirmedFaceTagslfaces()	1495
9.549.1.12	unconfirmedNameFaceTagslfaces()	1495
9.550	Digikam::FaceTagslface Class Reference	1496
9.550.1	Member Function Documentation	1498
9.550.1.1	fromVariant()	1498
9.550.1.2	typeForAttribute()	1498
9.551	Digikam::FaceUtils Class Reference	1499
9.551.1	Member Function Documentation	1501
9.551.1.1	addNormalTag()	1501
9.551.1.2	faceRectToDisplayRect()	1502
9.551.1.3	removeNormalTag()	1502
9.551.1.4	removeNormalTags()	1502
9.551.1.5	storeThumbnails()	1502
9.551.1.6	toFaceTagslfaces()	1502
9.551.1.7	writeUnconfirmedResults()	1503
9.552	Digikam::FacialRecognitionWrapper Class Reference	1503
9.552.1	Member Function Documentation	1504
9.552.1.1	addIdentity()	1504
9.552.1.2	allIdentities()	1504
9.552.1.3	findIdentity() [1/2]	1504
9.552.1.4	findIdentity() [2/2]	1504
9.552.1.5	recognizeFaces()	1505
9.552.1.6	setParameter()	1505
9.552.1.7	train() [1/2]	1505
9.552.1.8	train() [2/2]	1505
9.553	Digikam::FFmpegBinary Class Reference	1506
9.554	Digikam::FFmpegConfigHelper Class Reference	1508
9.554.1	Member Function Documentation	1509
9.554.1.1	getAudioCodecsProperties()	1509
9.554.1.2	getExtensionsProperties()	1509
9.554.1.3	getVideoCodecsProperties()	1509
9.555	Digikam::FFmpegLauncher Class Reference	1510

---

9.555.1 Member Function Documentation	1511
9.555.1.1 soundTrackLength()	1511
9.556 Digikam::FieldQueryBuilder Class Reference	1512
9.557 Digikam::FileActionItemInfoList Class Reference	1513
9.558 Digikam::FileActionMngr Class Reference	1515
9.558.1 Member Function Documentation	1516
9.558.1.1 transform	1516
9.559 Digikam::FileActionMngrDatabaseWorker Class Reference	1517
9.559.1 Member Function Documentation	1519
9.559.1.1 applyMetadata()	1519
9.559.1.2 assignColorLabel()	1519
9.559.1.3 assignPickLabel()	1519
9.559.1.4 assignRating()	1520
9.559.1.5 assignTags()	1520
9.559.1.6 copyAttributes()	1520
9.559.1.7 editGroup()	1520
9.559.1.8 removeTags()	1520
9.559.1.9 setExifOrientation()	1520
9.560 Digikam::FileActionMngrFileWorker Class Reference	1521
9.560.1 Member Function Documentation	1523
9.560.1.1 transform()	1523
9.560.1.2 writeMetadata()	1523
9.560.1.3 writeMetadataToFiles()	1523
9.560.1.4 writeOrientationToFiles()	1524
9.561 Digikam::FileActionProgress Class Reference	1524
9.562 Digikam::FileActionProgressItemContainer Class Reference	1527
9.563 Digikam::FileActionProgressItemCreator Class Reference	1528
9.564 Digikam::FilePropertiesOption Class Reference	1529
9.564.1 Member Function Documentation	1531
9.564.1.1 parseOperation()	1531
9.565 Digikam::FileReadLocker Class Reference	1531
9.566 Digikam::FileReadWriteLockKey Class Reference	1531
9.567 Digikam::FileSaveConflictBox Class Reference	1532
9.568 Digikam::FileSaveOptionsBox Class Reference	1533
9.568.1 Member Enumeration Documentation	1533
9.568.1.1 FORMAT	1533
9.568.2 Constructor & Destructor Documentation	1534
9.568.2.1 FileSaveOptionsBox()	1534
9.568.3 Member Function Documentation	1534
9.568.3.1 discoverFormat()	1534
9.569 Digikam::FileSaveOptionsDlg Class Reference	1535
9.570 Digikam::FilesDownloader Class Reference	1536

---

9.571 Digikam::FileWorkerInterface Class Reference . . . . .	1537
9.572 Digikam::FileWriteLocker Class Reference . . . . .	1539
9.573 Digikam::FilmContainer Class Reference . . . . .	1539
9.574 Digikam::FilmContainer::ListItem Class Reference . . . . .	1540
9.575 Digikam::FilmFilter Class Reference . . . . .	1541
9.575.1 Member Function Documentation . . . . .	1545
9.575.1.1 filterAction() . . . . .	1545
9.575.1.2 filterIdentifier() . . . . .	1545
9.575.1.3 readParameters() . . . . .	1545
9.576 Digikam::FilmGrainContainer Class Reference . . . . .	1545
9.577 Digikam::FilmGrainFilter Class Reference . . . . .	1546
9.577.1 Member Function Documentation . . . . .	1550
9.577.1.1 filterAction() . . . . .	1550
9.577.1.2 filterIdentifier() . . . . .	1550
9.577.1.3 readParameters() . . . . .	1550
9.578 Digikam::FilmGrainSettings Class Reference . . . . .	1550
9.579 Digikam::Filter Class Reference . . . . .	1551
9.580 Digikam::FilterAction Class Reference . . . . .	1552
9.580.1 Member Enumeration Documentation . . . . .	1554
9.580.1.1 Category . . . . .	1554
9.580.1.2 Flag . . . . .	1555
9.580.2 Member Function Documentation . . . . .	1555
9.580.2.1 description() . . . . .	1555
9.580.2.2 hasParameters() . . . . .	1555
9.580.2.3 identifier() . . . . .	1555
9.580.2.4 parameter() . . . . .	1556
9.580.2.5 version() . . . . .	1556
9.580.3 Member Data Documentation . . . . .	1556
9.580.3.1 m_category . . . . .	1556
9.581 Digikam::FilterActionFilter Class Reference . . . . .	1557
9.581.1 Member Function Documentation . . . . .	1561
9.581.1.1 appliedFilterActions() . . . . .	1561
9.581.1.2 completelyApplied() . . . . .	1561
9.581.1.3 filterAction() . . . . .	1561
9.581.1.4 filterIdentifier() . . . . .	1562
9.581.1.5 filterImage() . . . . .	1562
9.581.1.6 isComplexAction() . . . . .	1562
9.581.1.7 readParameters() . . . . .	1562
9.581.1.8 setContinueOnError() . . . . .	1562
9.582 Digikam::FiltersHistoryWidget Class Reference . . . . .	1563
9.583 Digikam::FilterSideBarWidget Class Reference . . . . .	1564
9.583.1 Detailed Description . . . . .	1566

---

9.583.2 Constructor & Destructor Documentation	1566
9.583.2.1 FilterSideBarWidget()	1566
9.583.3 Member Function Documentation	1566
9.583.3.1 doLoadState()	1566
9.583.3.2 doSaveState()	1567
9.583.3.3 setConfigGroup()	1567
9.583.3.4 signalTagFilterChanged	1567
9.584 Digikam::FilterStatusBar Class Reference	1568
9.585 Digikam::FindDuplicatesAlbum Class Reference	1568
9.586 Digikam::FindDuplicatesAlbumItem Class Reference	1570
9.587 Digikam::FindDuplicatesView Class Reference	1571
9.588 Digikam::FingerPrintsGenerator Class Reference	1572
9.588.1 Constructor & Destructor Documentation	1575
9.588.1.1 FingerPrintsGenerator()	1575
9.588.2 Member Function Documentation	1575
9.588.2.1 setUseMultiCoreCPU()	1575
9.589 Digikam::FingerprintsTask Class Reference	1576
9.590 Digikam::FirstRunDlg Class Reference	1578
9.591 Digikam::FocusPoint Class Reference	1578
9.591.1 Member Enumeration Documentation	1579
9.591.1.1 TypePoint	1579
9.591.2 Constructor & Destructor Documentation	1579
9.591.2.1 FocusPoint()	1579
9.591.3 Member Function Documentation	1579
9.591.3.1 setType()	1579
9.592 Digikam::FocusPointGroup Class Reference	1580
9.593 Digikam::FocusPointItem Class Reference	1582
9.594 Digikam::FocusPointsExtractor Class Reference	1585
9.594.1 Member Typedef Documentation	1586
9.594.1.1 ListAFPoints	1586
9.595 Digikam::FocusPointsWriter Class Reference	1586
9.596 Digikam::FrameOsd Class Reference	1586
9.597 Digikam::FrameOsdSettings Class Reference	1587
9.598 Digikam::FrameOsdWidget Class Reference	1588
9.599 Digikam::FrameUtils Class Reference	1588
9.600 Digikam::FreeRotationContainer Class Reference	1588
9.601 Digikam::FreeRotationFilter Class Reference	1590
9.601.1 Member Function Documentation	1594
9.601.1.1 filterAction()	1594
9.601.1.2 filterIdentifier()	1594
9.601.1.3 readParameters()	1594
9.602 Digikam::FreeRotationSettings Class Reference	1594

---

9.603 Digikam::FreeSpaceToolTip Class Reference	1596
9.603.1 Member Function Documentation	1597
9.603.1.1 repositionRect()	1597
9.603.1.2 tipContents()	1597
9.604 Digikam::FreeSpaceWidget Class Reference	1598
9.605 Digikam::FullObjectDetection Class Reference	1599
9.606 Digikam::FullScreenSettings Class Reference	1600
9.607 Digikam::FuzzySearchSideBarWidget Class Reference	1601
9.607.1 Member Function Documentation	1603
9.607.1.1 applySettings()	1603
9.607.1.2 changeAlbumFromHistory()	1603
9.607.1.3 doLoadState()	1603
9.607.1.4 doSaveState()	1603
9.607.1.5 getCaption()	1604
9.607.1.6 getIcon()	1604
9.607.1.7 setActive()	1604
9.608 Digikam::FuzzySearchView Class Reference	1605
9.608.1 Member Function Documentation	1607
9.608.1.1 doLoadState()	1607
9.608.1.2 doSaveState()	1607
9.608.1.3 setConfigGroup()	1607
9.609 Digikam::GeoCoordinates Class Reference	1607
9.609.1 Member Function Documentation	1608
9.609.1.1 fromMarbleCoordinates()	1608
9.610 Digikam::GeodeticCalculator Class Reference	1609
9.610.1 Constructor & Destructor Documentation	1611
9.610.1.1 GeodeticCalculator()	1611
9.610.2 Member Function Documentation	1611
9.610.2.1 azimuth()	1611
9.610.2.2 checkAzimuth()	1611
9.610.2.3 checkLatitude()	1612
9.610.2.4 checkLongitude()	1612
9.610.2.5 checkOrthodromicDistance()	1612
9.610.2.6 computeDirection()	1612
9.610.2.7 destinationGeographicPoint()	1613
9.610.2.8 meridianArcLength()	1613
9.610.2.9 meridianArcLengthRadians()	1613
9.610.2.10 orthodromicDistance()	1613
9.610.2.11 setDestinationGeographicPoint()	1614
9.610.2.12 setDirection()	1614
9.610.2.13 setStartingGeographicPoint()	1614
9.610.3 Member Data Documentation	1615



9.610.3.1 fo	1615
9.610.3.2 m_destinationValid	1615
9.610.3.3 m_directionValid	1615
9.610.3.4 m_lat1	1615
9.610.3.5 m_lat2	1615
9.610.3.6 m_TOLERANCE_CHECK	1615
9.611 Digikam::GeoDragDropHandler Class Reference	1616
9.612 Digikam::GeofaceCluster Class Reference	1616
9.613 Digikam::GeofaceGlobalObject Class Reference	1617
9.614 Digikam::GeofaceInternalWidgetInfo Class Reference	1619
9.614.1 Detailed Description	1620
9.615 Digikam::GeofaceSharedData Class Reference	1620
9.615.1 Member Function Documentation	1621
9.615.1.1 hasRegionSelection()	1621
9.616 Digikam::GeolocationFilter Class Reference	1622
9.617 Digikam::GeolocationSettings Class Reference	1623
9.617.1 Member Function Documentation	1624
9.617.1.1 instance()	1624
9.617.1.2 mainMarbleWidget()	1624
9.618 Digikam::GeolocationSettingsContainer Class Reference	1624
9.619 Digikam::GeoModelHelper Class Reference	1625
9.619.1 Detailed Description	1626
9.619.2 Member Function Documentation	1626
9.619.2.1 bestRepresentativeIndexFromList()	1626
9.619.2.2 itemCoordinates()	1627
9.619.2.3 itemIcon()	1627
9.619.2.4 model()	1627
9.619.2.5 onIndicesClicked()	1627
9.619.2.6 pixmapFromRepresentativeIndex()	1628
9.619.2.7 selectionModel()	1628
9.620 Digikam::GeoPluginAboutDlg Class Reference	1628
9.621 Digikam::GPCamera Class Reference	1629
9.621.1 Member Function Documentation	1631
9.621.1.1 cameraAbout()	1631
9.621.1.2 cameraDriverType()	1631
9.621.1.3 cameraManual()	1632
9.621.1.4 cameraMD5ID()	1632
9.621.1.5 cameraSummary()	1632
9.621.1.6 cancel()	1632
9.621.1.7 capture()	1632
9.621.1.8 deleteItem()	1632
9.621.1.9 doConnect()	1632

---

9.621.1.10 downloadItem()	1633
9.621.1.11 getFolders()	1633
9.621.1.12 getFreeSpace()	1633
9.621.1.13 getItemInfo()	1633
9.621.1.14 getItemsInfoList()	1633
9.621.1.15 getMetadata()	1633
9.621.1.16 getPreview()	1634
9.621.1.17 getThumbnail()	1634
9.621.1.18 setLockItem()	1634
9.621.1.19 uploadItem()	1634
9.622 Digikam::GPSBookmarkModelHelper Class Reference	1635
9.622.1 Member Function Documentation	1636
9.622.1.1 itemCoordinates()	1636
9.622.1.2 itemFlags()	1637
9.622.1.3 itemIcon()	1637
9.622.1.4 model()	1637
9.622.1.5 modelFlags()	1637
9.622.1.6 selectionModel()	1637
9.622.1.7 snapItemsTo()	1638
9.623 Digikam::GPSBookmarkOwner Class Reference	1638
9.624 Digikam::GPSCorrelatorWidget Class Reference	1639
9.625 Digikam::GPSDataContainer Class Reference	1640
9.626 Digikam::GPSDBJobInfo Class Reference	1641
9.627 Digikam::GPSDBJobsThread Class Reference	1643
9.627.1 Member Function Documentation	1645
9.627.1.1 GPSListing()	1645
9.628 Digikam::GPSGeofaceModelHelper Class Reference	1646
9.628.1 Member Function Documentation	1647
9.628.1.1 bestRepresentativeIndexFromList()	1647
9.628.1.2 itemCoordinates()	1648
9.628.1.3 model()	1648
9.628.1.4 modelFlags()	1648
9.628.1.5 onIndicesMoved()	1648
9.628.1.6 pixmapFromRepresentativeIndex()	1648
9.628.1.7 selectionModel()	1648
9.629 Digikam::GPSItemContainer Class Reference	1649
9.629.1 Member Function Documentation	1651
9.629.1.1 isTagListDirty()	1651
9.629.1.2 loadImageData()	1651
9.629.1.3 restoreGPSData()	1651
9.629.1.4 saveChanges()	1651
9.629.1.5 setTagList()	1651

---

9.630 Digikam::GPSItemDelegate Class Reference	1652
9.631 Digikam::GPSItemInfo Class Reference	1652
9.632 Digikam::GPSItemInfoSorter Class Reference	1653
9.633 Digikam::GPSItemList Class Reference	1654
9.634 Digikam::GPSItemListContextMenu Class Reference	1656
9.635 Digikam::GPSItemListDragDropHandler Class Reference	1657
9.635.1 Member Function Documentation	1658
9.635.1.1 createMimeData()	1658
9.636 Digikam::GPSItemModel Class Reference	1658
9.637 Digikam::GPSItemSortProxyModel Class Reference	1660
9.638 Digikam::GPSJob Class Reference	1661
9.639 Digikam::GPSLinkItemSelectionModel Class Reference	1663
9.639.1 Detailed Description	1664
9.640 Digikam::GPSMarkerTiler Class Reference	1664
9.640.1 Detailed Description	1667
9.640.2 Constructor & Destructor Documentation	1667
9.640.2.1 GPSMarkerTiler()	1667
9.640.3 Member Function Documentation	1667
9.640.3.1 bestRepresentativeIndexFromList()	1667
9.640.3.2 getGlobalGroupState()	1668
9.640.3.3 getTile()	1668
9.640.3.4 getTileGroupState()	1668
9.640.3.5 getTileMarkerCount()	1668
9.640.3.6 getTileRepresentativeMarker()	1668
9.640.3.7 getTileSelectedCount()	1669
9.640.3.8 indicesEqual()	1669
9.640.3.9 onIndicesClicked()	1669
9.640.3.10 pixmapFromRepresentativeIndex()	1669
9.640.3.11 prepareTiles()	1669
9.640.3.12 regenerateTiles()	1670
9.640.3.13 setActive()	1670
9.640.3.14 setPositiveFilterIsActive()	1670
9.640.3.15 slotNewModelData	1670
9.640.3.16 tileNew()	1670
9.641 Digikam::GPSModelIndexProxyMapper Class Reference	1671
9.641.1 Detailed Description	1672
9.641.2 Property Documentation	1672
9.641.2.1 isConnected	1672
9.642 Digikam::GPSSearchSideBarWidget Class Reference	1673
9.642.1 Member Function Documentation	1675
9.642.1.1 applySettings()	1675
9.642.1.2 changeAlbumFromHistory()	1675

9.642.1.3 doLoadState()	1675
9.642.1.4 doSaveState()	1675
9.642.1.5 getCaption()	1676
9.642.1.6 getIcon()	1676
9.642.1.7 setActive()	1676
9.643 Digikam::GPSSearchView Class Reference	1677
9.643.1 Constructor & Destructor Documentation	1679
9.643.1.1 GPSSearchView()	1679
9.643.2 Member Function Documentation	1679
9.643.2.1 doLoadState()	1679
9.643.2.2 doSaveState()	1679
9.643.2.3 setActive()	1679
9.643.2.4 setConfigGroup()	1680
9.644 Digikam::GPSUndoCommand Class Reference	1680
9.645 Digikam::GPSUndoCommand::UndoInfo Class Reference	1681
9.646 Digikam::Graph< VertexProperties, EdgeProperties > Class Template Reference	1681
9.646.1 Member Enumeration Documentation	1686
9.646.1.1 AdjacencyFlags	1686
9.646.2 Member Function Documentation	1686
9.646.2.1 edgeDifference()	1686
9.646.2.2 leaves()	1686
9.646.2.3 listPath()	1687
9.646.2.4 longestPathTouching()	1687
9.646.2.5 roots()	1687
9.646.2.6 rootsOf()	1687
9.646.2.7 shortestDistancesFrom()	1687
9.646.2.8 shortestPath()	1687
9.646.2.9 transitiveReduction()	1688
9.646.2.10 vertexCount()	1688
9.646.2.11 verticesBreadthFirst()	1688
9.646.2.12 verticesDepthFirstSorted()	1688
9.646.2.13 verticesDominatedBy()	1688
9.646.2.14 verticesDominatedByDepthFirstSorted()	1689
9.647 Digikam::Graph< VertexProperties, EdgeProperties >::DominatorTree Class Reference	1689
9.648 Digikam::Graph< VertexProperties, EdgeProperties >::Edge Class Reference	1689
9.649 Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch Class Reference	1690
9.649.1 Member Function Documentation	1690
9.649.1.1 depth_first_search_sorted()	1690
9.650 Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch::BreadthFirstSearchVisitor Class Reference	1691
9.651 Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch::CommonVisitor Class Reference	1692

9.652 Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch::DepthFirstSearchVisitor Class Reference	1693
9.653 Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch::lessThanMapEdgeTo←Target< GraphType, VertexLessThan > Class Template Reference	1694
9.654 Digikam::Graph< VertexProperties, EdgeProperties >::Path Class Reference	1694
9.654.1 Detailed Description	1694
9.654.2 Member Function Documentation	1695
9.654.2.1 longestPath()	1695
9.654.2.2 shortestPath()	1695
9.655 Digikam::Graph< VertexProperties, EdgeProperties >::Vertex Class Reference	1695
9.656 Digikam::GraphicsDImgItem Class Reference	1696
9.656.1 Member Function Documentation	1697
9.656.1.1 setImage()	1697
9.657 Digikam::GraphicsDImgView Class Reference	1698
9.657.1 Member Function Documentation	1700
9.657.1.1 scrollPointOnPoint()	1700
9.657.1.2 setItem()	1700
9.658 Digikam::GreycstorageContainer Class Reference	1700
9.659 Digikam::GreycstorageFilter Class Reference	1701
9.659.1 Member Enumeration Documentation	1705
9.659.1.1 MODE	1705
9.659.2 Constructor & Destructor Documentation	1705
9.659.2.1 GreycstorageFilter() [1/2]	1705
9.659.2.2 GreycstorageFilter() [2/2]	1705
9.659.3 Member Function Documentation	1705
9.659.3.1 cancelFilter()	1705
9.659.3.2 filterAction()	1705
9.659.3.3 filterIdentifier()	1706
9.659.3.4 readParameters()	1706
9.660 Digikam::GreycstorageSettings Class Reference	1706
9.661 Digikam::GroupedImagesFinder Class Reference	1707
9.661.1 Constructor & Destructor Documentation	1707
9.661.1.1 GroupedImagesFinder()	1707
9.662 Digikam::GroupIndicatorOverlay Class Reference	1708
9.662.1 Member Function Documentation	1711
9.662.1.1 checkIndex()	1711
9.662.1.2 createWidget()	1711
9.662.1.3 setActive()	1711
9.662.1.4 slotEntered()	1711
9.662.1.5 visualChange()	1712
9.663 Digikam::GroupIndicatorOverlayWidget Class Reference	1712
9.664 Digikam::GroupingViewImplementation Class Reference	1713
9.664.1 Member Function Documentation	1714

---

9.664.1.1 hasHiddenGroupedImages()	1714
9.665 Digikam::GroupItemFilterSettings Class Reference	1714
9.666 Digikam::GroupStateComputer Class Reference	1714
9.667 Digikam::Haar::Calculator Class Reference	1715
9.667.1 Member Function Documentation	1715
9.667.1.1 calcHaar()	1715
9.667.1.2 transform()	1715
9.668 Digikam::Haar::ImageData Class Reference	1715
9.669 Digikam::Haar::SignatureData Class Reference	1716
9.670 Digikam::Haar::SignatureMap Class Reference	1716
9.671 Digikam::Haar::WeightBin Class Reference	1716
9.671.1 Member Data Documentation	1717
9.671.1.1 m_bin	1717
9.672 Digikam::Haar::Weights Class Reference	1717
9.673 Digikam::Haarface Class Reference	1717
9.673.1 Member Enumeration Documentation	1719
9.673.1.1 RefImageSelMethod	1719
9.673.2 Member Function Documentation	1719
9.673.2.1 bestMatchesForImageWithThreshold() [1/2]	1719
9.673.2.2 bestMatchesForImageWithThreshold() [2/2]	1719
9.673.2.3 findDuplicates()	1720
9.673.2.4 loadQImage()	1720
9.673.2.5 rebuildDuplicatesAlbums()	1720
9.673.2.6 retrieveSignatureFromDB()	1720
9.673.2.7 setAlbumRootsToSearch()	1721
9.673.2.8 signatureAsText()	1721
9.674 Digikam::HaarProgressObserver Class Reference	1721
9.675 Digikam::HidingStateChanger Class Reference	1722
9.675.1 Constructor & Destructor Documentation	1725
9.675.1.1 HidingStateChanger()	1725
9.676 Digikam::Highlighter Class Reference	1725
9.677 Digikam::HistogramBox Class Reference	1726
9.678 Digikam::HistogramPainter Class Reference	1727
9.678.1 Detailed Description	1728
9.678.2 Constructor & Destructor Documentation	1728
9.678.2.1 HistogramPainter()	1728
9.678.3 Member Function Documentation	1728
9.678.3.1 enableHistogramGuideByColor()	1728
9.678.3.2 initFrom()	1729
9.678.3.3 render()	1729
9.678.3.4 setChannelType()	1729
9.678.3.5 setHighlightSelection()	1729

9.678.3.6 setHistogram()	1730
9.678.3.7 setRenderXGrid()	1730
9.678.3.8 setScale()	1730
9.678.3.9 setSelection()	1730
9.679 Digikam::HistogramWidget Class Reference	1731
9.679.1 Constructor & Destructor Documentation	1733
9.679.1.1 HistogramWidget()	1733
9.680 Digikam::HistoryEdgeProperties Class Reference	1733
9.680.1 Detailed Description	1733
9.681 Digikam::HistoryImageId Class Reference	1733
9.681.1 Member Enumeration Documentation	1735
9.681.1.1 Type	1735
9.681.2 Member Data Documentation	1735
9.681.2.1 m_originalUUID	1735
9.681.2.2 m_uuid	1735
9.682 Digikam::HistoryVertexProperties Class Reference	1735
9.682.1 Detailed Description	1736
9.683 Digikam::HotPixelContainer Class Reference	1736
9.684 Digikam::HotPixelFixer Class Reference	1737
9.684.1 Member Function Documentation	1741
9.684.1.1 filterAction()	1741
9.684.1.2 filterIdentifier()	1741
9.684.1.3 readParameters()	1741
9.685 Digikam::HotPixelProps Class Reference	1741
9.685.1 Member Function Documentation	1742
9.685.1.1 operator==(())	1742
9.686 Digikam::HotPixelSettings Class Reference	1742
9.687 Digikam::HotPixelsWeights Class Reference	1743
9.688 Digikam::HoverButtonDelegateOverlay Class Reference	1744
9.688.1 Member Function Documentation	1747
9.688.1.1 createButton()	1747
9.688.1.2 createWidget()	1747
9.688.1.3 setActive()	1747
9.688.1.4 updateButton()	1747
9.688.1.5 visualChange()	1747
9.689 Digikam::HSLContainer Class Reference	1748
9.690 Digikam::HSLFilter Class Reference	1749
9.690.1 Member Function Documentation	1753
9.690.1.1 filterAction()	1753
9.690.1.2 filterIdentifier()	1753
9.690.1.3 readParameters()	1753
9.691 Digikam::HSLSettings Class Reference	1753

---

9.692 Digikam::HSPreviewWidget Class Reference	1754
9.693 Digikam::HTMLWidget Class Reference	1755
9.694 Digikam::HTMLWidgetPage Class Reference	1756
9.695 Digikam::IccManager Class Reference	1757
9.695.1 Constructor & Destructor Documentation	1759
9.695.1.1 IccManager()	1759
9.695.2 Member Function Documentation	1759
9.695.2.1 needsPostLoadingManagement()	1759
9.695.2.2 transformDefault()	1759
9.695.2.3 transformForDisplay()	1759
9.696 Digikam::IccPostLoadingManager Class Reference	1760
9.696.1 Constructor & Destructor Documentation	1762
9.696.1.1 IccPostLoadingManager()	1762
9.696.2 Member Function Documentation	1762
9.696.2.1 postLoadingManage()	1762
9.697 Digikam::ICCPreviewWidget Class Reference	1762
9.698 Digikam::IccProfile Class Reference	1763
9.698.1 Member Enumeration Documentation	1764
9.698.1.1 ProfileType	1764
9.698.2 Member Function Documentation	1764
9.698.2.1 close()	1764
9.698.2.2 data()	1764
9.698.2.3 defaultSearchPaths()	1764
9.698.2.4 description()	1765
9.698.2.5 filePath()	1765
9.698.2.6 open()	1765
9.698.2.7 operator==(())	1765
9.698.2.8 sRGB()	1765
9.698.2.9 type()	1765
9.699 Digikam::ICCPProfileInfoDlg Class Reference	1766
9.700 Digikam::IccProfilesComboBox Class Reference	1767
9.700.1 Constructor & Destructor Documentation	1769
9.700.1.1 IccProfilesComboBox()	1769
9.700.2 Member Function Documentation	1769
9.700.2.1 addProfileSqueezed()	1769
9.700.2.2 setCurrentProfile()	1769
9.701 Digikam::IccProfilesMenuAction Class Reference	1770
9.701.1 Member Function Documentation	1771
9.701.1.1 addProfile()	1771
9.702 Digikam::IccProfilesSettings Class Reference	1772
9.703 Digikam::ICCPProfileWidget Class Reference	1774
9.703.1 Member Function Documentation	1776



9.703.1.1 getMetadataTitle()	1776
9.703.1.2 getTagDescription()	1776
9.703.1.3 getTagTitle()	1776
9.703.1.4 loadFromURL()	1776
9.704 Digikam::lccRenderingIntentComboBox Class Reference	1777
9.705 Digikam::lccSettings Class Reference	1778
9.705.1 Member Function Documentation	1779
9.705.1.1 instance()	1779
9.705.1.2 loadAllProfilesProperties()	1779
9.705.1.3 monitorProfile()	1780
9.706 Digikam::ICCSettingsContainer Class Reference	1780
9.706.1 Member Enumeration Documentation	1781
9.706.1.1 BehaviorEnum	1781
9.707 Digikam::lccTransform Class Reference	1781
9.707.1 Member Function Documentation	1782
9.707.1.1 apply() [1/2]	1782
9.707.1.2 apply() [2/2]	1782
9.707.1.3 close()	1782
9.707.1.4 setDoNotEmbedOutputProfile()	1782
9.707.1.5 setEmbeddedProfile()	1783
9.707.1.6 willHaveEffect()	1783
9.708 Digikam::lccTransformFilter Class Reference	1784
9.708.1 Member Function Documentation	1788
9.708.1.1 filterAction()	1788
9.708.1.2 filterIdentifier()	1788
9.708.1.3 filterImage()	1788
9.708.1.4 parametersSuccessfullyRead()	1788
9.708.1.5 progressInfo()	1789
9.708.1.6 readParameters()	1789
9.708.1.7 readParametersError()	1789
9.709 Digikam::Identity Class Reference	1789
9.709.1 Constructor & Destructor Documentation	1789
9.709.1.1 Identity()	1789
9.710 Digikam::IdentityProvider Class Reference	1790
9.710.1 Member Function Documentation	1791
9.710.1.1 addIdentity()	1791
9.710.1.2 findIdentity() [1/2]	1791
9.710.1.3 findIdentity() [2/2]	1791
9.711 Digikam::ImageChangeset Class Reference	1791
9.711.1 Constructor & Destructor Documentation	1791
9.711.1.1 ImageChangeset()	1791
9.712 Digikam::ImageCommonContainer Class Reference	1792

9.713 Digikam::ImageCurves Class Reference	1792
9.713.1 Member Enumeration Documentation	1793
9.713.1.1 CurveType	1793
9.713.2 Member Function Documentation	1794
9.713.2.1 channelToBinary()	1794
9.713.2.2 fillFromOtherCurves()	1794
9.713.2.3 setChannelFromBinary()	1794
9.713.2.4 setContainer()	1794
9.714 Digikam::ImageDialog Class Reference	1795
9.715 Digikam::ImageDialogIconProvider Class Reference	1796
9.716 Digikam::ImageDialogPreview Class Reference	1797
9.717 Digikam::ImageDialogToolTip Class Reference	1798
9.718 Digikam::ImageGuideWidget Class Reference	1800
9.719 Digikam::ImageHistogram Class Reference	1802
9.719.1 Member Function Documentation	1804
9.719.1.1 run()	1804
9.720 Digikam::ImageHistoryEntry Class Reference	1804
9.721 Digikam::ImageIface Class Reference	1805
9.721.1 Member Enumeration Documentation	1806
9.721.1.1 PreviewType	1806
9.721.2 Constructor & Destructor Documentation	1806
9.721.2.1 ImageIface()	1806
9.721.3 Member Function Documentation	1806
9.721.3.1 original()	1806
9.721.3.2 paint()	1806
9.721.3.3 previewReference()	1807
9.721.3.4 setOriginal()	1807
9.721.3.5 setPreview()	1807
9.721.3.6 setPreviewSize()	1807
9.721.3.7 setPreviewType()	1807
9.721.3.8 setSelection()	1807
9.722 Digikam::ImageLevels Class Reference	1808
9.723 Digikam::ImageListProvider Class Reference	1808
9.723.1 Detailed Description	1809
9.724 Digikam::ImageMetadataContainer Class Reference	1810
9.725 Digikam::ImagePreviewItem Class Reference	1811
9.726 Digikam::ImageQualityCalculator Class Reference	1812
9.727 Digikam::ImageQualityCalculator::ResultDetection Struct Reference	1813
9.728 Digikam::ImageQualityConfSelector Class Reference	1813
9.728.1 Member Enumeration Documentation	1814
9.728.1.1 SettingsType	1814
9.729 Digikam::ImageQualityContainer Class Reference	1814

9.730 Digikam::ImageQualityParser Class Reference . . . . .	1815
9.731 Digikam::ImageQualitySettings Class Reference . . . . .	1816
9.732 Digikam::ImageQualitySorter Class Reference . . . . .	1817
9.732.1 Member Enumeration Documentation . . . . .	1819
9.732.1.1 QualityScanMode . . . . .	1819
9.732.2 Constructor & Destructor Documentation . . . . .	1820
9.732.2.1 ImageQualitySorter() . . . . .	1820
9.732.3 Member Function Documentation . . . . .	1820
9.732.3.1 setUseMultiCoreCPU() . . . . .	1820
9.733 Digikam::ImageQualityTask Class Reference . . . . .	1821
9.734 Digikam::ImageQualityThread Class Reference . . . . .	1823
9.735 Digikam::ImageQualityThreadPool Class Reference . . . . .	1824
9.736 Digikam::ImageRegionItem Class Reference . . . . .	1825
9.737 Digikam::ImageRegionWidget Class Reference . . . . .	1827
9.737.1 Member Function Documentation . . . . .	1829
9.737.1.1 getOriginalRegionImage() . . . . .	1829
9.738 Digikam::ImageRelation Class Reference . . . . .	1829
9.739 Digikam::ImageSortFilterModel Class Reference . . . . .	1830
9.739.1 Member Function Documentation . . . . .	1832
9.739.1.1 imageFilterModel() . . . . .	1832
9.739.1.2 imageInfosSorted() . . . . .	1832
9.739.1.3 mapListToSource() . . . . .	1832
9.739.1.4 setDirectSourceItemModel() . . . . .	1832
9.739.1.5 setSourceModel() . . . . .	1832
9.740 Digikam::ImageTagChangeset Class Reference . . . . .	1833
9.740.1 Member Enumeration Documentation . . . . .	1833
9.740.1.1 Operation . . . . .	1833
9.740.2 Member Function Documentation . . . . .	1833
9.740.2.1 operator<<() . . . . .	1833
9.741 Digikam::ImageTagProperty Class Reference . . . . .	1834
9.742 Digikam::ImageTagPropertyName Class Reference . . . . .	1834
9.743 Digikam::ImageWindow Class Reference . . . . .	1835
9.743.1 Member Function Documentation . . . . .	1841
9.743.1.1 infoface() . . . . .	1841
9.743.1.2 versionManager() . . . . .	1841
9.744 Digikam::ImageZoomSettings Class Reference . . . . .	1841
9.744.1 Member Function Documentation . . . . .	1842
9.744.1.1 fitToSize() . . . . .	1842
9.744.1.2 originalImageSize() . . . . .	1842
9.744.1.3 setImageSize() . . . . .	1842
9.744.1.4 snappedZoomFactor() . . . . .	1843
9.744.1.5 snappedZoomStep() . . . . .	1843

---

9.744.1.6 zoomedSize()	1843
9.745 Digikam::ImportCategorizedView Class Reference	1844
9.745.1 Member Function Documentation	1849
9.745.1.1 activated()	1849
9.745.1.2 addOverlay()	1850
9.745.1.3 camItemInfoActivated	1850
9.745.1.4 deselected	1850
9.745.1.5 dragDropHandler()	1850
9.745.1.6 filterModel()	1850
9.745.1.7 importFilterModel()	1850
9.745.1.8 indexActivated()	1851
9.745.1.9 nextIndexHint()	1851
9.745.1.10 nextInOrder()	1851
9.745.1.11 selected	1851
9.745.1.12 showContextMenuOnIndex()	1851
9.746 Digikam::ImportCategoryDrawer Class Reference	1852
9.746.1 Member Function Documentation	1853
9.746.1.1 categoryHeight()	1853
9.746.1.2 drawCategory()	1854
9.747 Digikam::ImportContextMenuHelper Class Reference	1855
9.747.1 Constructor & Destructor Documentation	1856
9.747.1.1 ImportContextMenuHelper()	1856
9.747.2 Member Function Documentation	1856
9.747.2.1 addAction() [1/3]	1856
9.747.2.2 addAction() [2/3]	1857
9.747.2.3 addAction() [3/3]	1857
9.747.2.4 addAssignTagsMenu()	1857
9.747.2.5 addGroupMenu()	1858
9.747.2.6 addLabelsAction()	1858
9.747.2.7 addRemoveTagsMenu()	1858
9.747.2.8 addRotateMenu()	1859
9.747.2.9 addServicesMenu()	1859
9.747.2.10 addSubMenu()	1859
9.747.2.11 exec()	1859
9.747.2.12 setImportFilterModel()	1860
9.748 Digikam::ImportCoordinatesOverlay Class Reference	1861
9.748.1 Member Function Documentation	1863
9.748.1.1 checkIndex()	1863
9.748.1.2 createWidget()	1864
9.748.1.3 setActive()	1864
9.748.1.4 slotEntered()	1864
9.748.1.5 visualChange()	1864

---

9.749 Digikam::ImportDelegate Class Reference	1865
9.749.1 Member Function Documentation	1869
9.749.1.1 acceptsActivation()	1869
9.749.1.2 acceptsToolTip()	1870
9.749.1.3 clearCaches()	1870
9.749.1.4 imageInformationRect()	1870
9.749.1.5 invalidatePaintingCache()	1870
9.749.1.6 pixmapForDrag()	1870
9.749.1.7 pixmapRect()	1870
9.749.1.8 setDefaultViewOptions()	1871
9.749.1.9 setSpacing()	1871
9.749.1.10 updateContentWidth()	1871
9.749.1.11 updateRects()	1871
9.749.1.12 updateSizeRectsAndPixmaps()	1871
9.750 Digikam::ImportDownloadOverlay Class Reference	1872
9.750.1 Member Function Documentation	1874
9.750.1.1 checkIndex()	1874
9.750.1.2 createWidget()	1875
9.750.1.3 setActive()	1875
9.750.1.4 slotEntered()	1875
9.750.1.5 visualChange()	1875
9.751 Digikam::ImportDragDropHandler Class Reference	1876
9.751.1 Member Function Documentation	1877
9.751.1.1 accepts()	1877
9.751.1.2 createMimeData()	1877
9.751.1.3 dropEvent()	1877
9.751.1.4 mimeTypes()	1878
9.752 Digikam::ImportFilterComboBox Class Reference	1878
9.753 Digikam::ImportFilterDlg Class Reference	1879
9.754 Digikam::ImportFilterModel Class Reference	1881
9.754.1 Member Enumeration Documentation	1884
9.754.1.1 ImportFilterModelRoles	1884
9.754.2 Member Function Documentation	1884
9.754.2.1 camItemInfosAdded	1884
9.754.2.2 categoryIdentifier()	1884
9.754.2.3 compareCategories()	1884
9.754.2.4 importFilterModel()	1885
9.754.2.5 infosLessThan()	1885
9.754.2.6 setDirectSourceImportModel()	1885
9.754.2.7 subSortLessThan()	1886
9.755 Digikam::ImportIconView Class Reference	1887
9.755.1 Member Function Documentation	1893

---

9.755.1.1	activated()	1893
9.755.1.2	setThumbnailSize()	1894
9.755.1.3	showContextMenu()	1894
9.755.1.4	showContextMenuOnInfo()	1894
9.755.1.5	slotSetupChanged()	1894
9.756	Digikam::ImportItemModel Class Reference	1895
9.756.1	Member Enumeration Documentation	1898
9.756.1.1	ImportItemModelRoles	1898
9.756.2	Member Function Documentation	1899
9.756.2.1	addCamItemInfoSynchronously()	1899
9.756.2.2	allRefreshingFinished	1899
9.756.2.3	camItemInfo() [1/2]	1899
9.756.2.4	camItemInfo() [2/2]	1899
9.756.2.5	indexForUrl()	1899
9.756.2.6	isRefreshing()	1899
9.756.2.7	itemInfosAboutToBeAdded	1900
9.756.2.8	itemInfosAboutToBeRemoved	1900
9.756.2.9	itemInfosAdded	1900
9.756.2.10	itemInfosRemoved	1900
9.756.2.11	readyForIncrementalRefresh	1900
9.756.2.12	requestIncrementalRefresh()	1900
9.756.2.13	setCameraThumbsController()	1900
9.756.2.14	setKeepsFileUrlCache()	1901
9.756.2.15	setSendRemovalSignals()	1901
9.756.2.16	startIncrementalRefresh()	1901
9.756.2.17	startRefresh()	1901
9.757	Digikam::ImportItemPropertiesSideBarImport Class Reference	1902
9.757.1	Member Function Documentation	1905
9.757.1.1	applySettings()	1905
9.757.1.2	doLoadState()	1906
9.757.1.3	doSaveState()	1906
9.758	Digikam::ImportItemPropertiesTab Class Reference	1907
9.759	Digikam::ImportLockOverlay Class Reference	1909
9.759.1	Member Function Documentation	1911
9.759.1.1	checkIndex()	1911
9.759.1.2	createWidget()	1912
9.759.1.3	setActive()	1912
9.759.1.4	slotEntered()	1912
9.759.1.5	visualChange()	1912
9.760	Digikam::ImportNormalDelegate Class Reference	1913
9.760.1	Member Function Documentation	1917
9.760.1.1	updateRects()	1917

---

9.761 Digikam::ImportOverlayWidget Class Reference . . . . .	1918
9.762 Digikam::ImportPreviewView Class Reference . . . . .	1919
9.762.1 Member Function Documentation . . . . .	1921
9.762.1.1 acceptsMouseClicked() . . . . .	1921
9.763 Digikam::ImportRatingOverlay Class Reference . . . . .	1922
9.763.1 Member Function Documentation . . . . .	1925
9.763.1.1 createWidget() . . . . .	1925
9.763.1.2 hide() . . . . .	1925
9.763.1.3 setActive() . . . . .	1925
9.763.1.4 slotEntered() . . . . .	1925
9.763.1.5 visualChange() . . . . .	1926
9.763.1.6 widgetEnterEvent() . . . . .	1926
9.763.1.7 widgetLeaveEvent() . . . . .	1926
9.764 Digikam::ImportRenameParser Class Reference . . . . .	1927
9.765 Digikam::ImportRotateOverlay Class Reference . . . . .	1928
9.765.1 Member Function Documentation . . . . .	1931
9.765.1.1 checkIndex() . . . . .	1931
9.765.1.2 createButton() . . . . .	1931
9.765.1.3 setActive() . . . . .	1931
9.765.1.4 updateButton() . . . . .	1932
9.765.1.5 widgetEnterEvent() . . . . .	1932
9.765.1.6 widgetLeaveEvent() . . . . .	1932
9.766 Digikam::ImportRotateOverlayButton Class Reference . . . . .	1933
9.766.1 Member Function Documentation . . . . .	1935
9.766.1.1 icon() . . . . .	1935
9.766.1.2 sizeHint() . . . . .	1935
9.766.1.3 updateToolTip() . . . . .	1935
9.767 Digikam::ImportSettings Class Reference . . . . .	1936
9.768 Digikam::ImportSortFilterModel Class Reference . . . . .	1939
9.768.1 Member Function Documentation . . . . .	1941
9.768.1.1 camItemInfosSorted() . . . . .	1941
9.768.1.2 importFilterModel() . . . . .	1941
9.768.1.3 mapToSourceImportModel() . . . . .	1941
9.768.1.4 setDirectSourceImportModel() . . . . .	1941
9.769 Digikam::ImportStackedView Class Reference . . . . .	1942
9.769.1 Member Enumeration Documentation . . . . .	1943
9.769.1.1 StackedViewMode . . . . .	1943
9.770 Digikam::ImportThumbnailBar Class Reference . . . . .	1944
9.770.1 Member Function Documentation . . . . .	1950
9.770.1.1 setModelsFiltered() . . . . .	1950
9.770.1.2 slotSetupChanged() . . . . .	1950
9.771 Digikam::ImportThumbnailDelegate Class Reference . . . . .	1951

9.771.1 Member Function Documentation	1955
9.771.1.1 acceptsActivation()	1955
9.771.1.2 setDefaultViewOptions()	1956
9.771.1.3 updateContentWidth()	1956
9.771.1.4 updateRects()	1956
9.772 Digikam::ImportThumbnailModel Class Reference	1957
9.772.1 Constructor & Destructor Documentation	1961
9.772.1.1 ImportThumbnailModel()	1961
9.772.2 Member Function Documentation	1961
9.772.2.1 data()	1961
9.772.2.2 setCameraThumbsController()	1961
9.772.2.3 setData()	1961
9.772.2.4 setEmitDataChanged()	1961
9.773 Digikam::ImportUI Class Reference	1962
9.773.1 Member Function Documentation	1965
9.773.1.1 infoface()	1965
9.774 Digikam::ImportView Class Reference	1966
9.775 Digikam::InfoDlg Class Reference	1968
9.776 Digikam::InfraredContainer Class Reference	1969
9.777 Digikam::InfraredFilter Class Reference	1970
9.777.1 Member Function Documentation	1974
9.777.1.1 filterAction()	1974
9.777.1.2 filterIdentifier()	1974
9.777.1.3 readParameters()	1974
9.778 Digikam::InitializationObserver Class Reference	1975
9.779 Digikam::InsertBookmarksCommand Class Reference	1976
9.780 Digikam::InternalTagName Class Reference	1977
9.781 Digikam::InvertFilter Class Reference	1978
9.781.1 Member Function Documentation	1982
9.781.1.1 filterAction()	1982
9.781.1.2 filterIdentifier()	1982
9.781.1.3 readParameters()	1982
9.782 Digikam::IOFileSettings Class Reference	1982
9.782.1 Member Data Documentation	1983
9.782.1.1 JPEGSubSampling	1983
9.783 Digikam::IOJob Class Reference	1983
9.784 Digikam::IOJobData Class Reference	1984
9.785 Digikam::IOJobsManager Class Reference	1986
9.785.1 Member Function Documentation	1986
9.785.1.1 buildCollectionTrashCounters()	1986
9.785.1.2 instance()	1987
9.785.1.3 startDTrashItemsListingForCollection()	1987



9.785.1.4 startIOJobs()	1987
9.786 Digikam::IOJobsThread Class Reference	1988
9.786.1 Member Function Documentation	1990
9.786.1.1 copyOrMove()	1990
9.786.1.2 deleteFiles()	1990
9.786.1.3 emptyDTrashItems()	1990
9.786.1.4 errorsList()	1990
9.786.1.5 hasErrors()	1991
9.786.1.6 isCanceled()	1991
9.786.1.7 jobData()	1991
9.786.1.8 listDTrashItems()	1991
9.786.1.9 renameFile()	1991
9.786.1.10 restoreDTrashItems()	1992
9.787 Digikam::IptcCoreContactInfo Class Reference	1992
9.788 Digikam::IptcCoreLocationInfo Class Reference	1992
9.789 Digikam::IptcMetaEngineMergeHelper Class Reference	1993
9.790 Digikam::IptcWidget Class Reference	1994
9.790.1 Member Function Documentation	1996
9.790.1.1 getMetadataTitle()	1996
9.790.1.2 getTagDescription()	1996
9.790.1.3 getTagTitle()	1996
9.790.1.4 loadFromURL()	1996
9.791 Digikam::ItemAlbumFilterModel Class Reference	1997
9.791.1 Member Function Documentation	2001
9.791.1.1 compareInfosCategories() [1/2]	2001
9.791.1.2 compareInfosCategories() [2/2]	2001
9.791.1.3 setItemFilterSettings()	2001
9.792 Digikam::ItemAlbumModel Class Reference	2002
9.792.1 Member Function Documentation	2008
9.792.1.1 openAlbum	2008
9.792.1.2 slotImageChange	2008
9.793 Digikam::ItemAttributesWatch Class Reference	2008
9.793.1 Member Function Documentation	2009
9.793.1.1 signalFileMetadataChanged	2009
9.793.1.2 signalImageRatingChanged	2009
9.793.1.3 signalImagesChanged	2009
9.793.1.4 signalImageTagsChanged	2009
9.794 Digikam::ItemCategorizedView Class Reference	2010
9.794.1 Member Function Documentation	2016
9.794.1.1 activated()	2016
9.794.1.2 albumAt()	2016
9.794.1.3 dragDropHandler()	2016

---

9.794.1.4 filterModel()	2016
9.794.1.5 indexActivated()	2016
9.794.1.6 nextIndexHint()	2017
9.794.1.7 nextInOrder()	2017
9.794.1.8 showContextMenuOnIndex()	2017
9.795 Digikam::ItemCategoryDrawer Class Reference	2018
9.795.1 Member Function Documentation	2019
9.795.1.1 categoryHeight()	2019
9.795.1.2 drawCategory()	2020
9.796 Digikam::ItemChangeHint Class Reference	2020
9.796.1 Member Enumeration Documentation	2020
9.796.1.1 ChangeType	2020
9.797 Digikam::ItemComments Class Reference	2021
9.797.1 Member Enumeration Documentation	2022
9.797.1.1 LanguageChoiceBehavior	2022
9.797.1.2 UniqueBehavior	2023
9.797.2 Constructor & Destructor Documentation	2023
9.797.2.1 ItemComments()	2023
9.797.3 Member Function Documentation	2023
9.797.3.1 addComment()	2023
9.797.3.2 addHeadline()	2024
9.797.3.3 addTitle()	2024
9.797.3.4 apply()	2024
9.797.3.5 changeComment()	2024
9.797.3.6 commentForLanguage()	2024
9.797.3.7 defaultComment()	2024
9.797.3.8 replaceComments()	2025
9.797.3.9 setUniqueBehavior()	2025
9.797.3.10 type()	2025
9.798 Digikam::ItemCoordinatesOverlay Class Reference	2026
9.798.1 Member Function Documentation	2028
9.798.1.1 checkIndex()	2028
9.798.1.2 createWidget()	2029
9.798.1.3 setActive()	2029
9.798.1.4 slotEntered()	2029
9.798.1.5 visualChange()	2029
9.799 Digikam::ItemCopyMoveHint Class Reference	2029
9.799.1 Constructor & Destructor Documentation	2030
9.799.1.1 ItemCopyMoveHint()	2030
9.800 Digikam::ItemCopyright Class Reference	2030
9.800.1 Member Enumeration Documentation	2032
9.800.1.1 ReplaceMode	2032

---

9.800.2 Member Function Documentation	2032
9.800.2.1 contactInfo()	2032
9.800.2.2 copyrightNotice()	2032
9.800.2.3 creator()	2033
9.800.2.4 creatorJobTitle()	2033
9.800.2.5 fillTemplate()	2033
9.800.2.6 instructions()	2033
9.800.2.7 provider()	2033
9.800.2.8 rightsUsageTerms()	2033
9.800.2.9 setCopyrightNotice()	2034
9.800.2.10 setCreator()	2034
9.800.2.11 setFromTemplate()	2034
9.800.2.12 source()	2034
9.801 Digikam::ItemDelegate Class Reference	2035
9.801.1 Member Function Documentation	2039
9.801.1.1 acceptsActivation()	2039
9.801.1.2 acceptsToolTip()	2040
9.801.1.3 clearCaches()	2040
9.801.1.4 imageInformationRect()	2040
9.801.1.5 invalidatePaintingCache()	2040
9.801.1.6 pixmapForDrag()	2040
9.801.1.7 pixmapRect()	2040
9.801.1.8 setDefaultViewOptions()	2041
9.801.1.9 setSpacing()	2041
9.801.1.10 updateContentWidth()	2041
9.801.1.11 updateRects()	2041
9.801.1.12 updateSizeRectsAndPxmmaps()	2041
9.802 Digikam::ItemDelegateOverlay Class Reference	2042
9.802.1 Member Function Documentation	2043
9.802.1.1 affectsMultiple()	2043
9.802.1.2 mouseMoved()	2043
9.802.1.3 setActive()	2043
9.802.1.4 visualChange	2044
9.803 Digikam::ItemDelegateOverlayContainer Class Reference	2045
9.803.1 Constructor & Destructor Documentation	2046
9.803.1.1 ItemDelegateOverlayContainer()	2046
9.803.2 Member Function Documentation	2046
9.803.2.1 asDelegate()	2046
9.804 Digikam::ItemDescEditTab Class Reference	2047
9.805 Digikam::ItemDragDropHandler Class Reference	2050
9.805.1 Member Function Documentation	2051
9.805.1.1 accepts()	2051

---

9.805.1.2 createMimeData()	2051
9.805.1.3 dropEvent()	2052
9.805.1.4 mimeTypees()	2052
9.805.1.5 setReadOnlyDrop()	2052
9.806 Digikam::ItemExtendedProperties Class Reference	2052
9.806.1 Member Function Documentation	2053
9.806.1.1 intellectualGenre()	2053
9.806.1.2 jobId()	2053
9.806.1.3 location()	2053
9.806.1.4 scene()	2053
9.806.1.5 similarityTo()	2054
9.806.1.6 subjectCode()	2054
9.807 Digikam::ItemFaceDelegate Class Reference	2055
9.807.1 Member Function Documentation	2060
9.807.1.1 thumbnailPixmap()	2060
9.807.1.2 updateRects()	2060
9.808 Digikam::ItemFilterModel Class Reference	2061
9.808.1 Member Enumeration Documentation	2065
9.808.1.1 ItemFilterModelRoles	2065
9.808.2 Member Function Documentation	2065
9.808.2.1 categoryIdentifier()	2065
9.808.2.2 compareCategories()	2065
9.808.2.3 compareInfosCategories() [1/2]	2066
9.808.2.4 compareInfosCategories() [2/2]	2066
9.808.2.5 data()	2066
9.808.2.6 filterMatchesForText	2067
9.808.2.7 imageFilterModel()	2067
9.808.2.8 infosLessThan()	2067
9.808.2.9 setDayFilter	2067
9.808.2.10 setDirectSourceItemModel()	2067
9.808.2.11 setItemFilterSettings	2067
9.808.2.12 subSortLessThan()	2068
9.808.2.13 suggestedWatchFlags()	2068
9.809 Digikam::ItemFilterModelFilterer Class Reference	2069
9.809.1 Member Function Documentation	2071
9.809.1.1 process()	2071
9.810 Digikam::ItemFilterModelPrepareHook Class Reference	2071
9.811 Digikam::ItemFilterModelPreparer Class Reference	2072
9.811.1 Member Function Documentation	2074
9.811.1.1 process()	2074
9.812 Digikam::ItemFilterModelWorker Class Reference	2075
9.813 Digikam::ItemFilterSettings Class Reference	2077

9.813.1 Member Function Documentation	2078
9.813.1.1 matches()	2078
9.813.1.2 watchFlags()	2078
9.814 Digikam::ItemFiltersHistoryItemDelegate Class Reference	2079
9.815 Digikam::ItemFiltersHistoryModel Class Reference	2080
9.816 Digikam::ItemFiltersHistoryTreeItem Class Reference	2081
9.817 Digikam::ItemFullScreenOverlay Class Reference	2082
9.817.1 Member Function Documentation	2085
9.817.1.1 checkIndex()	2085
9.817.1.2 createButton()	2085
9.817.1.3 setActive()	2085
9.817.1.4 updateButton()	2085
9.817.1.5 widgetEnterEvent()	2086
9.817.1.6 widgetLeaveEvent()	2086
9.818 Digikam::ItemFullScreenOverlayButton Class Reference	2087
9.818.1 Member Function Documentation	2088
9.818.1.1 icon()	2088
9.818.1.2 sizeHint()	2089
9.818.1.3 updateToolTip()	2089
9.819 Digikam::ItemGPS Class Reference	2090
9.819.1 Member Function Documentation	2093
9.819.1.1 loadImageData()	2093
9.819.1.2 saveChanges()	2093
9.820 Digikam::ItemGPSModelHelper Class Reference	2094
9.820.1 Member Function Documentation	2095
9.820.1.1 bestRepresentativeIndexFromList()	2095
9.820.1.2 itemCoordinates()	2096
9.820.1.3 model()	2096
9.820.1.4 pixmapFromRepresentativeIndex()	2096
9.820.1.5 selectionModel()	2096
9.821 Digikam::ItemHistoryGraph Class Reference	2096
9.821.1 Member Enumeration Documentation	2097
9.821.1.1 HistoryLoadingFlag	2097
9.821.2 Member Function Documentation	2098
9.821.2.1 addHistory()	2098
9.821.2.2 addRelations()	2098
9.821.2.3 addScannedHistory()	2098
9.821.2.4 categorize()	2098
9.821.2.5 fromInfo()	2098
9.821.2.6 hasEdges()	2099
9.821.2.7 leafImages()	2099
9.821.2.8 reduceEdges()	2099

---

9.821.2.9 relationCloud()	2099
9.821.2.10 rootImages()	2099
9.822 Digikam::ItemHistoryGraphData Class Reference	2100
9.823 Digikam::ItemHistoryGraphModel Class Reference	2105
9.823.1 Member Function Documentation	2107
9.823.1.1 imageModel()	2107
9.823.1.2 imageModelIndex()	2107
9.823.1.3 indexForInfo()	2107
9.823.1.4 setHistory()	2107
9.824 Digikam::ItemIconView Class Reference	2108
9.824.1 Member Function Documentation	2112
9.824.1.1 allNeedGroupResolving()	2112
9.824.1.2 allUrls()	2112
9.824.1.3 selectedUrls()	2112
9.824.1.4 slotFitToWindow	2112
9.824.1.5 slotImageQualitySorter	2113
9.824.1.6 slotRemoveTag	2113
9.825 Digikam::ItemInfo Class Reference	2113
9.825.1 Detailed Description	2117
9.825.2 Constructor & Destructor Documentation	2117
9.825.2.1 ItemInfo() [1/2]	2117
9.825.2.2 ItemInfo() [2/2]	2117
9.825.3 Member Function Documentation	2117
9.825.3.1 addTagPaths()	2117
9.825.3.2 albumId()	2118
9.825.3.3 aspectRatio()	2118
9.825.3.4 comment()	2118
9.825.3.5 copyItem()	2118
9.825.3.6 dateTime()	2118
9.825.3.7 dimensions()	2119
9.825.3.8 faceCount()	2119
9.825.3.9 fileSize()	2119
9.825.3.10 fileUrl()	2119
9.825.3.11 getDatabaseFieldsRaw()	2119
9.825.3.12 getSuggestedNames()	2119
9.825.3.13 groupImage()	2120
9.825.3.14 id()	2120
9.825.3.15 imageComments()	2120
9.825.3.16 imageCopyright()	2120
9.825.3.17 imageExtendedProperties()	2120
9.825.3.18 imageHistory()	2120
9.825.3.19 longitudeNumber()	2120

9.825.3.20 modDateTime()	2121
9.825.3.21 name()	2121
9.825.3.22 removeTag()	2121
9.825.3.23 setDateTime()	2121
9.825.3.24 setMetadataTemplate()	2121
9.825.3.25 setModDateTime()	2122
9.825.3.26 setName()	2122
9.825.3.27 setTag()	2122
9.825.3.28 tagIds()	2122
9.825.3.29 title()	2122
9.825.3.30 unconfirmedFaceCount()	2123
9.825.3.31 uniqueHash()	2123
9.826 Digikam::ItemInfoAlbumsJob Class Reference	2123
9.827 Digikam::ItemInfoCache Class Reference	2124
9.827.1 Member Function Documentation	2125
9.827.1.1 cacheByName()	2125
9.827.1.2 infoForId()	2125
9.827.1.3 infoForPath()	2125
9.828 Digikam::ItemInfoData Class Reference	2126
9.829 Digikam::ItemInfoJob Class Reference	2128
9.830 Digikam::ItemInfoList Class Reference	2129
9.830.1 Member Function Documentation	2129
9.830.1.1 singleGroupMainItem()	2129
9.831 Digikam::ItemInfoReadLocker Class Reference	2130
9.832 Digikam::ItemInfoSet Class Reference	2130
9.833 Digikam::ItemInfoStatic Class Reference	2130
9.834 Digikam::ItemInfoTaskSplitter Class Reference	2131
9.835 Digikam::ItemInfoWriteLocker Class Reference	2133
9.836 Digikam::ItemListDragDropHandler Class Reference	2134
9.837 Digikam::ItemLister Class Reference	2134
9.837.1 Member Function Documentation	2135
9.837.1.1 listHaarSearch()	2135
9.837.1.2 listImageTagPropertySearch()	2135
9.837.1.3 listPALbum()	2136
9.837.1.4 listSearch()	2136
9.837.1.5 setListOnlyAvailable()	2136
9.837.1.6 setRecursive()	2136
9.838 Digikam::ItemListerJobGrowingPartsSendingReceiver Class Reference	2137
9.838.1 Member Function Documentation	2138
9.838.1.1 receive()	2138
9.839 Digikam::ItemListerJobPartsSendingReceiver Class Reference	2139
9.839.1 Member Function Documentation	2140

---

9.839.1.1 receive()	2140
9.840 Digikam::ItemListerJobReceiver Class Reference	2141
9.840.1 Member Function Documentation	2142
9.840.1.1 error()	2142
9.841 Digikam::ItemListerReceiver Class Reference	2143
9.842 Digikam::ItemListerRecord Class Reference	2144
9.843 Digikam::ItemListerValueListReceiver Class Reference	2145
9.843.1 Member Function Documentation	2146
9.843.1.1 error()	2146
9.843.1.2 receive()	2146
9.844 Digikam::ItemListModel Class Reference	2147
9.845 Digikam::ItemMarkerTiler Class Reference	2153
9.845.1 Member Function Documentation	2155
9.845.1.1 bestRepresentativeIndexFromList()	2155
9.845.1.2 getGlobalGroupState()	2155
9.845.1.3 getTile()	2155
9.845.1.4 getTileGroupState()	2155
9.845.1.5 getTileMarkerCount()	2155
9.845.1.6 getTileRepresentativeMarker()	2155
9.845.1.7 getTileSelectedCount()	2156
9.845.1.8 indicesEqual()	2156
9.845.1.9 onIndicesClicked()	2156
9.845.1.10 onIndicesMoved()	2156
9.845.1.11 pixmapFromRepresentativeIndex()	2156
9.845.1.12 prepareTiles()	2156
9.845.1.13 regenerateTiles()	2157
9.845.1.14 removeMarkerIndexFromGrid()	2157
9.845.1.15 setActive()	2158
9.845.1.16 tileNew()	2158
9.845.1.17 tilerFlags()	2158
9.846 Digikam::ItemMetadataAdjustmentHint Class Reference	2158
9.846.1 Member Enumeration Documentation	2159
9.846.1.1 AdjustmentStatus	2159
9.847 Digikam::ItemModel Class Reference	2160
9.847.1 Member Enumeration Documentation	2164
9.847.1.1 ItemModelRoles	2164
9.847.2 Member Function Documentation	2165
9.847.2.1 addItemInfo()	2165
9.847.2.2 addItemInfoSynchronously()	2165
9.847.2.3 allRefreshingFinished	2165
9.847.2.4 ensureHasItemInfo()	2165
9.847.2.5 imageInfo() [1/2]	2165



9.847.2.6 imageInfo() [2/2]	2165
9.847.2.7 imageInfosAboutToBeAdded	2166
9.847.2.8 imageInfosAboutToBeRemoved	2166
9.847.2.9 imageInfosAdded	2166
9.847.2.10 imageInfosCleared()	2166
9.847.2.11 imageInfosRemoved	2166
9.847.2.12 indexForPath()	2166
9.847.2.13 isRefreshing()	2167
9.847.2.14 readyForIncrementalRefresh	2167
9.847.2.15 requestIncrementalRefresh()	2167
9.847.2.16 retrievalItemInfo()	2167
9.847.2.17 setKeepsFilePathCache()	2167
9.847.2.18 setPreprocessor()	2167
9.847.2.19 setSendRemovalSignals()	2168
9.847.2.20 setWatchFlags()	2168
9.847.2.21 startIncrementalRefresh()	2168
9.847.2.22 startRefresh()	2168
9.848 Digikam::ItemPosition Class Reference	2168
9.848.1 Constructor & Destructor Documentation	2169
9.848.1.1 ItemPosition()	2169
9.848.2 Member Function Documentation	2170
9.848.2.1 apply()	2170
9.848.2.2 isEmpty()	2170
9.848.2.3 latitude()	2170
9.848.2.4 latitudeNumber()	2170
9.848.2.5 latitudeUserPresentableNumbers()	2170
9.848.2.6 remove()	2170
9.848.2.7 setLatitude() [1/2]	2171
9.848.2.8 setLatitude() [2/2]	2171
9.849 Digikam::ItemPreviewCanvas Class Reference	2172
9.850 Digikam::ItemPreviewView Class Reference	2175
9.850.1 Member Function Documentation	2177
9.850.1.1 acceptsMouseClicked()	2177
9.851 Digikam::ItemPropertiesColorsTab Class Reference	2178
9.852 Digikam::ItemPropertiesGPSTab Class Reference	2179
9.853 Digikam::ItemPropertiesHistoryTab Class Reference	2180
9.854 Digikam::ItemPropertiesMetadataTab Class Reference	2181
9.855 Digikam::ItemPropertiesSideBar Class Reference	2182
9.855.1 Member Function Documentation	2186
9.855.1.1 doLoadState()	2186
9.855.1.2 doSaveState()	2186
9.856 Digikam::ItemPropertiesSideBarDB Class Reference	2187

---

9.856.1 Member Function Documentation	2192
9.856.1.1 doLoadState()	2192
9.856.1.2 doSaveState()	2192
9.856.1.3 itemChanged()	2192
9.857 Digikam::ItemPropertiesTab Class Reference	2193
9.857.1 Member Function Documentation	2196
9.857.1.1 humanReadableBytesCount()	2196
9.857.1.2 shortenedTagPaths()	2196
9.858 Digikam::ItemPropertiesVersionsTab Class Reference	2197
9.859 Digikam::ItemQueryBuilder Class Reference	2198
9.859.1 Member Function Documentation	2198
9.859.1.1 setImageTagPropertiesJoined()	2198
9.860 Digikam::ItemQueryPostHook Class Reference	2198
9.861 Digikam::ItemQueryPostHooks Class Reference	2199
9.861.1 Member Function Documentation	2199
9.861.1.1 addHook()	2199
9.861.1.2 checkPosition()	2199
9.862 Digikam::ItemRatingOverlay Class Reference	2200
9.862.1 Member Function Documentation	2203
9.862.1.1 createWidget()	2203
9.862.1.2 hide()	2203
9.862.1.3 setActive()	2203
9.862.1.4 slotEntered()	2203
9.862.1.5 visualChange()	2204
9.862.1.6 widgetEnterEvent()	2204
9.862.1.7 widgetLeaveEvent()	2204
9.863 Digikam::ItemRotateOverlay Class Reference	2205
9.863.1 Member Function Documentation	2208
9.863.1.1 checkIndex()	2208
9.863.1.2 createButton()	2208
9.863.1.3 setActive()	2208
9.863.1.4 updateButton()	2208
9.863.1.5 widgetEnterEvent()	2209
9.863.1.6 widgetLeaveEvent()	2209
9.864 Digikam::ItemRotateOverlayButton Class Reference	2210
9.864.1 Member Function Documentation	2212
9.864.1.1 icon()	2212
9.864.1.2 sizeHint()	2212
9.864.1.3 updateToolTip()	2212
9.865 Digikam::ItemScanInfo Class Reference	2212
9.866 Digikam::ItemScanner Class Reference	2212
9.866.1 Constructor & Destructor Documentation	2215

9.866.1.1 ItemScanner() [1/3]	2215
9.866.1.2 ItemScanner() [2/3]	2215
9.866.1.3 ItemScanner() [3/3]	2215
9.866.2 Member Function Documentation	2216
9.866.2.1 commit()	2216
9.866.2.2 copiedFrom()	2216
9.866.2.3 creationDateFromFilesystem()	2216
9.866.2.4 fileModified()	2216
9.866.2.5 fillCommonContainer()	2216
9.866.2.6 fillVideoMetadataContainer()	2217
9.866.2.7 iptcCorePropertyName()	2217
9.866.2.8 itemScanInfo()	2217
9.866.2.9 loadFromDisk()	2217
9.866.2.10 newFileFullScan()	2217
9.866.2.11 resolvedImageHistory()	2217
9.866.2.12 resolveImageHistory()	2218
9.866.2.13 sameReferredImage()	2218
9.866.2.14 setCategory()	2218
9.866.2.15 sortByProximity()	2218
9.867 Digikam::ItemSelectionOverlay Class Reference	2219
9.867.1 Member Function Documentation	2222
9.867.1.1 createButton()	2222
9.867.1.2 setActive()	2222
9.867.1.3 updateButton()	2222
9.868 Digikam::ItemSelectionOverlayButton Class Reference	2223
9.868.1 Member Function Documentation	2224
9.868.1.1 icon()	2224
9.868.1.2 sizeHint()	2225
9.868.1.3 updateToolTip()	2225
9.869 Digikam::ItemSelectionPropertiesTab Class Reference	2226
9.870 Digikam::ItemShortInfo Class Reference	2228
9.871 Digikam::ItemSortCollator Class Reference	2228
9.871.1 Member Function Documentation	2229
9.871.1.1 instance()	2229
9.872 Digikam::ItemSortSettings Class Reference	2229
9.872.1 Member Enumeration Documentation	2230
9.872.1.1 CategorizationMode	2230
9.872.1.2 SortOrder	2230
9.872.1.3 SortRole	2231
9.872.2 Member Function Documentation	2231
9.872.2.1 compare()	2231
9.872.2.2 compareCategories()	2231

---

9.872.2.3 lessThan() [1/2]	2231
9.872.2.4 lessThan() [2/2]	2232
9.872.2.5 lessThanByOrder()	2232
9.872.2.6 watchFlags()	2232
9.873 Digikam::ItemTagPair Class Reference	2232
9.873.1 Constructor & Destructor Documentation	2233
9.873.1.1 ItemTagPair()	2233
9.873.2 Member Function Documentation	2233
9.873.2.1 addProperty()	2233
9.873.2.2 availablePairs()	2234
9.874 Digikam::ItemThumbnailBar Class Reference	2234
9.874.1 Member Function Documentation	2240
9.874.1.1 hasHiddenGroupedImages()	2240
9.874.1.2 setModelsFiltered()	2241
9.874.1.3 slotSetupChanged()	2241
9.875 Digikam::ItemThumbnailDelegate Class Reference	2242
9.875.1 Member Function Documentation	2246
9.875.1.1 acceptsActivation()	2246
9.875.1.2 setDefaultViewOptions()	2247
9.875.1.3 updateContentWidth()	2247
9.875.1.4 updateRects()	2247
9.876 Digikam::ItemThumbnailModel Class Reference	2248
9.876.1 Constructor & Destructor Documentation	2253
9.876.1.1 ItemThumbnailModel()	2253
9.876.2 Member Function Documentation	2253
9.876.2.1 data()	2253
9.876.2.2 imageInfosCleared()	2253
9.876.2.3 preloadThumbnails	2253
9.876.2.4 setData()	2253
9.876.2.5 setEmitDataChanged()	2254
9.876.2.6 setPreloadThumbnails()	2254
9.876.2.7 setThumbnailLoadThread()	2254
9.877 Digikam::ItemVersionsModel Class Reference	2255
9.878 Digikam::ItemViewCategorized Class Reference	2256
9.878.1 Member Function Documentation	2260
9.878.1.1 clicked	2260
9.878.1.2 filterModel()	2260
9.878.1.3 keyPressed	2260
9.878.1.4 mapIndexForDragDrop()	2260
9.878.1.5 nextIndexHint()	2261
9.878.1.6 pixmapForDrag()	2261
9.878.1.7 rowsRemoved()	2261

9.878.1.8 selectionChanged	2261
9.878.1.9 setScrollStepGranularity()	2261
9.878.1.10 setSpacing()	2261
9.878.1.11 showContextMenuOnIndex()	2262
9.878.1.12 showToolTip()	2262
9.879 Digikam::ItemViewDelegate Class Reference	2263
9.879.1 Member Function Documentation	2266
9.879.1.1 acceptsActivation()	2266
9.879.1.2 acceptsToolTip()	2266
9.879.1.3 asDelegate()	2267
9.879.1.4 gridSize()	2267
9.879.1.5 imageInformationRect()	2267
9.879.1.6 mouseMoved()	2267
9.879.1.7 pixmapRect()	2267
9.879.1.8 setDefaultViewOptions()	2267
9.879.1.9 setRatingEdited()	2268
9.879.1.10 setSpacing()	2268
9.879.1.11 setThumbnailSize()	2268
9.880 Digikam::ItemViewHoverButton Class Reference	2268
9.880.1 Member Function Documentation	2269
9.880.1.1 icon()	2269
9.880.1.2 sizeHint()	2269
9.880.1.3 updateToolTip()	2270
9.881 Digikam::ItemViewImportDelegate Class Reference	2271
9.881.1 Member Function Documentation	2274
9.881.1.1 acceptsActivation()	2274
9.881.1.2 acceptsToolTip()	2274
9.881.1.3 asDelegate()	2274
9.881.1.4 gridSize()	2275
9.881.1.5 imageInformationRect()	2275
9.881.1.6 invalidatePaintingCache()	2275
9.881.1.7 mouseMoved()	2275
9.881.1.8 pixmapRect()	2275
9.881.1.9 prepareRatingPxmmaps()	2275
9.881.1.10 setDefaultViewOptions()	2276
9.881.1.11 setRatingEdited()	2276
9.881.1.12 setSpacing()	2276
9.881.1.13 setThumbnailSize()	2276
9.882 Digikam::ItemViewToolTip Class Reference	2277
9.882.1 Member Function Documentation	2278
9.882.1.1 repositionRect()	2278
9.882.1.2 show()	2278

---

9.882.1.3 tipContents()	2278
9.883 Digikam::ItemViewUtilities Class Reference	2279
9.884 Digikam::ItemVisibilityController Class Reference	2281
9.884.1 Member Enumeration Documentation	2283
9.884.1.1 IncludeFadingOutMode	2283
9.884.1.2 State	2283
9.884.2 Member Function Documentation	2283
9.884.2.1 addItem()	2283
9.884.2.2 createAnimation()	2284
9.884.2.3 hasVisibleItems()	2284
9.884.2.4 hideAndRemoveItem	2284
9.884.2.5 setItemThatShallBeShown	2284
9.884.2.6 show	2284
9.884.2.7 showItem	2284
9.885 Digikam::ItemVisibilityControllerPropertyObject Class Reference	2285
9.885.1 Constructor & Destructor Documentation	2286
9.885.1.1 ItemVisibilityControllerPropertyObject()	2286
9.886 Digikam::JPEGUtils::digikam_source_mgr Struct Reference	2286
9.887 Digikam::JPEGUtils::JpegRotator Class Reference	2286
9.887.1 Constructor & Destructor Documentation	2287
9.887.1.1 JpegRotator()	2287
9.887.2 Member Function Documentation	2287
9.887.2.1 autoExifTransform()	2287
9.887.2.2 exifTransform() [1/2]	2287
9.887.2.3 exifTransform() [2/2]	2287
9.887.2.4 setCurrentOrientation()	2288
9.887.2.5 setDestinationFile()	2288
9.887.2.6 setDocumentName()	2288
9.888 Digikam::KDNodeBase Class Reference	2289
9.888.1 Member Function Documentation	2290
9.888.1.1 createNode()	2290
9.889 Digikam::KDNodeBase::NodeCompareResult Struct Reference	2290
9.890 Digikam::KDNodeOpenFace Class Reference	2291
9.890.1 Member Function Documentation	2292
9.890.1.1 createNode()	2292
9.890.1.2 nodeCompare()	2292
9.891 Digikam::KDNodeSFace Class Reference	2293
9.891.1 Member Function Documentation	2294
9.891.1.1 createNode()	2294
9.891.1.2 nodeCompare()	2294
9.892 Digikam::KDTreeBase Class Reference	2295
9.892.1 Constructor & Destructor Documentation	2295

---

9.892.1.1 KDTreeBase()	2295
9.892.2 Member Function Documentation	2296
9.892.2.1 add()	2296
9.892.2.2 createNode()	2296
9.892.2.3 getClosestNeighbors()	2296
9.893 Digikam::KDTreeOpenFace Class Reference	2297
9.894 Digikam::KDTreeSFace Class Reference	2298
9.895 Digikam::KeywordSearchReader Class Reference	2299
9.896 Digikam::KeywordSearchWriter Class Reference	2301
9.897 Digikam::LabelsSideBarWidget Class Reference	2303
9.897.1 Member Function Documentation	2305
9.897.1.1 applySettings()	2305
9.897.1.2 changeAlbumFromHistory()	2305
9.897.1.3 doLoadState()	2305
9.897.1.4 doSaveState()	2305
9.897.1.5 getCaption()	2306
9.897.1.6 getIcon()	2306
9.897.1.7 setActive()	2306
9.898 Digikam::LabelsTreeView Class Reference	2307
9.898.1 Member Function Documentation	2309
9.898.1.1 colorRectPixmap()	2309
9.898.1.2 doLoadState()	2309
9.898.1.3 doSaveState()	2309
9.898.1.4 goldenStarPixmap()	2309
9.898.1.5 isCheckable()	2309
9.898.1.6 isLoadingState()	2310
9.898.1.7 restoreSelectionFromHistory()	2310
9.898.1.8 selectedLabels()	2310
9.899 Digikam::LanguagesList Class Reference	2311
9.900 Digikam::LcmsLock Class Reference	2311
9.901 Digikam::LensDistortionFilter Class Reference	2312
9.901.1 Member Function Documentation	2316
9.901.1.1 filterAction()	2316
9.901.1.2 filterIdentifier()	2316
9.901.1.3 readParameters()	2316
9.902 Digikam::LensDistortionPixelAccess Class Reference	2316
9.902.1 Detailed Description	2316
9.903 Digikam::LensFunCameraSelector Class Reference	2317
9.904 Digikam::LensFunContainer Class Reference	2318
9.905 Digikam::LensFunFilter Class Reference	2319
9.905.1 Member Function Documentation	2323
9.905.1.1 filterAction()	2323

---

9.905.1.2 filterIdentifier()	2323
9.905.1.3 readParameters()	2323
9.906 Digikam::LensFunface Class Reference	2323
9.907 Digikam::LensFunSettings Class Reference	2324
9.908 Digikam::LevelsContainer Class Reference	2325
9.909 Digikam::LevelsFilter Class Reference	2326
9.909.1 Member Function Documentation	2330
9.909.1.1 filterAction()	2330
9.909.1.2 filterIdentifier()	2330
9.909.1.3 readParameters()	2330
9.910 Digikam::LibsInfoDlg Class Reference	2331
9.910.1 Constructor & Destructor Documentation	2332
9.910.1.1 LibsInfoDlg()	2332
9.911 Digikam::LightTablePreview Class Reference	2333
9.912 Digikam::LightTableThumbBar Class Reference	2337
9.913 Digikam::LightTableView Class Reference	2345
9.914 Digikam::LightTableWindow Class Reference	2347
9.914.1 Member Function Documentation	2350
9.914.1.1 infoface()	2350
9.914.1.2 loadItemInfos()	2350
9.914.1.3 slotApplicationSettingsChanged	2350
9.915 Digikam::ListItem Class Reference	2351
9.915.1 Member Function Documentation	2352
9.915.1.1 containsItem()	2352
9.916 Digikam::ListViewComboBox Class Reference	2353
9.916.1 Constructor & Destructor Documentation	2355
9.916.1.1 ListViewComboBox()	2355
9.916.2 Member Function Documentation	2355
9.916.2.1 installView()	2355
9.916.2.2 sendViewportEventToView()	2355
9.916.2.3 view()	2355
9.917 Digikam::LoadingCache Class Reference	2356
9.917.1 Member Function Documentation	2358
9.917.1.1 addLoadingProcess()	2358
9.917.1.2 fileChanged	2358
9.917.1.3 notifyFileChanged()	2358
9.917.1.4 putImage()	2358
9.917.1.5 retrieveThumbnail()	2358
9.917.1.6 setCacheSize()	2358
9.917.1.7 setFileWatch()	2359
9.917.1.8 setThumbnailCacheSize()	2359
9.918 Digikam::LoadingCache::CacheLock Class Reference	2359



---

9.918.1 Detailed Description	2359
9.919 Digikam::LoadingCacheFileWatch Class Reference	2360
9.919.1 Member Function Documentation	2361
9.919.1.1 notifyFileChanged()	2361
9.920 Digikam::LoadingCacheInterface Class Reference	2361
9.920.1 Member Function Documentation	2361
9.920.1.1 cleanCache()	2361
9.920.1.2 cleanThumbnailCache()	2361
9.920.1.3 setCacheOptions()	2362
9.921 Digikam::LoadingDescription Class Reference	2362
9.921.1 Member Enumeration Documentation	2363
9.921.1.1 ColorManagementSettings	2363
9.921.1.2 RawDecodingHint	2363
9.921.2 Constructor & Destructor Documentation	2364
9.921.2.1 LoadingDescription()	2364
9.921.3 Member Function Documentation	2364
9.921.3.1 lookupCacheKeys()	2364
9.921.3.2 needCheckRawDecoding()	2364
9.922 Digikam::LoadingDescription::PostProcessingParameters Class Reference	2364
9.923 Digikam::LoadingDescription::PreviewParameters Class Reference	2365
9.924 Digikam::LoadingProcess Class Reference	2366
9.925 Digikam::LoadingProcessListener Class Reference	2367
9.926 Digikam::LoadingTask Class Reference	2368
9.926.1 Member Function Documentation	2369
9.926.1.1 continueQuery()	2369
9.926.1.2 execute()	2370
9.926.1.3 progressInfo()	2370
9.926.1.4 type()	2370
9.927 Digikam::LoadSaveFileInfoProvider Class Reference	2370
9.927.1 Member Function Documentation	2371
9.927.1.1 dimensionsHint()	2371
9.927.1.2 orientationHint()	2371
9.928 Digikam::LoadSaveNotifier Class Reference	2372
9.928.1 Member Function Documentation	2373
9.928.1.1 thumbnailLoaded()	2373
9.929 Digikam::LoadSaveTask Class Reference	2374
9.930 Digikam::LoadSaveThread Class Reference	2376
9.930.1 Member Enumeration Documentation	2379
9.930.1.1 AccessMode	2379
9.930.1.2 NotificationPolicy	2379
9.930.2 Member Function Documentation	2379
9.930.2.1 imageLoaded()	2379

---

9.930.2.2 imageSaved()	2379
9.930.2.3 imageStartedLoading()	2380
9.930.2.4 imageStartedSaving()	2380
9.930.2.5 loadingProgress()	2380
9.930.2.6 moreCompleteLoadingAvailable()	2380
9.930.2.7 run()	2380
9.930.2.8 savingProgress()	2380
9.930.2.9 signalImageLoaded	2381
9.930.2.10 signalImageStartedLoading	2381
9.930.2.11 signalLoadingProgress	2381
9.930.2.12 signalMoreCompleteLoadingAvailable	2381
9.930.2.13 thumbnailLoaded()	2381
9.931 Digikam::LocalContrastContainer Class Reference	2382
9.932 Digikam::LocalContrastFilter Class Reference	2383
9.932.1 Member Function Documentation	2387
9.932.1.1 filterAction()	2387
9.932.1.2 filterIdentifier()	2387
9.932.1.3 readParameters()	2387
9.933 Digikam::LocalContrastSettings Class Reference	2387
9.934 Digikam::LocalizeConfig Class Reference	2388
9.935 Digikam::LocalizeContainer Class Reference	2389
9.935.1 Member Data Documentation	2389
9.935.1.1 ignoredWords	2389
9.936 Digikam::LocalizeSelector Class Reference	2390
9.937 Digikam::LocalizeSelectorList Class Reference	2391
9.938 Digikam::LocalizeSettings Class Reference	2392
9.938.1 Member Function Documentation	2393
9.938.1.1 instance()	2393
9.939 Digikam::LookupAltitude Class Reference	2394
9.940 Digikam::LookupAltitude::Request Class Reference	2395
9.941 Digikam::LookupAltitudeGeonames Class Reference	2396
9.941.1 Member Function Documentation	2397
9.941.1.1 backendHumanName()	2397
9.941.1.2 backendName()	2397
9.941.1.3 cancel()	2397
9.941.1.4 errorMessage()	2398
9.941.1.5 getRequest()	2398
9.941.1.6 getRequests()	2398
9.941.1.7 getStatus()	2398
9.941.1.8 startLookup()	2398
9.942 Digikam::LookupFactory Class Reference	2398
9.943 Digikam::MaintenanceData Class Reference	2399

---

9.944 Digikam::MaintenanceDlg Class Reference	2399
9.945 Digikam::MaintenanceMgr Class Reference	2400
9.946 Digikam::MaintenanceSettings Class Reference	2400
9.946.1 Member Data Documentation	2402
9.946.1.1 qualityScanMode	2402
9.947 Digikam::MaintenanceThread Class Reference	2403
9.947.1 Member Function Documentation	2405
9.947.1.1 signalAdvance	2405
9.948 Digikam::MaintenanceTool Class Reference	2405
9.948.1 Member Function Documentation	2407
9.948.1.1 setUseMultiCoreCPU()	2407
9.949 Digikam::MakerNoteWidget Class Reference	2408
9.949.1 Member Function Documentation	2410
9.949.1.1 getMetadataTitle()	2410
9.949.1.2 getTagDescription()	2410
9.949.1.3 getTagTitle()	2410
9.949.1.4 loadFromURL()	2410
9.950 Digikam::ManagedLoadSaveThread Class Reference	2411
9.950.1 Member Enumeration Documentation	2415
9.950.1.1 LoadingMode	2415
9.950.1.2 LoadingPolicy	2415
9.950.1.3 LoadingTaskFilter	2416
9.950.1.4 TerminationPolicy	2416
9.950.2 Member Function Documentation	2416
9.950.2.1 load()	2416
9.950.2.2 setLoadingPolicy()	2417
9.950.2.3 stopLoading()	2417
9.950.2.4 stopSaving()	2417
9.951 Digikam::MapBackend Class Reference	2418
9.951.1 Member Function Documentation	2420
9.951.1.1 centerOn()	2420
9.951.1.2 mapWidget()	2420
9.951.1.3 mouseModeChanged()	2420
9.951.1.4 setActive()	2420
9.952 Digikam::MapDragData Class Reference	2420
9.953 Digikam::MapDragDropHandler Class Reference	2421
9.953.1 Member Function Documentation	2422
9.953.1.1 accepts()	2422
9.953.1.2 createMimeData()	2422
9.953.1.3 dropEvent()	2422
9.954 Digikam::MapViewModelHelper Class Reference	2423
9.954.1 Member Function Documentation	2424

9.954.1.1	bestRepresentativeIndexFromList()	2424
9.954.1.2	itemCoordinates()	2425
9.954.1.3	model()	2425
9.954.1.4	onIndicesClicked()	2425
9.954.1.5	pixmapFromRepresentativeIndex()	2426
9.954.1.6	selectionModel()	2426
9.955	Digikam::MapWidget Class Reference	2426
9.955.1	Detailed Description	2430
9.955.2	Constructor & Destructor Documentation	2430
9.955.2.1	~MapWidget()	2430
9.955.3	Member Function Documentation	2430
9.955.3.1	addUngroupedModel()	2430
9.955.3.2	adjustBoundariesToGroupedMarkers()	2430
9.955.3.3	applyCacheToBackend()	2431
9.955.3.4	convertZoomToBackendZoom()	2431
9.955.3.5	dragEnterEvent()	2431
9.955.3.6	getColorInfos() [1/2]	2431
9.955.3.7	getColorInfos() [2/2]	2431
9.955.3.8	getDecoratedPixmapForCluster()	2432
9.955.3.9	removeUngroupedModel()	2432
9.955.3.10	setBackend()	2432
9.955.3.11	setGroupedModel()	2432
9.955.3.12	setSortKey()	2432
9.955.3.13	setThumbnailSize()	2432
9.955.3.14	slotClustersClicked	2432
9.955.3.15	slotClustersMoved	2432
9.955.3.16	slotItemDisplaySettingsChanged	2433
9.955.3.17	slotMouseModeChanged	2433
9.955.3.18	slotNewSelectionFromMap	2433
9.955.3.19	slotUpdateActionsEnabled	2433
9.955.3.20	updateClusters()	2433
9.956	Digikam::MapWidgetView Class Reference	2433
9.956.1	Constructor & Destructor Documentation	2436
9.956.1.1	MapWidgetView()	2436
9.956.2	Member Function Documentation	2436
9.956.2.1	currentCamItemInfo()	2436
9.956.2.2	currentItemInfo()	2436
9.956.2.3	doLoadState()	2436
9.956.2.4	doSaveState()	2436
9.956.2.5	getActiveState()	2436
9.956.2.6	setActive()	2436
9.957	Digikam::Mat Struct Reference	2437

---

9.957.1 Detailed Description	2437
9.958 Digikam::MdKeyListViewItem Class Reference	2437
9.959 Digikam::MediaPlayerView Class Reference	2438
9.960 Digikam::MetadataHub Class Reference	2439
9.960.1 Member Enumeration Documentation	2440
9.960.1.1 Status	2440
9.960.1.2 WriteMode	2440
9.960.2 Member Function Documentation	2441
9.960.2.1 cleanupTags()	2441
9.960.2.2 load()	2441
9.960.2.3 willWriteMetadata()	2441
9.960.2.4 write() [1/3]	2441
9.960.2.5 write() [2/3]	2442
9.960.2.6 write() [3/3]	2442
9.960.2.7 writeTags() [1/2]	2443
9.960.2.8 writeTags() [2/2]	2443
9.960.2.9 writeToBaloo()	2444
9.960.2.10 writeToMetadata()	2444
9.961 Digikam::MetadataHubMngr Class Reference	2445
9.962 Digikam::MetadataKeys Class Reference	2446
9.962.1 Member Function Documentation	2447
9.962.1.1 getDbValue()	2447
9.963 Digikam::MetadataListView Class Reference	2448
9.964 Digikam::MetadataListViewItem Class Reference	2449
9.965 Digikam::MetadataOption Class Reference	2450
9.965.1 Member Function Documentation	2452
9.965.1.1 parseOperation()	2452
9.966 Digikam::MetadataOptionDialog Class Reference	2453
9.967 Digikam::MetadataPage Class Reference	2454
9.968 Digikam::MetadataPanel Class Reference	2455
9.969 Digikam::MetadataRemover Class Reference	2457
9.969.1 Constructor & Destructor Documentation	2460
9.969.1.1 MetadataRemover()	2460
9.969.2 Member Function Documentation	2460
9.969.2.1 setUseMultiCoreCPU()	2460
9.970 Digikam::MetadataRemoveTask Class Reference	2461
9.971 Digikam::MetadataSelector Class Reference	2463
9.972 Digikam::MetadataSelectorItem Class Reference	2464
9.973 Digikam::MetadataSelectorView Class Reference	2465
9.974 Digikam::MetadataStatusBar Class Reference	2466
9.975 Digikam::MetadataSynchronizer Class Reference	2467
9.975.1 Constructor & Destructor Documentation	2470

---

9.975.1.1 MetadataSynchronizer()	2470
9.975.2 Member Function Documentation	2470
9.975.2.1 setUseMultiCoreCPU()	2470
9.976 Digikam::MetadataSyncTask Class Reference	2471
9.977 Digikam::MetadataWidget Class Reference	2473
9.978 Digikam::MetaEngine Class Reference	2475
9.978.1 Member Typedef Documentation	2484
9.978.1.1 AltLangMap	2484
9.978.1.2 TagsMap	2484
9.978.2 Member Enumeration Documentation	2484
9.978.2.1 Backend	2484
9.978.2.2 MetadataWritingMode	2485
9.978.3 Member Function Documentation	2485
9.978.3.1 addToXmpTagStringBag()	2485
9.978.3.2 applyChanges()	2485
9.978.3.3 backendName()	2485
9.978.3.4 convertDegreeAngleToDouble()	2485
9.978.3.5 convertFromGPSCoordinateString() [1/2]	2486
9.978.3.6 convertFromGPSCoordinateString() [2/2]	2486
9.978.3.7 convertToGPSCoordinateString()	2486
9.978.3.8 convertToRational()	2486
9.978.3.9 convertToRationalSmallDenominator()	2487
9.978.3.10 convertToUserPresentableNumbers()	2487
9.978.3.11 createExifUserStringFromValue()	2487
9.978.3.12 detectLanguageAlt()	2487
9.978.3.13 exportChanges()	2487
9.978.3.14 getComments()	2488
9.978.3.15 getCommentsDecoded()	2488
9.978.3.16 getDigitizationDateTime()	2488
9.978.3.17 getExifComment()	2488
9.978.3.18 getExifEncoded()	2488
9.978.3.19 getExifTagComment()	2488
9.978.3.20 getExifTagData()	2489
9.978.3.21 getExifTagLong() [1/2]	2489
9.978.3.22 getExifTagLong() [2/2]	2489
9.978.3.23 getExifTagRational()	2489
9.978.3.24 getExifTagsDataList()	2489
9.978.3.25 getExifTagString()	2490
9.978.3.26 getExifTagVariant()	2490
9.978.3.27 getExifThumbnail()	2490
9.978.3.28 getGPSInfo()	2490
9.978.3.29 getGPSLatitudeNumber()	2490

---

9.978.3.30 getGPSLatitudeString()	2490
9.978.3.31 getIptc()	2491
9.978.3.32 getIptcKeywords()	2491
9.978.3.33 getIptcSubCategories()	2491
9.978.3.34 getIptcSubjects()	2491
9.978.3.35 getIptcTagData()	2491
9.978.3.36 getIptcTagsDataList()	2491
9.978.3.37 getIptcTagsStringList()	2492
9.978.3.38 getIptcTagString()	2492
9.978.3.39 getItemColorWorkSpace()	2492
9.978.3.40 getItemDateTime()	2492
9.978.3.41 getItemDimensions()	2492
9.978.3.42 getItemOrientation()	2492
9.978.3.43 getItemPreview()	2492
9.978.3.44 getMimeType()	2493
9.978.3.45 getPixelSize()	2493
9.978.3.46 getXmp()	2493
9.978.3.47 getXmpKeywords()	2493
9.978.3.48 getXmpSubCategories()	2493
9.978.3.49 getXmpSubjects()	2493
9.978.3.50 getXmpTagsDataList()	2494
9.978.3.51 getXmpTagString()	2494
9.978.3.52 getXmpTagStringBag()	2494
9.978.3.53 getXmpTagStringLangAlt()	2494
9.978.3.54 getXmpTagStringListLangAlt()	2494
9.978.3.55 getXmpTagStringSeq()	2495
9.978.3.56 getXmpTagVariant()	2495
9.978.3.57 initializeExiv2()	2495
9.978.3.58 load()	2495
9.978.3.59 loadFromData()	2495
9.978.3.60 loadFromDataAndMerge()	2496
9.978.3.61 loadFromSidecarAndMerge()	2496
9.978.3.62 metadataWritingMode()	2496
9.978.3.63 registerXmpNameSpace()	2496
9.978.3.64 removeExifTag()	2496
9.978.3.65 removeFromXmpTagStringBag()	2497
9.978.3.66 removeGPSInfo()	2497
9.978.3.67 removeIptcTag()	2497
9.978.3.68 removeXmpKeywords()	2497
9.978.3.69 removeXmpSubCategories()	2497
9.978.3.70 removeXmpSubjects()	2497
9.978.3.71 removeXmpTag()	2498

9.978.3.72 rotateExifQImage()	2498
9.978.3.73 save()	2498
9.978.3.74 setComments()	2498
9.978.3.75 setExif()	2498
9.978.3.76 setExifComment()	2498
9.978.3.77 setExifTagData()	2499
9.978.3.78 setExifTagLong()	2499
9.978.3.79 setExifTagRational()	2499
9.978.3.80 setExifTagString()	2499
9.978.3.81 setExifTagURational()	2499
9.978.3.82 setExifTagUShort()	2499
9.978.3.83 setExifTagVariant()	2500
9.978.3.84 setExifThumbnail()	2500
9.978.3.85 setGPSInfo() [1/3]	2500
9.978.3.86 setGPSInfo() [2/3]	2500
9.978.3.87 setGPSInfo() [3/3]	2500
9.978.3.88 setImageDateTime()	2501
9.978.3.89 setIptc()	2501
9.978.3.90 setIptcKeywords()	2501
9.978.3.91 setIptcSubCategories()	2501
9.978.3.92 setIptcSubjects()	2501
9.978.3.93 setIptcTagData()	2501
9.978.3.94 setIptcTagsStringList()	2502
9.978.3.95 setIptcTagString()	2502
9.978.3.96 setItemColorWorkSpace()	2502
9.978.3.97 setItemDimensions()	2502
9.978.3.98 setItemOrientation()	2502
9.978.3.99 setItemPreview()	2502
9.978.3.100 setItemProgramId()	2503
9.978.3.101 setMetadataWritingMode()	2503
9.978.3.102 setTiffThumbnail()	2503
9.978.3.103 setUpdateFileTimeStamp()	2503
9.978.3.104 setWriteRawFiles()	2503
9.978.3.105 setXmp()	2503
9.978.3.106 setXmpKeywords()	2504
9.978.3.107 setXmpSubCategories()	2504
9.978.3.108 setXmpSubjects()	2504
9.978.3.109 setXmpTagString() [1/2]	2504
9.978.3.110 setXmpTagString() [2/2]	2504
9.978.3.111 setXmpTagStringBag()	2504
9.978.3.112 setXmpTagStringLangAlt()	2505
9.978.3.113 setXmpTagStringListLangAlt()	2505



9.978.3.114 setXmpTagStringSeq()	2505
9.978.3.115 sidecarFilePathForFile()	2505
9.978.3.116 supportBmff()	2505
9.979 Digikam::MetaEngineData Class Reference	2505
9.980 Digikam::MetaEngineMergeHelper< Data, Key, KeyString, KeyStringList > Class Template Reference	2506
9.980.1 Member Function Documentation	2506
9.980.1.1 exclusiveMerge()	2506
9.980.1.2 mergeFields()	2506
9.981 Digikam::MetaEnginePreviews Class Reference	2507
9.981.1 Member Function Documentation	2507
9.981.1.1 dataSize()	2507
9.981.1.2 image()	2507
9.982 Digikam::MetaEngineRotation Class Reference	2508
9.982.1 Member Enumeration Documentation	2509
9.982.1.1 TransformationAction	2509
9.982.2 Member Function Documentation	2510
9.982.2.1 exifOrientation()	2510
9.982.2.2 transformations()	2510
9.983 Digikam::MetaEngineSettings Class Reference	2510
9.983.1 Member Function Documentation	2511
9.983.1.1 instance()	2511
9.984 Digikam::MetaEngineSettingsContainer Class Reference	2511
9.984.1 Detailed Description	2512
9.984.2 Member Enumeration Documentation	2512
9.984.2.1 RotationBehaviorFlag	2512
9.985 Digikam::MigrateFromDigikam4Page Class Reference	2513
9.986 Digikam::MimeFilter Class Reference	2514
9.986.1 Member Enumeration Documentation	2515
9.986.1.1 TypeMimeFilter	2515
9.987 Digikam::MixerContainer Class Reference	2515
9.988 Digikam::MixerFilter Class Reference	2516
9.988.1 Member Function Documentation	2520
9.988.1.1 filterAction()	2520
9.988.1.2 filterIdentifier()	2520
9.988.1.3 readParameters()	2520
9.989 Digikam::MixerSettings Class Reference	2520
9.990 Digikam::MLClassifierFoundation Class Reference	2522
9.991 Digikam::MLClassifierFoundation::VotingGroups Class Reference	2523
9.992 Digikam::MLClassifierFoundation::VotingGroups::VoteTally Struct Reference	2523
9.993 Digikam::MLPipelineFoundation Class Reference	2524
9.993.1 Member Enumeration Documentation	2526
9.993.1.1 MLPipelineStage	2526

---

9.993.2 Member Function Documentation	2526
9.993.2.1 cancel()	2526
9.994 Digikam::MLPipelineFoundation::_MLPipelinePerformanceProfile Struct Reference	2527
9.995 Digikam::MLPipelinePackageFoundation Class Reference	2528
9.996 Digikam::MLPipelinePackageNotify Class Reference	2529
9.997 Digikam::ModelCompleter Class Reference	2530
9.997.1 Member Function Documentation	2530
9.997.1.1 setItemModel()	2530
9.998 Digikam::ModelIndexBasedComboBox Class Reference	2532
9.998.1 Constructor & Destructor Documentation	2533
9.998.1.1 ModelIndexedComboBox()	2533
9.999 Digikam::ModelMenu Class Reference	2533
9.999.1 Member Function Documentation	2535
9.999.1.1 prePopulated()	2535
9.1000 Digikam::Modifier Class Reference	2535
9.1000.1 Member Function Documentation	2537
9.1000.1.1 parseOperation()	2537
9.1001 Digikam::MonthWidget Class Reference	2538
9.1002 Digikam::MysqlAdminBinary Class Reference	2539
9.1003 Digikam::MysqlInitBinary Class Reference	2542
9.1004 Digikam::MysqlServerBinary Class Reference	2545
9.1005 Digikam::MysqlUpgradeBinary Class Reference	2548
9.1006 Digikam::NamespaceEditDlg Class Reference	2550
9.1007 Digikam::NamespaceEntry Class Reference	2551
9.1008 Digikam::NamespaceListView Class Reference	2552
9.1009 Digikam::NetworkManager Class Reference	2553
9.1009.1 Member Function Documentation	2554
9.1009.1.1 instance()	2554
9.1010 Digikam::NewItemFinder Class Reference	2555
9.1010.1 Member Enumeration Documentation	2557
9.1010.1.1 FinderMode	2557
9.1011 Digikam::NoDuplicatesImportFilterModel Class Reference	2559
9.1012 Digikam::NoDuplicatesItemFilterModel Class Reference	2562
9.1013 Digikam::NoiseDetector Class Reference	2564
9.1013.1 Member Function Documentation	2565
9.1013.1.1 detect()	2565
9.1014 Digikam::NonDeterministicRandomData Class Reference	2565
9.1014.1 Constructor & Destructor Documentation	2566
9.1014.1.1 NonDeterministicRandomData()	2566
9.1015 Digikam::NormalizeFilter Class Reference	2567
9.1015.1 Member Function Documentation	2571
9.1015.1.1 filterAction()	2571

---

9.1015.1.2 filterIdentifier() . . . . .	2571
9.1015.1.3 readParameters() . . . . .	2571
9.1016 Digikam::NormalSearchTreeView Class Reference . . . . .	2571
9.1016.1 Detailed Description . . . . .	2577
9.1016.2 Constructor & Destructor Documentation . . . . .	2577
9.1016.2.1 NormalSearchTreeView() . . . . .	2577
9.1016.3 Member Function Documentation . . . . .	2578
9.1016.3.1 addCustomContextMenuActions() . . . . .	2578
9.1016.3.2 copySearch . . . . .	2578
9.1016.3.3 editSearch . . . . .	2578
9.1016.3.4 handleCustomContextMenuAction() . . . . .	2578
9.1017 Digikam::NRContainer Class Reference . . . . .	2578
9.1017.1 Member Data Documentation . . . . .	2579
9.1017.1.1 thresholds . . . . .	2579
9.1018 Digikam::NREstimate Class Reference . . . . .	2580
9.1018.1 Member Function Documentation . . . . .	2584
9.1018.1.1 setLogFilesPath() . . . . .	2584
9.1018.1.2 startAnalyse() . . . . .	2584
9.1019 Digikam::NRFilter Class Reference . . . . .	2585
9.1019.1 Member Function Documentation . . . . .	2589
9.1019.1.1 filterAction() . . . . .	2589
9.1019.1.2 filterIdentifier() . . . . .	2589
9.1019.1.3 readParameters() . . . . .	2589
9.1020 Digikam::NRSettings Class Reference . . . . .	2589
9.1021 Digikam::OilPaintFilter Class Reference . . . . .	2591
9.1021.1 Member Function Documentation . . . . .	2595
9.1021.1.1 filterAction() . . . . .	2595
9.1021.1.2 filterIdentifier() . . . . .	2595
9.1021.1.3 readParameters() . . . . .	2595
9.1022 Digikam::OnlineVersionChecker Class Reference . . . . .	2595
9.1022.1 Member Function Documentation . . . . .	2596
9.1022.1.1 bundleProperties() . . . . .	2596
9.1023 Digikam::OnlineVersionDlg Class Reference . . . . .	2597
9.1024 Digikam::OnlineVersionDwnl Class Reference . . . . .	2598
9.1025 Digikam::OpenCVDNNFaceDetector Class Reference . . . . .	2598
9.1025.1 Member Function Documentation . . . . .	2599
9.1025.1.1 detectFaces() . . . . .	2599
9.1025.1.2 recommendedImageSizeForDetection() . . . . .	2600
9.1026 Digikam::OpenCVDNNFaceRecognizer Class Reference . . . . .	2600
9.1026.1 Member Enumeration Documentation . . . . .	2600
9.1026.1.1 Classifier . . . . .	2600
9.1026.2 Member Function Documentation . . . . .	2601

---

9.1026.2.1 recognize() [1/2]	2601
9.1026.2.2 recognize() [2/2]	2601
9.1027 Digikam::OpenfacePreprocessor Class Reference	2601
9.1028 Digikam::OpenFilePage Class Reference	2602
9.1029 Digikam::Option Class Reference	2603
9.1029.1 Member Function Documentation	2604
9.1029.1.1 parseOperation()	2604
9.1030 Digikam::OverlayWidget Class Reference	2605
9.1030.1 Detailed Description	2607
9.1031 Digikam::PackageLoadingDescriptionList Class Reference	2607
9.1032 Digikam::PALbum Class Reference	2608
9.1032.1 Member Function Documentation	2610
9.1032.1.1 databaseUrl()	2610
9.1033 Digikam::PanIconFrame Class Reference	2611
9.1033.1 Member Function Documentation	2612
9.1033.1.1 close	2612
9.1033.1.2 resizeEvent()	2612
9.1033.1.3 setMainWidget()	2612
9.1034 Digikam::PanIconWidget Class Reference	2613
9.1034.1 Member Function Documentation	2614
9.1034.1.1 signalSelectionMoved	2614
9.1035 Digikam::ParallelAdapter< A > Class Template Reference	2615
9.1035.1 Constructor & Destructor Documentation	2617
9.1035.1.1 ParallelAdapter()	2617
9.1035.2 Member Function Documentation	2617
9.1035.2.1 asQObject()	2617
9.1035.2.2 mocMetaObject()	2617
9.1035.2.3 staticMetacallPointer()	2617
9.1035.2.4 WorkerObjectQtMetacall()	2618
9.1036 Digikam::ParallelPipes Class Reference	2618
9.1037 Digikam::ParallelWorkers Class Reference	2620
9.1037.1 Constructor & Destructor Documentation	2621
9.1037.1.1 ParallelWorkers()	2621
9.1037.2 Member Function Documentation	2622
9.1037.2.1 asQObject()	2622
9.1037.2.2 mocMetaObject()	2622
9.1037.2.3 WorkerObjectQtMetacall()	2622
9.1038 Digikam::Parser Class Reference	2623
9.1038.1 Member Function Documentation	2624
9.1038.1.1 parseStringsValid()	2624
9.1039 Digikam::ParseResults Class Reference	2624
9.1040 Digikam::ParseSettings Class Reference	2625

---

9.1041 Digikam::PeopleSideBarWidget Class Reference	2626
9.1041.1 Member Function Documentation	2628
9.1041.1.1 applySettings()	2628
9.1041.1.2 changeAlbumFromHistory()	2628
9.1041.1.3 doLoadState()	2628
9.1041.1.4 doSaveState()	2628
9.1041.1.5 getCaption()	2629
9.1041.1.6 getIcon()	2629
9.1041.1.7 setActive()	2629
9.1042 Digikam::PersistentWidgetDelegateOverlay Class Reference	2630
9.1042.1 Constructor & Destructor Documentation	2633
9.1042.1.1 PersistentWidgetDelegateOverlay()	2633
9.1042.2 Member Function Documentation	2633
9.1042.2.1 hide()	2633
9.1042.2.2 setActive()	2633
9.1042.2.3 setFocusOnWidget()	2633
9.1042.2.4 setPersistent	2633
9.1042.2.5 showOnIndex()	2634
9.1042.2.6 slotEntered()	2634
9.1042.2.7 slotLayoutChanged()	2634
9.1042.2.8 slotReset()	2634
9.1042.2.9 slotRowsRemoved()	2634
9.1042.2.10 slotViewportEntered()	2634
9.1042.2.11 viewportLeaveEvent()	2635
9.1043 Digikam::PhotoInfoContainer Class Reference	2635
9.1044 Digikam::PickLabelFilter Class Reference	2636
9.1045 Digikam::PickLabelMenuAction Class Reference	2638
9.1046 Digikam::PickLabelSelector Class Reference	2639
9.1047 Digikam::PickLabelWidget Class Reference	2640
9.1047.1 Member Function Documentation	2641
9.1047.1.1 setButtonsExclusive()	2641
9.1047.1.2 setPickLabels()	2642
9.1048 Digikam::PlaceholderWidget Class Reference	2642
9.1049 Digikam::PointTransformAffine Class Reference	2642
9.1050 Digikam::PositionKeys Class Reference	2643
9.1050.1 Constructor & Destructor Documentation	2644
9.1050.1.1 PositionKeys()	2644
9.1050.2 Member Function Documentation	2644
9.1050.2.1 getDbValue()	2644
9.1051 Digikam::PreviewList Class Reference	2645
9.1052 Digikam::PreviewListItem Class Reference	2646
9.1053 Digikam::PreviewLoadingTask Class Reference	2647

---

9.1053.1 Member Function Documentation	2649
9.1053.1.1 execute()	2649
9.1054 Digikam::PreviewLoadThread Class Reference	2650
9.1054.1 Constructor & Destructor Documentation	2655
9.1054.1.1 PreviewLoadThread()	2655
9.1054.2 Member Function Documentation	2655
9.1054.2.1 load() [1/2]	2655
9.1054.2.2 load() [2/2]	2655
9.1054.2.3 loadFast()	2655
9.1054.2.4 loadFastButLarge()	2655
9.1054.2.5 loadFastSynchronously()	2656
9.1054.2.6 loadHighQuality()	2656
9.1055 Digikam::PreviewPage Class Reference	2657
9.1056 Digikam::PreviewSettings Class Reference	2658
9.1056.1 Member Enumeration Documentation	2658
9.1056.1.1 Quality	2658
9.1057 Digikam::PreviewThreadWrapper Class Reference	2659
9.1058 Digikam::PreviewToolBar Class Reference	2660
9.1058.1 Member Enumeration Documentation	2661
9.1058.1.1 PreviewMode	2661
9.1059 Digikam::ProcessLauncher Class Reference	2662
9.1060 Digikam::ProgressItem Class Reference	2663
9.1060.1 Member Function Documentation	2665
9.1060.1.1 advance()	2665
9.1060.1.2 canBeCanceled()	2665
9.1060.1.3 hasThumbnail()	2665
9.1060.1.4 id()	2665
9.1060.1.5 label()	2665
9.1060.1.6 parent()	2666
9.1060.1.7 progress()	2666
9.1060.1.8 progressItemAdded	2666
9.1060.1.9 progressItemCanceled	2666
9.1060.1.10 progressItemCompleted	2666
9.1060.1.11 progressItemLabel	2667
9.1060.1.12 progressItemProgress	2667
9.1060.1.13 progressItemStatus	2667
9.1060.1.14 progressItemThumbnail	2667
9.1060.1.15 progressItemUsesBusyIndicator	2668
9.1060.1.16 setComplete()	2668
9.1060.1.17 setLabel()	2668
9.1060.1.18 setProgress()	2668
9.1060.1.19 setShowAtStart()	2669

9.1060.1.20 setStatus()	2669
9.1060.1.21 setThumbnail()	2669
9.1060.1.22 setUsesBusyIndicator()	2669
9.1060.1.23 showAtStart()	2669
9.1060.1.24 status()	2670
9.1060.1.25 usesBusyIndicator()	2670
9.1061 Digikam::ProgressManager Class Reference	2670
9.1061.1 Detailed Description	2672
9.1061.2 Member Function Documentation	2672
9.1061.2.1 addProgressItem()	2672
9.1061.2.2 createProgressItem() [1/4]	2672
9.1061.2.3 createProgressItem() [2/4]	2673
9.1061.2.4 createProgressItem() [3/4]	2673
9.1061.2.5 createProgressItem() [4/4]	2674
9.1061.2.6 findItemById()	2674
9.1061.2.7 getUniqueId()	2675
9.1061.2.8 instance()	2675
9.1061.2.9 isEmpty()	2675
9.1061.2.10 progressItemAdded	2675
9.1061.2.11 progressItemCanceled	2675
9.1061.2.12 progressItemCompleted	2676
9.1061.2.13 progressItemLabel	2676
9.1061.2.14 progressItemProgress	2676
9.1061.2.15 progressItemStatus	2676
9.1061.2.16 progressItemThumbnail	2676
9.1061.2.17 progressItemUsesBusyIndicator	2677
9.1061.2.18 showProgressView	2677
9.1061.2.19 singleItem()	2677
9.1061.2.20 slotAbortAll	2677
9.1061.2.21 slotStandardCancelHandler	2677
9.1062 Digikam::ProgressView Class Reference	2678
9.1063 Digikam::ProxyClickLineEdit Class Reference	2680
9.1063.1 Constructor & Destructor Documentation	2682
9.1063.1.1 ProxyClickLineEdit()	2682
9.1064 Digikam::ProxyLineEdit Class Reference	2683
9.1064.1 Constructor & Destructor Documentation	2684
9.1064.1.1 ProxyLineEdit()	2684
9.1065 Digikam::QListImageListProvider Class Reference	2685
9.1065.1 Member Function Documentation	2686
9.1065.1.1 atEnd()	2686
9.1065.1.2 image()	2686
9.1065.1.3 images()	2686

---

9.1065.1.4 proceed()	2686
9.1065.1.5 setImages()	2686
9.1065.1.6 setUnpairedImages()	2686
9.1065.1.7 size()	2687
9.1066 Digikam::QMapForAdaptors< Key, Value > Class Template Reference	2687
9.1067 Digikam::QueueListView Class Reference	2688
9.1067.1 Member Enumeration Documentation	2689
9.1067.1.1 ItemListType	2689
9.1068 Digikam::QueueListViewItem Class Reference	2690
9.1069 Digikam::QueueMgrWindow Class Reference	2691
9.1069.1 Member Function Documentation	2694
9.1069.1.1 infolface()	2694
9.1070 Digikam::QueuePool Class Reference	2695
9.1071 Digikam::QueuePoolBar Class Reference	2697
9.1072 Digikam::QueueSettings Class Reference	2697
9.1073 Digikam::QueueSettingsView Class Reference	2698
9.1074 Digikam::QueueToolTip Class Reference	2699
9.1075 Digikam::RainDropFilter Class Reference	2701
9.1075.1 Member Function Documentation	2705
9.1075.1.1 filterAction()	2705
9.1075.1.2 filterIdentifier()	2705
9.1075.1.3 readParameters()	2705
9.1076 Digikam::RandomNumberGenerator Class Reference	2705
9.1076.1 Detailed Description	2706
9.1076.2 Constructor & Destructor Documentation	2706
9.1076.2.1 RandomNumberGenerator()	2706
9.1076.3 Member Function Documentation	2706
9.1076.3.1 currentSeed()	2706
9.1076.3.2 number()	2706
9.1076.3.3 reseed()	2706
9.1076.3.4 seed()	2707
9.1076.3.5 seedByTime()	2707
9.1076.3.6 seedNonDeterministic()	2707
9.1077 Digikam::RangeDialog Class Reference	2707
9.1078 Digikam::RangeModifier Class Reference	2709
9.1078.1 Member Function Documentation	2711
9.1078.1.1 parseOperation()	2711
9.1079 Digikam::RatingBox Class Reference	2712
9.1080 Digikam::RatingComboBox Class Reference	2714
9.1080.1 Member Enumeration Documentation	2715
9.1080.1.1 RatingValue	2715
9.1081 Digikam::RatingComboBoxDelegate Class Reference	2716



9.1082 Digikam::RatingComboBoxModel Class Reference	2717
9.1083 Digikam::RatingComboBoxWidget Class Reference	2718
9.1084 Digikam::RatingFilter Class Reference	2721
9.1085 Digikam::RatingFilterWidget Class Reference	2723
9.1086 Digikam::RatingMenuAction Class Reference	2725
9.1087 Digikam::RatingStarDrawer Class Reference	2726
9.1088 Digikam::RatingWidget Class Reference	2727
9.1089 Digikam::RawCameraDlg Class Reference	2729
9.1090 Digikam::RawPage Class Reference	2730
9.1091 Digikam::RawProcessingFilter Class Reference	2731
9.1091.1 Detailed Description	2736
9.1091.2 Constructor & Destructor Documentation	2736
9.1091.2.1 RawProcessingFilter() [1/2]	2736
9.1091.2.2 RawProcessingFilter() [2/2]	2736
9.1091.3 Member Function Documentation	2737
9.1091.3.1 filterAction()	2737
9.1091.3.2 filterIdentifier()	2737
9.1091.3.3 filterImage()	2737
9.1091.3.4 readParameters()	2737
9.1091.3.5 setObserver()	2737
9.1091.3.6 setSettings()	2737
9.1092 Digikam::RecognitionBenchmarker Class Reference	2738
9.1092.1 Member Function Documentation	2740
9.1092.1.1 result()	2740
9.1093 Digikam::RecognitionBenchmarker::Statistics Class Reference	2740
9.1094 Digikam::RecognitionPreprocessor Class Reference	2741
9.1094.1 Member Function Documentation	2741
9.1094.1.1 preprocess()	2741
9.1095 Digikam::RecognitionTrainingProvider Class Reference	2741
9.1095.1 Member Function Documentation	2742
9.1095.1.1 images()	2742
9.1095.1.2 newImages()	2743
9.1096 Digikam::RecognitionTrainingUpdateQueue Class Reference	2743
9.1097 Digikam::RecognitionWorker Class Reference	2744
9.1097.1 Member Function Documentation	2746
9.1097.1.1 aboutToDeactivate()	2746
9.1098 Digikam::RedEye::RegressionTree Struct Reference	2747
9.1098.1 Member Function Documentation	2747
9.1098.1.1 operator>()	2747
9.1099 Digikam::RedEye::ShapePredictor Class Reference	2747
9.1100 Digikam::RedEye::SplitFeature Struct Reference	2748
9.1101 Digikam::RedEyeCorrectionContainer Class Reference	2748

---

9.1102 Digikam::RedEyeCorrectionFilter Class Reference	2749
9.1102.1 Member Function Documentation	2753
9.1102.1.1 filterAction()	2753
9.1102.1.2 filterIdentifier()	2753
9.1103 Digikam::RedEyeCorrectionSettings Class Reference	2753
9.1104 Digikam::RefocusFilter Class Reference	2755
9.1104.1 Member Function Documentation	2759
9.1104.1.1 filterAction()	2759
9.1104.1.2 filterIdentifier()	2759
9.1104.1.3 readParameters()	2759
9.1105 Digikam::RefocusMatrix Class Reference	2759
9.1106 Digikam::RegionFrameItem Class Reference	2760
9.1106.1 Member Function Documentation	2763
9.1106.1.1 setHudWidget()	2763
9.1106.1.2 setViewportRect	2763
9.1107 Digikam::RemoveBookmarksCommand Class Reference	2764
9.1108 Digikam::RemoveDoublesModifier Class Reference	2765
9.1108.1 Member Function Documentation	2767
9.1108.1.1 parseOperation()	2767
9.1109 Digikam::RemoveFilterAction Class Reference	2768
9.1110 Digikam::RenameCustomizer Class Reference	2769
9.1111 Digikam::RenameFileJob Class Reference	2770
9.1112 Digikam::ReplaceDialog Class Reference	2772
9.1113 Digikam::ReplaceModifier Class Reference	2773
9.1113.1 Member Function Documentation	2775
9.1113.1.1 parseOperation()	2775
9.1114 Digikam::RestoreDTrashItemsJob Class Reference	2776
9.1115 Digikam::RGBBackend Class Reference	2778
9.1115.1 Member Function Documentation	2778
9.1115.1.1 backendName()	2778
9.1115.1.2 callRGBBackend()	2779
9.1115.1.3 getErrorMessage()	2779
9.1116 Digikam::RGInfo Class Reference	2779
9.1117 Digikam::RGTagModel Class Reference	2780
9.1117.1 Detailed Description	2782
9.1117.2 Constructor & Destructor Documentation	2782
9.1117.2.1 RGTagModel()	2782
9.1117.3 Member Function Documentation	2783
9.1117.3.1 addDataInTree()	2783
9.1117.3.2 addExternalTags()	2783
9.1117.3.3 addNewData()	2783
9.1117.3.4 addNewTag()	2784

9.1117.3.5 addSpacerTag()	2784
9.1117.3.6 branchFromIndex()	2784
9.1117.3.7 climbTreeAndGetSpacers()	2784
9.1117.3.8 deleteAllSpacersOrNewTags()	2785
9.1117.3.9 deleteTag()	2785
9.1117.3.10 findAndDeleteSpacersOrNewTags()	2785
9.1117.3.11 fromSourceIndex()	2785
9.1117.3.12 getSpacerAddress()	2786
9.1117.3.13 getSpacers()	2786
9.1117.3.14 getTagType()	2786
9.1117.3.15 readdNewTags()	2786
9.1117.3.16 readdTag()	2787
9.1117.3.17 toSourceIndex()	2787
9.1118 Digikam::RGWidget Class Reference	2787
9.1118.1 Constructor & Destructor Documentation	2789
9.1118.1.1 RGWidget()	2789
9.1118.2 Member Function Documentation	2789
9.1118.2.1 readSettingsFromGroup()	2789
9.1118.2.2 saveSettingsToGroup()	2789
9.1118.2.3 setUIEnabled()	2789
9.1118.2.4 signalProgressChanged	2790
9.1118.2.5 signalSetUIEnabled	2790
9.1118.2.6 signalUndoCommand	2790
9.1119 Digikam::RubberItem Class Reference	2791
9.1120 Digikam::Rule Class Reference	2794
9.1120.1 Member Function Documentation	2795
9.1120.1.1 addToken()	2795
9.1120.1.2 escapeToken()	2795
9.1120.1.3 isValid()	2796
9.1120.1.4 parseOperation()	2796
9.1120.1.5 regExp()	2796
9.1120.1.6 registerButton()	2797
9.1120.1.7 registerMenu()	2797
9.1120.1.8 reset()	2798
9.1120.1.9 setUseTokenMenu()	2798
9.1120.1.10 tokens()	2798
9.1121 Digikam::RuleDialog Class Reference	2798
9.1122 Digikam::SafeTemporaryFile Class Reference	2799
9.1123 Digikam::SAlbum Class Reference	2799
9.1123.1 Member Function Documentation	2802
9.1123.1.1 databaseUrl()	2802
9.1123.1.2 getTemporaryHaarTitle()	2802

---

9.1123.1.3 getTemporaryTitle()	2803
9.1123.1.4 isTemporarySearch()	2803
9.1124 Digikam::SaveProperties Class Reference	2803
9.1125 Digikam::SavingContext Class Reference	2804
9.1126 Digikam::SavingTask Class Reference	2805
9.1126.1 Member Function Documentation	2806
9.1126.1.1 continueQuery()	2806
9.1126.1.2 execute()	2806
9.1126.1.3 progressInfo()	2806
9.1126.1.4 type()	2806
9.1127 Digikam::ScanController Class Reference	2807
9.1127.1 Member Function Documentation	2810
9.1127.1.1 abortInitialization()	2810
9.1127.1.2 beginFileMetadataWrite()	2810
9.1127.1.3 cancelAllAndSuspendCollectionScan()	2810
9.1127.1.4 cancelCompleteScan()	2810
9.1127.1.5 completeCollectionScan()	2810
9.1127.1.6 databaseInitialization()	2810
9.1127.1.7 finishFileMetadataWrite()	2811
9.1127.1.8 hintAtModificationOfItems()	2811
9.1127.1.9 hintAtMoveOrCopyOfAlbum()	2811
9.1127.1.10 hintAtMoveOrCopyOfItems()	2811
9.1127.1.11 restartCollectionScan()	2811
9.1127.1.12 resumeCollectionScan()	2811
9.1127.1.13 scheduleCollectionScan()	2812
9.1127.1.14 scheduleCollectionScanExternal()	2812
9.1127.1.15 scheduleCollectionScanRelaxed()	2812
9.1127.1.16 shutDown()	2812
9.1127.1.17 suspendCollectionScan()	2812
9.1127.1.18 updateUniqueHash()	2812
9.1128 Digikam::ScanController::FileMetadataWrite Class Reference	2812
9.1128.1 Detailed Description	2813
9.1129 Digikam::ScanStateFilter Class Reference	2814
9.1129.1 Member Function Documentation	2816
9.1129.1.1 run()	2816
9.1130 Digikam::ScriptingSettings Class Reference	2817
9.1131 Digikam::SearchChangeset Class Reference	2817
9.1132 Digikam::SearchesDBJobInfo Class Reference	2818
9.1133 Digikam::SearchesDBJobsThread Class Reference	2820
9.1133.1 Member Function Documentation	2822
9.1133.1.1 searchesListing()	2822
9.1134 Digikam::SearchesJob Class Reference	2823

---

9.1135 Digikam::SearchField Class Reference . . . . .	2825
9.1135.1 Member Function Documentation . . . . .	2826
9.1135.1.1 createField() . . . . .	2826
9.1135.1.2 isVisible() . . . . .	2826
9.1135.1.3 setVisible() . . . . .	2826
9.1135.1.4 write() . . . . .	2826
9.1136 Digikam::SearchFieldAlbum Class Reference . . . . .	2827
9.1136.1 Member Function Documentation . . . . .	2829
9.1136.1.1 read() . . . . .	2829
9.1136.1.2 reset() . . . . .	2829
9.1136.1.3 setupValueWidgets() . . . . .	2829
9.1136.1.4 setValueWidgetsVisible() . . . . .	2829
9.1136.1.5 valueWidgetRects() . . . . .	2830
9.1136.1.6 write() . . . . .	2830
9.1137 Digikam::SearchFieldCheckBox Class Reference . . . . .	2831
9.1137.1 Member Function Documentation . . . . .	2833
9.1137.1.1 read() . . . . .	2833
9.1137.1.2 reset() . . . . .	2833
9.1137.1.3 setupValueWidgets() . . . . .	2833
9.1137.1.4 setValueWidgetsVisible() . . . . .	2833
9.1137.1.5 valueWidgetRects() . . . . .	2834
9.1137.1.6 write() . . . . .	2834
9.1138 Digikam::SearchFieldChoice Class Reference . . . . .	2835
9.1138.1 Member Function Documentation . . . . .	2837
9.1138.1.1 read() . . . . .	2837
9.1138.1.2 reset() . . . . .	2837
9.1138.1.3 setupValueWidgets() . . . . .	2837
9.1138.1.4 setValueWidgetsVisible() . . . . .	2838
9.1138.1.5 valueWidgetRects() . . . . .	2838
9.1138.1.6 write() . . . . .	2838
9.1139 Digikam::SearchFieldColorDepth Class Reference . . . . .	2839
9.1139.1 Member Function Documentation . . . . .	2841
9.1139.1.1 read() . . . . .	2841
9.1139.1.2 setupValueWidgets() . . . . .	2841
9.1140 Digikam::SearchFieldComboBox Class Reference . . . . .	2842
9.1140.1 Member Function Documentation . . . . .	2844
9.1140.1.1 reset() . . . . .	2844
9.1140.1.2 setupValueWidgets() . . . . .	2844
9.1140.1.3 setValueWidgetsVisible() . . . . .	2844
9.1140.1.4 valueWidgetRects() . . . . .	2844
9.1140.1.5 write() . . . . .	2845
9.1141 Digikam::SearchFieldGroup Class Reference . . . . .	2845

---

9.1142 Digikam::SearchFieldGroupLabel Class Reference . . . . .	2846
9.1143 Digikam::SearchFieldKeyword Class Reference . . . . .	2848
9.1143.1 Member Function Documentation . . . . .	2850
9.1143.1.1 read() . . . . .	2850
9.1143.1.2 write() . . . . .	2850
9.1144 Digikam::SearchFieldLabels Class Reference . . . . .	2851
9.1144.1 Member Function Documentation . . . . .	2853
9.1144.1.1 read() . . . . .	2853
9.1144.1.2 reset() . . . . .	2853
9.1144.1.3 setupValueWidgets() . . . . .	2853
9.1144.1.4 setValueWidgetsVisible() . . . . .	2853
9.1144.1.5 valueWidgetRects() . . . . .	2854
9.1144.1.6 write() . . . . .	2854
9.1145 Digikam::SearchFieldMonthDay Class Reference . . . . .	2855
9.1145.1 Member Function Documentation . . . . .	2857
9.1145.1.1 read() . . . . .	2857
9.1145.1.2 reset() . . . . .	2857
9.1145.1.3 setupValueWidgets() . . . . .	2857
9.1145.1.4 setValueWidgetsVisible() . . . . .	2857
9.1145.1.5 valueWidgetRects() . . . . .	2858
9.1145.1.6 write() . . . . .	2858
9.1146 Digikam::SearchFieldPageOrientation Class Reference . . . . .	2859
9.1146.1 Member Function Documentation . . . . .	2861
9.1146.1.1 read() . . . . .	2861
9.1146.1.2 setupValueWidgets() . . . . .	2861
9.1147 Digikam::SearchFieldRangeDate Class Reference . . . . .	2862
9.1147.1 Member Function Documentation . . . . .	2864
9.1147.1.1 read() . . . . .	2864
9.1147.1.2 reset() . . . . .	2864
9.1147.1.3 setupValueWidgets() . . . . .	2864
9.1147.1.4 setValueWidgetsVisible() . . . . .	2865
9.1147.1.5 valueWidgetRects() . . . . .	2865
9.1147.1.6 write() . . . . .	2865
9.1148 Digikam::SearchFieldRangeDouble Class Reference . . . . .	2866
9.1148.1 Member Function Documentation . . . . .	2868
9.1148.1.1 read() . . . . .	2868
9.1148.1.2 reset() . . . . .	2868
9.1148.1.3 setupValueWidgets() . . . . .	2868
9.1148.1.4 setValueWidgetsVisible() . . . . .	2869
9.1148.1.5 valueWidgetRects() . . . . .	2869
9.1148.1.6 write() . . . . .	2869
9.1149 Digikam::SearchFieldRangeInt Class Reference . . . . .	2870

---

9.1149.1 Member Function Documentation . . . . .	2872
9.1149.1.1 read() . . . . .	2872
9.1149.1.2 reset() . . . . .	2872
9.1149.1.3 setupValueWidgets() . . . . .	2872
9.1149.1.4 setValueWidgetsVisible() . . . . .	2873
9.1149.1.5 valueWidgetRects() . . . . .	2873
9.1149.1.6 write() . . . . .	2873
9.1150 Digikam::SearchFieldRangeTime Class Reference . . . . .	2874
9.1150.1 Member Function Documentation . . . . .	2876
9.1150.1.1 read() . . . . .	2876
9.1150.1.2 reset() . . . . .	2876
9.1150.1.3 setupValueWidgets() . . . . .	2876
9.1150.1.4 setValueWidgetsVisible() . . . . .	2876
9.1150.1.5 valueWidgetRects() . . . . .	2877
9.1150.1.6 write() . . . . .	2877
9.1151 Digikam::SearchFieldRating Class Reference . . . . .	2878
9.1151.1 Member Function Documentation . . . . .	2880
9.1151.1.1 read() . . . . .	2880
9.1151.1.2 reset() . . . . .	2880
9.1151.1.3 setupValueWidgets() . . . . .	2880
9.1151.1.4 setValueWidgetsVisible() . . . . .	2880
9.1151.1.5 valueWidgetRects() . . . . .	2881
9.1151.1.6 write() . . . . .	2881
9.1152 Digikam::SearchFieldText Class Reference . . . . .	2882
9.1152.1 Member Function Documentation . . . . .	2884
9.1152.1.1 read() . . . . .	2884
9.1152.1.2 reset() . . . . .	2884
9.1152.1.3 setupValueWidgets() . . . . .	2884
9.1152.1.4 setValueWidgetsVisible() . . . . .	2884
9.1152.1.5 valueWidgetRects() . . . . .	2885
9.1152.1.6 write() . . . . .	2885
9.1153 Digikam::SearchFilterModel Class Reference . . . . .	2885
9.1153.1 Member Function Documentation . . . . .	2889
9.1153.1.1 isFiltering() . . . . .	2889
9.1153.1.2 matches() . . . . .	2889
9.1154 Digikam::SearchGroup Class Reference . . . . .	2890
9.1154.1 Member Function Documentation . . . . .	2892
9.1154.1.1 addGroupToLayout() . . . . .	2892
9.1154.1.2 createSearchGroup() . . . . .	2892
9.1155 Digikam::SearchGroupLabel Class Reference . . . . .	2893
9.1156 Digikam::SearchInfo Class Reference . . . . .	2894
9.1157 Digikam::SearchModel Class Reference . . . . .	2895

---

9.1157.1 Member Function Documentation . . . . .	2901
9.1157.1.1 albumData() . . . . .	2901
9.1157.1.2 albumForId() . . . . .	2901
9.1157.1.3 setReplaceNames() . . . . .	2901
9.1158 Digikam::SearchModificationHelper Class Reference . . . . .	2901
9.1158.1 Detailed Description . . . . .	2903
9.1158.2 Constructor & Destructor Documentation . . . . .	2903
9.1158.2.1 SearchModificationHelper() . . . . .	2903
9.1158.3 Member Function Documentation . . . . .	2903
9.1158.3.1 createFuzzySearchFromDropped() . . . . .	2903
9.1158.3.2 createFuzzySearchFromImage() . . . . .	2904
9.1158.3.3 createFuzzySearchFromSketch() . . . . .	2904
9.1158.3.4 slotCreateFuzzySearchFromDropped . . . . .	2904
9.1158.3.5 slotCreateFuzzySearchFromImage . . . . .	2905
9.1158.3.6 slotCreateFuzzySearchFromSketch . . . . .	2905
9.1158.3.7 slotCreateTimeLineSearch . . . . .	2906
9.1158.3.8 slotSearchDelete . . . . .	2906
9.1158.3.9 slotSearchRename . . . . .	2906
9.1159 Digikam::SearchSideBarWidget Class Reference . . . . .	2907
9.1159.1 Member Function Documentation . . . . .	2909
9.1159.1.1 applySettings() . . . . .	2909
9.1159.1.2 changeAlbumFromHistory() . . . . .	2909
9.1159.1.3 doLoadState() . . . . .	2909
9.1159.1.4 doSaveState() . . . . .	2909
9.1159.1.5 getCaption() . . . . .	2910
9.1159.1.6 getIcon() . . . . .	2910
9.1159.1.7 setActive() . . . . .	2910
9.1160 Digikam::SearchTabHeader Class Reference . . . . .	2911
9.1161 Digikam::SearchTextBar Class Reference . . . . .	2912
9.1161.1 Detailed Description . . . . .	2914
9.1161.2 Member Enumeration Documentation . . . . .	2914
9.1161.2.1 HighlightState . . . . .	2914
9.1161.3 Member Function Documentation . . . . .	2914
9.1161.3.1 doLoadState() . . . . .	2914
9.1161.3.2 doSaveState() . . . . .	2914
9.1161.3.3 getCurrentHighlightState() . . . . .	2915
9.1161.3.4 setCaseSensitive() . . . . .	2915
9.1161.3.5 setHighlightOnResult() . . . . .	2915
9.1162 Digikam::SearchTextBarDb Class Reference . . . . .	2915
9.1162.1 Detailed Description . . . . .	2918
9.1162.2 Member Function Documentation . . . . .	2918
9.1162.2.1 setFilterModel() . . . . .	2918



9.1162.2.2 setModel() [1/2]	2919
9.1162.2.3 setModel() [2/2]	2919
9.1163 Digikam::SearchTextFilterSettings Class Reference	2919
9.1164 Digikam::SearchTextSettings Class Reference	2920
9.1165 Digikam::SearchTreeView Class Reference	2921
9.1166 Digikam::SearchView Class Reference	2927
9.1166.1 Member Function Documentation	2929
9.1166.1.1 addGroupToLayout()	2929
9.1166.1.2 bottomBarPixmap()	2929
9.1166.1.3 createSearchGroup()	2929
9.1166.1.4 groupLabelPixmap()	2929
9.1167 Digikam::SearchViewBottomBar Class Reference	2930
9.1168 Digikam::SearchViewThemedPartsCache Class Reference	2931
9.1169 Digikam::SearchWindow Class Reference	2932
9.1169.1 Member Function Documentation	2933
9.1169.1.1 readSearch()	2933
9.1169.1.2 reset()	2933
9.1169.1.3 searchEdited	2933
9.1170 Digikam::SearchXmlCachingReader Class Reference	2934
9.1171 Digikam::SearchXmlReader Class Reference	2937
9.1171.1 Member Function Documentation	2938
9.1171.1.1 defaultFieldOperator()	2938
9.1171.1.2 fieldOperator()	2939
9.1171.1.3 groupCaption()	2939
9.1171.1.4 groupOperator()	2939
9.1171.1.5 readNext()	2939
9.1171.1.6 readToStartOfElement()	2939
9.1171.1.7 value()	2939
9.1172 Digikam::SearchXmlWriter Class Reference	2940
9.1172.1 Member Function Documentation	2942
9.1172.1.1 finish()	2942
9.1172.1.2 finishField()	2942
9.1172.1.3 finishGroup()	2942
9.1172.1.4 setDefaultFieldOperator()	2942
9.1172.1.5 setGroupOperator()	2942
9.1172.1.6 writeField()	2942
9.1172.1.7 writeGroup()	2943
9.1172.1.8 xml()	2943
9.1173 Digikam::SequenceNumberDialog Class Reference	2943
9.1174 Digikam::SequenceNumberOption Class Reference	2945
9.1174.1 Member Function Documentation	2947
9.1174.1.1 parseOperation()	2947

9.1175 Digikam::Setup Class Reference . . . . .	2948
9.1175.1 Member Function Documentation . . . . .	2951
9.1175.1.1 execDialog() . . . . .	2951
9.1175.1.2 execSinglePage() . . . . .	2951
9.1176 Digikam::SetupAlbumView Class Reference . . . . .	2951
9.1177 Digikam::SetupCamera Class Reference . . . . .	2952
9.1178 Digikam::SetupCategory Class Reference . . . . .	2953
9.1179 Digikam::SetupCollectionDelegate Class Reference . . . . .	2954
9.1179.1 Member Function Documentation . . . . .	2956
9.1179.1.1 createItemWidgets() . . . . .	2956
9.1179.1.2 updateItemWidgets() . . . . .	2956
9.1180 Digikam::SetupCollectionModel Class Reference . . . . .	2958
9.1180.1 Member Enumeration Documentation . . . . .	2960
9.1180.1.1 SetupCollectionDataRole . . . . .	2960
9.1180.2 Constructor & Destructor Documentation . . . . .	2961
9.1180.2.1 SetupCollectionModel() . . . . .	2961
9.1180.3 Member Function Documentation . . . . .	2961
9.1180.3.1 slotAppendPressed . . . . .	2961
9.1180.3.2 slotCategoryButtonPressed . . . . .	2961
9.1181 Digikam::SetupCollectionModel::Item Class Reference . . . . .	2961
9.1182 Digikam::SetupCollections Class Reference . . . . .	2962
9.1183 Digikam::SetupCollectionView Class Reference . . . . .	2963
9.1184 Digikam::SetupDatabase Class Reference . . . . .	2964
9.1185 Digikam::SetupEditor Class Reference . . . . .	2965
9.1186 Digikam::SetupEditorIface Class Reference . . . . .	2966
9.1187 Digikam::SetupGeolocation Class Reference . . . . .	2967
9.1188 Digikam::SetupICC Class Reference . . . . .	2968
9.1188.1 Constructor & Destructor Documentation . . . . .	2968
9.1188.1.1 SetupICC() . . . . .	2968
9.1189 Digikam::SetupImageQualitySorter Class Reference . . . . .	2969
9.1190 Digikam::SetupIOFiles Class Reference . . . . .	2970
9.1191 Digikam::SetupLightTable Class Reference . . . . .	2970
9.1192 Digikam::SetupMetadata Class Reference . . . . .	2971
9.1193 Digikam::SetupMime Class Reference . . . . .	2972
9.1194 Digikam::SetupMisc Class Reference . . . . .	2973
9.1195 Digikam::SetupPlugins Class Reference . . . . .	2974
9.1196 Digikam::SetupRaw Class Reference . . . . .	2975
9.1197 Digikam::SetupTemplate Class Reference . . . . .	2976
9.1198 Digikam::SetupToolTip Class Reference . . . . .	2977
9.1199 Digikam::SetupVersioning Class Reference . . . . .	2978
9.1200 Digikam::SharedLoadingTask Class Reference . . . . .	2979
9.1200.1 Member Function Documentation . . . . .	2981

9.1200.1.1	accessMode()	2981
9.1200.1.2	addListener()	2981
9.1200.1.3	cacheKey()	2981
9.1200.1.4	completed()	2981
9.1200.1.5	execute()	2981
9.1200.1.6	loadSaveNotifier()	2982
9.1200.1.7	notifyNewLoadingProcess()	2982
9.1200.1.8	progressInfo()	2982
9.1200.1.9	querySendNotifyEvent()	2982
9.1200.1.10	removeListener()	2982
9.1200.1.11	setResult()	2982
9.1201	Digikam::SharedLoadSaveThread Class Reference	2983
9.1202	Digikam::SharedQueue< T > Class Template Reference	2987
9.1203	Digikam::SharpContainer Class Reference	2987
9.1204	Digikam::SharpenFilter Class Reference	2989
9.1204.1	Member Function Documentation	2993
9.1204.1.1	filterAction()	2993
9.1204.1.2	filterIdentifier()	2993
9.1204.1.3	readParameters()	2993
9.1205	Digikam::SharpSettings Class Reference	2993
9.1206	Digikam::ShearFilter Class Reference	2995
9.1206.1	Member Function Documentation	2999
9.1206.1.1	filterAction()	2999
9.1206.1.2	filterIdentifier()	2999
9.1206.1.3	readParameters()	2999
9.1207	Digikam::ShowHideVersionsOverlay Class Reference	3000
9.1207.1	Member Function Documentation	3003
9.1207.1.1	checkIndex()	3003
9.1207.1.2	createButton()	3003
9.1207.1.3	setActive()	3003
9.1207.1.4	updateButton()	3003
9.1208	Digikam::Sidebar Class Reference	3004
9.1208.1	Detailed Description	3007
9.1208.2	Constructor & Destructor Documentation	3007
9.1208.2.1	Sidebar()	3007
9.1208.3	Member Function Documentation	3008
9.1208.3.1	activeNextTab()	3008
9.1208.3.2	activePreviousTab()	3008
9.1208.3.3	appendTab()	3008
9.1208.3.4	backup()	3008
9.1208.3.5	doLoadState()	3008
9.1208.3.6	doSaveState()	3008

---

9.1208.3.7 restore()	3009
9.1209 Digikam::SidebarSplitter Class Reference	3009
9.1209.1 Member Function Documentation	3010
9.1209.1.1 restoreState() [1/2]	3010
9.1209.1.2 restoreState() [2/2]	3010
9.1209.1.3 saveState() [1/2]	3010
9.1209.1.4 saveState() [2/2]	3011
9.1209.1.5 setSize()	3011
9.1210 Digikam::SidebarWidget Class Reference	3011
9.1210.1 Constructor & Destructor Documentation	3013
9.1210.1.1 SidebarWidget()	3013
9.1210.2 Member Function Documentation	3013
9.1210.2.1 applySettings()	3013
9.1210.2.2 changeAlbumFromHistory()	3013
9.1210.2.3 getCaption()	3013
9.1210.2.4 getIcon()	3013
9.1210.2.5 setActive()	3014
9.1211 Digikam::SidecarFinder Class Reference	3014
9.1212 Digikam::SimilarityDb Class Reference	3014
9.1212.1 Member Function Documentation	3015
9.1212.1.1 clearImageSimilarity()	3015
9.1212.1.2 getDirtyOrMissingFingerprints()	3015
9.1212.1.3 getDirtyOrMissingFingerprintURLs()	3016
9.1212.1.4 getImageSimilarity()	3016
9.1212.1.5 getImageSimilarityAlgorithms()	3016
9.1212.1.6 getLegacySetting()	3017
9.1212.1.7 getSetting()	3017
9.1212.1.8 hasDirtyOrMissingFingerprint()	3017
9.1212.1.9 hasFingerprint()	3018
9.1212.1.10 hasFingerprints() [1/2]	3018
9.1212.1.11 hasFingerprints() [2/2]	3018
9.1212.1.12 integrityCheck()	3018
9.1212.1.13 registeredImageIds()	3019
9.1212.1.14 removeImageFingerprint()	3019
9.1212.1.15 removeImageSimilarity() [1/2]	3019
9.1212.1.16 removeImageSimilarity() [2/2]	3019
9.1212.1.17 setSetting()	3020
9.1213 Digikam::SimilarityDbAccess Class Reference	3020
9.1213.1 Constructor & Destructor Documentation	3020
9.1213.1.1 SimilarityDbAccess()	3020
9.1213.2 Member Function Documentation	3021
9.1213.2.1 checkReadyForUse()	3021

---

9.1213.2.2	<a href="#">initDbEngineErrorHandler()</a>	3021
9.1213.2.3	<a href="#">isInitialized()</a>	3021
9.1213.2.4	<a href="#">parameters()</a>	3021
9.1213.2.5	<a href="#">setLastError()</a>	3021
9.1213.2.6	<a href="#">setParameters()</a>	3021
9.1214	<a href="#">Digikam::SimilarityDbBackend Class Reference</a>	3022
9.1214.1	<a href="#">Member Function Documentation</a>	3026
9.1214.1.1	<a href="#">initSchema()</a>	3026
9.1215	<a href="#">Digikam::SimilarityDbSchemaUpdater Class Reference</a>	3026
9.1216	<a href="#">Digikam::SimpleTreeModel Class Reference</a>	3027
9.1217	<a href="#">Digikam::SimpleTreeModel::Item Class Reference</a>	3028
9.1218	<a href="#">Digikam::SinglePhotoPreviewLayout Class Reference</a>	3029
9.1218.1	<a href="#">Member Function Documentation</a>	3030
9.1218.1.1	<a href="#">addItem()</a>	3030
9.1219	<a href="#">Digikam::SketchWidget Class Reference</a>	3031
9.1219.1	<a href="#">Member Function Documentation</a>	3032
9.1219.1.1	<a href="#">setSketchImageFromXML()</a>	3032
9.1220	<a href="#">Digikam::SlideVideo Class Reference</a>	3033
9.1221	<a href="#">Digikam::SoftProofDialog Class Reference</a>	3034
9.1222	<a href="#">Digikam::SolidHardwareDlg Class Reference</a>	3035
9.1223	<a href="#">Digikam::SpellCheckConfig Class Reference</a>	3036
9.1224	<a href="#">Digikam::SqueezedComboBox Class Reference</a>	3036
9.1224.1	<a href="#">Detailed Description</a>	3038
9.1224.2	<a href="#">Constructor &amp; Destructor Documentation</a>	3038
9.1224.2.1	<a href="#">SqueezedComboBox()</a>	3038
9.1224.3	<a href="#">Member Function Documentation</a>	3038
9.1224.3.1	<a href="#">addSqueezedItem()</a>	3038
9.1224.3.2	<a href="#">contains()</a>	3039
9.1224.3.3	<a href="#">findOriginalText()</a>	3039
9.1224.3.4	<a href="#">insertSqueezedItem()</a>	3039
9.1224.3.5	<a href="#">insertSqueezedList()</a>	3039
9.1224.3.6	<a href="#">item()</a>	3040
9.1224.3.7	<a href="#">itemHighlighted()</a>	3040
9.1224.3.8	<a href="#">setCurrent()</a>	3040
9.1225	<a href="#">Digikam::StackedView Class Reference</a>	3041
9.1226	<a href="#">Digikam::StartScanPage Class Reference</a>	3043
9.1227	<a href="#">Digikam::StateSavingObject Class Reference</a>	3044
9.1227.1	<a href="#">Detailed Description</a>	3045
9.1227.2	<a href="#">Member Enumeration Documentation</a>	3045
9.1227.2.1	<a href="#">StateSavingDepth</a>	3045
9.1227.3	<a href="#">Constructor &amp; Destructor Documentation</a>	3046
9.1227.3.1	<a href="#">StateSavingObject()</a>	3046

---

9.1227.4 Member Function Documentation . . . . .	3046
9.1227.4.1 doLoadState() . . . . .	3046
9.1227.4.2 doSaveState() . . . . .	3046
9.1227.4.3 entryName() . . . . .	3047
9.1227.4.4 getConfigGroup() . . . . .	3047
9.1227.4.5 getStateSavingDepth() . . . . .	3047
9.1227.4.6 setConfigGroup() . . . . .	3047
9.1227.4.7 setEntryPrefix() . . . . .	3048
9.1227.4.8 setStateSavingDepth() . . . . .	3048
9.1228 Digikam::StatusBarProgressWidget Class Reference . . . . .	3049
9.1229 Digikam::StatusProgressBar Class Reference . . . . .	3050
9.1230 Digikam::StayPoppedUpComboBox Class Reference . . . . .	3052
9.1230.1 Constructor & Destructor Documentation . . . . .	3053
9.1230.1.1 StayPoppedUpComboBox() . . . . .	3053
9.1230.2 Member Function Documentation . . . . .	3053
9.1230.2.1 installView() . . . . .	3053
9.1230.2.2 sendViewportEventToView() . . . . .	3054
9.1231 Digikam::StretchFilter Class Reference . . . . .	3055
9.1231.1 Member Function Documentation . . . . .	3059
9.1231.1.1 filterAction() . . . . .	3059
9.1231.1.2 filterIdentifier() . . . . .	3059
9.1231.1.3 readParameters() . . . . .	3059
9.1232 Digikam::StyleSheetDebugger Class Reference . . . . .	3059
9.1232.1 Constructor & Destructor Documentation . . . . .	3060
9.1232.1.1 StyleSheetDebugger() . . . . .	3060
9.1233 Digikam::SubjectData Class Reference . . . . .	3060
9.1234 Digikam::SubjectEdit Class Reference . . . . .	3061
9.1235 Digikam::SubjectWidget Class Reference . . . . .	3063
9.1236 Digikam::SyncJob Class Reference . . . . .	3064
9.1237 Digikam::SystemSettings Class Reference . . . . .	3065
9.1237.1 Member Enumeration Documentation . . . . .	3065
9.1237.1.1 ProxyType . . . . .	3065
9.1238 Digikam::SystemSettingsWidget Class Reference . . . . .	3066
9.1239 Digikam::TableView Class Reference . . . . .	3067
9.1239.1 Member Function Documentation . . . . .	3070
9.1239.1.1 doLoadState() . . . . .	3070
9.1239.1.2 doSaveState() . . . . .	3070
9.1239.1.3 invertSelection() . . . . .	3070
9.1239.1.4 selectAll() . . . . .	3070
9.1239.1.5 slotAwayFromSelection . . . . .	3070
9.1239.1.6 slotDeleteSelected . . . . .	3070
9.1239.1.7 slotSetCurrentWhenAvailable . . . . .	3070

---

9.1240 Digikam::TableViewColumn Class Reference	3071
9.1240.1 Member Function Documentation	3072
9.1240.1.1 columnAffectedByChangeset()	3072
9.1240.1.2 compare()	3072
9.1240.1.3 data()	3072
9.1240.1.4 getColumnFlags()	3073
9.1240.1.5 paint()	3073
9.1240.1.6 sizeHint()	3073
9.1240.1.7 updateThumbnailSize()	3073
9.1241 Digikam::TableViewColumnConfiguration Class Reference	3073
9.1242 Digikam::TableViewColumnConfigurationWidget Class Reference	3074
9.1243 Digikam::TableViewColumnDescription Class Reference	3074
9.1244 Digikam::TableViewColumnFactory Class Reference	3075
9.1245 Digikam::TableViewColumnProfile Class Reference	3076
9.1245.1 Member Function Documentation	3076
9.1245.1.1 loadSettings()	3076
9.1246 Digikam::TableViewColumns::ColumnAudioVideoProperties Class Reference	3077
9.1246.1 Member Function Documentation	3079
9.1246.1.1 compare()	3079
9.1246.1.2 data()	3079
9.1246.1.3 getColumnFlags()	3079
9.1246.1.4 getTitle()	3079
9.1246.1.5 setConfiguration()	3080
9.1247 Digikam::TableViewColumns::ColumnDigikamProperties Class Reference	3081
9.1247.1 Member Function Documentation	3083
9.1247.1.1 columnAffectedByChangeset()	3083
9.1247.1.2 compare()	3083
9.1247.1.3 data()	3083
9.1247.1.4 getColumnFlags()	3083
9.1247.1.5 getDescription()	3084
9.1247.1.6 getTitle()	3084
9.1248 Digikam::TableViewColumns::ColumnFileConfigurationWidget Class Reference	3084
9.1248.1 Member Function Documentation	3085
9.1248.1.1 getNewConfiguration()	3085
9.1249 Digikam::TableViewColumns::ColumnFileProperties Class Reference	3086
9.1249.1 Member Function Documentation	3088
9.1249.1.1 compare()	3088
9.1249.1.2 data()	3088
9.1249.1.3 getColumnFlags()	3088
9.1249.1.4 getConfigurationWidget()	3088
9.1249.1.5 getTitle()	3089
9.1249.1.6 setConfiguration()	3089

---

9.1250 Digikam::TableViewColumns::ColumnGeoConfigurationWidget Class Reference	3089
9.1250.1 Member Function Documentation	3090
9.1250.1.1 getNewConfiguration()	3090
9.1251 Digikam::TableViewColumns::ColumnGeoProperties Class Reference	3091
9.1251.1 Member Function Documentation	3093
9.1251.1.1 compare()	3093
9.1251.1.2 data()	3093
9.1251.1.3 getColumnFlags()	3093
9.1251.1.4 getConfigurationWidget()	3094
9.1251.1.5 getTitle()	3094
9.1251.1.6 setConfiguration()	3094
9.1252 Digikam::TableViewColumns::ColumnItemProperties Class Reference	3095
9.1252.1 Member Function Documentation	3097
9.1252.1.1 compare()	3097
9.1252.1.2 data()	3097
9.1252.1.3 getColumnFlags()	3097
9.1252.1.4 getTitle()	3097
9.1253 Digikam::TableViewColumns::ColumnPhotoConfigurationWidget Class Reference	3098
9.1253.1 Member Function Documentation	3099
9.1253.1.1 getNewConfiguration()	3099
9.1254 Digikam::TableViewColumns::ColumnPhotoProperties Class Reference	3100
9.1254.1 Member Function Documentation	3102
9.1254.1.1 compare()	3102
9.1254.1.2 data()	3102
9.1254.1.3 getColumnFlags()	3102
9.1254.1.4 getConfigurationWidget()	3102
9.1254.1.5 getTitle()	3103
9.1254.1.6 setConfiguration()	3103
9.1255 Digikam::TableViewColumns::ColumnThumbnail Class Reference	3104
9.1255.1 Member Function Documentation	3106
9.1255.1.1 data()	3106
9.1255.1.2 getColumnFlags()	3106
9.1255.1.3 getTitle()	3106
9.1255.1.4 paint()	3106
9.1255.1.5 sizeHint()	3106
9.1255.1.6 updateThumbnailSize()	3107
9.1256 Digikam::TableViewConfigurationDialog Class Reference	3107
9.1257 Digikam::TableViewItemDelegate Class Reference	3108
9.1257.1 Member Function Documentation	3108
9.1257.1.1 sizeHint()	3108
9.1258 Digikam::TableViewModel Class Reference	3109
9.1258.1 Member Function Documentation	3111



9.1258.1.1 addColumnAt()	3111
9.1258.1.2 flags()	3111
9.1258.1.3 indexFromImageId()	3111
9.1258.1.4 infoFromItem()	3111
9.1258.1.5 loadColumnProfile()	3111
9.1258.1.6 parent()	3111
9.1258.1.7 sort()	3111
9.1259 Digikam::TableViewModel::Item Class Reference	3112
9.1260 Digikam::TableViewSelectionModeSyncer Class Reference	3112
9.1260.1 Constructor & Destructor Documentation	3113
9.1260.1.1 TableViewSelectionModeSyncer()	3113
9.1261 Digikam::TableViewShared Class Reference	3113
9.1262 Digikam::TableViewTreeView Class Reference	3114
9.1262.1 Detailed Description	3115
9.1262.2 Member Function Documentation	3115
9.1262.2.1 dragDropHandler()	3115
9.1262.2.2 hasHiddenGroupedImages()	3115
9.1262.2.3 mapIndexForDragDrop()	3116
9.1262.2.4 pixmapForDrag()	3116
9.1263 Digikam::TagChangeset Class Reference	3116
9.1263.1 Member Enumeration Documentation	3116
9.1263.1.1 Operation	3116
9.1264 Digikam::TagCheckView Class Reference	3117
9.1264.1 Member Function Documentation	3123
9.1264.1.1 addCustomContextMenuActions()	3123
9.1264.1.2 checkedTagsChanged	3124
9.1264.1.3 doLoadState()	3124
9.1264.1.4 doSaveState()	3124
9.1264.1.5 setCheckNewTags()	3124
9.1265 Digikam::TagCompleter Class Reference	3125
9.1265.1 Member Function Documentation	3125
9.1265.1.1 setSupportingTagModel()	3125
9.1266 Digikam::TagData Struct Reference	3126
9.1267 Digikam::TagDragDropHandler Class Reference	3126
9.1267.1 Member Function Documentation	3127
9.1267.1.1 accepts()	3127
9.1267.1.2 createMimeData()	3127
9.1267.1.3 dropEvent()	3128
9.1267.1.4 mimeTypees()	3128
9.1268 Digikam::TagEditDlg Class Reference	3128
9.1268.1 Member Function Documentation	3129
9.1268.1.1 createTAlbum()	3129

---

9.1269 Digikam::TagFilterView Class Reference	3129
9.1269.1 Detailed Description	3136
9.1269.2 Constructor & Destructor Documentation	3136
9.1269.2.1 TagFilterView()	3136
9.1269.3 Member Function Documentation	3137
9.1269.3.1 addCustomContextMenuActions()	3137
9.1269.3.2 handleCustomContextMenuAction()	3137
9.1270 Digikam::TagFolderView Class Reference	3138
9.1270.1 Constructor & Destructor Documentation	3143
9.1270.1.1 TagFolderView()	3143
9.1270.2 Member Function Documentation	3144
9.1270.2.1 addCustomContextMenuActions()	3144
9.1270.2.2 contextMenuEvent()	3144
9.1270.2.3 contextMenuTitle()	3144
9.1270.2.4 handleCustomContextMenuAction()	3145
9.1270.2.5 setContextMenuItems()	3145
9.1270.2.6 setShowDeleteFaceTagsAction()	3145
9.1270.2.7 setShowFindDuplicateAction()	3145
9.1271 Digikam::TaggingAction Class Reference	3146
9.1271.1 Constructor & Destructor Documentation	3146
9.1271.1.1 TaggingAction()	3146
9.1272 Digikam::TaggingActionFactory Class Reference	3146
9.1272.1 Member Enumeration Documentation	3147
9.1272.1.1 NameMatchMode	3147
9.1272.2 Member Function Documentation	3148
9.1272.2.1 setConstraintInterface()	3148
9.1273 Digikam::TaggingActionFactory::ConstraintInterface Class Reference	3148
9.1274 Digikam::TagInfo Class Reference	3149
9.1275 Digikam::TagList Class Reference	3149
9.1275.1 Member Function Documentation	3150
9.1275.1.1 restoreSettings()	3150
9.1276 Digikam::TagMngrListModel Class Reference	3150
9.1276.1 Member Function Documentation	3151
9.1276.1.1 addItem()	3151
9.1276.1.2 dropMimeData()	3151
9.1277 Digikam::TagMngrListView Class Reference	3152
9.1278 Digikam::TagMngrTreeView Class Reference	3153
9.1278.1 Member Function Documentation	3159
9.1278.1.1 contextMenuEvent()	3159
9.1278.1.2 setContextMenuItems()	3159
9.1279 Digikam::TagModel Class Reference	3160
9.1279.1 Member Function Documentation	3166

---

9.1279.1.1 albumData()	3166
9.1279.1.2 albumForId()	3166
9.1279.1.3 decorationRoleData()	3166
9.1279.1.4 fontRoleData()	3166
9.1280 Digikam::TagModificationHelper Class Reference	3167
9.1280.1 Detailed Description	3169
9.1280.2 Constructor & Destructor Documentation	3169
9.1280.2.1 TagModificationHelper()	3169
9.1280.3 Member Function Documentation	3169
9.1280.3.1 bindMultipleTags()	3169
9.1280.3.2 bindTag()	3169
9.1280.3.3 boundMultipleTags()	3170
9.1280.3.4 boundTag()	3170
9.1280.3.5 slotFaceTagDelete	3170
9.1280.3.6 slotMultipleFaceTagDel	3170
9.1280.3.7 slotMultipleTagDel	3170
9.1280.3.8 slotMultipleTagsToFaceTags	3171
9.1280.3.9 slotTagDelete	3171
9.1280.3.10 slotTagEdit	3171
9.1280.3.11 slotTagNew [1/2]	3171
9.1280.3.12 slotTagNew [2/2]	3172
9.1280.3.13 slotTagToFaceTag	3172
9.1281 Digikam::TagProperties Class Reference	3172
9.1281.1 Constructor & Destructor Documentation	3173
9.1281.1.1 TagProperties()	3173
9.1281.2 Member Function Documentation	3173
9.1281.2.1 addProperty()	3173
9.1281.2.2 getOrCreate()	3173
9.1281.2.3 value()	3173
9.1282 Digikam::TagPropertiesFilterModel Class Reference	3174
9.1282.1 Member Function Documentation	3177
9.1282.1.1 isFiltering()	3177
9.1282.1.2 matches()	3177
9.1283 Digikam::TagProperty Class Reference	3178
9.1284 Digikam::TagPropertyName Class Reference	3178
9.1285 Digikam::TagPropWidget Class Reference	3178
9.1286 Digikam::TagRegion Class Reference	3179
9.1286.1 Constructor & Destructor Documentation	3180
9.1286.1.1 TagRegion()	3180
9.1286.2 Member Function Documentation	3180
9.1286.2.1 absoluteToRelative()	3180
9.1286.2.2 adjustToOrientation()	3181

---

9.1286.2.3 intersects()	3181
9.1286.2.4 reverseToOrientation()	3181
9.1286.2.5 toVariant()	3181
9.1287 Digikam::TagsActionMngr Class Reference	3182
9.1287.1 Member Function Documentation	3183
9.1287.1.1 registerLabelsActions()	3183
9.1287.1.2 registerTagsActionCollections()	3183
9.1287.1.3 updateTagShortcut()	3183
9.1288 Digikam::TagsCache Class Reference	3184
9.1288.1 Member Enumeration Documentation	3186
9.1288.1.1 LeadingSlashPolicy	3186
9.1288.2 Member Function Documentation	3187
9.1288.2.1 canBeWrittenToMetadata()	3187
9.1288.2.2 colorLabelForTag()	3187
9.1288.2.3 colorLabelFromTags()	3187
9.1288.2.4 createTag()	3187
9.1288.2.5 getOrCreateTag()	3187
9.1288.2.6 getOrCreateTagWithProperty()	3188
9.1288.2.7 hasProperty()	3188
9.1288.2.8 parentTags()	3188
9.1288.2.9 pickLabelForTag()	3188
9.1288.2.10 pickLabelFromTags()	3188
9.1288.2.11 properties()	3188
9.1288.2.12 propertyValue()	3189
9.1288.2.13 shortenedTagPaths()	3189
9.1288.2.14 tagAdded	3189
9.1288.2.15 tagForColorLabel()	3189
9.1288.2.16 tagForName()	3189
9.1288.2.17 tagForPath()	3189
9.1288.2.18 tagForPickLabel()	3190
9.1288.2.19 tagName()	3190
9.1288.2.20 tagPath()	3190
9.1288.2.21 tagsForName()	3190
9.1288.2.22 tagsWithProperty()	3190
9.1288.2.23 tagsWithPropertyCached()	3190
9.1289 Digikam::TagsDBJobInfo Class Reference	3191
9.1290 Digikam::TagsDBJobsThread Class Reference	3192
9.1290.1 Member Function Documentation	3194
9.1290.1.1 tagsListing()	3194
9.1291 Digikam::TagsEdit Class Reference	3195
9.1292 Digikam::TagShortInfo Class Reference	3196
9.1293 Digikam::TagsJob Class Reference	3196

---

9.1294 Digikam::TagsLineEditOverlay Class Reference	3198
9.1294.1 Member Function Documentation	3201
9.1294.1.1 createWidget()	3201
9.1294.1.2 hide()	3201
9.1294.1.3 setActive()	3201
9.1294.1.4 slotEntered()	3201
9.1294.1.5 visualChange()	3202
9.1295 Digikam::TagsManager Class Reference	3202
9.1295.1 Member Function Documentation	3204
9.1295.1.1 doLoadState()	3204
9.1295.1.2 doSaveState()	3204
9.1296 Digikam::TagsManagerFilterModel Class Reference	3205
9.1296.1 Member Function Documentation	3208
9.1296.1.1 matches()	3208
9.1297 Digikam::TagsPopupMenu Class Reference	3209
9.1297.1 Member Enumeration Documentation	3209
9.1297.1.1 Mode	3209
9.1298 Digikam::TagTreeView Class Reference	3210
9.1299 Digikam::TagTreeViewSelectComboBox Class Reference	3216
9.1300 Digikam::TagViewSideBarWidget Class Reference	3220
9.1300.1 Member Function Documentation	3222
9.1300.1.1 applySettings()	3222
9.1300.1.2 changeAlbumFromHistory()	3222
9.1300.1.3 doLoadState()	3222
9.1300.1.4 doSaveState()	3222
9.1300.1.5 getCaption()	3223
9.1300.1.6 getIcon()	3223
9.1300.1.7 setActive()	3223
9.1301 Digikam::TAlbum Class Reference	3223
9.1301.1 Member Function Documentation	3226
9.1301.1.1 databaseUrl()	3226
9.1301.1.2 tagPath()	3226
9.1302 Digikam::Template Class Reference	3227
9.1302.1 Member Data Documentation	3228
9.1302.1.1 m_templateTitle	3228
9.1303 Digikam::TemplateList Class Reference	3228
9.1304 Digikam::TemplateListItem Class Reference	3229
9.1305 Digikam::TemplateManager Class Reference	3230
9.1306 Digikam::TemplatePanel Class Reference	3231
9.1307 Digikam::TemplateSelector Class Reference	3232
9.1308 Digikam::TemplateViewer Class Reference	3234
9.1309 Digikam::TextFilter Class Reference	3236

---

9.1310 Digikam::TextureContainer Class Reference	3237
9.1311 Digikam::TextureFilter Class Reference	3238
9.1311.1 Member Function Documentation	3242
9.1311.1.1 filterAction()	3242
9.1311.1.2 filterIdentifier()	3242
9.1311.1.3 readParameters()	3242
9.1312 Digikam::TextureSettings Class Reference	3242
9.1313 Digikam::ThemeManager Class Reference	3243
9.1314 Digikam::ThreadManager Class Reference	3244
9.1315 Digikam::ThumbBarDock Class Reference	3245
9.1315.1 Detailed Description	3246
9.1315.2 Member Function Documentation	3246
9.1315.2.1 reinitialize()	3246
9.1315.2.2 shouldBeVisible()	3246
9.1316 Digikam::ThumbnailAligningDelegate Class Reference	3247
9.1317 Digikam::ThumbnailCreator Class Reference	3247
9.1317.1 Constructor & Destructor Documentation	3248
9.1317.1.1 ThumbnailCreator()	3248
9.1317.2 Member Function Documentation	3248
9.1317.2.1 deleteThumbnailsFromDisk()	3248
9.1317.2.2 errorString()	3249
9.1317.2.3 loadDetail()	3249
9.1317.2.4 setExifRotate()	3249
9.1317.2.5 setLoadingProperties()	3249
9.1317.2.6 setOnlyLargeThumbnails()	3249
9.1317.2.7 setRemoveAlphaChannel()	3249
9.1317.2.8 setThumbnailSize()	3250
9.1317.2.9 store()	3250
9.1317.2.10 storedSize()	3250
9.1318 Digikam::ThumbnailIdentifier Class Reference	3250
9.1319 Digikam::ThumbnailImageCatcher Class Reference	3251
9.1319.1 Constructor & Destructor Documentation	3252
9.1319.1.1 ThumbnailImageCatcher()	3252
9.1319.2 Member Function Documentation	3252
9.1319.2.1 cancel	3252
9.1319.2.2 enqueue()	3252
9.1319.2.3 setActive	3253
9.1320 Digikam::ThumbnailInfo Class Reference	3253
9.1320.1 Member Data Documentation	3254
9.1320.1.1 isAccessible	3254
9.1320.1.2 mimeType	3254
9.1320.1.3 modificationDate	3254

9.1320.1.4 orientationHint . . . . .	3254
9.1321 Digikam::ThumbnailInfoProvider Class Reference . . . . .	3255
9.1322 Digikam::ThumbnailLoadingTask Class Reference . . . . .	3256
9.1322.1 Member Function Documentation . . . . .	3258
9.1322.1.1 execute() . . . . .	3258
9.1322.1.2 postProcess() . . . . .	3258
9.1323 Digikam::ThumbnailLoadThread Class Reference . . . . .	3259
9.1323.1 Member Function Documentation . . . . .	3265
9.1323.1.1 defaultThread() . . . . .	3265
9.1323.1.2 deleteThumbnail() . . . . .	3265
9.1323.1.3 find() [1/2] . . . . .	3265
9.1323.1.4 find() [2/2] . . . . .	3265
9.1323.1.5 findGroup() . . . . .	3265
9.1323.1.6 initializeNoThumbnailStorage() . . . . .	3265
9.1323.1.7 initializeThumbnailDatabase() . . . . .	3266
9.1323.1.8 lastDescriptions() . . . . .	3266
9.1323.1.9 load() . . . . .	3266
9.1323.1.10 maximumThumbnailSize() . . . . .	3266
9.1323.1.11 pregenerateGroup() . . . . .	3266
9.1323.1.12 preload() . . . . .	3266
9.1323.1.13 setDisplayingWidget() . . . . .	3267
9.1323.1.14 setHighlightPixmap() . . . . .	3267
9.1323.1.15 setPixmapRequested() . . . . .	3267
9.1323.1.16 setSendSurrogatePixmap() . . . . .	3267
9.1323.1.17 setThumbnailSize() . . . . .	3267
9.1323.1.18 signalThumbnailLoaded . . . . .	3268
9.1323.1.19 storeDetailThumbnail() . . . . .	3268
9.1323.1.20 thumbnailCreator() . . . . .	3268
9.1323.1.21 thumbnailLoaded() . . . . .	3268
9.1323.1.22 thumbnailsAvailable . . . . .	3268
9.1323.1.23 thumbnailToPixmapSize() . . . . .	3269
9.1324 Digikam::ThumbnailSize Class Reference . . . . .	3269
9.1324.1 Member Enumeration Documentation . . . . .	3269
9.1324.1.1 Size . . . . .	3269
9.1325 Digikam::ThumbsDb Class Reference . . . . .	3270
9.1325.1 Member Function Documentation . . . . .	3270
9.1325.1.1 findByFilePath() . . . . .	3270
9.1326 Digikam::ThumbsDbAccess Class Reference . . . . .	3271
9.1326.1 Constructor & Destructor Documentation . . . . .	3271
9.1326.1.1 ThumbsDbAccess() . . . . .	3271
9.1326.2 Member Function Documentation . . . . .	3271
9.1326.2.1 setLastError() . . . . .	3271

9.1327 Digikam::ThumbsDbBackend Class Reference . . . . .	3272
9.1327.1 Member Function Documentation . . . . .	3276
9.1327.1.1 initSchema() . . . . .	3276
9.1328 Digikam::ThumbsDbInfo Class Reference . . . . .	3276
9.1329 Digikam::ThumbsDbInfoProvider Class Reference . . . . .	3276
9.1329.1 Member Function Documentation . . . . .	3277
9.1329.1.1 thumbnailInfo() . . . . .	3277
9.1330 Digikam::ThumbsDbSchemaUpdater Class Reference . . . . .	3277
9.1331 Digikam::ThumbsGenerator Class Reference . . . . .	3278
9.1331.1 Constructor & Destructor Documentation . . . . .	3280
9.1331.1.1 ThumbsGenerator() [1/2] . . . . .	3280
9.1331.1.2 ThumbsGenerator() [2/2] . . . . .	3281
9.1331.2 Member Function Documentation . . . . .	3281
9.1331.2.1 setUseMultiCoreCPU() . . . . .	3281
9.1332 Digikam::ThumbsTask Class Reference . . . . .	3281
9.1333 Digikam::TileGrouper Class Reference . . . . .	3283
9.1333.1 Member Function Documentation . . . . .	3283
9.1333.1.1 updateClusters() . . . . .	3283
9.1334 Digikam::TileIndex Class Reference . . . . .	3283
9.1335 Digikam::TimeAdjustContainer Class Reference . . . . .	3284
9.1336 Digikam::TimeAdjustSettings Class Reference . . . . .	3285
9.1336.1 Member Function Documentation . . . . .	3286
9.1336.1.1 detAdjustmentByClockPhotoUrl() . . . . .	3286
9.1337 Digikam::TimelineSideBarWidget Class Reference . . . . .	3287
9.1337.1 Member Function Documentation . . . . .	3289
9.1337.1.1 applySettings() . . . . .	3289
9.1337.1.2 changeAlbumFromHistory() . . . . .	3289
9.1337.1.3 doLoadState() . . . . .	3289
9.1337.1.4 doSaveState() . . . . .	3289
9.1337.1.5 getCaption() . . . . .	3290
9.1337.1.6 getIcon() . . . . .	3290
9.1337.1.7 setActive() . . . . .	3290
9.1338 Digikam::TimeLineWidget Class Reference . . . . .	3291
9.1338.1 Member Enumeration Documentation . . . . .	3292
9.1338.1.1 ScaleMode . . . . .	3292
9.1338.1.2 SelectionMode . . . . .	3292
9.1339 Digikam::TimeZoneComboBox Class Reference . . . . .	3293
9.1340 Digikam::Token Class Reference . . . . .	3293
9.1340.1 Detailed Description . . . . .	3294
9.1340.2 Member Function Documentation . . . . .	3295
9.1340.2.1 action() . . . . .	3295
9.1340.2.2 description() . . . . .	3295



9.1340.2.3 id() . . . . .	3295
9.1341 Digikam::TonalityContainer Class Reference . . . . .	3295
9.1342 Digikam::TonalityFilter Class Reference . . . . .	3296
9.1342.1 Member Function Documentation . . . . .	3300
9.1342.1.1 filterAction() . . . . .	3300
9.1342.1.2 filterIdentifier() . . . . .	3300
9.1342.1.3 readParameters() . . . . .	3300
9.1343 Digikam::ToolListViewGroup Class Reference . . . . .	3300
9.1344 Digikam::ToolListViewItem Class Reference . . . . .	3301
9.1345 Digikam::ToolSettingsView Class Reference . . . . .	3302
9.1346 Digikam::ToolsListView Class Reference . . . . .	3303
9.1347 Digikam::ToolsView Class Reference . . . . .	3304
9.1348 Digikam::TooltipCreator Class Reference . . . . .	3305
9.1349 Digikam::TooltipDialog Class Reference . . . . .	3305
9.1350 Digikam::TooltipsPage Class Reference . . . . .	3306
9.1351 Digikam::TrackCorrelator Class Reference . . . . .	3307
9.1352 Digikam::TrackCorrelator::Correlation Class Reference . . . . .	3308
9.1353 Digikam::TrackCorrelator::CorrelationOptions Class Reference . . . . .	3308
9.1354 Digikam::TrackCorrelatorThread Class Reference . . . . .	3309
9.1355 Digikam::TrackListModel Class Reference . . . . .	3310
9.1355.1 Member Function Documentation . . . . .	3311
9.1355.1.1 headerData() . . . . .	3311
9.1355.1.2 index() . . . . .	3311
9.1356 Digikam::TrackManager Class Reference . . . . .	3311
9.1356.1 Member Typedef Documentation . . . . .	3312
9.1356.1.1 Id . . . . .	3312
9.1356.2 Member Function Documentation . . . . .	3312
9.1356.2.1 clear() . . . . .	3312
9.1357 Digikam::TrackManager::Track Class Reference . . . . .	3313
9.1358 Digikam::TrackManager::TrackPoint Class Reference . . . . .	3313
9.1359 Digikam::TrackReader Class Reference . . . . .	3314
9.1360 Digikam::TrackReader::TrackReadResult Class Reference . . . . .	3314
9.1361 Digikam::TrainerWorker Class Reference . . . . .	3315
9.1361.1 Member Function Documentation . . . . .	3317
9.1361.1.1 aboutToDeactivate() . . . . .	3317
9.1362 Digikam::TrainingDataProvider Class Reference . . . . .	3318
9.1362.1 Detailed Description . . . . .	3318
9.1362.2 Member Function Documentation . . . . .	3318
9.1362.2.1 images() . . . . .	3318
9.1362.2.2 newImages() . . . . .	3319
9.1363 Digikam::TransactionItem Class Reference . . . . .	3320
9.1363.1 Member Function Documentation . . . . .	3321

---

9.1363.1.1 setStatus()	3321
9.1364 Digikam::TransactionItemView Class Reference	3322
9.1365 Digikam::TransitionMngr Class Reference	3323
9.1366 Digikam::TransitionPreview Class Reference	3323
9.1367 Digikam::TrashView Class Reference	3324
9.1367.1 Member Function Documentation	3325
9.1367.1.1 getThumbnailSize()	3325
9.1367.1.2 lastSelectedItemUrl()	3325
9.1367.1.3 model()	3325
9.1367.1.4 setThumbnailSize()	3325
9.1367.1.5 statusBarText()	3325
9.1368 Digikam::TreeBranch Class Reference	3326
9.1369 Digikam::TreeProxyModel Class Reference	3326
9.1370 Digikam::TreeViewComboBox Class Reference	3327
9.1370.1 Constructor & Destructor Documentation	3328
9.1370.1.1 TreeViewComboBox()	3328
9.1370.2 Member Function Documentation	3329
9.1370.2.1 installView()	3329
9.1370.2.2 sendViewportEventToView()	3329
9.1370.2.3 view()	3329
9.1371 Digikam::TreeViewLineEditComboBox Class Reference	3330
9.1371.1 Constructor & Destructor Documentation	3332
9.1371.1.1 TreeViewLineEditComboBox()	3332
9.1371.2 Member Function Documentation	3332
9.1371.2.1 installLineEdit()	3332
9.1371.2.2 installView()	3332
9.1371.2.3 setLineEditText()	3332
9.1372 Digikam::TrimmedModifier Class Reference	3333
9.1372.1 Member Function Documentation	3335
9.1372.1.1 parseOperation()	3335
9.1373 Digikam::TwoProgressItemsContainer Class Reference	3336
9.1374 Digikam::UMSCamera Class Reference	3337
9.1374.1 Member Function Documentation	3339
9.1374.1.1 cameraAbout()	3339
9.1374.1.2 cameraDriverType()	3339
9.1374.1.3 cameraManual()	3339
9.1374.1.4 cameraMD5ID()	3339
9.1374.1.5 cameraSummary()	3340
9.1374.1.6 cancel()	3340
9.1374.1.7 capture()	3340
9.1374.1.8 deleteItem()	3340
9.1374.1.9 doConnect()	3340

9.1374.1.10 downloadItem()	3340
9.1374.1.11 getFolders()	3340
9.1374.1.12 getFreeSpace()	3341
9.1374.1.13 getItemInfo()	3341
9.1374.1.14 getItemInfoList()	3341
9.1374.1.15 getMetadata()	3341
9.1374.1.16 getPreview()	3341
9.1374.1.17 getThumbnail()	3342
9.1374.1.18 setLockItem()	3342
9.1374.1.19 uploadItem()	3342
9.1375 Digikam::UndoAction Class Reference	3343
9.1376 Digikam::UndoActionIrreversible Class Reference	3344
9.1377 Digikam::UndoActionReversible Class Reference	3345
9.1378 Digikam::UndoCache Class Reference	3346
9.1379 Digikam::UndoManager Class Reference	3346
9.1380 Digikam::UndoMetadataContainer Class Reference	3346
9.1381 Digikam::UndoState Class Reference	3347
9.1382 Digikam::UniqueModifier Class Reference	3348
9.1382.1 Member Function Documentation	3350
9.1382.1.1 parseOperation()	3350
9.1382.1.2 reset()	3350
9.1383 Digikam::UnsharpMaskFilter Class Reference	3351
9.1383.1 Member Function Documentation	3355
9.1383.1.1 filterAction()	3355
9.1383.1.2 filterIdentifier()	3355
9.1383.1.3 readParameters()	3355
9.1384 Digikam::VersionFileInfo Class Reference	3355
9.1385 Digikam::VersionFileOperation Class Reference	3355
9.1385.1 Member Enumeration Documentation	3356
9.1385.1.1 Task	3356
9.1385.2 Constructor & Destructor Documentation	3356
9.1385.2.1 VersionFileOperation()	3356
9.1386 Digikam::VersioningPromptUserSaveDialog Class Reference	3357
9.1387 Digikam::VersionItemFilterSettings Class Reference	3357
9.1388 Digikam::VersionManager Class Reference	3358
9.1389 Digikam::VersionManagerSettings Class Reference	3358
9.1390 Digikam::VersionNamingScheme Class Reference	3359
9.1390.1 Member Function Documentation	3360
9.1390.1.1 baseName()	3360
9.1390.1.2 directory()	3360
9.1390.1.3 incrementedCounter()	3361
9.1390.1.4 initialCounter()	3361

9.1390.1.5 intermediateFileName()	3361
9.1390.1.6 versionFileName()	3361
9.1391 Digikam::VersionsDelegate Class Reference	3362
9.1391.1 Member Function Documentation	3364
9.1391.1.1 asDelegate()	3364
9.1391.1.2 requestNotification	3364
9.1392 Digikam::VersionsTreeView Class Reference	3365
9.1392.1 Constructor & Destructor Documentation	3367
9.1392.1.1 ~VersionsTreeView()	3367
9.1392.2 Member Function Documentation	3367
9.1392.2.1 dragDropHandler()	3367
9.1392.2.2 mapIndexForDragDrop()	3367
9.1392.2.3 pixmapForDrag()	3367
9.1393 Digikam::VersionsWidget Class Reference	3368
9.1394 Digikam::VideoFrame Class Reference	3369
9.1395 Digikam::VideoInfoContainer Class Reference	3369
9.1396 Digikam::VideoMetadataContainer Class Reference	3369
9.1397 Digikam::VideoStripFilter Class Reference	3370
9.1398 Digikam::VideoThumbDecoder Class Reference	3370
9.1399 Digikam::VideoThumbnailer Class Reference	3370
9.1400 Digikam::VideoThumbWriter Class Reference	3370
9.1401 Digikam::VidPlayerDlg Class Reference	3371
9.1402 Digikam::VidSlideSettings Class Reference	3371
9.1402.1 Member Enumeration Documentation	3374
9.1402.1.1 VidBitRate	3374
9.1402.1.2 VidCodec	3374
9.1402.1.3 VidFormat	3375
9.1402.1.4 VidStd	3375
9.1402.1.5 VidType	3375
9.1403 Digikam::VidSlideTask Class Reference	3377
9.1404 Digikam::VidSlideThread Class Reference	3379
9.1405 Digikam::VisibilityController Class Reference	3381
9.1405.1 Member Function Documentation	3382
9.1405.1.1 addObject()	3382
9.1406 Digikam::VisibilityObject Class Reference	3382
9.1407 Digikam::WBContainer Class Reference	3383
9.1408 Digikam::WBFilter Class Reference	3384
9.1408.1 Member Function Documentation	3388
9.1408.1.1 autoWBAdjustementFromColor()	3388
9.1408.1.2 filterAction()	3388
9.1408.1.3 filterIdentifier()	3388
9.1408.1.4 filterImage()	3388

---

9.1408.1.5 readParameters() . . . . .	3388
9.1409 Digikam::WBSettings Class Reference . . . . .	3389
9.1410 Digikam::WebBrowserDlg Class Reference . . . . .	3390
9.1411 Digikam::WebWidget Class Reference . . . . .	3391
9.1412 Digikam::WelcomePage Class Reference . . . . .	3392
9.1413 Digikam::WelcomePageView Class Reference . . . . .	3393
9.1414 Digikam::WelcomePageViewPage Class Reference . . . . .	3394
9.1415 Digikam::WorkerObject Class Reference . . . . .	3395
9.1415.1 Member Enumeration Documentation . . . . .	3396
9.1415.1.1 DeactivatingMode . . . . .	3396
9.1415.2 Constructor & Destructor Documentation . . . . .	3396
9.1415.2.1 WorkerObject() . . . . .	3396
9.1415.3 Member Function Documentation . . . . .	3397
9.1415.3.1 aboutToDeactivate() . . . . .	3397
9.1415.3.2 aboutToQuitLoop() . . . . .	3397
9.1415.3.3 connectAndSchedule() . . . . .	3397
9.1415.3.4 deactivate . . . . .	3397
9.1415.3.5 setPriority() . . . . .	3397
9.1415.3.6 shutDown() . . . . .	3398
9.1416 Digikam::Workflow Class Reference . . . . .	3398
9.1417 Digikam::WorkflowDlg Class Reference . . . . .	3398
9.1418 Digikam::WorkflowItem Class Reference . . . . .	3399
9.1419 Digikam::WorkflowList Class Reference . . . . .	3400
9.1420 Digikam::WorkflowManager Class Reference . . . . .	3401
9.1420.1 Member Function Documentation . . . . .	3402
9.1420.1.1 load() . . . . .	3402
9.1421 Digikam::WorkingWidget Class Reference . . . . .	3402
9.1422 Digikam::WSAlbum Class Reference . . . . .	3403
9.1423 Digikam::WSComboBoxIntermediate Class Reference . . . . .	3403
9.1423.1 Member Function Documentation . . . . .	3404
9.1423.1.1 setIntermediate() . . . . .	3404
9.1424 Digikam::WSLoginDialog Class Reference . . . . .	3404
9.1425 Digikam::WSNewAlbumDialog Class Reference . . . . .	3405
9.1426 Digikam::WSSelectUserDlg Class Reference . . . . .	3406
9.1427 Digikam::WSSettings Class Reference . . . . .	3407
9.1428 Digikam::WSSettingsWidget Class Reference . . . . .	3409
9.1429 Digikam::WSToolDialog Class Reference . . . . .	3411
9.1430 Digikam::WSToolUtils Class Reference . . . . .	3412
9.1431 Digikam::XbelReader Class Reference . . . . .	3412
9.1432 Digikam::XbelWriter Class Reference . . . . .	3413
9.1433 Digikam::XmpMetaEngineMergeHelper Class Reference . . . . .	3414
9.1434 Digikam::XmpWidget Class Reference . . . . .	3415

---

9.1434.1 Member Function Documentation	3417
9.1434.1.1 getMetadataTitle()	3417
9.1434.1.2 getTagDescription()	3417
9.1434.1.3 getTagTitle()	3417
9.1434.1.4 loadFromURL()	3417
9.1435 ShowFoto::NoDuplicatesShowfotoFilterModel Class Reference	3418
9.1436 ShowFoto::Showfoto Class Reference	3421
9.1436.1 Member Function Documentation	3426
9.1436.1.1 infolface()	3426
9.1437 ShowFoto::ShowfotoCategorizedView Class Reference	3427
9.1437.1 Member Function Documentation	3432
9.1437.1.1 addOverlay()	3432
9.1437.1.2 deselected	3432
9.1437.1.3 dragDropHandler()	3433
9.1437.1.4 filterModel()	3433
9.1437.1.5 indexActivated()	3433
9.1437.1.6 nextIndexHint()	3433
9.1437.1.7 nextInOrder()	3433
9.1437.1.8 selected	3433
9.1437.1.9 showContextMenuOnIndex()	3434
9.1437.1.10 showfotoFilterModel()	3434
9.1437.1.11 showfotoItemInfoActivated	3434
9.1438 ShowFoto::ShowfotoCoordinatesOverlay Class Reference	3435
9.1438.1 Member Function Documentation	3437
9.1438.1.1 checkIndex()	3437
9.1438.1.2 createWidget()	3438
9.1438.1.3 setActive()	3438
9.1438.1.4 slotEntered()	3438
9.1438.1.5 visualChange()	3438
9.1439 ShowFoto::ShowfotoCoordinatesOverlayWidget Class Reference	3438
9.1440 ShowFoto::ShowfotoDelegate Class Reference	3440
9.1440.1 Member Function Documentation	3444
9.1440.1.1 acceptsActivation()	3444
9.1440.1.2 acceptsToolTip()	3444
9.1440.1.3 clearCaches()	3444
9.1440.1.4 imageInformationRect()	3444
9.1440.1.5 pixmapForDrag()	3444
9.1440.1.6 pixmapRect()	3445
9.1440.1.7 setDefaultViewOptions()	3445
9.1440.1.8 updateContentWidth()	3445
9.1440.1.9 updateRects()	3445
9.1440.1.10 updateSizeRectsAndPxmmaps()	3445

---

9.1441 ShowFoto::ShowfotoDragDropHandler Class Reference	3446
9.1441.1 Member Function Documentation	3447
9.1441.1.1 accepts()	3447
9.1441.1.2 createMimeData()	3447
9.1441.1.3 dropEvent()	3447
9.1441.1.4 mimeTypes()	3448
9.1442 ShowFoto::ShowfotoFilterModel Class Reference	3449
9.1442.1 Member Enumeration Documentation	3452
9.1442.1.1 ShowfotoFilterModelRoles	3452
9.1442.2 Member Function Documentation	3453
9.1442.2.1 categoryIdentifier()	3453
9.1442.2.2 compareCategories()	3453
9.1442.2.3 infosLessThan()	3454
9.1442.2.4 setDirectSourceShowfotoModel()	3454
9.1442.2.5 showfotoFilterModel()	3454
9.1442.2.6 subSortLessThan()	3454
9.1443 ShowFoto::ShowfotoFolderViewBar Class Reference	3455
9.1444 ShowFoto::ShowfotoFolderViewBookmarkDlg Class Reference	3457
9.1445 ShowFoto::ShowfotoFolderViewBookmarkItem Class Reference	3458
9.1446 ShowFoto::ShowfotoFolderViewBookmarkList Class Reference	3459
9.1447 ShowFoto::ShowfotoFolderViewBookmarks Class Reference	3460
9.1448 ShowFoto::ShowfotoFolderViewList Class Reference	3461
9.1448.1 Member Enumeration Documentation	3461
9.1448.1.1 FolderViewRole	3461
9.1449 ShowFoto::ShowfotoFolderViewModel Class Reference	3462
9.1450 ShowFoto::ShowfotoFolderViewSideBar Class Reference	3463
9.1450.1 Member Function Documentation	3465
9.1450.1.1 doLoadState()	3465
9.1450.1.2 doSaveState()	3465
9.1451 ShowFoto::ShowfotoFolderViewToolTip Class Reference	3466
9.1452 ShowFoto::ShowfotoFolderViewUndo Class Reference	3467
9.1453 ShowFoto::ShowfotoInfoface Class Reference	3468
9.1453.1 Member Function Documentation	3470
9.1453.1.1 openSetupPage()	3470
9.1454 ShowFoto::ShowfotoItemInfo Class Reference	3470
9.1454.1 Member Data Documentation	3471
9.1454.1.1 size	3471
9.1455 ShowFoto::ShowfotoItemModel Class Reference	3472
9.1455.1 Member Enumeration Documentation	3475
9.1455.1.1 ShowfotoItemModelRoles	3475
9.1455.2 Member Function Documentation	3475
9.1455.2.1 addShowfotoItemInfoSynchronously()	3475

---

9.1455.2.2 allRefreshingFinished	3476
9.1455.2.3 indexForUrl()	3476
9.1455.2.4 itemInfosAboutToBeAdded	3476
9.1455.2.5 itemInfosAboutToBeRemoved	3476
9.1455.2.6 itemInfosAdded	3476
9.1455.2.7 itemInfosRemoved	3476
9.1455.2.8 readyForIncrementalRefresh	3477
9.1455.2.9 requestIncrementalRefresh()	3477
9.1455.2.10 setKeepsFileUrlCache()	3477
9.1455.2.11 setSendRemovalSignals()	3477
9.1455.2.12 showfotoItemInfo() [1/2]	3477
9.1455.2.13 showfotoItemInfo() [2/2]	3477
9.1455.2.14 showfotoItemInfosCleared()	3477
9.1455.2.15 startIncrementalRefresh()	3478
9.1456 ShowFoto::ShowfotoItemSortSettings Class Reference	3478
9.1456.1 Member Enumeration Documentation	3479
9.1456.1.1 SortOrder	3479
9.1456.2 Member Function Documentation	3479
9.1456.2.1 compare()	3479
9.1456.2.2 compareCategories()	3479
9.1456.2.3 lessThan() [1/2]	3479
9.1456.2.4 lessThan() [2/2]	3480
9.1457 ShowFoto::ShowfotoItemViewDelegate Class Reference	3481
9.1457.1 Member Function Documentation	3484
9.1457.1.1 acceptsActivation()	3484
9.1457.1.2 acceptsToolTip()	3484
9.1457.1.3 asDelegate()	3484
9.1457.1.4 gridSize()	3484
9.1457.1.5 imageInformationRect()	3485
9.1457.1.6 mouseMoved()	3485
9.1457.1.7 pixmapRect()	3485
9.1457.1.8 setDefaultViewOptions()	3485
9.1457.1.9 setSpacing()	3485
9.1457.1.10 setThumbnailSize()	3486
9.1458 ShowFoto::ShowfotoKineticScroller Class Reference	3486
9.1458.1 Detailed Description	3487
9.1459 ShowFoto::ShowfotoNormalDelegate Class Reference	3487
9.1459.1 Member Function Documentation	3491
9.1459.1.1 updateRects()	3491
9.1460 ShowFoto::ShowfotoSettings Class Reference	3492
9.1461 ShowFoto::ShowfotoSetup Class Reference	3494
9.1461.1 Member Function Documentation	3497



---

9.1461.1.1 execSinglePage() . . . . .	3497
9.1462 ShowFoto::ShowfotoSetupMetadata Class Reference . . . . .	3497
9.1463 ShowFoto::ShowfotoSetupMisc Class Reference . . . . .	3498
9.1464 ShowFoto::ShowfotoSetupPlugins Class Reference . . . . .	3499
9.1465 ShowFoto::ShowfotoSetupRaw Class Reference . . . . .	3500
9.1466 ShowFoto::ShowfotoSetupToolTip Class Reference . . . . .	3501
9.1467 ShowFoto::ShowfotoSortFilterModel Class Reference . . . . .	3502
9.1467.1 Member Function Documentation . . . . .	3504
9.1467.1.1 mapToSourceShowfotoModel() . . . . .	3504
9.1467.1.2 setDirectSourceShowfotoModel() . . . . .	3504
9.1467.1.3 showfotoFilterModel() . . . . .	3504
9.1467.1.4 showfotoItemInfosSorted() . . . . .	3504
9.1468 ShowFoto::ShowfotoStackViewFavoriteItem Class Reference . . . . .	3505
9.1468.1 Member Enumeration Documentation . . . . .	3506
9.1468.1.1 FavoriteType . . . . .	3506
9.1468.2 Member Function Documentation . . . . .	3506
9.1468.2.1 hierarchyFromParent() . . . . .	3506
9.1469 ShowFoto::ShowfotoStackViewFavoriteItemDlg Class Reference . . . . .	3507
9.1470 ShowFoto::ShowfotoStackViewFavoriteList Class Reference . . . . .	3508
9.1470.1 Member Function Documentation . . . . .	3509
9.1470.1.1 setFilter() . . . . .	3509
9.1470.1.2 signalSearchResult . . . . .	3509
9.1471 ShowFoto::ShowfotoStackViewFavorites Class Reference . . . . .	3510
9.1472 ShowFoto::ShowfotoStackViewItem Class Reference . . . . .	3511
9.1473 ShowFoto::ShowfotoStackViewList Class Reference . . . . .	3512
9.1473.1 Member Enumeration Documentation . . . . .	3513
9.1473.1.1 StackViewRole . . . . .	3513
9.1474 ShowFoto::ShowfotoStackViewSideBar Class Reference . . . . .	3514
9.1474.1 Member Function Documentation . . . . .	3516
9.1474.1.1 doLoadState() . . . . .	3516
9.1474.1.2 doSaveState() . . . . .	3516
9.1475 ShowFoto::ShowfotoStackViewToolTip Class Reference . . . . .	3517
9.1476 ShowFoto::ShowfotoThumbnailBar Class Reference . . . . .	3519
9.1476.1 Member Function Documentation . . . . .	3525
9.1476.1.1 setModelsFiltered() . . . . .	3525
9.1477 ShowFoto::ShowfotoThumbnailDelegate Class Reference . . . . .	3526
9.1477.1 Member Function Documentation . . . . .	3530
9.1477.1.1 acceptsActivation() . . . . .	3530
9.1477.1.2 setDefaultViewOptions() . . . . .	3530
9.1477.1.3 updateContentWidth() . . . . .	3530
9.1477.1.4 updateRects() . . . . .	3530
9.1478 ShowFoto::ShowfotoThumbnailModel Class Reference . . . . .	3531

9.1478.1 Constructor & Destructor Documentation . . . . .	3535
9.1478.1.1 ShowfotoThumbnailModel() . . . . .	3535
9.1478.2 Member Function Documentation . . . . .	3535
9.1478.2.1 data() . . . . .	3535
9.1478.2.2 setData() . . . . .	3535
9.1478.2.3 setEmitDataChanged() . . . . .	3535
9.1478.2.4 setPreloadThumbnails() . . . . .	3536
9.1478.2.5 setThumbnailLoadThread() . . . . .	3536
9.1478.2.6 showfotoItemInfosCleared() . . . . .	3536
<b>Index</b>	<b>3537</b>

# Chapter 1

## digikam project API reference.

digikam is an advanced open-source digital photo management application that runs on Linux, Windows, and macOS.

digikam is an advanced open-source digital photo management application that runs on Linux, Windows, and macOS.

### Author

(c) 2001-2025 digiKam team.

## 1.1 Source Code Directories

digikam is split into a number of components, each ones located to a dedicated directory. The main namespace is [Digikam](#) for the digiKam application and all sub components. A second namespace is ShowFoto for the stand alone version of digiKam image editor.

See below the complete list of directories used by the project:

SOURCE TREE-VIEW	DETAILS
. AUTHORS	List of developers and contributors to the project
. bootstrap.api	Script to build API documentation (HTML + PDF)
. bootstrap.linux	Configuration script to compile under Linux
. bootstrap.local	Configuration script to compile a local version under Linux
. bootstrap.macports	Configuration script to compile under macOS with Macports
. bootstrap.homebrew	Configuration script to compile under macOS with HomeBrew
. bootstrap.vcpkg	Configuration script to compile under Windows with VPCKG
. bootstrap.tarball	Script to build the release tarball
. build	Temporary directory created by bootstrap script to host compiled files
. ChangeLog	Note about how to list source code changes since the project origin
. CMakeLists.txt	Main Cmake script including lead compilation rules for the project
. COPYING	Link to main project license
. LICENSES	All licenses used in the project
. Mainpage.dox	API documentation main page based on Doxygen
. Messages.sh	Script to extract strings for translators
. NEWS	Notice to resume all project changes done at release time

SOURCE TREE-VIEW	DETAILS
. README.DEVEL	Read me file for developers
. README.md	First start helper documentation
. README.BUNDLES	Read me for Linux, macOS, and Windows bundles support
. build	Directory to store compiled files and binary targets
. core	All source code are hosted in this directory
. . app	Lead application component
. . . date	All date relevant views
. . . dragdrop	Drag and drop helper classes
. . . filters	Tags filter widgets
. . . items	Item management classes
. . . . delegate	Item view delegate
. . . . overlays	Item overlays
. . . . thumbbar	Item thumbbar widget
. . . . utils	Item utility classes
. . . . views	Item view classes
. . . main	Main digiKam application
. . . utils	Generic utility classes
. . . views	Views classes
. . . . preview	Item preview classes
. . . . sidebar	Left sidebar contents
. . . . stack	Stacked-view show in central place of main digiKam window
. . . . tableview	Table-view classes
. . . . utils	View utility classes
. . cmake	Extra Cmake scripts will be hosted here
. . . modules	Cmake scripts to find extra dependencies
. . . templates	Cmake template files used at configuration time
. . data	Application data files will be hosted here
. . . about	Welcome page files (HTML + CSS)
. . . colorschemes	GUI Color scheme files
. . . database	Database XML configuration files
. . . facesengine	Face detection and recognition data files
. . . filters	Image filters data files
. . . geolocation	Geolocation tool data files
. . . hotplug	Hotplug Linux integration files
. . . htmlgallery	HTML gallery tool data files
. . . icons	Application icons
. . . metadata	Metadata tool data files
. . . pics	Application pictures
. . . printcreator	Print Creator tool data files
. . . profiles	Basis open source ICC color profiles
. . . scripts	Miscs maintenance scripts
. . dplugins	All digiKam plugins will be hosted in this directory
. . . bqm	All Batch Queue Manager plugins
. . . . colors	All color adjustments plugins
. . . . convert	All file convert plugins
. . . . custom	All user-custom processing plugins
. . . . decorate	All decorate item plugins
. . . . enhance	All enhance item plugins
. . . . filters	All filter item plugins

SOURCE TREE-VIEW	DETAILS
. . . . metadata	All metadata edit plugins
. . . . transform	All transform item plugins
. . . editor	All Image Editor plugins
. . . . colors	All color adjustments plugins
. . . . decorate	All decorate item plugins
. . . . enhance	All enhance item plugins
. . . . file	All file processing plugins
. . . . filters	All filter item plugins
. . . . transform	All transform item plugins
. . . generic	All generic plugins
. . . . import	Tools to import items
. . . . metadata	Plugins to change items metadata
. . . . tools	Plugins hosted in Tools main menu
. . . . view	Plugins to display items
. . . . webservices	All plugins to import and export items to remote web-services
. . . rawimport	All Raw import plugins
. . . dimg	All DImg image loader plugins
. . libs	digiKam core sub-components (few are shared with Showfoto)
. . . album	All classes use to manage digiKam albums operations and properties
. . . database	All low level database interface is here
. . . . collection	All classes relevant of collections management
. . . . coredb	The core database interface used to host all image properties
. . . . dbjobs	All database multi-threaded jobs
. . . . engine	The low level database engine classes
. . . . haar	The similarity low level algorithms to compute image finger-prints
. . . . history	The item history classes for the database
. . . . item	The database item classes, including containers, lister, and scanner
. . . . models	The database model classes
. . . . server	The Mysql internal server
. . . . similaritydb	The similarity database
. . . . tags	The database tags management classes
. . . . thumbsdb	The thumbnails database
. . . . utils	Miscs tools and widgets used with database
. . . dialogs	Common dialogs
. . . dimg	The Qt digiKam image data container support ICC and 16 bits color depth
. . . . filters	All image filters will be hosted here. All support 16 bits color depth
. . . . . auto	Auto colors correction filters
. . . . . bcg	Brightness-Contrast-Gamma filter
. . . . . bw	Black and White image converter, including infrared filter
. . . . . cb	Colors balance filter
. . . . . curves	Colors curves filter
. . . . . decorate	Decorate filters
. . . . . film	Analog film emulation filters
. . . . . fx	Special effect filters
. . . . . greycstoration	Cimg based restoration filter
. . . . . hsl	Hue-Saturation-Lightness filter
. . . . . icc	Icc color profile filters
. . . . . imgqsort	The image quality sort algorithms
. . . . . lc	Local contrast filter (pseudo HDR)

SOURCE TREE-VIEW	DETAILS
. . . . . lens	Lens corrections filters, including Qt Lensfun interface
. . . . . levels	Color levels filter
. . . . . nr	Wavelets noise reduction filter
. . . . . redeye	Red-eyes parser and fixer
. . . . . sharp	Image sharp filter, including Unsharped-mask and Refocus
. . . . . transform	All image transformation filters
. . . . . wb	White balance filter
. . . . imagehistory	Image history interface for image container
. . . . loaders	All DImg image loaders interface
. . . metadataengine	The metadata wrapper based on Exiv2 for image and FFMpeg for video
. . . dngwriter	Qt classes to convert RAW files to DNG format
. . . . extra	DNG and XMP sdks from Adobe
. . . dplugins	All shared dplugins classes are hosted here
. . . . core	Low level classes for plugins definitions
. . . . iface	Low level classes for host interface definitions
. . . . setup	Classes to setup plugins in configuration panel
. . . . webservices	Common classes for Webservices tools
. . . . widgets	Common widget sfor plugins
. . . dtrash	digiKam trash manager full independent of desktop trash
. . . facesengine	Face detection and recognition engine + Faces database implementations
. . . . detection	Face detection modules
. . . . . opencv-dnn	Deep-learning classes based on OpenCV to detect face
. . . . . common	Face containers
. . . . . facedb	Faces database classes
. . . . preprocessing	Face pre-processing classes
. . . . . recognition	Preprocessor recognition module
. . . . . shape-predictor	Shape predictor algorithms
. . . . recognition	Face recognition modules
. . . . . opencv-dnn	Deep-learning classes based on OpenCV to recognize face
. . . fileactionmanager	Classes to connect database and metadata actions to file operations
. . . filters	Widgets to filter items by metadata properties
. . . imageproperties	All widgets used in right side-bar from all main views
. . . iojobs	Multithreaded jobs manager used with files operations
. . . jpegutils	Utilities to process JPEG files
. . . . libjpeg	JPEG loss-less transform private implementations from libjpeg
. . . kmemoryinfo	Qt backend to analyze system memory information
. . . models	Qt models used with item views
. . . notificationmanager	Multi-desktop notifications wrapper
. . . pgfutils	Qt Classes to work with PGF image format
. . . progressmanager	Multi-level operations progress widget
. . . rawengine	Qt classes to work with libraw decoder
. . . . libraw	Internal Libraw sdk
. . . settings	digiKam settings manager
. . . tags	Classes to play with tags
. . . . tagsmanager	Tags manager view
. . . template	Metadata template support
. . . threadimageio	Classes to process thumbs and preview extraction including video support
. . . threads	Classes to manage and chain threads using multi-core
. . . timeadjust	Common classes time adjustments tools

SOURCE TREE-VIEW	DETAILS
. . . transitionmngr	Frames transitions manager
. . . versionmanager	Classes to manage versioning operations
. . . video	Classes to play with video contents
. . . widgets	To host plenty of widgets used everywhere
. . . . colors	Colors relevant views
. . . . combo	Combo-box helper classes
. . . . common	Uncategorized widgets
. . . . files	File operation classes
. . . . fonts	Font management classes
. . . . graphicsview	Graphics-view implementation (model-view)
. . . . iccprofiles	ICC color profiles widgets
. . . . imagehistory	Image history widgets
. . . . itemview	Item-view implementations (model-view)
. . . . layout	Layout helper classes
. . . . mainview	Common top-level view implementations
. . . . metadata	Metadata widgets
. . . . range	Range helper classes
. . showfoto	Stand alone image editor
. . . main	Main Showfoto application
. . . setup	Showfoto Setup views
. . . thumbbar	Showfoto thumb-bar views
. . tests	Unit tests
. . utilities	digiKam utilities and advanced tools (few are shared with showfoto)
. . . advancedrename	Advance rename tool
. . . extrasupport	Extra desktop features support as Baloo search engine
. . . facemanagement	Face management classes and tools
. . . firstrun	First-run assistant to configure lead digiKam settings
. . . fuzzysearch	Similarity search tools
. . . geolocation	All geo-location tools are located here
. . . . editor	Tool to edit items geo-location
. . . . geoiface	All shared classes used by geo-location tools
. . . . geomapwrapper	Legacy helper classes for geo-location support
. . . . mapsearches	Tool to perform map searches
. . . imageeditor	The famous digiKam image editor, a lots of classes shared with Showfoto
. . . . core	Core implementation including canvas and tools interface
. . . . dialogs	Image editor dialogs
. . . . editor	The core image editors classes
. . . . main	The main digiKam image editor view, not shared with Showfoto
. . . . widgets	All common widgets
. . . import	The import tools, including USB Mass Storage and Gphoto2 support
. . . . backend	Camera backends
. . . . dialogs	Import tools dialogs
. . . . items	Import item classes
. . . . main	Import tool main view
. . . . models	Import model classes
. . . . views	Import view classes
. . . . widgets	Import common widgets
. . . lighttable	The Light-table tool to compare images side by side
. . . maintenance	The digiKam tool to maintain the database contents

SOURCE TREE-VIEW	DETAILS
. . . queuemanager	The famous Batch Queue Manager tool
. . . . main	The main BQM view
. . . . manager	The multi-core manager to run tools in background
. . . . tools	All BQM tools classed by functions
. . . . views	The BQM internal views
. . . searchwindow	The powerful advanced search tool
. . . setup	All digiKam setup panel, with few ones shared with Showfoto
. . . . album	Album configuration views
. . . . camera	Camera configuration views
. . . . collections	Collection configuration views
. . . . editor	Image Editor configuration views
. . . . metadata	Metadata configuration views
. . . slideshow	The simple slideshow tool
. po	Program translations
. project	Extra project parts
. . bundles	Bundles build scripts
. . . 3rdparty	External components required to build bundles
. . . CD	Continuous deployment configurations
. . . appimage	Linux AppImage
. . . homebrew	macOS package (HomeBrew version)
. . . macports	macOS package (Macports version)
. . . vcpkg	Windows installer
. . documents	Project documentations
. . reports	Static analyzers report scripts for Continuous Integration
. . scripts	3rdparty source code management scripts

## 1.2 External Dependencies

### 1.2.1 Dependencies To Checkout All Source Code

- Git <http://git-scm.com>

### 1.2.2 Dependencies To Process Translations Files (optional)

- Gettext <https://www.gnu.org/software/gettext> (including Msgfmt to compile po files to mo files)

### 1.2.3 Dependencies To Compile And Link Source Code

The full list of mandatory (X) and (optional) external dependencies required to compile and link digiKam source code is listed below.

Dependency	Requirement	Qt5 Version	Qt6 Version	External Links	Remarks	Notes
CMake	X	>= 3.16.0	>= 3.22.0	<a href="#">url</a>		



Dependency	Requirement	Qt5 Version	Qt6 Version	External Links	Remarks	Notes
ECM	X	>= 5.55.0	>= 5.240.0	<a href="#">url</a>	Qt6 support implemented in KDE framework >= 5.91.0	
Qt::Core	X	>= 5.14	>= 6.4	<a href="#">url</a>		
Qt::Gui	X	>= 5.14	>= 6.4	<a href="#">url</a>		
Qt::Widgets	X	>= 5.14	>= 6.4	<a href="#">url</a>		
Qt::Network	X	>= 5.14	>= 6.4	<a href="#">url</a>		
Qt::↔ NetworkAuth	X	>= 5.14	>= 6.4	<a href="#">url</a>		
Qt::Sql	X	>= 5.14	>= 6.4	<a href="#">url</a>	Including Qt::Sqlite and Qt↔::Mysql plugins	
Qt::Xml	X	>= 5.14	>= 6.4	<a href="#">url</a>		
Qt::↔ Concurrent	X	>= 5.14	>= 6.4	<a href="#">url</a>		
Qt::Print↔ Support	X	>= 5.14	>= 6.4	<a href="#">url</a>		
Qt::Svg	X	>= 5.14	>= 6.4	<a href="#">url</a>		
Qt::Web↔ Engine	X	>= 5.14	>= 6.4	<a href="#">url</a>	To render web contents (ENABLE_↔ QWEBENGINE=on)	
Qt::Xml↔ Patterns	optional	>= 5.14	—	<a href="#">url</a>	To parse and validate Xml	Used by Rajce plugin. Module removed with Qt6.
Qt::X11↔ Extras	optional	>= 5.14	—	<a href="#">url</a>	For color management support under Linux	Module removed with Qt6.
Qt::DBus	optional	>= 5.14	>= 6.4	<a href="#">url</a>	Optional: only for Linux Desktop	
Qt::OpenGL	optional	>= 5.14	>= 6.4	<a href="#">url</a>	For Presentation tool	
Qt::Open↔ GLWidgets	optional	—	>= 6.4	<a href="#">url</a>	For Presentation tool	With Qt6, OpenGL is separated in 2 modules: core and widgets.
Qt::↔ Multimedia	optional	—	>= 6.4	<a href="#">url</a>	For Presentation tool	With Qt6, OpenGL is separated in 2 modules: core and widgets.

Dependency	Requirement	Qt5 Version	Qt6 Version	External Links	Remarks	Notes
Qt::Test	optional	>= 5.14	>= 6.4	<a href="#">url</a>	To compile test codes (BUILD_↔ TESTING=on)	
Qt::Qml	optional	>= 5.14	>= 6.4	<a href="#">url</a>	To compile test codes (BUILD_↔ TESTING=on) O2 unit tests	
Qt::WebView	optional	>= 5.14	>= 6.4	<a href="#">url</a>	To compile test codes (BUILD_↔ TESTING=on) O2 unit tests	
KF::Config	X	>= 5.95.0	>= 5.240.0	<a href="#">url</a>		
KF::XmlGui	X	>= 5.95.0	>= 5.240.0	<a href="#">url</a>		
KF::l18n	X	>= 5.95.0	>= 5.240.0	<a href="#">url</a>		
KF::↔ Window↔ System	X	>= 5.95.0	>= 5.240.0	<a href="#">url</a>		
KF::Service	X	>= 5.95.0	>= 5.240.0	<a href="#">url</a>	TODO: make optional for Linux desktop (DFile↔ Operations)	
KF::Solid	X	>= 5.95.0	>= 5.240.0	<a href="#">url</a>		
KF::Core↔ Addons	X	>= 5.95.0	>= 5.240.0	<a href="#">url</a>	Needs for KAbout↔ Data and KMemory↔ Info	
KF::Notify↔ Config	optional	>= 5.95.0	>= 5.240.0	<a href="#">url</a>	For Linux desktop application notify configuration	
KF::↔ Notifications	optional	>= 5.95.0	>= 5.240.0	<a href="#">url</a>	For Linux desktop notifications integrations	
KF::↔ Thread↔ Weaver	optional	>= 5.95.0	>= 5.240.0	<a href="#">url</a>	For panorama tool	
KF::Icon↔ Themes	optional	>= 5.95.0	>= 5.240.0	<a href="#">url</a>	Optional: only for Linux Desktop (KIcon↔ Dialog)	
KF::File↔ MetaData	optional	>= 5.95.0	>= 5.240.0	<a href="#">url</a>	Plasma desktop files indexer support	(ENABLE_↔ KFILEMETADATASUPPORT=ON Disabled by default.

Dependency	Requirement	Qt5 Version	Qt6 Version	External Links	Remarks	Notes
KF::CalendarCore	optional	>= 5.95.0	>= 5.240.0	<a href="#">url</a>	For calendar tool to setup special events	
KF::KIO	optional	>= 5.95.0	>= 5.240.0	<a href="#">url</a>	Optional: only for Linux Desktop	
KF::Sonnet	optional	>= 5.95.0	>= 5.240.0	<a href="#">url</a>	To perform spell-checking in text widget (aka caption)	
KF::AkonadiContact	optional	>= 5.95.0	>= 5.240.0	<a href="#">url</a>	Plasma desktop address-book support	(ENABLE_AKONADICONTACTSUPPORT= Disabled by default.
libopencv	X	>= 4.8		<a href="#">url</a>	OpenCV 4 recommended	DNN module required for face management
libtiff	X	>= 4.0		<a href="#">url</a>	For DImg TIFF image loader	
libpng	X	>= 1.6		<a href="#">url</a>	For DImg PNG image loader	
libjpeg	X	>= 8		<a href="#">url</a>	jpeglib >= 8.0 required by RawEngine for DNG support	
libboost	X	>= 1.55.0		<a href="#">url</a>	For Versioning support	
liblcms	X	>= 2.x		<a href="#">url</a>	For Color Management support	
libexpat	X	>= 2.1.0		<a href="#">url</a>	For RAW to DNG converter	
libexiv2	X	>= 0.27.0		<a href="#">url</a>	Metadata low level management.	
libjpegxl	optional	>= 0.7		<a href="#">url</a>	For DNG-Writer and RawEngine	To decode and encode DNG files
libheif	optional	>= 1.6.0		<a href="#">url</a>	For HEIF file format support.	Library must be compiled with libde265 (read) and optionally with libx265 (write).

Dependency	Requirement	Qt5 Version	Qt6 Version	External Links	Remarks	Notes
libx265	optional	>= 2.2		<a href="#">url</a>	For HEIC encoding support	
libxml2	optional	>= 2.7.0		<a href="#">url</a>	For Html↔ Gallery tool	
libxslt	optional	>= 1.1.0		<a href="#">url</a>	For Html↔ Gallery tool	
Flex	optional	>= 2.5.0		<a href="#">url</a>	For Panorama tool	
Bison	optional	>= 2.5.0		<a href="#">url</a>	For Panorama tool	
libmesa	optional	>= 11.0		<a href="#">url</a>	For Presentation tools (Linux only)	
libksane	optional	>= 21.12.0	22.04.2	<a href="#">url</a>	Digital scanner support	
libjasper	optional	>= 1.900.1		<a href="#">url</a>	For JPEG-2000 support	
libeigen3	optional	>= 3.2		<a href="#">url</a>	For Refocus tool	See if Clapack from OpenCV can be used instead
liblensfun	optional	>= 0.2.8		<a href="#">url</a>	For Lens↔ Correction tool	
libglib2	optional	>= 2.0.0		<a href="#">url</a>	For Liquid rescale tool	
libgphoto2	optional	>= 2.5		<a href="#">url</a>	Digital camera drivers support. Need libusb-1	
libgomp	optional	>= 5.0		<a href="#">url</a>	OpenMP support for RawEngine	
libimagemagick	optional	>= 6.7.0		<a href="#">url</a>	Image↔ Magick codecs support for DImg image loader	Version >= 7.0 recommended
libffmpeg	optional	>= 5.x		<a href="#">url</a>	To play video and audio (ENABLE_↔ MEDIAPLAYER=on)	libavformat, libavutil, libavcodec used to extract video metadata. QtAVPlayer
libvaapi	optional	>= 2.4		<a href="#">url</a>	To play video and audio (ENABLE_↔ MEDIAPLAYER=on)	Intel Video support in QtAVPlayer

## 1.3 Get Source Code

### 1.3.1 Software Components

digikam project use a single git repository from GitLab to host whole source code base. The project page is given below:

<https://invent.kde.org/graphics/digikam>

The digikam handbook source code is hosted in a separate GitLab repository:

<https://invent.kde.org/documentation/digikam-doc>

## 1.4 Development Environment

If you are a developer with push access to the git repositories, it is strongly recommended to use the "kde:" prefix and let git use the read-only mirrors for pulling.

If you did not clone this repository from "kde:", do it again:

```
git config --global url.git://anongit.kde.org/.insteadof kde:
git config --global url.ssh://git@git.kde.org/.pushinsteadof kde:
git clone kde:digikam
```

See below an example of .gitconfig file working with a developer account:

```
[url "git://anongit.kde.org/"]
    insteadof = kde://

[url "git@git.kde.org:"]
    pushinsteadof = kde://

[url "ssh://git@git.kde.org/"]
    pushinsteadof = kde://

[alias]
    up = pull --rebase -v --stat
    ci = commit -a -v

[core]
    editor = mcedit

[user]
    name = my name
    email = my email

[push]
    default = tracking

[color]
    # turn on color
    diff = auto
    status = auto
    branch = auto
    interactive = auto
    ui = auto

[color "branch"]
    current = green bold
    local = green
    remote = red bold

[color "diff"]
    meta = yellow bold
    frag = magenta bold
    old = red bold
    new = green bold

[color "status"]
    added = green bold
    changed = yellow bold
    untracked = red

[color "sh"]
    branch = yellow
    [color "sh"]
```

## 1.5 Cmake Configuration Options

To configure the project with CMake, use dedicated "bootstrap" script for your platform where all available configuration options are present with default values.

There are two configuration sections : the top level and the core.

### 1.5.1 Top Level Configuration

- Packaging options:
  - **DIGIKAMSC\_COMPILE\_DIGIKAM** : Build digiKam core (default=ON).
  - **DIGIKAMSC\_COMPILE\_PO** : Build application translations files. (default=OFF).
- Developers only options:
  - **BUILD\_TESTING=ON** : Build tests code (default=ON).
  - **BUILD\_WITH\_QT6=ON** : Build with Qt6 framework, else Qt5 (default=OFF).

### 1.5.2 Core Configuration

- Extra feature support options:
  - **ENABLE\_KFILEMETADATASUPPORT** : Build digiKam with KDE files indexer support (default=OFF).
  - **ENABLE\_AKONADICONTACTSUPPORT** : Build digiKam with KDE Mail Contacts support (default=OFF).
  - **ENABLE\_GEOLOCATION** : Build digiKam with Geolocation support (default=ON).
  - **ENABLE\_MEDIAPLAYER** : Build digiKam with Media Player support (default=ON).
  - **ENABLE\_DBUS** : Build digiKam with DBUS support (default=ON).
  - **ENABLE\_APPSTYLES** : Build digiKam with support for changing the widget application style (default=OFF).
  - **ENABLE\_KIO** : Build digiKam with KIO support (default=ON).
- Database options
  - **ENABLE\_MYSQLSUPPORT** : Build digiKam with MySQL database support (default=ON).
  - **ENABLE\_INTERNALMYSQL** : Build digiKam with internal MySQL server executable (default=ON).
- Showfoto application options
  - **ENABLE\_SHOWFOTO** : Build Showfoto stand-alone image editor application (default=ON).
- Developers only options:
  - **ENABLE\_DIGIKAM\_MODELTEST** : Enable ModelTest on some models for debugging (default=OFF).
  - **ENABLE\_SANITIZERS** : Enable ASAN and UBSAN sanitizers when available (default=OFF).
  - **BUILD\_WITH\_CCACHE** : Use ccache to speed up compilations (default=OFF)

## 1.6 Setup Local Compilation and Run-Time

This section describes how to install digiKam from the git repository, while keeping a system-wide digiKam install.

This procedure is based on the configure script **bootstrap.local**

1. Set the root directory for your git install in bootstrap.local (DIGIKAM\_INSTALL\_PREFIX variable)
2. If you want a clean build directory, set CLEANROOT to 1
3. Type the following command in your terminal:

```
$ ./bootstrap.local # or "./bootstrap.local --eclipse" if you intend to use Eclipse
$ cd build
$ make
$ make install
$ KDESYCOCA="/your/root/directory/var/tmp/kde-$USER/ksycoca5" kbuildsycoca5
```

To run digikam, use the following commands:

```
$ export KDESYCOCA=/your/root/directory/var/tmp/kde-$USER/ksycoca5
$ export QT_PLUGIN_PATH=/your/root/directory/lib64/plugins:/your/root/directory/lib/plugins:$QT_PLUGIN_PATH
$ export XDG_DATA_DIRS=/your/root/directory/share:$XDG_DATA_DIRS
$ /your/root/directory/bin/digikam
```

The same applies for all binaries in /your/root/directory/bin/

If your shell is bash, you can edit your .bashrc file (in \$HOME) and add the following alias:

```
DIGIKAMROOT="/your/root/directory"
alias digikam-dev="KDESYCOCA=\$DIGIKAMROOT/var/tmp/kde-$USER/ksycoca5
XDG_DATA_DIRS=\$DIGIKAMROOT/share:\$XDG_DATA_DIRS
QT_PLUGIN_PATH=\$DIGIKAMROOT/lib64/plugins:\$DIGIKAMROOT/lib/plugins:\$QT_PLUGIN_PATH
\$DIGIKAMROOT/bin/digikam"
```

then you can start your newly installed digikam with

```
$ digikam-dev
```

## 1.7 Debug Traces At Run-Time

digiKam uses categorized logging at run-time. By default, all debug messages are printed on the console. To disable output, you can either fine-grained control by using one or more logging categories listed below.

Note: under Windows, to catch all debug messages you need to install an extra Microsoft application named DebugView available at this url: <http://technet.microsoft.com/en-us/sysinternals/bb896647>. ↩  
aspix

### 1.7.1 Logging Using an Environment Variable

You can set the environment variable **QT\_LOGGING\_RULES**. Rules are divided by semicolons.

E.g. you can start digiKam like this on the command line with thumbnails and core database messages disabled:

```
export QT_LOGGING_RULES='digiKam.thumbsdb=false;digiKam.coredb=false'
digiKam
```

## 1.7.2 Logging Categories in digiKam

All logging categories are listed in

```

/* =====
 *
 * This file is a part of digiKam project
 * https://www.digikam.org
 *
 * Date      : 2014-09-08
 * Description : digiKam debug spaces
 *
 * SPDX-FileCopyrightText: 2014      by Laurent Montel <montel at kde dot org>
 * SPDX-FileCopyrightText: 2015      by Mohamed_Anwer <m_dot_anwer at gmx dot com>
 * SPDX-FileCopyrightText: 2014-2025 by Gilles Caulier <caulier dot gilles at gmail dot com>
 *
 * SPDX-License-Identifier: GPL-2.0-or-later
 *
 * ===== */

#include "digikam_debug.h"

// Local includes

#include "digikam_config.h"

Q_LOGGING_CATEGORY(DIGIKAM_GENERAL_LOG,           "digikam.general",           QtInfoMsg)
Q_LOGGING_CATEGORY(DIGIKAM_WIDGETS_LOG,          "digikam.widgets",          QtInfoMsg)
Q_LOGGING_CATEGORY(DIGIKAM_IOJOB_LOG,            "digikam.iojob",            QtInfoMsg)
Q_LOGGING_CATEGORY(DIGIKAM_SHOWFOTO_LOG,         "digikam.showfoto",         QtInfoMsg)
Q_LOGGING_CATEGORY(DIGIKAM_WEBSERVICES_LOG,      "digikam.webservices",      QtInfoMsg)
Q_LOGGING_CATEGORY(DIGIKAM_DATABASESERVER_LOG,   "digikam.databaseserver",   QtInfoMsg)
Q_LOGGING_CATEGORY(DIGIKAM_IMPORTUI_LOG,         "digikam.import",           QtInfoMsg)
Q_LOGGING_CATEGORY(DIGIKAM_METAENGINE_LOG,       "digikam.metaengine",       QtInfoMsg)
Q_LOGGING_CATEGORY(DIGIKAM_RAWENGINE_LOG,        "digikam.rawengine",        QtInfoMsg)
Q_LOGGING_CATEGORY(DIGIKAM_FACEENGINE_LOG,       "digikam.facesengine",      QtInfoMsg)
Q_LOGGING_CATEGORY(DIGIKAM_AUTOTAGSENGINE_LOG,   "digikam.autotagsengine",    QtInfoMsg)
Q_LOGGING_CATEGORY(DIGIKAM_GEOIFACE_LOG,         "digikam.geoface",          QtInfoMsg)
Q_LOGGING_CATEGORY(DIGIKAM_GEOENGINE_LOG,        "digikam.geocore",          QtInfoMsg)
Q_LOGGING_CATEGORY(DIGIKAM_DNNMODELNGR_LOG,      "digikam.dnnmodelmanager",   QtInfoMsg)

Q_LOGGING_CATEGORY(DIGIKAM_TESTS_LOG,            "digikam.tests",            QtInfoMsg)

Q_LOGGING_CATEGORY(DIGIKAM_DPLUGIN_RAWIMPORT_LOG, "digikam.dplugin.rawimport", QtInfoMsg)
Q_LOGGING_CATEGORY(DIGIKAM_DPLUGIN_GENERIC_LOG,  "digikam.dplugin.generic",  QtInfoMsg)
Q_LOGGING_CATEGORY(DIGIKAM_DPLUGIN_EDITOR_LOG,   "digikam.dplugin.editor",   QtInfoMsg)
Q_LOGGING_CATEGORY(DIGIKAM_DPLUGIN_BQM_LOG,      "digikam.dplugin.bqm",      QtInfoMsg)
Q_LOGGING_CATEGORY(DIGIKAM_DPLUGIN_LOG,          "digikam.dplugin",          QtInfoMsg)

Q_LOGGING_CATEGORY(DIGIKAM_DATABASE_LOG,         "digikam.database",         QtInfoMsg)
Q_LOGGING_CATEGORY(DIGIKAM_DBENGINE_LOG,         "digikam.dbengine",         QtInfoMsg)
Q_LOGGING_CATEGORY(DIGIKAM_DBJOB_LOG,            "digikam.dbjob",            QtInfoMsg)
Q_LOGGING_CATEGORY(DIGIKAM_COREDB_LOG,          "digikam.coredb",           QtInfoMsg)
Q_LOGGING_CATEGORY(DIGIKAM_FACEDB_LOG,          "digikam.facedb",           QtInfoMsg)
Q_LOGGING_CATEGORY(DIGIKAM_THUMBSDB_LOG,        "digikam.thumbsdb",         QtInfoMsg)
Q_LOGGING_CATEGORY(DIGIKAM_SIMILARITYDB_LOG,     "digikam.similaritydb",     QtInfoMsg)

// NOTE: per default only warnings and more severe messages are logged for other than general category

Q_LOGGING_CATEGORY(DIGIKAM_DIMG_LOG,             "digikam.dimg",             QtInfoMsg)
Q_LOGGING_CATEGORY(DIGIKAM_DIMG_LOG_JPEG,        "digikam.dimg.jpeg",        QtWarningMsg)
Q_LOGGING_CATEGORY(DIGIKAM_DIMG_LOG_JP2K,        "digikam.dimg.jp2k",        QtWarningMsg)
Q_LOGGING_CATEGORY(DIGIKAM_DIMG_LOG_PGF,         "digikam.dimg.pgf",         QtWarningMsg)
Q_LOGGING_CATEGORY(DIGIKAM_DIMG_LOG_PNG,         "digikam.dimg.png",         QtWarningMsg)
Q_LOGGING_CATEGORY(DIGIKAM_DIMG_LOG_PPM,         "digikam.dimg.ppm",         QtWarningMsg)
Q_LOGGING_CATEGORY(DIGIKAM_DIMG_LOG_TIFF,        "digikam.dimg.tiff",        QtWarningMsg)
Q_LOGGING_CATEGORY(DIGIKAM_DIMG_LOG_RAW,         "digikam.dimg.raw",         QtWarningMsg)
Q_LOGGING_CATEGORY(DIGIKAM_DIMG_LOG_QIMAGE,     "digikam.dimg.qimage",     QtWarningMsg)
Q_LOGGING_CATEGORY(DIGIKAM_DIMG_LOG_HEIF,       "digikam.dimg.heif",       QtWarningMsg)
Q_LOGGING_CATEGORY(DIGIKAM_DIMG_LOG_MAGICK,      "digikam.dimg.magick",      QtWarningMsg)

Q_LOGGING_CATEGORY(DIGIKAM_MEDIASRV_LOG,         "digikam.mediaserver",      QtInfoMsg)
Q_LOGGING_CATEGORY(DIGIKAM_MEDIASRV_LOG_INFO,    "digikam.mediaserver.info", QtInfoMsg)
Q_LOGGING_CATEGORY(DIGIKAM_MEDIASRV_LOG_DEBUG,   "digikam.mediaserver.debug", QtInfoMsg)
Q_LOGGING_CATEGORY(DIGIKAM_MEDIASRV_LOG_WARN,    "digikam.mediaserver.warn", QtWarningMsg)
Q_LOGGING_CATEGORY(DIGIKAM_MEDIASRV_LOG_CRITICAL, "digikam.mediaserver.critical", QtWarningMsg)
Q_LOGGING_CATEGORY(DIGIKAM_MEDIASRV_LOG_FATAL,   "digikam.mediaserver.fatal", QtCriticalMsg)

void digikamSetDebugFilterRules(bool on)
{
    if (on)
    {
        QLoggingCategory::setFilterRules(QLatin1String("digikam.*=true\n"
            "digikam.dimg.jpeg=false\n"
            "digikam.dimg.jp2k=false\n"
            "digikam.dimg.pgf=false\n"
            "digikam.dimg.png=false\n"
            "digikam.dimg.ppm=false\n"
            "digikam.dimg.tiff=false\n"
            "digikam.dimg.raw=false\n"
            "digikam.dimg.qimage=false\n"
            "digikam.dimg.heif=false\n"
            "digikam.dimg.magick=false\n"
            "digikam.geocore=false"));
        // to much verbose
    }
    // at the console
}

```



```
}
```

source code.

### 1.7.3 Further Reading

For more details see the Qt framework documentation about logging categories available at this url: <https://doc.qt.io/qt-5/qloggingcategory.html#details>

## 1.8 Cmake compilation rules

### 1.8.1 Introduction

The whole project is written mostly in C++/Qt and the Cmake framework is used to compile under Linux, macOS, and Windows. The Cmake rules have been configured to reduce the linking overhead and improve CPU utilization with modular design.

Independent Cmake configuration is presents in following folders:

- root source dir
- core
- doc

The Cmake rules will build the following targets:

- digikamcore shared lib
- digikamdatabase shared lib
- digikamgui shared lib
- digikam executable
- showfoto executable
- plugin shared libraries (dplugins)
- various test executables - if testing is enabled
- various unit-tests - if testing is enabled

Each of them depend on various sources which must be compiled before. A complete description of source code direction is given to the sourcedirs section.

## 1.8.2 CMake Implementation Details

### 1.8.2.1 Include Directories

Local include directories are all managed by this snippet of code:

```
set(DK_INCLUDES_ALL "")
HEADER_DIRECTORIES(DK_LOCAL_INCLUDES_RAW)
```

The libjpeg- folders are all included, so we need to delete them all and include the correct one only:

```
# This macro will set all paths which do not contain libjpeg-
# We will add later the directory we need
```

```
FOREACH(var ${DK_LOCAL_INCLUDES_RAW})
    STRING(REGEX MATCH "libjpeg-" item ${var})
    IF(item STREQUAL "")
        LIST(APPEND DK_LOCAL_INCLUDES ${var})
    ENDF (item)
ENDFOREACH(var)

set(DK_LOCAL_INCLUDES ${DK_LOCAL_INCLUDES}
    libs/jpegutils/${DIGIKAM_LIBJPEG_DIR})

include_directories(${DK_LOCAL_INCLUDES})
```

There is no need for manual intervention to add new includes, even if you add a new folder, just keep in mind to use:

```
#include "tagmnglistitem.h"
```

instead of :

```
#include "models/tagmnglistitem.h"
```

### 1.8.2.2 Shared Libraries

To avoid linking overhead and make a better use of sources there are some dynamic libs as these one:

- digikamcore : core components used by almost all executables as digiKam and Showfoto.
- digikamdatabase : database components, also used together with digikamcore but only for digiKam

Please add sources to digikam core or digikam database only if they don't depend on any big component from digikam main executable. These two shared libs must be kept small because they link in a lot of places

### 1.8.2.3 Static Libraries

Currently cmake configuration features a lots of shared libraries as:

- metadataedit
- geolocationedit
- digikamfaceengine

This libraries are linked in digikam main executable and some tests tools.

Avoid making static libraries if possible, and use OBJECT libraries instead. Only make STATIC libraries which does not depend on other digikam code. Also make sure you put the PRIVATE parameter when setting the target\_link\_libraries.

```
target_link_libraries(digikamcore
    PRIVATE
    Qt{QT_VERSION_MAJOR}::Core
    Qt{QT_VERSION_MAJOR}::Gui
    Qt{QT_VERSION_MAJOR}::Widgets
)
```

### 1.8.2.4 Object Libraries

While static libraries are still collection of objects, CMake offer a better approach by allowing to specify an OBJECT library:

```
set(libslideshow_SRCS
    slidetoolbar.cpp
    slideosd.cpp
    slideproperties.cpp
    slideimage.cpp
    slideerror.cpp
    slideend.cpp
    slideshow.cpp
    slidehelp.cpp
    slideshowsettings.cpp
)

add_library(slideshow_src OBJECT ${libslideshow_SRCS})
```

OBJECT library is a cmake internal implementation feature and allow to easily manage sources. Here is an example of how to make a shared lib using OBJECT libraries:

```
add_library(digikamcore
    SHARED
    ${TARGET_OBJECTS:slideshow_src} # the lib we made few lines above
    ${TARGET_OBJECTS:digikamdatabasecore_src}
    ${TARGET_OBJECTS:dimg_src}
    ....
)
```

## 1.9 Contribute To The Code

This section's purpose is to guide contributors and developers to help on the digiKam project.

### 1.9.1 Starting With Open-Source

Before to contribute to digiKam project, please take a look to this link which provide 10 golden rules for starting with open source project:

[http://schlitt.info/opensource/blog/0541\\_10\\_golden\\_rules\\_for\\_starting\\_with\\_open\\_source.html](http://schlitt.info/opensource/blog/0541_10_golden_rules_for_starting_with_open_source.html)

### 1.9.2 Source Code Formatting

Adhere to this style guide strictly while adding new code to digiKam or working on existing code.

#### 1.9.2.1 Indentation length

Indent with 4 spaces exactly.

For example:

```
void function()
{
    ....int a;
    ....for (int i = 0 ; i < 10 ; ++i)
    ....{
    .....a = i;
```

Emacs by default will indent to 4 spaces vim users add this to you .vimrc set tabstop=4

### 1.9.2.2 Tabs vs Spaces

Absolutely no tabs. Use a sensible editor which will convert tabs to spaces. This will reduce unnecessary changes in your git commits.

Emacs by default will convert tab to spaces. For vim users, add this to your .vimrc set expandtab

### 1.9.2.3 Line length

Line length should never exceed 80 chars (unless really necessary - these cases are rare). Having long lines greatly reduces readability of code

### 1.9.2.4 Bracketing

In all cases, {} brackets should start on a newline and should be aligned with previous line (follow the indentation spaces). For example.

```
class A
{ //new line
...

for (int i = 0 ; i < 10 ; ++i)
{ //new line

if (a == foobar)
{ //new line
...
}
else
{ // new line
..
}
```

### 1.9.2.5 Positioning of Access modifiers

public, private, protected, public slots, ... should be aligned to the beginning of the line with no margin

```
class A
{
public: // aligned to left
...
private Q_SLOTS: // aligned to left
```

Follow a consistent order in defining these attributes. The recommended order is public, protected (functions), private (functions), signals, public slots, protected slots, private slots, private (variables)

## 1.9.3 Class, file and Variable names

### 1.9.3.1 Class and filenames

- filenames should always be in lower-case
- class names should match the filenames. Capitalize the first letter and other letters logically to improve readability

### 1.9.3.2 Protected Member variables

- protected member variable names should always be of the form `m_varName`.
- Capitalize logically so that it becomes easy to read it. Do not capitalize the first letter after `_` (Use `m_varName` not `m_VarName`)
- variable names should be indicative of their functionality and also of the type they belong too if they are instances of qt widgets. For example, `QCheckBox* m_autoRotateCheckBox`;

### 1.9.3.3 Non-Member variables

- non-member variables should follow the same naming convention as the member variables, except for the leading `m_`

### 1.9.3.4 Private Member variables

- private member variables must be stored in a `d` private container to reduce compilation time and improve binary compatibility between digiKam components. See more information how to use a 'd' private class at this url:

[https://community.kde.org/Policies/Library\\_Code\\_Policy](https://community.kde.org/Policies/Library_Code_Policy)

## 1.9.4 Comments and Whitespace

Use whitespaces liberally to improve readability. Add blank lines between logical sections of the code.

Comment as much as possible. Position comments at the beginning of the section/line you want to comment, NEVER at the end of the line, excepted for special cases for ex to describe enum values.

```
// put your comments here
a = (b == foobar) ? 1 : -1;

a = (b == foobar) ? 1 : -1; // you are asking for trouble by putting comments here
```

## 1.9.5 Header Files

- Add copyright to top of every file. Use the same header than others digiKam source code.
- Add double inclusion protection

```
#pragma once

class AnotherNiceClass
{
...
}
```

- Use forward declarations as much as possible.

```
class QFileInfo;

class A
{
...QFileInfo* m_fileInfo = nullptr;
```

## 1.9.6 Automatic source code formatting

The above coding style guidelines can be automatically applied with `astyle` (<http://astyle.sourceforge.net/>).

Run it in the directory where the files are located that should be formatted.

To apply the coding guidelines with `astyle` is to use the `fileformatter.py` script in `project/scripts` directory. This script will also clean up the source tree and remove backup files that had been created by `astyle`, if the appropriate command line argument is given.

To handle the command easier, create a bash function in `~/bashrc`, e.g.

```
dkfrmcode()
{
    astyle --style=allman \
          --indent=spaces=4 \
          --convert-tabs \
          --indent-switches \
          --break-blocks \
          --break-closing-brackets \
          --pad-header \
          --align-pointer=type \
          --indent-coll-comments \
          --add-brackets \
          --min-conditional-indent=0 \
          `find $1 -type f -name '*.cpp'` `find $1 -type f -name '*.c'` `find $1 -type f -name '*.h'`
}

```

You can pass a parameter to the function, in this case the first parameter is the directory, where files should be formatted.

Examples:

1. Run `astyle` in the current directory  
`$> dkfrmcode`
2. Run `astyle` in a different directory  
`$> dkfrmcode /home/user/code/git/digikam/`

## 1.9.7 General recommendations

Please take a look into this contrib page tips before to write code/patches for digiKam project : <http://techbase.kde.org/Contribute>

Use the same `.cpp/.h` header than the rest of digiKam project.

Use a decent editor which does auto-indentation/syntax-highlighting for you, as Kate or QtCreator

There are excellent initializer scripts in the `kdesdk` package for `xemacs` and `vim` which can substantially increase your productivity.

Just to give a taste of what i can do with `emacs` (and `kdesdk`):

automatically insert copyright (and `ifdefs`) in new files. insertion of class function definitions for declared class functions in header with one keystroke switch between header and declaration files with one keystroke go to corresponding definition/declaration with one keystroke tab completion of variable/function names already declared.

## 1.9.8 GDB Backtrace

If you found a context to crash digiKam, you can provide a backtrace using GDB debugger. digiKam need to be compiled with all debug info else the backtrace will not suitable. There is a configure option for that:

```
$> cmake . -DCMAKE_BUILD_TYPE=debugfull
$> make
$> su
$> make install/fast
```

To make a backtrace with GDB use following command:

```
$ gdb digikam
> catch throw
> run
> ...
> _crash here_
> ...
> bt
> _the backtrace is here_
> quit
```

Post this backtrace at the right place (Bugzilla or development mailing list) for investigation by developers.

For Windows users, take a look on this tutorial :

[http://techbase.kde.org/Development/Tutorials/Debugging/Debugging\\_on\\_MS\\_Windows](http://techbase.kde.org/Development/Tutorials/Debugging/Debugging_on_MS_Windows)

## 1.9.9 Memory Leak

To check any memory leak problem in digiKam, valgrind is your friend ( <http://valgrind.org>) Try this command line to use with valgrind :

```
valgrind --tool=memcheck --leak-check=full --error-limit=no --suppressions=project/reports/digikam.supp digikam
```

NOTE: digikam.supp file is available in digikam/project sub-folder.

## 1.9.10 Profiling With Cachegrind

Valgrind also includes a tool to find out in which parts of your code time is spent.

```
valgrind --tool=callgrind digikam
```

Profiling can be disabled at startup to limit the output to the code you are interested in. Start with

```
valgrind --tool=callgrind --instr-atstart=no digikam
```

and prepare the situation you want to profile. Then, in another console, start profiling with "callgrind\_control -i on" and, after the situation has passed, request a profile dump with "callgrind\_control -d". The resulting callgrind.out files need to be viewed with the kcachegrind program, e.g.:

```
kcachegrind callgrind.out.16693.1
```

## 1.9.11 Unit Testing / Automated Testing

Unit Testing is great way to ensure that software units (in OOP this normally means classes) work as expected. Wikipedia gives a good introduction to Unit Testing:

[http://en.wikipedia.org/wiki/Unit\\_testing](http://en.wikipedia.org/wiki/Unit_testing)

It is also worth to follow most of QTest API rules with digiKam:

<http://doc.qt.io/qt-5/qtest-tutorial.html>

The digiKam test suite is located under tests and will be compiled if BUILD\_TESTING is turned ON at cmake configuration time. After compiling the source code the tests can be executed via

```
make test
```

The console output while running the tests is stored in Testing/Temporary/LastTest.log in the CMake binary dir.

All tests are simple binaries that can be executed separately if needed.

### 1.9.12 Checking For Corrupt Qt Signal Slot Connection

Use this alias for running digikam:

```
alias digikamdbg="digikam 2>&1 | tee - /tmp/digikam.out; echo -e \"\n\nPossible connection errors:\n\n\";
cat /tmp/digikam.out | grep -A2 'Object::connect'"
```

It will print a list of connection warnings after terminating the program. Moreover the complete console log of the last session is stored in /tmp/digikam.out.

### 1.9.13 Finding Duplicated Code

Code duplication should be avoided as bugs have to be fixed for every piece of duplicated code. The current duplication can be analyzed eg. with Simian: <http://www.redhillconsulting.com.au/products/simian/>

In the digikam checkout directory run:

```
java -jar simian.jar `find . -regex '.*\.(cpp|\.h)' | grep -v 3rdparty`
```

This prints out a list of duplicated code segments.

### 1.9.14 API Documentation Validation, User Documentation Validation, Source Code Checking

The following site check on a daily basis for the a.m. errors: [www.englishbreakfastnetwork.org/krazy/](http://www.englishbreakfastnetwork.org/krazy/)

It can be very useful, in particular before major releases. Don't trust it blindly! Sometimes they propose too advanced modifications that are no compatible with the prevailing include files.

### 1.9.15 Usability Issues

OpenUsability project has define default menu structure and keyboard shortcuts:

[http://wiki.openusability.org/guidelines/index.php/Appendices:Keyboard\\_Shortcuts](http://wiki.openusability.org/guidelines/index.php/Appendices:Keyboard_Shortcuts)

### 1.9.16 Generate API Documentation

To generate API documentation, you need to install:

- Doxygen program ( <http://www.doxygen.org>)
- Dot program ( <http://www.graphviz.org>)
- TeXLive package full version to generate the API doc in PDF ( <https://tug.org/texlive/>)

Under Ubuntu run:

```
sudo apt install doxygen graphviz texlive-full
```

After cmake generated a Makefile you can call 'make doc'. A new subfolder named 'html' will be created. Warning, this can take a while.

For finding documentation errors, doxygen generates a warning log file at the cmake binary dir called 'doxygen-warn.log'.



### 1.9.17 Speed Up The Code-Compile-Test Cycle

Assuming you have setup your environment in `~/bashrc` as is suggested for development, you can add something like this to your `~/bashrc`:

```
function digikam_start
{
LD_LIBRARY_PATH=${KDE_BUILD}/extragear/graphics/lib:${LD_LIBRARY_PATH}
  ${KDE_BUILD}/extragear/graphics/digikam/digikam/digikam
}

function digikam_start_gdb
{
LD_LIBRARY_PATH=${KDE_BUILD}/extragear/graphics/lib:${LD_LIBRARY_PATH} gdb
  ${KDE_BUILD}/extragear/graphics/digikam/digikam/digikam
}
```

This allows you to run digikam after compiling without the need of a "make install", even if you changed code in the libraries.

### 1.9.18 Working With Branches From Git Repository

Example to create a local 'dplugins' development branch based on master:

```
git checkout master
git checkout -b development/dplugins
```

Example to delete the local 'dplugins' development branch:

```
git checkout master
git branch -d development/dplugins
```

Example to create a remote 'dplugins' development branch from the local branch:

```
git push -u origin development/dplugins
```

Example to delete the remote 'dplugins' development branch:

```
git push origin :development/dplugins
```

### 1.9.19 Sync a Branch With Master From Git Repository

It typical to use a dedicated development branch in Git to keep the master code stable for production. To synchronize branches with master, use these commands in your local branch checkout:

```
$>git checkout master
$>git pull --rebase
$>git checkout -b MY_DEVEL_BRANCH GIT_REMOTE_PATH
  Branch 'MY_DEVEL_BRANCH' set up to track remote branch path 'GIT_REMOTE_PATH' from origin.
  To list GIT_REMOTE_PATH, use 'git branch -a' command
  Switched to a new branch 'MY_DEVEL_BRANCH'
$>git merge master
  Merge made by the 'recursive' strategy.
  ...
$>git push
  ...
```

The first 2 lines make sure that your local master repository is up to date. The 3rd line creates the local development branch "MY\_DEVEL\_BRANCH". If you have already created this branch, just run "git checkout MY\_DEVEL\_BRANCH". Merging between master and "MY\_DEVEL\_BRANCH" branch is done in the 4th line. Git might ask you to resolve conflicts here. When it's done, it will ask you to provide a commit message. Finally you push your merge into the remote repository.



## Chapter 2

# CODE\_OF\_CONDUCT

The digiKam project is part of the KDE community. The KDE Code of Conduct applies to the digiKam community as well:

<https://www.kde.org/code-of-conduct/>

In case of problems within the digiKam or wider KDE community, please contact the KDE e.V. Community Working Group:

<https://ev.kde.org/workinggroups/cwg.php>



## Chapter 3

# COMMIT POLICY

Everybody is welcome to committing small fixes and one-liners without prior notification to the maintainer, provided that the following rules are followed:

1. Please keep your commits as small and as atomic as possible.
2. Do not push both formatting and code changes in the same commit.
3. Do not fix coding style and code issues in the same commit.

For larger commits, please use Gitlab functionalities or send a patch to [bugzilla](#). A rule of thumb to check whether your commit is a major commit is if it affects more than 5 lines of code.

Break down larger fixes into smaller commits. Even if you push the commits with one "git push", git preserves your commit info.

For the todo list, see bugzilla for details which is the complete story of the project.

### 3.1 CODING STYLE

See the coding style detailed on [developer documentation](#) [available at this url](#)

### 3.2 DOCUMENTATION

See The API documentation generated with Doxygen. You will find all information about the dependencies, the configuration options, the rules to compile and install, and all pointers to contribute to the project as a developer.

Install Doxygen and Graphviz, and then run make doc from build directory. You can also run doxygen command line tool directly from this folder or consult the Mainpage.dox file.

[Online version archive](#)

### 3.3 LICENSING

See the guideline of licenses policy:

[SPDX](#) [KDE](#)

License files are hosted in LICENSE/ directory.

Usual usages:

BSD-3 : cmake, sh, rb, py, pl. GPL2+/LGPL2+ : cpp, c, h, xsl, conf, nsh, nsl. CC0-1 : yml. MIT : css, js.



# Chapter 4

## README

digiKam - Professional Photo Management with the Power of Open Source

CI Job	Status
Gitlab Builds	
Coverity Scan	

If you are reading this on Github, be aware that this is just a mirror. Our real code repository [is located here](#)

Developers, if you want to contribute, see the online [API documentation here](#)

NOTE: this project support Qt5 and Qt6 frameworks.

### 4.1 About

digiKam is an advanced open-source digital photo management application that runs on Linux, Windows, and MacOS. The application provides a comprehensive set of tools for importing, managing, editing, and sharing photos and RAW files.

[!\[\] \(https://c1.staticflickr.com/5/4216/35354951072\\_a034561b5e\\_c.jpg "Albums View and Image Editor"\)](https://c1.staticflickr.com/5/4216/35354951072_a034561b5e_c.jpg)

You can use digiKam's import capabilities to easily transfer photos, RAW files, and videos directly from your camera and external storage devices (SD cards, USB disks, etc.). The application allows you to configure import settings and rules that process and organize imported items on-the-fly.

[!\[\] \(https://c1.staticflickr.com/1/703/32558229094\\_3d7ec01d3a\\_c.jpg "Map View displaying rated items and Batch Queue Manager in action"\)](https://c1.staticflickr.com/1/703/32558229094_3d7ec01d3a_c.jpg)

digiKam organizes photos, RAW files, and videos into albums. But the application also features powerful tagging tools that allow you to assign tags, ratings, and labels to photos and raw files. You can then use filtering functionality to quickly find items that match specific criteria.

[!\[\] \(https://c2.staticflickr.com/4/3726/32557269024\\_ae870b0466\\_c.jpg "Search items by date range and Geolocation editor"\)](https://c2.staticflickr.com/4/3726/32557269024_ae870b0466_c.jpg)

In addition to filtering functionality, digiKam features powerful searching capabilities that let you search the photo library by a wide range of criteria. You can search photos by tags, labels, rating, data, location, and even specific EXIF, IPTC, or XMP metadata.

[!\[\] \(https://c1.staticflickr.com/1/306/32217007615\\_db6f9d116a\\_c.jpg](https://c1.staticflickr.com/1/306/32217007615_db6f9d116a_c.jpg) "Search by Tags with preview mode and Metadata Editor in action")

You can also combine several criteria for more advanced searches. digiKam rely on Exiv2 library to handle metadata tag contents from files to populate the photo library.

[!\[\] \(https://c1.staticflickr.com/5/4795/40743725771\\_0b69dca743\\_c.jpg](https://c1.staticflickr.com/5/4795/40743725771_0b69dca743_c.jpg) "Advanced search tool and video file result played as preview")

digiKam can handle RAW files, and the application uses the excellent LibRaw library for decoding raw files. The library is actively maintained and regularly updated to include support for the latest camera models.

[!\[\] \(https://c1.staticflickr.com/1/300/31407487553\\_a14abd0418\\_c.jpg](https://c1.staticflickr.com/1/300/31407487553_a14abd0418_c.jpg) "Find by items similarity and Light Table in cation to compare side by side")

The application provides a comprehensive set of editing tools. This includes basic tools for adjusting colors, cropping, and sharpening as well as advanced tools for, curves adjustment, panorama stitching, and much more. A special tool based on Lensfun library permit to apply lens corrections automatically on images.

[!\[\] \(https://c1.staticflickr.com/5/4649/40430534662\\_097b46a270\\_c.jpg](https://c1.staticflickr.com/5/4649/40430534662_097b46a270_c.jpg) "Panorama tool stiching photo")

Extended functionality in digiKam is implemented via a set of tools, dedicated especially to import and export contents to remote web-services.

digiKam is based in part on the work of the Independent JPEG Group.

## 4.2 Authors

See [AUTHORS](AUTHORS) file for details.

## 4.3 Related URLs

- [digiKam project web site](#)
- [digiKam handbook git repository](#)
- [digiKam web site git repository](#)
- [digiKam unit-test data git repository](#)

## 4.4 Contact

If you have questions, comments, and suggestions, write an email to:

[digikam-users@kde.org](mailto:digikam-users@kde.org)

If you want to contribute to digiKam developments write an email to:

[digikam-devel@kde.org](mailto:digikam-devel@kde.org)

IRC channel from irc.libera.chat server: #digikam (or use [web chat](#))



## 4.5 Bug reports

IMPORTANT: the bug reports and wishlist entries are hosted by the Bugzilla system which can be reached from the standard Help menu of digiKam. A mail will automatically be sent to the digiKam development mailing list. There is no need to contact directly the digiKam mailing list for a bug report or a devel wish.

The current bugs and devel wishes reported to the bugzilla servers can be seen at this url:

- [digiKam](#)

Extra Bugzilla servers for shared libs used by digiKam :

- [LibRaw library](#)
- [Lensfun library](#)
- [GPhoto2 library](#)
- [Exiv2 library](#)

## 4.6 Compilation and Installation

From the **developer documentation** [available at this url](#), see the sections:

- **External Dependencies**
- **Get Source Code**
- **Development Environment**
- **Cmake Configuration Options**
- **Setup Local Compilation and Run-Time**

## 4.7 Donate Money

If you love digiKam, you can help developers to buy new photo devices to test and implement new features. Thanks in advance for your generous donations.

For more information, look [at this url](#)



# Chapter 5

## Namespace Index

### 5.1 Namespace List

Here is a list of all documented namespaces with brief descriptions:

- [Digikam](#) NOTE: This is because of the [CollectionManager](#) private slot . . . . . 95
- [Digikam::Matrix](#) If the picture is displayed according to the exif orientation tag, the user will request rotating operations relative to what he sees, and that is the picture rotated according to the EXIF tag . . . 145



# Chapter 6

## Hierarchical Index

### 6.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

A	
Digikam::ParallelAdapter< A > . . . . .	2615
Digikam::AlbumPointer< Digikam::SAlbum > . . . . .	317
Digikam::AlbumPointer< Digikam::TAlbum > . . . . .	317
boost::default_bfs_visitor	
Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch::BreadthFirstSearchVisitor . . .	1691
boost::default_dfs_visitor	
Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch::DepthFirstSearchVisitor . . .	1693
Digikam::AbstractAlbumTreeView::ContextMenuElement . . . . .	162
Digikam::AbstractMarkerTiler::ClickInfo . . . . .	199
Digikam::AbstractMarkerTiler::NonEmptyIterator . . . . .	199
Digikam::AbstractMarkerTiler::Tile . . . . .	200
Digikam::ActionData . . . . .	213
Digikam::Album . . . . .	254
Digikam::DAlbum . . . . .	720
Digikam::PAlbum . . . . .	2608
Digikam::SAlbum . . . . .	2799
Digikam::TAlbum . . . . .	3223
Digikam::AlbumChangeset . . . . .	262
Digikam::AlbumCopyMoveHint . . . . .	263
Digikam::AlbumInfo . . . . .	280
Digikam::AlbumIterator . . . . .	280
Digikam::AlbumPointer< T > . . . . .	317
Digikam::AlbumRootChangeset . . . . .	320
Digikam::AlbumRootInfo . . . . .	320
Digikam::AlbumShortInfo . . . . .	347
Digikam::AlbumSimplified . . . . .	347
Digikam::AntiVignettingContainer . . . . .	371
Digikam::AssignedBatchTools . . . . .	386
Digikam::AutoTagsAssign . . . . .	419
Digikam::AutoTagsScanSettings . . . . .	425
Digikam::BalooInfo . . . . .	452
Digikam::BatchToolSet . . . . .	463
Digikam::BCGContainer . . . . .	465
Digikam::BdEngineBackend::QueryState . . . . .	479

Digikam::BorderContainer	504
Digikam::BWSepiaContainer	515
Digikam::CameraMessageBox	534
Digikam::CameraNameHelper	534
Digikam::CameraType	540
Digikam::CamItemInfo	540
Digikam::CamItemSortSettings	542
Digikam::CaptionValues	552
Digikam::CBContainer	558
Digikam::ChoiceSearchModel::Entry	580
Digikam::CMat	584
Digikam::CollectionImageChangeset	584
Digikam::CollectionLocation	586
Digikam::CollectionScannerHintContainer	602
Digikam::CollectionScannerObserver	603
Digikam::InitializationObserver	1975
Digikam::ScanController	2807
Digikam::ColorFXContainer	604
Digikam::CommentInfo	618
Digikam::ContentAwareContainer	622
Digikam::CopyrightInfo	639
Digikam::CoreDB	640
Digikam::CoreDbAccess	678
Digikam::CoreDbAccessUnlock	681
Digikam::CoreDbDownloadHistory	687
Digikam::CoreDbNameFilter	688
Digikam::CoreDbOperationGroup	688
Digikam::CoreDbPrivilegesChecker	689
Digikam::CoreDbSchemaUpdater	689
Digikam::CoreDbTransaction	689
Digikam::CurvesContainer	700
Digikam::DAAlbumInfo	723
Digikam::DatabaseFields::DatabaseFieldsEnumIterator< FieldName >	725
Digikam::DatabaseFields::DatabaseFieldsEnumIteratorSetOnly< FieldName >	726
Digikam::DatabaseFields::FieldMetaInfo< FieldName >	726
Digikam::DatabaseFields::Set	728
Digikam::DatabaseServerError	737
Digikam::DateFormat	760
Digikam::DbEngineAccess	779
Digikam::DbEngineAction	779
Digikam::DbEngineActionElement	779
Digikam::DbEngineActionType	779
Digikam::DbEngineConfig	780
Digikam::DbEngineConfigSettings	780
Digikam::DbEngineConfigSettingsLoader	781
Digikam::DbEngineErrorAnswer	782
Digikam::DbEngineLocking	785
Digikam::DbEngineParameters	785
Digikam::DBJobInfo	800
Digikam::AlbumsDBJobInfo	321
Digikam::DatesDBJobInfo	765
Digikam::GPSDBJobInfo	1641
Digikam::SearchesDBJobInfo	2818
Digikam::TagsDBJobInfo	3191
Digikam::DbKeysCollection	806
Digikam::CommonKeys	619
Digikam::MetadataKeys	2446
Digikam::PositionKeys	2643

Digikam::DColor	833
Digikam::DColorComposer	835
Digikam::DeltaTime	925
Digikam::DFileOperations	938
Digikam::DImageHistory	976
Digikam::DImageHistory::Entry	979
Digikam::DImg	979
Digikam::DImgBuiltinFilter	995
Digikam::DImgFilterGenerator	1002
Digikam::BasicDImgFilterGenerator< T >	455
Digikam::DImgFilterManager	1004
Digikam::DImgLoader	1006
Digikam::DImgLoaderObserver	1009
Digikam::lccTransformFilter	1784
Digikam::LoadingTask	2368
Digikam::SharedLoadingTask	2979
Digikam::PreviewLoadingTask	2647
Digikam::ThumbnailLoadingTask	3256
Digikam::SavingTask	2805
Digikam::DisjointMetadataDataFields	1047
Digikam::DItemInfo	1057
Digikam::DMessageBox	1071
Digikam::DMetadataSettingsContainer	1096
Digikam::DNGWriter	1115
Digikam::DNNBaseDetectorModel	1117
Digikam::DNNResnetDetector	1142
Digikam::DNNYoloDetector	1147
Digikam::DNNFaceDetectorBase	1119
Digikam::DNNFaceDetectorSSD	1121
Digikam::DNNFaceDetectorYOLO	1123
Digikam::DNNFaceDetectorYuNet	1125
Digikam::DNNFaceExtractorBase	1127
Digikam::DNNOpenFaceExtractor	1139
Digikam::DNNSFaceExtractor	1144
Digikam::DNNModelBase	1129
Digikam::DNNModelConfig	1130
Digikam::DNNModelNet	1134
Digikam::DNNModelSFace	1135
Digikam::DNNModelYuNet	1137
Digikam::DNNModelInfoContainer	1131
Digikam::DOnlineTranslatorOption	1179
Digikam::DownloadInfo	1185
Digikam::DownloadSettings	1185
Digikam::DPixelsAliasFilter	1186
Digikam::DPluginAuthor	1198
Digikam::DragDropModelImplementation	1247
Digikam::ImportItemModel	1895
Digikam::ImportThumbnailModel	1957
Digikam::ItemHistoryGraphModel	2105
Digikam::ItemModel	2160
Digikam::ItemThumbnailModel	2248
Digikam::ItemAlbumModel	2002
Digikam::ItemListModel	2147
ShowFoto::ShowfotoItemModel	3472
ShowFoto::ShowfotoThumbnailModel	3531
Digikam::DragDropViewImplementation	1249

Digikam::ItemViewCategorized	2256
Digikam::ImportCategorizedView	1844
Digikam::ImportIconView	1887
Digikam::ImportThumbnailBar	1944
Digikam::ItemCategorizedView	2010
Digikam::DigikamItemView	968
Digikam::ItemThumbnailBar	2234
Digikam::LightTableThumbBar	2337
ShowFoto::ShowfotoCategorizedView	3427
ShowFoto::ShowfotoThumbnailBar	3519
Digikam::TableViewTreeView	3114
Digikam::VersionsTreeView	3365
Digikam::DRawDecoderSettings	1258
Digikam::DRawDecoding	1266
Digikam::DRawInfo	1267
Digikam::DServiceInfo	1277
Digikam::DServiceMenu	1278
Digikam::DToolTipStyleSheet	1292
Digikam::DTrash	1293
Digikam::DTrashItemInfo	1294
Digikam::DWItemDelegatePool	1310
Digikam::DWItemDelegatePoolPrivate	1311
Digikam::EffectMngr	1351
Digikam::Ellipsoid	1352
Digikam::ExifToolProcess::Result	1392
Digikam::ExposureSettingsContainer	1399
Digikam::FaceDb	1403
Digikam::FaceDbAccess	1405
Digikam::FaceDbAccessUnlock	1406
Digikam::FaceDbOperationGroup	1411
Digikam::FaceDbSchemaUpdater	1412
Digikam::FaceDetector	1412
Digikam::FaceItemRetriever	1420
Digikam::FacePipelinePackage	1449
Digikam::FacePipelineExtendedPackage	1443
Digikam::FacePreprocessor	1464
Digikam::RecognitionPreprocessor	2741
Digikam::FaceScanSettings	1478
Digikam::FaceTags	1489
Digikam::FaceTagsEditor	1491
Digikam::FaceUtils	1499
Digikam::FaceTagsIface	1496
Digikam::FacePipelineFaceTagsIface	1445
Digikam::FacialRecognitionWrapper	1503
Digikam::FFMpegConfigHelper	1508
Digikam::FieldQueryBuilder	1512
Digikam::FileActionProgressItemCreator	1528
Digikam::FileReadLocker	1531
Digikam::FileReadWriteLockKey	1531
Digikam::FileWriteLocker	1539
Digikam::FilmContainer	1539
Digikam::FilmGrainContainer	1545
Digikam::Filter	1551
Digikam::FilterAction	1552
Digikam::DImgThreadedFilter::DefaultFilterAction< Filter >	1026
Digikam::FocusPoint	1578



Digikam::FrameOsd	1586
Digikam::FrameOsdSettings	1587
Digikam::FrameUtils	1588
Digikam::FreeRotationContainer	1588
Digikam::FullObjectDetection	1599
Digikam::GeoCoordinates	1607
Digikam::GeodeticCalculator	1609
Digikam::GeofaceCluster	1616
Digikam::GeofaceInternalWidgetInfo	1619
Digikam::GeolocationSettingsContainer	1624
Digikam::GPSDataContainer	1640
Digikam::GPSItemContainer	1649
Digikam::ItemGPS	2090
Digikam::GPSItemInfo	1652
Digikam::GPSUndoCommand::UndoInfo	1681
Digikam::Graph< VertexProperties, EdgeProperties >	1681
Digikam::ItemHistoryGraphData	2100
Digikam::Graph< VertexProperties, EdgeProperties >::DominatorTree	1689
Digikam::Graph< VertexProperties, EdgeProperties >::Edge	1689
Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch	1690
Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch::CommonVisitor	1692
Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch::BreadthFirstSearchVisitor	1691
Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch::DepthFirstSearchVisitor	1693
Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch::lessThanMapEdgeToTarget< GraphType, VertexLessThan >	1694
Digikam::Graph< VertexProperties, EdgeProperties >::Path	1694
Digikam::Graph< VertexProperties, EdgeProperties >::Vertex	1695
Digikam::GreycstorageContainer	1700
Digikam::GroupedImagesFinder	1707
Digikam::GroupingViewImplementation	1713
Digikam::DigikamItemView	968
Digikam::ItemThumbnailBar	2234
Digikam::TableViewTreeView	3114
Digikam::GroupItemFilterSettings	1714
Digikam::GroupStateComputer	1714
Digikam::Haar::Calculator	1715
Digikam::Haar::ImageData	1715
Digikam::Haar::SignatureData	1716
Digikam::Haar::SignatureMap	1716
Digikam::Haar::WeightBin	1716
Digikam::Haar::Weights	1717
Digikam::HaarIface	1717
Digikam::HaarProgressObserver	1721
Digikam::DuplicatesProgressObserver	1304
Digikam::HistoryEdgeProperties	1733
Digikam::HistoryImageId	1733
Digikam::HistoryVertexProperties	1735
Digikam::HotPixelContainer	1736
Digikam::HotPixelProps	1741
Digikam::HotPixelsWeights	1743
Digikam::HSLContainer	1748
Digikam::IccManager	1757
Digikam::IccPostLoadingManager	1760
Digikam::IccProfile	1763
Digikam::ICCSettingsContainer	1780
Digikam::IccTransform	1781
Digikam::Identity	1789

Digikam::IdentityProvider	1790
Digikam::ImageChangeset	1791
Digikam::ImageCommonContainer	1792
Digikam::ImageCurves	1792
Digikam::ImageHistoryEntry	1804
Digikam::ImageIface	1805
Digikam::ImageLevels	1808
Digikam::ImageListProvider	1808
Digikam::EmptyImageListProvider	1366
Digikam::QListImageListProvider	2685
Digikam::ImageMetadataContainer	1810
Digikam::ImageQualityCalculator	1812
Digikam::ImageQualityCalculator::ResultDetection	1813
Digikam::ImageQualityContainer	1814
Digikam::ImageRelation	1829
Digikam::ImageTagChangeset	1833
Digikam::ImageTagProperty	1834
Digikam::ImageTagPropertyName	1834
Digikam::ImageZoomSettings	1841
Digikam::InfraredContainer	1969
Digikam::InternalTagName	1977
Digikam::IOFileSettings	1982
Digikam::IOJobData	1984
Digikam::IptcCoreContactInfo	1992
Digikam::IptcCoreLocationInfo	1992
Digikam::ItemChangeHint	2020
Digikam::ItemComments	2021
Digikam::ItemCopyMoveHint	2029
Digikam::ItemCopyright	2030
Digikam::ItemDelegateOverlayContainer	2045
Digikam::ItemViewDelegate	2263
Digikam::ItemDelegate	2035
Digikam::DigikamItemDelegate	963
Digikam::ItemFaceDelegate	2055
Digikam::ItemThumbnailDelegate	2242
Digikam::ItemViewImportDelegate	2271
Digikam::ImportDelegate	1865
Digikam::ImportNormalDelegate	1913
Digikam::ImportThumbnailDelegate	1951
Digikam::VersionsDelegate	3362
ShowFoto::ShowfotoItemViewDelegate	3481
ShowFoto::ShowfotoDelegate	3440
ShowFoto::ShowfotoNormalDelegate	3487
ShowFoto::ShowfotoThumbnailDelegate	3526
Digikam::ItemExtendedProperties	2052
Digikam::ItemFilterModelPrepareHook	2071
Digikam::ItemFilterSettings	2077
Digikam::ItemFiltersHistoryTreeItem	2081
Digikam::ItemHistoryGraph	2096
Digikam::ItemInfo	2113
Digikam::ItemInfoSet	2130
Digikam::ItemInfoStatic	2130
Digikam::ItemLister	2134
Digikam::ItemListerReceiver	2143
Digikam::ItemListerValueListReceiver	2145
Digikam::ItemListerJobReceiver	2141
Digikam::ItemListerJobPartsSendingReceiver	2139

Digikam::ItemListerJobGrowingPartsSendingReceiver . . . . .	2137
Digikam::ItemListerRecord . . . . .	2144
Digikam::ItemMetadataAdjustmentHint . . . . .	2158
Digikam::ItemPosition . . . . .	2168
Digikam::ItemQueryBuilder . . . . .	2198
Digikam::ItemQueryPostHook . . . . .	2198
Digikam::ItemQueryPostHooks . . . . .	2199
Digikam::ItemScanInfo . . . . .	2212
Digikam::ItemScanner . . . . .	2212
Digikam::ItemShortInfo . . . . .	2228
Digikam::ItemSortSettings . . . . .	2229
Digikam::ItemTagPair . . . . .	2232
Digikam::JPEGUtils::digikam_source_mgr . . . . .	2286
Digikam::JPEGUtils::JpegRotator . . . . .	2286
Digikam::KDNodeBase . . . . .	2289
Digikam::KDNodeOpenFace . . . . .	2291
Digikam::KDNodeSFace . . . . .	2293
Digikam::KDNodeBase::NodeCompareResult . . . . .	2290
Digikam::KDTreeBase . . . . .	2295
Digikam::KDTreeOpenFace . . . . .	2297
Digikam::KDTreeSFace . . . . .	2298
Digikam::LcmsLock . . . . .	2311
Digikam::LensDistortionPixelAccess . . . . .	2316
Digikam::LensFunContainer . . . . .	2318
Digikam::LensFuniface . . . . .	2323
Digikam::LevelsContainer . . . . .	2325
Digikam::LoadingCache::CacheLock . . . . .	2359
Digikam::LoadingCacheInterface . . . . .	2361
Digikam::LoadingDescription . . . . .	2362
Digikam::LoadingDescription::PostProcessingParameters . . . . .	2364
Digikam::LoadingDescription::PreviewParameters . . . . .	2365
Digikam::LoadingProcess . . . . .	2366
Digikam::SharedLoadingTask . . . . .	2979
Digikam::LoadingProcessListener . . . . .	2367
Digikam::SharedLoadingTask . . . . .	2979
Digikam::LoadSaveFileInfoProvider . . . . .	2370
Digikam::DatabaseLoadSaveFileInfoProvider . . . . .	729
Digikam::LoadSaveNotifier . . . . .	2372
Digikam::LoadSaveThread . . . . .	2376
Digikam::ManagedLoadSaveThread . . . . .	2411
Digikam::PreviewLoadThread . . . . .	2650
Digikam::FacePreviewLoader . . . . .	1465
Digikam::SharedLoadSaveThread . . . . .	2983
Digikam::ThumbnailLoadThread . . . . .	3259
Digikam::LoadSaveTask . . . . .	2374
Digikam::LoadingTask . . . . .	2368
Digikam::SavingTask . . . . .	2805
Digikam::LocalContrastContainer . . . . .	2382
Digikam::LocalizeContainer . . . . .	2389
Digikam::LookupAltitude::Request . . . . .	2395
Digikam::LookupFactory . . . . .	2398
Digikam::MaintenanceData . . . . .	2399
Digikam::MaintenanceSettings . . . . .	2400
Digikam::Mat . . . . .	2437
Digikam::MetadataHub . . . . .	2439
Digikam::MetaEngine . . . . .	2475

Digikam::DMetadata . . . . .	1075
Digikam::MetaEngineData . . . . .	2505
Digikam::MetaEngineMergeHelper< Data, Key, KeyString, KeyStringList > . . . . .	2506
Digikam::MetaEnginePreviews . . . . .	2507
Digikam::MetaEngineRotation . . . . .	2508
Digikam::MetaEngineSettingsContainer . . . . .	2511
Digikam::MixerContainer . . . . .	2515
Digikam::MLClassifierFoundation . . . . .	2522
Digikam::FaceClassifierBase . . . . .	1402
Digikam::FaceClassifier . . . . .	1400
Digikam::MLClassifierFoundation::VotingGroups . . . . .	2523
Digikam::MLClassifierFoundation::VotingGroups::VoteTally . . . . .	2523
Digikam::MLPipelineFoundation::_MLPipelinePerformanceProfile . . . . .	2527
Digikam::MLPipelinePackageFoundation . . . . .	2528
Digikam::FacePipelinePackageBase . . . . .	1450
Digikam::NamespaceEntry . . . . .	2551
Digikam::NRContainer . . . . .	2578
Digikam::OpenCVDNNFaceDetector . . . . .	2598
Digikam::OpenCVDNNFaceRecognizer . . . . .	2600
Digikam::OpenfacePreprocessor . . . . .	2601
Digikam::ParallelWorkers . . . . .	2620
Digikam::ParallelAdapter< A > . . . . .	2615
Digikam::Parser . . . . .	2623
Digikam::DefaultRenameParser . . . . .	912
Digikam::ImportRenameParser . . . . .	1927
Digikam::ParseResults . . . . .	2624
Digikam::ParseSettings . . . . .	2625
Digikam::PhotoInfoContainer . . . . .	2635
Digikam::PointTransformAffine . . . . .	2642
Digikam::PreviewSettings . . . . .	2658
Digikam::QueueSettings . . . . .	2697
Digikam::RandomNumberGenerator . . . . .	2705
Digikam::RatingStarDrawer . . . . .	2726
Digikam::RatingComboBoxDelegate . . . . .	2716
Digikam::RatingComboBoxWidget . . . . .	2718
Digikam::RecognitionBenchmarkers::Statistics . . . . .	2740
Digikam::RecognitionTrainingUpdateQueue . . . . .	2743
Digikam::RedEye::RegressionTree . . . . .	2747
Digikam::RedEye::ShapePredictor . . . . .	2747
Digikam::RedEye::SplitFeature . . . . .	2748
Digikam::RedEyeCorrectionContainer . . . . .	2748
Digikam::RefocusMatrix . . . . .	2759
Digikam::RGInfo . . . . .	2779
Digikam::SaveProperties . . . . .	2803
Digikam::SavingContext . . . . .	2804
Digikam::ScanController::FileMetadataWrite . . . . .	2812
Digikam::SearchChangeset . . . . .	2817
Digikam::SearchInfo . . . . .	2894
Digikam::SearchTextSettings . . . . .	2920
Digikam::SearchTextFilterSettings . . . . .	2919
Digikam::SearchViewThemedPartsCache . . . . .	2931
Digikam::SearchView . . . . .	2927
Digikam::SetupCollectionModel::Item . . . . .	2961
Digikam::SharedQueue< T > . . . . .	2987
Digikam::SharpContainer . . . . .	2987
Digikam::SidecarFinder . . . . .	3014

Digikam::SimilarityDb	3014
Digikam::SimilarityDbAccess	3020
Digikam::SimilarityDbSchemaUpdater	3026
Digikam::SimpleTreeModel::Item	3028
Digikam::StateSavingObject	3044
Digikam::AbstractAlbumTreeView	153
Digikam::AbstractCountingAlbumTreeView	189
Digikam::AbstractCheckableAlbumTreeView	177
Digikam::AlbumTreeView	354
Digikam::AlbumSelectTreeView	339
Digikam::AlbumSelectionTreeView	330
Digikam::SearchTreeView	2921
Digikam::EditableSearchTreeView	1324
Digikam::NormalSearchTreeView	2571
Digikam::TagTreeView	3210
Digikam::TagFolderView	3138
Digikam::TagCheckView	3117
Digikam::TagFilterView	3129
Digikam::TagMngrTreeView	3153
Digikam::DateTreeView	771
Digikam::AutoTagsScanWidget	426
Digikam::DateFolderView	754
Digikam::FaceScanWidget	1480
Digikam::FilterSideBarWidget	1564
Digikam::FuzzySearchView	1605
Digikam::GPSSearchView	1677
Digikam::LabelsTreeView	2307
Digikam::MapWidgetView	2433
Digikam::SearchTextBar	2912
Digikam::SearchTextBarDb	2915
Digikam::Sidebar	3004
Digikam::ImportItemPropertiesSideBarImport	1902
Digikam::ItemPropertiesSideBar	2182
Digikam::ItemPropertiesSideBarDB	2187
Digikam::SidebarWidget	3011
Digikam::AlbumFolderViewSideBarWidget	274
Digikam::DateFolderViewSideBarWidget	757
Digikam::FuzzySearchSideBarWidget	1601
Digikam::GPSSearchSideBarWidget	1673
Digikam::LabelsSideBarWidget	2303
Digikam::PeopleSideBarWidget	2626
Digikam::SearchSideBarWidget	2907
Digikam::TagViewSideBarWidget	3220
Digikam::TimelineSideBarWidget	3287
Digikam::TableView	3067
Digikam::TagsManager	3202
ShowFoto::ShowfotoFolderViewSideBar	3463
ShowFoto::ShowfotoStackViewSideBar	3514
Digikam::SubjectData	3060
Digikam::SystemSettings	3065
Digikam::TableViewColumnConfiguration	3073
Digikam::TableViewColumnDescription	3074
Digikam::TableViewColumnProfile	3076
Digikam::TableViewModel::Item	3112
Digikam::TableViewShared	3113
Digikam::TagChangeset	3116
Digikam::TagData	3126

Digikam::TaggingAction	3146
Digikam::TaggingActionFactory	3146
Digikam::TaggingActionFactory::ConstraintInterface	3148
Digikam::TagInfo	3149
Digikam::TagProperties	3172
Digikam::TagProperty	3178
Digikam::TagPropertyName	3178
Digikam::TagRegion	3179
Digikam::TagShortInfo	3196
Digikam::Template	3227
Digikam::TextureContainer	3237
Digikam::ThumbnailCreator	3247
Digikam::ThumbnailIdentifier	3250
Digikam::ThumbnailInfo	3253
Digikam::ThumbnailInfoProvider	3255
Digikam::ThumbsDbInfoProvider	3276
Digikam::ThumbnailSize	3269
Digikam::ThumbsDb	3270
Digikam::ThumbsDbAccess	3271
Digikam::ThumbsDbInfo	3276
Digikam::ThumbsDbSchemaUpdater	3277
Digikam::TileIndex	3283
Digikam::TimeAdjustContainer	3284
Digikam::TonalityContainer	3295
Digikam::TooltipCreator	3305
Digikam::TrackCorrelator::Correlation	3308
Digikam::TrackCorrelator::CorrelationOptions	3308
Digikam::TrackManager::Track	3313
Digikam::TrackManager::TrackPoint	3313
Digikam::TrackReader::TrackReadResult	3314
Digikam::TrainingDataProvider	3318
Digikam::RecognitionTrainingProvider	2741
Digikam::TransitionMngr	3323
Digikam::TreeBranch	3326
Digikam::UndoAction	3343
Digikam::UndoActionIrreversible	3344
Digikam::UndoActionReversible	3345
Digikam::UndoCache	3346
Digikam::UndoManager	3346
Digikam::UndoMetadataContainer	3346
Digikam::UndoState	3347
Digikam::VersionFileInfo	3355
Digikam::VersionFileOperation	3355
Digikam::VersionItemFilterSettings	3357
Digikam::VersionManager	3358
Digikam::VersionManagerSettings	3358
Digikam::VersionNamingScheme	3359
Digikam::DefaultVersionNamingScheme	.917
Digikam::VideoFrame	3369
Digikam::VideoInfoContainer	3369
Digikam::VideoMetadataContainer	3369
Digikam::VideoStripFilter	3370
Digikam::VideoThumbDecoder	3370
Digikam::VideoThumbnailer	3370
Digikam::VideoThumbWriter	3370
Digikam::VidSlideSettings	3371
Digikam::VisibilityObject	3382

Digikam::SearchField	2825
Digikam::SearchFieldAlbum	2827
Digikam::SearchFieldCheckBox	2831
Digikam::SearchFieldChoice	2835
Digikam::SearchFieldComboBox	2842
Digikam::SearchFieldColorDepth	2839
Digikam::SearchFieldPageOrientation	2859
Digikam::SearchFieldLabels	2851
Digikam::SearchFieldMonthDay	2855
Digikam::SearchFieldRangeDate	2862
Digikam::SearchFieldRangeDouble	2866
Digikam::SearchFieldRangeInt	2870
Digikam::SearchFieldRangeTime	2874
Digikam::SearchFieldRating	2878
Digikam::SearchFieldText	2882
Digikam::SearchFieldKeyword	2848
Digikam::WBContainer	3383
Digikam::Workflow	3398
Digikam::WSAlbum	3403
Digikam::WSToolUtils	3412
dng_host	
Digikam::DNGWriterHost	1116
KXmlGuiWindow	
Digikam::DXmlGuiWindow	1314
Digikam::DigikamApp	959
Digikam::EditorWindow	1345
Digikam::ImageWindow	1835
ShowFoto::Showfoto	3421
Digikam::ImportUI	1962
Digikam::LightTableWindow	2347
Digikam::QueueMgrWindow	2691
Marble::LayerInterface	
Digikam::BackendMarbleLayer	449
Digikam::MetaEngineMergeHelper< Exiv2::ExifData, Exiv2::ExifKey, QLatin1String >	2506
Digikam::ExifMetaEngineMergeHelper	1372
Digikam::MetaEngineMergeHelper< Exiv2::IptcData, Exiv2::IptcKey, QLatin1String >	2506
Digikam::IptcMetaEngineMergeHelper	1993
Digikam::MetaEngineMergeHelper< Exiv2::XmpData, Exiv2::XmpKey, QLatin1String >	2506
Digikam::XmpMetaEngineMergeHelper	3414
QAbstractButton	
Digikam::CoordinatesOverlayWidget	637
Digikam::GroupIndicatorOverlayWidget	1712
Digikam::ImportOverlayWidget	1918
Digikam::ItemViewHoverButton	2268
Digikam::FaceRejectionOverlayButton	1476
Digikam::ImportRotateOverlayButton	1933
Digikam::ItemFullScreenOverlayButton	2087
Digikam::ItemRotateOverlayButton	2210
Digikam::ItemSelectionOverlayButton	2223
ShowFoto::ShowfotoCoordinatesOverlayWidget	3438
QAbstractItemDelegate	
Digikam::ComboBoxDelegate	617
Digikam::DItemDelegate	1054
Digikam::ItemViewDelegate	2263
Digikam::ItemViewImportDelegate	2271
ShowFoto::ShowfotoItemViewDelegate	3481
Digikam::DWItemDelegate	1306



Digikam::SetupCollectionDelegate . . . . .	2954
QAbstractItemModel	
Digikam::AbstractAlbumModel . . . . .	148
Digikam::AbstractSpecificAlbumModel . . . . .	203
Digikam::AbstractCountingAlbumModel . . . . .	183
Digikam::AbstractCheckableAlbumModel . . . . .	169
Digikam::AlbumModel . . . . .	303
Digikam::SearchModel . . . . .	2895
Digikam::TagModel . . . . .	3160
Digikam::DateAlbumModel . . . . .	747
Digikam::BookmarksModel . . . . .	503
Digikam::DConfigDlgModel . . . . .	858
Digikam::DConfigDlgWdgModel . . . . .	881
Digikam::GPSItemModel . . . . .	1658
Digikam::ItemFiltersHistoryModel . . . . .	2080
Digikam::ItemHistoryGraphModel . . . . .	2105
Digikam::RGTagModel . . . . .	2780
Digikam::SetupCollectionModel . . . . .	2958
Digikam::SimpleTreeModel . . . . .	3027
Digikam::TableViewModel . . . . .	3109
Digikam::TagMngrListModel . . . . .	3150
Digikam::TrackListModel . . . . .	3310
QAbstractListModel	
Digikam::ChoiceSearchModel . . . . .	578
Digikam::ImportItemModel . . . . .	1895
Digikam::ItemModel . . . . .	2160
Digikam::ItemVersionsModel . . . . .	2255
Digikam::RatingComboBoxModel . . . . .	2717
ShowFoto::ShowfotoItemModel . . . . .	3472
QAbstractSlider	
Digikam::DSelector . . . . .	1274
Digikam::DColorValueSelector . . . . .	839
QAbstractTableModel	
Digikam::DTrashItemModel . . . . .	1295
QAction	
Digikam::DPluginAction . . . . .	1196
Digikam::RemoveFilterAction . . . . .	2768
Digikam::RemoveFilterAction . . . . .	2768
QByteArray	
Digikam::NonDeterministicRandomData . . . . .	2565
QComboBox	
Digikam::AdvancedRenameInput . . . . .	241
Digikam::CountrySelector . . . . .	698
Digikam::DDateEdit . . . . .	889
Digikam::GeolocationFilter . . . . .	1622
Digikam::lccRenderingIntentComboBox . . . . .	1777
Digikam::ImportFilterComboBox . . . . .	1878
Digikam::MimeTypeFilter . . . . .	2514
Digikam::ModelIndexBasedComboBox . . . . .	2532
Digikam::RatingComboBox . . . . .	2714
Digikam::StayPoppedUpComboBox . . . . .	3052
Digikam::ListViewComboBox . . . . .	2353
Digikam::ChoiceSearchComboBox . . . . .	575
Digikam::TreeViewComboBox . . . . .	3327
Digikam::TreeViewLineEditComboBox . . . . .	3330
Digikam::AlbumSelectComboBox . . . . .	325
Digikam::AbstractAlbumTreeViewSelectComboBox . . . . .	164



Digikam::AlbumTreeViewSelectComboBox . . . . .	359
Digikam::TagTreeViewSelectComboBox . . . . .	3216
Digikam::AddTagsComboBox . . . . .	232
Digikam::SqueezedComboBox . . . . .	3036
Digikam::IccProfilesComboBox . . . . .	1767
Digikam::TimeZoneComboBox . . . . .	3293
Digikam::WSComboBoxIntermediate . . . . .	3403
QCompleter	
Digikam::ModelCompleter . . . . .	2530
Digikam::TagCompleter . . . . .	3125
QDBusAbstractAdaptor	
CoreDbWatchAdaptor . . . . .	147
QDialog	
Digikam::AddBookmarkDialog . . . . .	230
Digikam::AdvancedRenameDialog . . . . .	240
Digikam::AlbumPropsEdit . . . . .	319
Digikam::AlbumSelectDialog . . . . .	329
Digikam::BookmarksDialog . . . . .	498
Digikam::CameraFolderDialog . . . . .	526
Digikam::CameraInfoDialog . . . . .	530
Digikam::CameraSelection . . . . .	538
Digikam::CaptureDlg . . . . .	552
Digikam::ChangeFaceRecognitionModelDlg . . . . .	565
Digikam::ClockPhotoDialog . . . . .	583
Digikam::ColorCorrectionDlg . . . . .	604
Digikam::DConfigDlg . . . . .	844
Digikam::Setup . . . . .	2948
ShowFoto::ShowfotoSetup . . . . .	3494
Digikam::DPluginAboutDlg . . . . .	1195
Digikam::DPluginDialog . . . . .	1212
Digikam::WSToolDialog . . . . .	3411
Digikam::DProgressDlg . . . . .	1244
Digikam::AdvancedRenameProcessDialog . . . . .	246
Digikam::DatabaseMigrationDialog . . . . .	730
Digikam::DbShrinkDialog . . . . .	812
Digikam::DeleteDialog . . . . .	920
Digikam::FileSaveOptionsDlg . . . . .	1535
Digikam::FilesDownloader . . . . .	1536
Digikam::GeoPluginAboutDlg . . . . .	1628
Digikam::ICCProfileInfoDlg . . . . .	1766
Digikam::ImportFilterDlg . . . . .	1879
Digikam::InfoDlg . . . . .	1968
Digikam::DBStatDlg . . . . .	813
Digikam::LibsInfoDlg . . . . .	2331
Digikam::RawCameraDlg . . . . .	2729
Digikam::SolidHardwareDlg . . . . .	3035
Digikam::MaintenanceDlg . . . . .	2399
Digikam::NamespaceEditDlg . . . . .	2550
Digikam::OnlineVersionDlg . . . . .	2597
Digikam::RuleDialog . . . . .	2798
Digikam::DatabaseOptionDialog . . . . .	734
Digikam::DateOptionDialog . . . . .	764
Digikam::DefaultValueDialog . . . . .	913
Digikam::MetadataOptionDialog . . . . .	2453
Digikam::RangeDialog . . . . .	2707
Digikam::ReplaceDialog . . . . .	2772
Digikam::SequenceNumberDialog . . . . .	2943
Digikam::SoftProofDialog . . . . .	3034

Digikam::TableViewConfigurationDialog . . . . .	3107
Digikam::TagEditDlg . . . . .	3128
Digikam::ToolTipDialog . . . . .	3305
Digikam::VersioningPromptUserSaveDialog . . . . .	3357
Digikam::VidPlayerDlg . . . . .	3371
Digikam::WSLoginDialog . . . . .	3404
Digikam::WSNewAlbumDialog . . . . .	3405
Digikam::WSSelectUserDlg . . . . .	3406
Digikam::WebBrowserDlg . . . . .	3390
Digikam::WorkflowDlg . . . . .	3398
ShowFoto::ShowfotoFolderViewBookmarkDlg . . . . .	3457
ShowFoto::ShowfotoStackViewFavoriteItemDlg . . . . .	3507
QDockWidget	
Digikam::ThumbBarDock . . . . .	3245
QDoubleSpinBox	
Digikam::CustomStepsDoubleSpinBox . . . . .	712
QFileDialog	
Digikam::DFileDialog . . . . .	937
QFileIconProvider	
Digikam::ImageDialogIconProvider . . . . .	1796
QFileSystemModel	
ShowFoto::ShowfotoFolderViewModel . . . . .	3462
QFrame	
Digikam::AssignNameWidget . . . . .	396
Digikam::DDatePicker . . . . .	892
Digikam::DHBox . . . . .	952
Digikam::DDateTimeEdit . . . . .	904
Digikam::DFileSelector . . . . .	939
Digikam::DFontSelect . . . . .	949
Digikam::DVBox . . . . .	1305
Digikam::CaptionEdit . . . . .	549
Digikam::ColorLabelWidget . . . . .	615
Digikam::ColorLabelFilter . . . . .	611
Digikam::DateFolderView . . . . .	754
Digikam::FilterSideBarWidget . . . . .	1564
Digikam::IccProfilesSettings . . . . .	1772
Digikam::ItemDescEditTab . . . . .	2047
Digikam::PickLabelWidget . . . . .	2640
Digikam::PickLabelFilter . . . . .	2636
Digikam::RatingBox . . . . .	2712
Digikam::TransactionItem . . . . .	3320
ShowFoto::ShowfotoFolderViewBar . . . . .	3455
Digikam::DZoomBar . . . . .	1322
Digikam::ImportView . . . . .	1966
Digikam::ItemIconView . . . . .	2108
Digikam::LocalizeSelector . . . . .	2390
Digikam::OverlayWidget . . . . .	2605
Digikam::ProgressView . . . . .	2678
Digikam::RatingFilter . . . . .	2721
Digikam::TemplateSelector . . . . .	3232
Digikam::TextFilter . . . . .	3236
Digikam::DLineWidget . . . . .	1070
Digikam::DMultiTabBarFrame . . . . .	1109
Digikam::DNotificationPopup . . . . .	1149
Digikam::DNotificationWidget . . . . .	1159
Digikam::DPopupFrame . . . . .	1236
Digikam::LightTableView . . . . .	2345
Digikam::PanIconFrame . . . . .	2611

Digikam::PlaceholderWidget . . . . .	2642
Digikam::StatusBarProgressWidget . . . . .	3049
QGraphicsItem	
Digikam::DSelectedItem . . . . .	1273
QGraphicsObject	
Digikam::ClickDragReleaseItem . . . . .	582
Digikam::DImgChildItem . . . . .	998
Digikam::RegionFrameItem . . . . .	2760
Digikam::FacelItem . . . . .	1417
Digikam::FocusPointItem . . . . .	1582
Digikam::RubberItem . . . . .	2791
Digikam::GraphicsDImgItem . . . . .	1696
Digikam::DImgPreviewItem . . . . .	1012
Digikam::ItemPreviewCanvas . . . . .	2172
Digikam::ImagePreviewItem . . . . .	1811
Digikam::ImageRegionItem . . . . .	1825
QGraphicsView	
Digikam::DPreviewImage . . . . .	1239
Digikam::GraphicsDImgView . . . . .	1698
Digikam::Canvas . . . . .	545
Digikam::ImageRegionWidget . . . . .	1827
Digikam::ImportPreviewView . . . . .	1919
Digikam::ItemPreviewView . . . . .	2175
Digikam::LightTablePreview . . . . .	2333
QGroupBox	
Digikam::FullScreenSettings . . . . .	1600
QHash	
Digikam::DatabaseFields::Hash< QVariant > . . . . .	726
Digikam::DatabaseFields::Hash< T > . . . . .	726
QItemDelegate	
Digikam::GPSItemDelegate . . . . .	1652
Digikam::RatingComboBoxDelegate . . . . .	2716
Digikam::TableViewItemDelegate . . . . .	3108
QItemSelectionModel	
Digikam::GPSLinkItemSelectionModel . . . . .	1663
QLabel	
Digikam::DActiveLabel . . . . .	718
Digikam::DAdjustableLabel . . . . .	719
Digikam::DSqueezedClickLabel . . . . .	1283
Digikam::DTextLabelName . . . . .	1290
Digikam::DTextLabelValue . . . . .	1291
Digikam::DClickLabel . . . . .	832
Digikam::DCursorTracker . . . . .	888
Digikam::DItemToolTip . . . . .	1065
Digikam::BlackFrameToolTip . . . . .	483
Digikam::FreeSpaceToolTip . . . . .	1596
Digikam::ImageDialogToolTip . . . . .	1798
Digikam::ItemViewToolTip . . . . .	2277
Digikam::QueueToolTip . . . . .	2699
ShowFoto::ShowfotoFolderViewToolTip . . . . .	3466
ShowFoto::ShowfotoStackViewToolTip . . . . .	3517
Digikam::EffectPreview . . . . .	1352
Digikam::TransitionPreview . . . . .	3323
Digikam::WorkingWidget . . . . .	3402
QLayout	
Digikam::DynamicLayout . . . . .	1318
QLineEdit	
Digikam::AddTagsLineEdit . . . . .	237

Digikam::ProxyLineEdit . . . . .	2683
Digikam::ProxyClickLineEdit . . . . .	2680
Digikam::SearchTextBar . . . . .	2912
QList	
Digikam::AlbumPointerList< T > . . . . .	318
Digikam::FacePipelineFaceTagsIfaceList . . . . .	1448
Digikam::FileActionItemInfoList . . . . .	1513
Digikam::ItemInfoTaskSplitter . . . . .	2131
Digikam::ItemInfoList . . . . .	2129
Digikam::PackageLoadingDescriptionList . . . . .	2607
QListView	
Digikam::DCategorizedView . . . . .	824
Digikam::ActionCategorizedView . . . . .	211
Digikam::ItemViewCategorized . . . . .	2256
Digikam::NamespaceListView . . . . .	2552
QListWidget	
Digikam::DTextList . . . . .	1292
Digikam::PreviewList . . . . .	2645
QListWidgetItem	
Digikam::FilmContainer::ListItem . . . . .	1540
Digikam::PreviewListItem . . . . .	2646
QMainWindow	
Digikam::TagsManager . . . . .	3202
QMap	
Digikam::QMapForAdaptors< Vertex, Vertex > . . . . .	2687
Digikam::QMapForAdaptors< Vertex, int > . . . . .	2687
Digikam::CaptionsMap . . . . .	550
Digikam::QMapForAdaptors< Key, Value > . . . . .	2687
QMenu	
Digikam::ColorLabelMenuAction . . . . .	613
Digikam::DDatePickerPopup . . . . .	897
Digikam::lccProfilesMenuAction . . . . .	1770
Digikam::ModelMenu . . . . .	2533
Digikam::BookmarksMenu . . . . .	500
Digikam::PickLabelMenuAction . . . . .	2638
Digikam::RatingMenuAction . . . . .	2725
Digikam::TagsPopupMenu . . . . .	3209
QMimeData	
Digikam::DAlbumDrag . . . . .	722
Digikam::DCameraDragObject . . . . .	817
Digikam::DCameraItemListDrag . . . . .	818
Digikam::DItemDrag . . . . .	1056
Digikam::DTagListDrag . . . . .	1284
Digikam::MapDragData . . . . .	2420
QObject	
Digikam::AbstractDetector . . . . .	193
Digikam::AestheticDetector . . . . .	252
Digikam::BlurDetector . . . . .	485
Digikam::CompressionDetector . . . . .	621
Digikam::ExposureDetector . . . . .	1398
Digikam::NoiseDetector . . . . .	2564
Digikam::AbstractItemDragDropHandler . . . . .	194
Digikam::ImportDragDropHandler . . . . .	1876
Digikam::ItemDragDropHandler . . . . .	2050
ShowFoto::ShowfotoDragDropHandler . . . . .	3446
Digikam::AbstractMarkerTiler . . . . .	196
Digikam::GPSMarkerTiler . . . . .	1664
Digikam::ItemMarkerTiler . . . . .	2153

Digikam::ActionJob	217
Digikam::ActionTask	220
Digikam::AutotagsAssignmentTask	423
Digikam::DBJob	798
Digikam::AlbumsJob	348
Digikam::DatesJob	769
Digikam::GPSJob	1661
Digikam::SearchesJob	2823
Digikam::TagsJob	3196
Digikam::DatabaseTask	740
Digikam::FingerprintsTask	1576
Digikam::IOJob	1983
Digikam::BuildTrashCountersJob	514
Digikam::CopyOrMoveJob	638
Digikam::DTrashItemsListingJob	1299
Digikam::DeleteJob	923
Digikam::EmptyDTrashItemsJob	1364
Digikam::RenameFileJob	2770
Digikam::RestoreDTrashItemsJob	2776
Digikam::ImageQualityTask	1821
Digikam::MetadataRemoveTask	2461
Digikam::MetadataSyncTask	2471
Digikam::ThumbsTask	3281
Digikam::VidSlideTask	3377
Digikam::AdvancedRenameManager	244
Digikam::Akonadiface	253
Digikam::AlbumHistory	278
Digikam::AlbumLabelsSearchHandler	281
Digikam::AlbumManager	282
Digikam::AlbumModelDragDropHandler	310
Digikam::AlbumDragDropHandler	265
Digikam::TagDragDropHandler	3126
Digikam::AlbumModificationHelper	312
Digikam::AlbumThumbnailLoader	350
Digikam::AlbumWatch	362
Digikam::ApplicationSettings	377
Digikam::BalooWrap	452
Digikam::BatchTool	457
Digikam::BatchToolsFactory	464
Digikam::BdEngineBackend	471
Digikam::CoreDbBackend	682
Digikam::FaceDbBackend	1407
Digikam::SimilarityDbBackend	3022
Digikam::ThumbsDbBackend	3272
Digikam::BlackFrameListViewItem	481
Digikam::BlackFrameParser	482
Digikam::BookmarkNode	497
Digikam::BookmarksManager	499
Digikam::CameraList	533
Digikam::CameraThumbsCtrl	539
Digikam::CollectionManager	589
Digikam::CollectionScanner	597
Digikam::ContextMenuHelper	627
Digikam::CoreDbCopyManager	686
Digikam::CoreDbWatch	696
Digikam::DAboutData	715
Digikam::DBJobsManager	801
Digikam::DBinaryIface	790

Digikam::ExifToolBinary . . . . .	1374
Digikam::FFmpegBinary . . . . .	1506
Digikam::MysqlAdminBinary . . . . .	2539
Digikam::MysqlInitBinary . . . . .	2542
Digikam::MysqlServerBinary . . . . .	2545
Digikam::MysqlUpgradeBinary . . . . .	2548
Digikam::DCategoryDrawer . . . . .	827
Digikam::ImportCategoryDrawer . . . . .	1852
Digikam::ItemCategoryDrawer . . . . .	2018
Digikam::DConfigDlgMngr . . . . .	852
Digikam::DConfigDlgWdgItem . . . . .	878
Digikam::DIO . . . . .	1037
Digikam::DInfoInterface . . . . .	1030
Digikam::DBInfolface . . . . .	793
Digikam::BqmInfolface . . . . .	511
Digikam::DMetalInfolface . . . . .	1097
ShowFoto::ShowfotoInfolface . . . . .	3468
Digikam::DKCamera . . . . .	1066
Digikam::GPCamera . . . . .	1629
Digikam::UMSCamera . . . . .	3337
Digikam::DMetadataSettings . . . . .	1095
Digikam::DModelFactory . . . . .	1101
Digikam::DNNModelManager . . . . .	1133
Digikam::DOnlineTranslator . . . . .	1167
Digikam::DOnlineTts . . . . .	1180
Digikam::DPlugin . . . . .	1190
Digikam::DPluginBqm . . . . .	1199
Digikam::DPluginDImg . . . . .	1213
Digikam::DPluginEditor . . . . .	1218
Digikam::DPluginGeneric . . . . .	1221
Digikam::DPluginRawImport . . . . .	1228
Digikam::DPluginLoader . . . . .	1224
Digikam::DRawDecoder . . . . .	1252
Digikam::DWorkingPixmap . . . . .	1313
Digikam::DatabaseServerStarter . . . . .	738
Digikam::DbEngineErrorHandler . . . . .	783
Digikam::DbEngineGuiErrorHandler . . . . .	784
Digikam::DbHeaderListItem . . . . .	789
Digikam::DisjointMetadata . . . . .	1042
Digikam::DynamicThread . . . . .	1319
Digikam::DImgThreadedFilter . . . . .	1019
Digikam::AntiVignettingFilter . . . . .	372
Digikam::AutoLevelsFilter . . . . .	415
Digikam::BCGFilter . . . . .	466
Digikam::BWSepiaFilter . . . . .	517
Digikam::BlurFXFilter . . . . .	492
Digikam::BlurFilter . . . . .	487
Digikam::BorderFilter . . . . .	505
Digikam::CBFilter . . . . .	559
Digikam::CharcoalFilter . . . . .	566
Digikam::ColorFXFilter . . . . .	605
Digikam::ContentAwareFilter . . . . .	623
Digikam::CurvesFilter . . . . .	702
Digikam::DImgThreadedAnalyser . . . . .	1015
Digikam::AutoCrop . . . . .	404
Digikam::NREstimate . . . . .	2580
Digikam::DistortionFXFilter . . . . .	1049

Digikam::EmbossFilter . . . . .	1359
Digikam::EqualizeFilter . . . . .	1368
Digikam::FilmFilter . . . . .	1541
Digikam::FilmGrainFilter . . . . .	1546
Digikam::FilterActionFilter . . . . .	1557
Digikam::FreeRotationFilter . . . . .	1590
Digikam::GreycstorationFilter . . . . .	1701
Digikam::HSLFilter . . . . .	1749
Digikam::HotPixelFixer . . . . .	1737
Digikam::IccTransformFilter . . . . .	1784
Digikam::InfraredFilter . . . . .	1970
Digikam::InvertFilter . . . . .	1978
Digikam::LensDistortionFilter . . . . .	2312
Digikam::LensFunFilter . . . . .	2319
Digikam::LevelsFilter . . . . .	2326
Digikam::LocalContrastFilter . . . . .	2383
Digikam::MixerFilter . . . . .	2516
Digikam::NRFilter . . . . .	2585
Digikam::NormalizeFilter . . . . .	2567
Digikam::OilPaintFilter . . . . .	2591
Digikam::RainDropFilter . . . . .	2701
Digikam::RawProcessingFilter . . . . .	2731
Digikam::RedEyeCorrectionFilter . . . . .	2749
Digikam::RefocusFilter . . . . .	2755
Digikam::SharpenFilter . . . . .	2989
Digikam::ShearFilter . . . . .	2995
Digikam::StretchFilter . . . . .	3055
Digikam::TextureFilter . . . . .	3238
Digikam::TonalityFilter . . . . .	3296
Digikam::UnsharpMaskFilter . . . . .	3351
Digikam::WBFilter . . . . .	3384
Digikam::AutoExpoFilter . . . . .	409
Digikam::ImageHistogram . . . . .	1802
Digikam::LoadSaveThread . . . . .	2376
Digikam::ScanStateFilter . . . . .	2814
Digikam::EditorCore . . . . .	1331
Digikam::EditorTool . . . . .	1336
Digikam::EditorToolThreaded . . . . .	1341
Digikam::EditorToolIface . . . . .	1338
Digikam::ExifToolListViewGroup . . . . .	1379
Digikam::ExifToolParser . . . . .	1382
Digikam::FaceGroup . . . . .	1414
Digikam::FacePipeline . . . . .	1421
Digikam::FaceUtils . . . . .	1499
Digikam::FileActionMngr . . . . .	1515
Digikam::FileActionProgressItemContainer . . . . .	1527
Digikam::FocusPointGroup . . . . .	1580
Digikam::FocusPointsExtractor . . . . .	1585
Digikam::FocusPointsWriter . . . . .	1586
Digikam::GPSBookmarkOwner . . . . .	1638
Digikam::GPSItemInfoSorter . . . . .	1653
Digikam::GPSItemListContextMenu . . . . .	1656
Digikam::GPSModelIndexProxyMapper . . . . .	1671
Digikam::GeoDragDropHandler . . . . .	1616
Digikam::MapDragDropHandler . . . . .	2421
Digikam::GeofaceGlobalObject . . . . .	1617
Digikam::GeoModelHelper . . . . .	1625
Digikam::GPSBookmarkModelHelper . . . . .	1635

Digikam::GPSGeofaceModelHelper . . . . .	1646
Digikam::ItemGPSModelHelper . . . . .	2094
Digikam::MapViewModelHelper . . . . .	2423
Digikam::GeolocationSettings . . . . .	1623
Digikam::GreycstorationSettings . . . . .	1706
Digikam::HistogramPainter . . . . .	1727
Digikam::IOJobsManager . . . . .	1986
Digikam::lccSettings . . . . .	1778
Digikam::ImageDialog . . . . .	1795
Digikam::ImageDialogIconProvider . . . . .	1796
Digikam::ImageQualityParser . . . . .	1815
Digikam::ImageQualityThreadPool . . . . .	1824
Digikam::ImportContextMenuHelper . . . . .	1855
Digikam::ImportSettings . . . . .	1936
Digikam::ItemAttributesWatch . . . . .	2008
Digikam::ItemDelegateOverlay . . . . .	2042
Digikam::AbstractWidgetDelegateOverlay . . . . .	206
Digikam::GroupIndicatorOverlay . . . . .	1708
Digikam::HoverButtonDelegateOverlay . . . . .	1744
Digikam::ActionVersionsOverlay . . . . .	226
Digikam::FaceRejectionOverlay . . . . .	1471
Digikam::ImportRotateOverlay . . . . .	1928
Digikam::ItemFullScreenOverlay . . . . .	2082
Digikam::ItemRotateOverlay . . . . .	2205
Digikam::ItemSelectionOverlay . . . . .	2219
Digikam::ShowHideVersionsOverlay . . . . .	3000
Digikam::ImportCoordinatesOverlay . . . . .	1861
Digikam::ImportDownloadOverlay . . . . .	1872
Digikam::ImportLockOverlay . . . . .	1909
Digikam::ImportRatingOverlay . . . . .	1922
Digikam::ItemCoordinatesOverlay . . . . .	2026
Digikam::ItemRatingOverlay . . . . .	2200
Digikam::PersistentWidgetDelegateOverlay . . . . .	2630
Digikam::AssignNameOverlay . . . . .	390
Digikam::TagsLineEditOverlay . . . . .	3198
ShowFoto::ShowfotoCoordinatesOverlay . . . . .	3435
Digikam::ItemInfoAlbumsJob . . . . .	2123
Digikam::ItemInfoCache . . . . .	2124
Digikam::ItemInfoJob . . . . .	2128
Digikam::ItemListDragDropHandler . . . . .	2134
Digikam::GPSItemListDragDropHandler . . . . .	1657
Digikam::ItemSortCollator . . . . .	2228
Digikam::ItemViewUtilities . . . . .	2279
Digikam::ItemVisibilityController . . . . .	2281
Digikam::HidingStateChanger . . . . .	1722
Digikam::AssignNameWidgetStates . . . . .	400
Digikam::ItemVisibilityControllerPropertyObject . . . . .	2285
Digikam::AnimatedVisibility . . . . .	370
Digikam::ListItem . . . . .	2351
Digikam::LoadingCache . . . . .	2356
Digikam::LoadingCacheFileWatch . . . . .	2360
Digikam::LocalizeSettings . . . . .	2392
Digikam::LookupAltitude . . . . .	2394
Digikam::LookupAltitudeGeonames . . . . .	2396
Digikam::MLPipelineFoundation . . . . .	2524
Digikam::FacePipelineBase . . . . .	1426
Digikam::FacePipelineDetect . . . . .	1430



Digikam::FacePipelineDetectRecognize . . . . .	1434
Digikam::FacePipelineEdit . . . . .	1438
Digikam::FacePipelineRecognize . . . . .	1452
Digikam::FacePipelineReset . . . . .	1456
Digikam::FacePipelineRetrain . . . . .	1460
Digikam::MaintenanceMngr . . . . .	2400
Digikam::MapBackend . . . . .	2418
Digikam::BackendGoogleMaps . . . . .	434
Digikam::BackendMarble . . . . .	441
Digikam::MdKeyListViewItem . . . . .	2437
Digikam::MetaEngineSettings . . . . .	2510
Digikam::MetadataHubMngr . . . . .	2445
Digikam::MetadataPanel . . . . .	2455
Digikam::NetworkManager . . . . .	2553
Digikam::OnlineVersionChecker . . . . .	2595
Digikam::OnlineVersionDwnl . . . . .	2598
Digikam::ParallelPipes . . . . .	2618
Digikam::PreviewThreadWrapper . . . . .	2659
Digikam::ProgressItem . . . . .	2663
Digikam::AlbumParser . . . . .	315
Digikam::FileActionProgress . . . . .	1524
Digikam::MaintenanceTool . . . . .	2405
Digikam::AutotagsAssignment . . . . .	420
Digikam::DbCleaner . . . . .	776
Digikam::DuplicatesFinder . . . . .	1301
Digikam::FacesDetector . . . . .	1483
Digikam::FacesEngine . . . . .	1486
Digikam::FingerPrintsGenerator . . . . .	1572
Digikam::ImageQualitySorter . . . . .	1817
Digikam::MetadataRemover . . . . .	2457
Digikam::MetadataSynchronizer . . . . .	2467
Digikam::NewItemFinder . . . . .	2555
Digikam::ThumbsGenerator . . . . .	3278
Digikam::ProgressManager . . . . .	2670
Digikam::RGBBackend . . . . .	2778
Digikam::BackendGeonamesRG . . . . .	428
Digikam::BackendGeonamesUSRG . . . . .	430
Digikam::BackendOsmRG . . . . .	449
Digikam::Rule . . . . .	2794
Digikam::Modifier . . . . .	2535
Digikam::CaseModifier . . . . .	554
Digikam::DefaultValueModifier . . . . .	914
Digikam::RangeModifier . . . . .	2709
Digikam::RemoveDoublesModifier . . . . .	2765
Digikam::ReplaceModifier . . . . .	2773
Digikam::TrimmedModifier . . . . .	3333
Digikam::UniqueModifier . . . . .	3348
Digikam::Option . . . . .	2603
Digikam::CameraNameOption . . . . .	535
Digikam::DatabaseOption . . . . .	731
Digikam::DateOption . . . . .	761
Digikam::DirectoryNameOption . . . . .	1039
Digikam::FilePropertiesOption . . . . .	1529
Digikam::MetadataOption . . . . .	2450
Digikam::SequenceNumberOption . . . . .	2945
Digikam::SearchField . . . . .	2825
Digikam::SearchModificationHelper . . . . .	2901
Digikam::SetupRaw . . . . .	2975

Digikam::SinglePhotoPreviewLayout	3029
Digikam::SyncJob	3064
Digikam::TableViewColumn	3071
Digikam::TableViewColumns::ColumnAudioVideoProperties	3077
Digikam::TableViewColumns::ColumnDigikamProperties	3081
Digikam::TableViewColumns::ColumnFileProperties	3086
Digikam::TableViewColumns::ColumnGeoProperties	3091
Digikam::TableViewColumns::ColumnItemProperties	3095
Digikam::TableViewColumns::ColumnPhotoProperties	3100
Digikam::TableViewColumns::ColumnThumbnail	3104
Digikam::TableViewColumnFactory	3075
Digikam::TableViewSelectionModeSyncer	3112
Digikam::TagModificationHelper	3167
Digikam::TagsActionMngr	3182
Digikam::TagsCache	3184
Digikam::TemplateManager	3230
Digikam::ThemeManager	3243
Digikam::ThreadManager	3244
Digikam::ThumbnailImageCatcher	3251
Digikam::TileGroupier	3283
Digikam::Token	3293
Digikam::TrackCorrelator	3307
Digikam::TrackManager	3311
Digikam::TrackReader	3314
Digikam::VisibilityController	3381
Digikam::WSSettings	3407
Digikam::WorkerObject	3395
Digikam::DatabaseWorkerInterface	742
Digikam::FileActionMngrDatabaseWorker	1517
Digikam::DatabaseWriter	745
Digikam::DetectionBenchmarkier	927
Digikam::DetectionWorker	930
Digikam::FileWorkerInterface	1537
Digikam::FileActionMngrFileWorker	1521
Digikam::ItemFilterModelWorker	2075
Digikam::ItemFilterModelFilterer	2069
Digikam::ItemFilterModelPreparer	2072
Digikam::RecognitionBenchmarkier	2738
Digikam::RecognitionWorker	2744
Digikam::TrainerWorker	3315
Digikam::WorkflowManager	3401
ShowFoto::ShowfotoKineticScroller	3486
ShowFoto::ShowfotoSettings	3492
QPlainTextEdit	
Digikam::AdvancedRenameLineEdit	242
Digikam::DPlainTextEdit	1187
QProcess	
Digikam::ExifToolProcess	1388
QProgressBar	
Digikam::DProgressWdg	1245
QProgressDialog	
Digikam::DBusyDlg	815
QPushButton	
Digikam::ColorLabelSelector	614
Digikam::DColorSelector	837
Digikam::DMultiTabBarButton	1107
Digikam::DMultiTabBarTab	1110
Digikam::DetByClockPhotoButton	926

Digikam::PickLabelSelector . . . . .	2639
QReadLocker	
Digikam::ItemInfoReadLocker . . . . .	2130
QRunnable	
Digikam::ActionJob . . . . .	217
QScrollArea	
Digikam::DExpanderBox . . . . .	932
Digikam::DExpanderBoxExclusive . . . . .	935
Digikam::DRawDecoderWidget . . . . .	1263
Digikam::ImportItemPropertiesTab . . . . .	1907
Digikam::ItemPropertiesTab . . . . .	2193
Digikam::ItemSelectionPropertiesTab . . . . .	2226
Digikam::TemplateViewer . . . . .	3234
Digikam::EditorToolSettings . . . . .	1339
Digikam::FuzzySearchView . . . . .	1605
Digikam::ICCPreviewWidget . . . . .	1762
Digikam::ImageDialogPreview . . . . .	1797
Digikam::SetupAlbumView . . . . .	2951
Digikam::SetupCamera . . . . .	2952
Digikam::SetupCategory . . . . .	2953
Digikam::SetupCollections . . . . .	2962
Digikam::SetupDatabase . . . . .	2964
Digikam::SetupEditor . . . . .	2965
Digikam::SetupEditorIface . . . . .	2966
Digikam::SetupGeolocation . . . . .	2967
Digikam::SetupICC . . . . .	2968
Digikam::SetupIOFiles . . . . .	2970
Digikam::SetupImageQualitySorter . . . . .	2969
Digikam::SetupLightTable . . . . .	2970
Digikam::SetupMetadata . . . . .	2971
Digikam::SetupMime . . . . .	2972
Digikam::SetupMisc . . . . .	2973
Digikam::SetupPlugins . . . . .	2974
Digikam::SetupTemplate . . . . .	2976
Digikam::SetupToolTip . . . . .	2977
Digikam::SetupVersioning . . . . .	2978
Digikam::SubjectWidget . . . . .	3063
Digikam::SubjectEdit . . . . .	3061
Digikam::TagsEdit . . . . .	3195
Digikam::TimeAdjustSettings . . . . .	3285
Digikam::TransactionItemView . . . . .	3322
ShowFoto::ShowfotoSetupMetadata . . . . .	3497
ShowFoto::ShowfotoSetupMisc . . . . .	3498
ShowFoto::ShowfotoSetupPlugins . . . . .	3499
ShowFoto::ShowfotoSetupRaw . . . . .	3500
ShowFoto::ShowfotoSetupToolTip . . . . .	3501
QSharedData	
Digikam::FacePipelineExtendedPackage . . . . .	1443
Digikam::GeofaceSharedData . . . . .	1620
Digikam::ItemHistoryGraphData . . . . .	2100
Digikam::ItemInfoData . . . . .	2126
Digikam::MLPipelinePackageNotify . . . . .	2529
Digikam::TwoProgressItemsContainer . . . . .	3336
Digikam::FileActionProgressItemContainer . . . . .	1527
QSortFilterProxyModel	
Digikam::AddBookmarkProxyModel . . . . .	231
Digikam::AlbumFilterModel . . . . .	268
Digikam::CheckableAlbumFilterModel . . . . .	570

Digikam::SearchFilterModel . . . . .	2885
Digikam::TagPropertiesFilterModel . . . . .	3174
Digikam::TagsManagerFilterModel . . . . .	3205
Digikam::DCategorizedSortFilterProxyModel . . . . .	819
Digikam::ActionSortFilterProxyModel . . . . .	218
Digikam::ImageSortFilterModel . . . . .	1830
Digikam::ItemFilterModel . . . . .	2061
Digikam::ItemAlbumFilterModel . . . . .	1997
Digikam::NoDuplicatesItemFilterModel . . . . .	2562
Digikam::ImportSortFilterModel . . . . .	1939
Digikam::ImportFilterModel . . . . .	1881
Digikam::NoDuplicatesImportFilterModel . . . . .	2559
ShowFoto::ShowfotoSortFilterModel . . . . .	3502
ShowFoto::NoDuplicatesShowfotoFilterModel . . . . .	3418
ShowFoto::ShowfotoFilterModel . . . . .	3449
Digikam::GPSItemSortProxyModel . . . . .	1660
Digikam::TreeProxyModel . . . . .	3326
QSpinBox	
Digikam::CustomStepsIntSpinBox . . . . .	713
QSplashScreen	
Digikam::DSplashScreen . . . . .	1282
QSplitter	
Digikam::SidebarSplitter . . . . .	3009
QSqlQuery	
Digikam::DbEngineSqlQuery . . . . .	788
QStackedWidget	
Digikam::DPreviewManager . . . . .	1242
Digikam::EditorStackView . . . . .	1334
Digikam::ExifToolWidget . . . . .	1393
Digikam::FileSaveOptionsBox . . . . .	1533
Digikam::ImportStackedView . . . . .	1942
Digikam::MediaPlayerView . . . . .	2438
Digikam::StackedView . . . . .	3041
Digikam::StatusProgressBar . . . . .	3050
Digikam::ToolSettingsView . . . . .	3302
QStandardItemModel	
Digikam::CategorizedItemModel . . . . .	557
Digikam::ActionItemModel . . . . .	214
QStyledItemDelegate	
Digikam::ItemFiltersHistoryItemDelegate . . . . .	2079
Digikam::ThumbnailAligningDelegate . . . . .	3247
Digikam::VersionsDelegate . . . . .	3362
QSyntaxHighlighter	
Digikam::Highlighter . . . . .	1725
QTabBar	
Digikam::QueuePoolBar . . . . .	2697
QTabWidget	
Digikam::AlbumSelectTabs . . . . .	339
Digikam::AutoTagsScanWidget . . . . .	426
Digikam::FaceScanWidget . . . . .	1480
Digikam::ItemPropertiesColorsTab . . . . .	2178
Digikam::ItemPropertiesMetadataTab . . . . .	2181
Digikam::ItemPropertiesVersionsTab . . . . .	2197
Digikam::QueuePool . . . . .	2695
Digikam::QueueSettingsView . . . . .	2698
Digikam::TemplatePanel . . . . .	3231
Digikam::ToolsView . . . . .	3304
QTemporaryFile	

Digikam::SafeTemporaryFile	2799
QTextBrowser	
Digikam::DTextBrowser	1285
QTextEdit	
Digikam::DTextEdit	1286
QThread	
Digikam::ActionThreadBase	224
Digikam::ActionThread	222
Digikam::DBJobsThread	804
Digikam::AlbumsDBJobsThread	322
Digikam::DatesDBJobsThread	766
Digikam::GPSDBJobsThread	1643
Digikam::SearchesDBJobsThread	2820
Digikam::TagsDBJobsThread	3192
Digikam::IOJobsThread	1988
Digikam::MaintenanceThread	2403
Digikam::VidSlideThread	3379
Digikam::CameraController	524
Digikam::CameraHistoryUpdater	529
Digikam::DBusSignalListenerThread	814
Digikam::DBusyThread	816
Digikam::CameraAutoDetectThread	522
Digikam::DatabaseCopyThread	725
Digikam::DatabaseServer	736
Digikam::DbEngineConnectionChecker	781
Digikam::ExifToolThread	1392
Digikam::ImageQualityThread	1823
Digikam::ProcessLauncher	2662
Digikam::FFmpegLauncher	1510
Digikam::ScanController	2807
Digikam::TrackCorrelatorThread	3309
QTreeView	
Digikam::AbstractAlbumTreeView	153
Digikam::GPSItemList	1654
Digikam::SetupCollectionTreeView	2963
Digikam::TableViewTreeView	3114
Digikam::TagMngrListView	3152
Digikam::VersionsTreeView	3365
ShowFoto::ShowfotoFolderViewList	3461
QTreeWidget	
Digikam::AssignedListView	387
Digikam::BlackFrameListView	480
Digikam::CameraFolderView	528
Digikam::CameraItemView	532
Digikam::DBinarySearch	792
Digikam::DHistoryView	953
Digikam::DItemsListView	1062
Digikam::DPluginConfView	1202
Digikam::DPluginConfViewBqm	1204
Digikam::DPluginConfViewDImg	1206
Digikam::DPluginConfViewEditor	1208
Digikam::DPluginConfViewGeneric	1210
Digikam::DbKeySelector	809
Digikam::DeleteItemList	922
Digikam::ExifToolListView	1378
Digikam::FindDuplicatesAlbum	1568
Digikam::LabelsTreeView	2307
Digikam::LanguagesList	2311

Digikam::MetadataListView	2448
Digikam::MetadataSelector	2463
Digikam::QueueListView	2688
Digikam::TemplateList	3228
Digikam::ToolsListView	3303
Digikam::WorkflowList	3400
ShowFoto::ShowfotoFolderViewBookmarkList	3459
ShowFoto::ShowfotoStackViewFavoriteList	3508
ShowFoto::ShowfotoStackViewList	3512
QTreeWidgetItem	
Digikam::AdvancedRenameListItem	243
Digikam::AssignedListViewItem	388
Digikam::BlackFrameListViewItem	481
Digikam::CameraFolderItem	527
Digikam::CameraItem	531
Digikam::DItemsListViewItem	1063
Digikam::DbHeaderListItem	789
Digikam::DbKeySelectorItem	810
Digikam::DeleteItem	921
Digikam::ExifToolListViewGroup	1379
Digikam::ExifToolListViewItem	1380
Digikam::FindDuplicatesAlbumItem	1570
Digikam::MdKeyListViewItem	2437
Digikam::MetadataListViewItem	2449
Digikam::MetadataSelectorItem	2464
Digikam::QueueListViewItem	2690
Digikam::TemplateListItem	3229
Digikam::ToolListViewGroup	3300
Digikam::ToolListViewItem	3301
Digikam::WorkflowItem	3399
ShowFoto::ShowfotoFolderViewBookmarkItem	3458
ShowFoto::ShowfotoStackViewFavoriteItem	3505
ShowFoto::ShowfotoStackViewItem	3511
QUndoCommand	
Digikam::ChangeBookmarkCommand	564
Digikam::GPSUndoCommand	1680
Digikam::RemoveBookmarksCommand	2764
Digikam::InsertBookmarksCommand	1976
ShowFoto::ShowfotoFolderViewUndo	3467
QUrl	
Digikam::CoreDbUrl	691
QWebEnginePage	
Digikam::HTMLWidgetPage	1756
Digikam::WelcomePageViewPage	3394
QWebEngineView	
Digikam::HTMLWidget	1755
Digikam::WebWidget	3391
Digikam::WelcomePageView	3393
QWebView	
Digikam::WebWidget	3391
QWidget	
Digikam::AbstractSearchGroupContainer	201
Digikam::SearchGroup	2890
Digikam::SearchView	2927
Digikam::AdvancedMetadataTab	239
Digikam::AdvancedRenameWidget	248
Digikam::AdvancedSettings	251
Digikam::AlbumCustomizer	264

Digikam::AlbumSelectWidget	346
Digikam::AlbumSelectors	336
Digikam::AltLangStrEdit	364
Digikam::AnimatedClearButton	368
Digikam::AntiVignettingSettings	376
Digikam::AudPlayerWdg	403
Digikam::BCGSettings	470
Digikam::BWSepiaSettings	521
Digikam::BorderSettings	509
Digikam::CBSettings	563
Digikam::CIETongueWidget	581
Digikam::CaptureWidget	553
Digikam::ColorFXSettings	609
Digikam::ColorGradientWidget	610
Digikam::CurvesBox	699
Digikam::CurvesSettings	707
Digikam::CurvesWidget	709
Digikam::DAbstractSliderSpinBox	716
Digikam::DDoubleSliderSpinBox	909
Digikam::DSliderSpinBox	1279
Digikam::DArrowClickLabel	724
Digikam::DComboBox	843
Digikam::DConfigDlgTitle	859
Digikam::DConfigDlgView	866
Digikam::DConfigDlgWdg	871
Digikam::DDateTable	900
Digikam::DDoubleNumInput	907
Digikam::DFontProperties	942
Digikam::DGradientSlider	951
Digikam::DImgLoaderSettings	1010
Digikam::DIntNumInput	1034
Digikam::DIntRangeBox	1035
Digikam::DItemsList	1059
Digikam::DLabelExpander	1069
Digikam::DMultiTabBar	1102
Digikam::Sidebar	3004
Digikam::DNGConvertSettings	1113
Digikam::DNGSettings	1114
Digikam::DPluginSetup	1231
Digikam::DPointSelect	1232
Digikam::DHueSaturationSelector	954
Digikam::DSaveSettingsWidget	1272
Digikam::DatabaseSettingsWidget	739
Digikam::DbKeySelectorView	811
Digikam::DeleteWidget	925
Digikam::DragHandle	1251
Digikam::ExifToolConfPanel	1376
Digikam::ExifToolErrorView	1377
Digikam::ExifToolLoadingView	1381
Digikam::FileSaveConflictBox	1532
Digikam::FilmGrainSettings	1550
Digikam::FilterStatusBar	1568
Digikam::FiltersHistoryWidget	1563
Digikam::FindDuplicatesView	1571
Digikam::FrameOsdWidget	1588
Digikam::FreeRotationSettings	1594
Digikam::FreeSpaceWidget	1598
Digikam::GPSCorrelatorWidget	1639

Digikam::GPSSearchView . . . . .	1677
Digikam::HSLSettings . . . . .	1753
Digikam::HSPreviewWidget . . . . .	1754
Digikam::HistogramBox . . . . .	1726
Digikam::HistogramWidget . . . . .	1731
Digikam::HotPixelSettings . . . . .	1742
Digikam::ImageGuideWidget . . . . .	1800
Digikam::ImageQualityConfSelector . . . . .	1813
Digikam::ImageQualitySettings . . . . .	1816
Digikam::ItemPropertiesGPSTab . . . . .	2179
Digikam::ItemPropertiesHistoryTab . . . . .	2180
Digikam::LensFunCameraSelector . . . . .	2317
Digikam::LensFunSettings . . . . .	2324
Digikam::LocalContrastSettings . . . . .	2387
Digikam::LocalizeConfig . . . . .	2388
Digikam::LocalizeSelectorList . . . . .	2391
Digikam::MapWidget . . . . .	2426
Digikam::MapWidgetView . . . . .	2433
Digikam::MetadataSelectorView . . . . .	2465
Digikam::MetadataStatusBar . . . . .	2466
Digikam::MetadataWidget . . . . .	2473
Digikam::ExifWidget . . . . .	1395
Digikam::ICCProfileWidget . . . . .	1774
Digikam::IptcWidget . . . . .	1994
Digikam::MakerNoteWidget . . . . .	2408
Digikam::XmpWidget . . . . .	3415
Digikam::MixerSettings . . . . .	2520
Digikam::MonthWidget . . . . .	2538
Digikam::NRSettings . . . . .	2589
Digikam::PanIconWidget . . . . .	2613
Digikam::PreviewToolBar . . . . .	2660
Digikam::RGWidget . . . . .	2787
Digikam::RatingWidget . . . . .	2727
Digikam::RatingComboBoxWidget . . . . .	2718
Digikam::RatingFilterWidget . . . . .	2723
Digikam::RedEyeCorrectionSettings . . . . .	2753
Digikam::RenameCustomizer . . . . .	2769
Digikam::ScriptingSettings . . . . .	2817
Digikam::SearchFieldGroup . . . . .	2845
Digikam::SearchFieldGroupLabel . . . . .	2846
Digikam::SearchGroupLabel . . . . .	2893
Digikam::SearchTabHeader . . . . .	2911
Digikam::SearchViewBottomBar . . . . .	2930
Digikam::SearchWindow . . . . .	2932
Digikam::SharpSettings . . . . .	2993
Digikam::SidebarWidget . . . . .	3011
Digikam::SketchWidget . . . . .	3031
Digikam::SlideVideo . . . . .	3033
Digikam::SpellCheckConfig . . . . .	3036
Digikam::StyleSheetDebugger . . . . .	3059
Digikam::SystemSettingsWidget . . . . .	3066
Digikam::TableView . . . . .	3067
Digikam::TableViewColumnConfigurationWidget . . . . .	3074
Digikam::TableViewColumns::ColumnFileConfigurationWidget . . . . .	3084
Digikam::TableViewColumns::ColumnGeoConfigurationWidget . . . . .	3089
Digikam::TableViewColumns::ColumnPhotoConfigurationWidget . . . . .	3098
Digikam::TagList . . . . .	3149
Digikam::TagPropWidget . . . . .	3178



Digikam::TextureSettings . . . . .	3242
Digikam::TimeLineWidget . . . . .	3291
Digikam::TrashView . . . . .	3324
Digikam::VersionsWidget . . . . .	3368
Digikam::WBSettings . . . . .	3389
Digikam::WSSettingsWidget . . . . .	3409
ShowFoto::ShowfotoFolderViewBookmarks . . . . .	3460
ShowFoto::ShowfotoFolderViewSideBar . . . . .	3463
ShowFoto::ShowfotoStackViewFavorites . . . . .	3510
ShowFoto::ShowfotoStackViewSideBar . . . . .	3514
QWidgetAction	
Digikam::DLogoAction . . . . .	1071
QWizard	
Digikam::DWizardDlg . . . . .	1312
Digikam::FirstRunDlg . . . . .	1578
QWizardPage	
Digikam::DWizardPage . . . . .	1312
Digikam::CollectionPage . . . . .	595
Digikam::DatabasePage . . . . .	735
Digikam::MetadataPage . . . . .	2454
Digikam::MigrateFromDigikam4Page . . . . .	2513
Digikam::OpenFilePage . . . . .	2602
Digikam::PreviewPage . . . . .	2657
Digikam::RawPage . . . . .	2730
Digikam::StartScanPage . . . . .	3043
Digikam::TooltipsPage . . . . .	3306
Digikam::WelcomePage . . . . .	3392
QWriteLocker	
Digikam::ItemInfoWriteLocker . . . . .	2133
QXmlStreamReader	
Digikam::SearchXmlReader . . . . .	2937
Digikam::KeywordSearchReader . . . . .	2299
Digikam::SearchXmlCachingReader . . . . .	2934
Digikam::XbelReader . . . . .	3412
QXmlStreamWriter	
Digikam::SearchXmlWriter . . . . .	2940
Digikam::KeywordSearchWriter . . . . .	2301
Digikam::XbelWriter . . . . .	3413
Digikam::SharedQueue< QString > . . . . .	2987
ShowFoto::ShowfotoItemInfo . . . . .	3470
ShowFoto::ShowfotoItemSortSettings . . . . .	3478



# Chapter 7

## Class Index

### 7.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

<a href="#">CoreDbWatchAdaptor</a>	147
<a href="#">Digikam::AbstractAlbumModel</a>	148
<a href="#">Digikam::AbstractAlbumTreeView</a>	
Base class for all tree views that display Album-based content provided by an <a href="#">AbstractSpecificAlbumModel</a>	
153	
<a href="#">Digikam::AbstractAlbumTreeView::ContextMenuElement</a>	
Add a context menu element which can add actions to the context menu when the menu is generated	
162	
<a href="#">Digikam::AbstractAlbumTreeViewSelectComboBox</a>	164
<a href="#">Digikam::AbstractCheckableAlbumModel</a>	169
<a href="#">Digikam::AbstractCheckableAlbumTreeView</a>	177
<a href="#">Digikam::AbstractCountingAlbumModel</a>	183
<a href="#">Digikam::AbstractCountingAlbumTreeView</a>	189
<a href="#">Digikam::AbstractDetector</a>	193
<a href="#">Digikam::AbstractItemDragDropHandler</a>	194
<a href="#">Digikam::AbstractMarkerTiler</a>	196
<a href="#">Digikam::AbstractMarkerTiler::ClickInfo</a>	199
<a href="#">Digikam::AbstractMarkerTiler::NonEmptyIterator</a>	199
<a href="#">Digikam::AbstractMarkerTiler::Tile</a>	200
<a href="#">Digikam::AbstractSearchGroupContainer</a>	201
<a href="#">Digikam::AbstractSpecificAlbumModel</a>	203
<a href="#">Digikam::AbstractWidgetDelegateOverlay</a>	206
<a href="#">Digikam::ActionCategorizedView</a>	211
<a href="#">Digikam::ActionData</a>	213
<a href="#">Digikam::ActionItemModel</a>	214
<a href="#">Digikam::ActionJob</a>	217
<a href="#">Digikam::ActionSortFilterProxyModel</a>	218
<a href="#">Digikam::ActionTask</a>	220
<a href="#">Digikam::ActionThread</a>	222
<a href="#">Digikam::ActionThreadBase</a>	224
<a href="#">Digikam::ActionVersionsOverlay</a>	226
<a href="#">Digikam::AddBookmarkDialog</a>	230
<a href="#">Digikam::AddBookmarkProxyModel</a>	
Proxy model that filters out the bookmarks so only the folders are left behind	
231	
<a href="#">Digikam::AddTagsComboBox</a>	232

Digikam::AddTagsLineEdit	237
Digikam::AdvancedMetadataTab	239
Digikam::AdvancedRenameDialog	240
Digikam::AdvancedRenameInput	241
Digikam::AdvancedRenameLineEdit	242
Digikam::AdvancedRenameListItem	243
Digikam::AdvancedRenameManager	244
Digikam::AdvancedRenameProcessDialog	246
Digikam::AdvancedRenameWidget	248
Digikam::AdvancedSettings	251
Digikam::AestheticDetector	252
Digikam::Akonadiiface	253
Digikam::Album	
Abstract base class for all album types	254
Digikam::AlbumChangeset	262
Digikam::AlbumCopyMoveHint	263
Digikam::AlbumCustomizer	264
Digikam::AlbumDragDropHandler	265
Digikam::AlbumFilterModel	268
Digikam::AlbumFolderViewSideBarWidget	274
Digikam::AlbumHistory	
Manages the history of the last visited albums	278
Digikam::AlbumInfo	
A container class for transporting album information from the database to <a href="#">AlbumManager</a>	280
Digikam::AlbumIterator	
Iterate over all children of this <a href="#">Album</a>	280
Digikam::AlbumLabelsSearchHandler	281
Digikam::AlbumManager	
<a href="#">AlbumManager</a> manages albums: does listing of albums and controls the lifetime of it	282
Digikam::AlbumModel	303
Digikam::AlbumModelDragDropHandler	310
Digikam::AlbumModificationHelper	
Utility class providing methods to modify physical albums ( <a href="#">PAlbum</a> ) in a way useful to implement views	312
Digikam::AlbumParser	315
Digikam::AlbumPointer< T >	
You can use <a href="#">AlbumPointer</a> to store a guarded pointer to <a href="#">Album</a> or one of the subclasses (use template parameter)	317
Digikam::AlbumPointerList< T >	318
Digikam::AlbumPropsEdit	319
Digikam::AlbumRootChangeset	320
Digikam::AlbumRootInfo	320
Digikam::AlbumsDBJobInfo	321
Digikam::AlbumsDBJobsThread	322
Digikam::AlbumSelectComboBox	325
Digikam::AlbumSelectDialog	329
Digikam::AlbumSelectionTreeView	
Album tree view used in the left sidebar to select <a href="#">PAlbums</a> and perform operations on them via a context menu	330
Digikam::AlbumSelectors	336
Digikam::AlbumSelectTabs	339
Digikam::AlbumSelectTreeView	
Enables a simple context menu only for creating a new album	339
Digikam::AlbumSelectWidget	346
Digikam::AlbumShortInfo	347
Digikam::AlbumSimplified	
This class is used when parsing response of <code>listAlbums()</code>	347
Digikam::AlbumsJob	348

Digikam::AlbumThumbnailLoader	350
Digikam::AlbumTreeView	354
Digikam::AlbumTreeViewSelectComboBox	359
Digikam::AlbumWatch	362
Digikam::AltLangStrEdit	364
Digikam::AnimatedClearButton	368
Digikam::AnimatedVisibility	370
Digikam::AntiVignettingContainer	371
Digikam::AntiVignettingFilter	372
Digikam::AntiVignettingSettings	376
Digikam::ApplicationSettings	377
Digikam::AssignedBatchTools	
Container to assign Batch tools and settings to an item by Url	386
Digikam::AssignedListView	387
Digikam::AssignedListViewItem	388
Digikam::AssignNameOverlay	390
Digikam::AssignNameWidget	396
Digikam::AssignNameWidgetStates	400
Digikam::AudPlayerWdg	403
Digikam::AutoCrop	404
Digikam::AutoExpoFilter	409
Digikam::AutoLevelsFilter	415
Digikam::AutoTagsAssign	419
Digikam::AutotagsAssignment	420
Digikam::AutotagsAssignmentTask	423
Digikam::AutoTagsScanSettings	425
Digikam::AutoTagsScanWidget	426
Digikam::BackendGeonamesRG	
This class calls Geonames' reverse geocoding service	428
Digikam::BackendGeonamesUSRG	
This class calls Geonames' get address service available only for USA locations	430
Digikam::BackendGoogleMaps	434
Digikam::BackendMarble	441
Digikam::BackendMarbleLayer	449
Digikam::BackendOsmRG	
This class calls Open Street Map's reverse geocoding service	449
Digikam::BalooInfo	452
Digikam::BalooWrap	
Singleton class which offer functionality for reading and writing image comment, tags and rating from Baloo to digiKam and from digiKam to Baloo	452
Digikam::BasicDImgFilterGenerator< T >	455
Digikam::BatchTool	457
Digikam::BatchToolSet	
A container of associated batch tool and settings	463
Digikam::BatchToolsFactory	464
Digikam::BCGContainer	465
Digikam::BCGFilter	466
Digikam::BCGSettings	470
Digikam::BdEngineBackend	471
Digikam::BdEngineBackend::QueryState	479
Digikam::BlackFrameListView	480
Digikam::BlackFrameListViewItem	481
Digikam::BlackFrameParser	482
Digikam::BlackFrameToolTip	483
Digikam::BlurDetector	485
Digikam::BlurFilter	487
Digikam::BlurFXFilter	492
Digikam::BookmarkNode	497

Digikam::BookmarksDialog	498
Digikam::BookmarksManager	
Bookmark manager, owner of the bookmarks, loads, saves and basic tasks	499
Digikam::BookmarksMenu	
Menu that is dynamically populated from the bookmarks	500
Digikam::BookmarksModel	
BookmarksModel is a QAbstractItemModel wrapper around the BookmarkManager	503
Digikam::BorderContainer	504
Digikam::BorderFilter	505
Digikam::BorderSettings	509
Digikam::BqmInfoface	511
Digikam::BuildTrashCountersJob	514
Digikam::BWSepiaContainer	515
Digikam::BWSepiaFilter	517
Digikam::BWSepiaSettings	521
Digikam::CameraAutoDetectThread	522
Digikam::CameraController	524
Digikam::CameraFolderDialog	526
Digikam::CameraFolderItem	527
Digikam::CameraFolderView	528
Digikam::CameraHistoryUpdater	529
Digikam::CameraInfoDialog	530
Digikam::CameraItem	531
Digikam::CameraItemList	532
Digikam::CameraList	533
Digikam::CameraMessageBox	534
Digikam::CameraNameHelper	534
Digikam::CameraNameOption	535
Digikam::CameraSelection	538
Digikam::CameraThumbsCtrl	539
Digikam::CameraType	540
Digikam::CamItemInfo	540
Digikam::CamItemSortSettings	542
Digikam::Canvas	545
Digikam::CaptionEdit	549
Digikam::CaptionsMap	
A map used to store a list of Alternative Language values + author and date properties The map key is the language code following RFC3066 notation (like "fr-FR" for French), and the CaptionsMap value all caption properties	550
Digikam::CaptionValues	552
Digikam::CaptureDlg	552
Digikam::CaptureWidget	553
Digikam::CaseModifier	554
Digikam::CategorizedItemModel	557
Digikam::CBContainer	558
Digikam::CBFilter	559
Digikam::CBSettings	563
Digikam::ChangeBookmarkCommand	564
Digikam::ChangeFaceRecognitionModelDlg	565
Digikam::CharcoalFilter	566
Digikam::CheckableAlbumFilterModel	
Filter model for checkable album models that allows more filtering options based on check state	570
Digikam::ChoiceSearchComboBox	575
Digikam::ChoiceSearchModel	578
Digikam::ChoiceSearchModel::Entry	580
Digikam::CIETongueWidget	581
Digikam::ClickDragReleaseItem	582
Digikam::ClockPhotoDialog	583

Digikam::CMat	
CMat:	584
Digikam::CollectionImageChangeset	584
Digikam::CollectionLocation	586
Digikam::CollectionManager	589
Digikam::CollectionPage	595
Digikam::CollectionScanner	597
Digikam::CollectionScannerHintContainer	602
Digikam::CollectionScannerObserver	603
Digikam::ColorCorrectionDlg	604
Digikam::ColorFXContainer	604
Digikam::ColorFXFilter	605
Digikam::ColorFXSettings	609
Digikam::ColorGradientWidget	610
Digikam::ColorLabelFilter	611
Digikam::ColorLabelMenuAction	613
Digikam::ColorLabelSelector	614
Digikam::ColorLabelWidget	615
Digikam::ComboBoxDelegate	617
Digikam::CommentInfo	618
Digikam::CommonKeys	619
Digikam::CompressionDetector	621
Digikam::ContentAwareContainer	622
Digikam::ContentAwareFilter	623
Digikam::ContextMenuHelper	
A helper class to add actions and special menus to the context menu	627
Digikam::CoordinatesOverlayWidget	637
Digikam::CopyOrMoveJob	638
Digikam::CopyrightInfo	639
Digikam::CoreDB	640
Digikam::CoreDbAccess	
The <a href="#">CoreDbAccess</a> provides access to the database: Create an instance of this class on the stack to retrieve a pointer to the database	678
Digikam::CoreDbAccessUnlock	681
Digikam::CoreDbBackend	682
Digikam::CoreDbCopyManager	686
Digikam::CoreDbDownloadHistory	687
Digikam::CoreDbNameFilter	688
Digikam::CoreDbOperationGroup	
When you intend to execute a number of write operations to the database, group them while holding a <a href="#">CoreDbOperationGroup</a>	688
Digikam::CoreDbPrivilegesChecker	689
Digikam::CoreDbSchemaUpdater	689
Digikam::CoreDbTransaction	
Convenience class: You can create a <a href="#">CoreDbTransaction</a> object for a scope for which you want to declare a database commit	689
Digikam::CoreDbUrl	691
Digikam::CoreDbWatch	696
Digikam::CountrySelector	698
Digikam::CurvesBox	699
Digikam::CurvesContainer	700
Digikam::CurvesFilter	702
Digikam::CurvesSettings	707
Digikam::CurvesWidget	709
Digikam::CustomStepsDoubleSpinBox	712
Digikam::CustomStepsIntSpinBox	713
Digikam::DAboutData	715
Digikam::DAbstractSliderSpinBox	716

<a href="#">Digikam::DActiveLabel</a>	
A widget to host an image into a label with an active url which can be open to default web browser using simple mouse click	718
<a href="#">Digikam::DAdjustableLabel</a>	
A label to show text adjusted to widget size	719
<a href="#">Digikam::DAlbum</a>	
A Date <a href="#">Album</a> representation	720
<a href="#">Digikam::DAlbumDrag</a>	
Provides a drag object for an album	722
<a href="#">Digikam::DAlbumInfo</a>	723
<a href="#">Digikam::DArrowClickLabel</a>	724
<a href="#">Digikam::DatabaseCopyThread</a>	725
<a href="#">Digikam::DatabaseFields::DatabaseFieldsEnumIterator&lt; <a href="#">FieldName</a> &gt;</a>	
You can iterate over each of the Enumerations defined above: <a href="#">ImagesIterator</a> , <a href="#">ImageMetadataIterator</a> etc	725
<a href="#">Digikam::DatabaseFields::DatabaseFieldsEnumIteratorSetOnly&lt; <a href="#">FieldName</a> &gt;</a>	
An iterator that iterates only over the flags which are set	726
<a href="#">Digikam::DatabaseFields::FieldMetaInfo&lt; <a href="#">FieldName</a> &gt;</a>	726
<a href="#">Digikam::DatabaseFields::Hash&lt; T &gt;</a>	
This class provides a hash on all <a href="#">DatabaseFields</a> enums, allowing to use the enum values as independent keys	726
<a href="#">Digikam::DatabaseFields::Set</a>	
This class provides a set of all <a href="#">DatabaseFields</a> enums, without resorting to a <a href="#">QSet</a>	728
<a href="#">Digikam::DatabaseLoadSaveFileInfoProvider</a>	729
<a href="#">Digikam::DatabaseMigrationDialog</a>	730
<a href="#">Digikam::DatabaseOption</a>	731
<a href="#">Digikam::DatabaseOptionDialog</a>	734
<a href="#">Digikam::DatabasePage</a>	735
<a href="#">Digikam::DatabaseServer</a>	736
<a href="#">Digikam::DatabaseServerError</a>	737
<a href="#">Digikam::DatabaseServerStarter</a>	738
<a href="#">Digikam::DatabaseSettingsWidget</a>	739
<a href="#">Digikam::DatabaseTask</a>	740
<a href="#">Digikam::DatabaseWorkerInterface</a>	742
<a href="#">Digikam::DatabaseWriter</a>	745
<a href="#">Digikam::DateAlbumModel</a>	
A model for date based albums	747
<a href="#">Digikam::DateFolderView</a>	754
<a href="#">Digikam::DateFolderViewSideBarWidget</a>	757
<a href="#">Digikam::DateFormat</a>	760
<a href="#">Digikam::DateOption</a>	761
<a href="#">Digikam::DateOptionDialog</a>	764
<a href="#">Digikam::DatesDBJobInfo</a>	765
<a href="#">Digikam::DatesDBJobsThread</a>	766
<a href="#">Digikam::DatesJob</a>	769
<a href="#">Digikam::DateTreeView</a>	771
<a href="#">Digikam::DbCleaner</a>	776
<a href="#">Digikam::DbEngineAccess</a>	
Access to the database: Create an instance of this class on the stack to retrieve a pointer to the database	779
<a href="#">Digikam::DbEngineAction</a>	779
<a href="#">Digikam::DbEngineActionElement</a>	779
<a href="#">Digikam::DbEngineActionType</a>	
The <a href="#">DbEngineActionType</a> is used by the <a href="#">BdEngineBackend</a> to wrap another data object within an sql statement and controls whether it should be used as field entry or as value (prepared to an sql statement with positional binding)	779
<a href="#">Digikam::DbEngineConfig</a>	780
<a href="#">Digikam::DbEngineConfigSettings</a>	780



Digikam::DbEngineConfigSettingsLoader	781
Digikam::DbEngineConnectionChecker	781
Digikam::DbEngineErrorAnswer	782
Digikam::DbEngineErrorHandler	783
Digikam::DbEngineGuiErrorHandler	784
Digikam::DbEngineLocking	785
Digikam::DbEngineParameters	
This class encapsulates all parameters needed to establish a connection to a database (inspired by the API of Qt::Sql)	785
Digikam::DbEngineSqlQuery	788
Digikam::DbHeaderListItem	789
Digikam::DBinaryIface	790
Digikam::DBinarySearch	
This class has nothing to do with a binary search, it is a widget to search for binaries	792
Digikam::DBInfolface	793
Digikam::DBJob	798
Digikam::DBJobInfo	800
Digikam::DBJobsManager	801
Digikam::DBJobsThread	804
Digikam::DbKeysCollection	
A class for managing / grouping database keys	806
Digikam::DbKeySelector	809
Digikam::DbKeySelectorItem	810
Digikam::DbKeySelectorView	811
Digikam::DbShrinkDialog	812
Digikam::DBStatDlg	813
Digikam::DBusSignalListenerThread	814
Digikam::DBusyDlg	815
Digikam::DBusyThread	816
Digikam::DCameraDragObject	
Provides a drag object for a camera object	817
Digikam::DCameratemListDrag	
Provides a drag object for a list of camera items	818
Digikam::DCategorizedSortFilterProxyModel	
This class lets you categorize a view	819
Digikam::DCategorizedView	
Item view for listing items	824
Digikam::DCategoryDrawer	
The category drawing is performed by this class	827
Digikam::DClickLabel	832
Digikam::DColor	833
Digikam::DColorComposer	835
Digikam::DColorSelector	
A widget to choose a color from a palette	837
Digikam::DColorValueSelector	839
Digikam::DComboBox	843
Digikam::DConfigDlg	
A dialog base class which can handle multiple pages	844
Digikam::DConfigDlgMgr	
Means of automatically retrieving, saving and resetting basic settings	852
Digikam::DConfigDlgModel	
A base class for a model used by <a href="#">DConfigDlgView</a>	858
Digikam::DConfigDlgTitle	
This class provides a widget often used for <a href="#">DConfigDlg</a> titles	859
Digikam::DConfigDlgView	
A base class which can handle multiple pages	866
Digikam::DConfigDlgWdg	
Page widget with many layouts (faces)	871

Digikam::DConfigDlgWdgItem	
DConfigDlgWdgItem is used by DConfigDlgWdg and represents a page	878
Digikam::DConfigDlgWdgModel	
This page model is used by	881
Digikam::DCursorTracker	
This class implements a window which looks like a tool tip	888
Digikam::DDateEdit	
A date editing widget that consists of an editable combo box	889
Digikam::DDatePicker	
Provides a widget for calendar date input	892
Digikam::DDatePickerPopup	
This menu helps the user to select a date quickly	897
Digikam::DDateTable	
This is a support class for the DDatePicker class	900
Digikam::DDateTimeEdit	
This class is basically the same as the KDE Date Time widget with the exception that a QTime↔	
Edit is placed directly besides it	904
Digikam::DDoubleNumInput	907
Digikam::DDoubleSliderSpinBox	909
Digikam::DefaultRenameParser	912
Digikam::DefaultValueDialog	913
Digikam::DefaultValueModifier	914
Digikam::DefaultVersionNamingScheme	917
Digikam::DeleteDialog	920
Digikam::DeleteItem	921
Digikam::DeleteItemList	922
Digikam::DeleteJob	923
Digikam::DeleteWidget	925
Digikam::DeltaTime	
Container that hold the time difference for clock photo dialog	925
Digikam::DetByClockPhotoButton	926
Digikam::DetectionBenchmark	927
Digikam::DetectionWorker	930
Digikam::DExpanderBox	932
Digikam::DExpanderBoxExclusive	935
Digikam::DFileDialog	937
Digikam::DFileOperations	938
Digikam::DFileSelector	
A widget to chose a single local file or path	939
Digikam::DFontProperties	942
Digikam::DFontSelect	949
Digikam::DGradientSlider	951
Digikam::DHBox	
An Horizontal widget to host children widgets	952
Digikam::DHistoryView	953
Digikam::DHueSaturationSelector	954
Digikam::DigikamApp	959
Digikam::DigikamItemDelegate	963
Digikam::DigikamItemView	968
Digikam::DImageHistory	976
Digikam::DImageHistory::Entry	979
Digikam::DImg	979
Digikam::DImgBuiltinFilter	995
Digikam::DImgChildItem	998
Digikam::DImgFilterGenerator	1002
Digikam::DImgFilterManager	1004
Digikam::DImgLoader	1006
Digikam::DImgLoaderObserver	1009

Digikam::DImgLoaderSettings	1010
Digikam::DImgPreviewItem	1012
Digikam::DImgThreadedAnalyser	1015
Digikam::DImgThreadedFilter	1019
Digikam::DImgThreadedFilter::DefaultFilterAction< Filter >	
Convenience class to spare the few repeating lines of code	1026
Digikam::DInfoInterface	1030
Digikam::DIntNumInput	1034
Digikam::DIntRangeBox	1035
Digikam::DIO	1037
Digikam::DirectoryNameOption	1039
Digikam::DisjointMetadata	1042
Digikam::DisjointMetadataDataFields	
This class was split from DisjointMetadata::Private to allow to use the automatic C++ copy constructor (DisjointMetadata::Private contains a QMutex and is thus non-copyable)	1047
Digikam::DistortionFXFilter	1049
Digikam::DItemDelegate	1054
Digikam::DItemDrag	
Provides a drag object with additional information for internal drag&drop	1056
Digikam::DItemInfo	
DItemInfo is a class to get item information from host application (Showfoto or digiKam) The interface is re-implemented in host and depend how item information must be retrieved (from a database or by file metadata)	1057
Digikam::DItemsList	1059
Digikam::DItemsListView	1062
Digikam::DItemsListViewItem	1063
Digikam::DItemToolTip	1065
Digikam::DKCamera	1066
Digikam::DLabelExpander	1069
Digikam::DLineWidget	
A widget to show an horizontal or vertical line separator	1070
Digikam::DLogoAction	1071
Digikam::DMessageBox	1071
Digikam::DMetadata	1075
Digikam::DMetadataSettings	1095
Digikam::DMetadataSettingsContainer	
The class DMetadataSettingsContainer is designed to dynamically add namespaces	1096
Digikam::DMetaInfoface	1097
Digikam::DModelFactory	
This class is simply a factory of all models that build the core of the digikam application	1101
Digikam::DMultiTabBar	
A Widget for horizontal and vertical tabs	1102
Digikam::DMultiTabBarButton	1107
Digikam::DMultiTabBarFrame	1109
Digikam::DMultiTabBarTab	1110
Digikam::DNGConvertSettings	1113
Digikam::DNGSettings	1114
Digikam::DNGWriter	1115
Digikam::DNGWriterHost	1116
Digikam::DNNBaseDetectorModel	1117
Digikam::DNNFaceDetectorBase	1119
Digikam::DNNFaceDetectorSSD	1121
Digikam::DNNFaceDetectorYOLO	1123
Digikam::DNNFaceDetectorYuNet	1125
Digikam::DNNFaceExtractorBase	1127
Digikam::DNNModelBase	1129
Digikam::DNNModelConfig	1130
Digikam::DNNModelInfoContainer	1131

Digikam::DNNModelManager	1133
Digikam::DNNModelNet	1134
Digikam::DNNModelSFace	1135
Digikam::DNNModelYuNet	1137
Digikam::DNNOpenFaceExtractor	1139
Digikam::DNNResnetDetector	1142
Digikam::DNNSFaceExtractor	1144
Digikam::DNNYoloDetector	1147
Digikam::DNotificationPopup	
A dialog-like popup that displays messages without interrupting the user	1149
Digikam::DNotificationWidget	
This widget can be used to provide inline positive or negative feedback, or to implement opportunistic interactions	1159
Digikam::DOnlineTranslator	
Provides translation data	1167
Digikam::DOnlineTranslatorOption	
Contains translation options for a single word	1179
Digikam::DOnlineTts	
Provides TTS URL generation	1180
Digikam::DownloadInfo	1185
Digikam::DownloadSettings	1185
Digikam::DPixelsAliasFilter	1186
Digikam::DPlainTextEdit	
A text edit widget based on QPlainTextEdit with spell checker capabilities based on Sonnet (optional)	1187
Digikam::DPlugin	
A digiKam external plugin abstract class	1190
Digikam::DPluginAboutDlg	1195
Digikam::DPluginAction	1196
Digikam::DPluginAuthor	1198
Digikam::DPluginBqm	1199
Digikam::DPluginConfView	1202
Digikam::DPluginConfViewBqm	1204
Digikam::DPluginConfViewDImg	1206
Digikam::DPluginConfViewEditor	1208
Digikam::DPluginConfViewGeneric	1210
Digikam::DPluginDialog	1212
Digikam::DPluginDImg	1213
Digikam::DPluginEditor	1218
Digikam::DPluginGeneric	1221
Digikam::DPluginLoader	
The class that handles digiKam's external plugins	1224
Digikam::DPluginRawImport	1228
Digikam::DPluginSetup	1231
Digikam::DPointSelect	1232
Digikam::DPopupFrame	1236
Digikam::DPreviewImage	1239
Digikam::DPreviewManager	1242
Digikam::DProgressDlg	1244
Digikam::DProgressWdg	1245
Digikam::DragDropModellImplementation	1247
Digikam::DragDropViewImplementation	1249
Digikam::DragHandle	
An alternative handle for QDockWidget's that looks like a toolbar handle	1251
Digikam::DRawDecoder	1252
Digikam::DRawDecoderSettings	1258
Digikam::DRawDecoderWidget	1263
Digikam::DRawDecoding	1266

Digikam::DRawInfo	1267
Digikam::DSaveSettingsWidget	1272
Digikam::DSelectionItem	1273
Digikam::DSelector	
DSelector is the base class for other widgets which provides the ability to choose from a one-dimensional range of values	1274
Digikam::DServiceInfo	1277
Digikam::DServiceMenu	1278
Digikam::DSliderSpinBox	1279
Digikam::DSplashScreen	1282
Digikam::DSqueezedClickLabel	1283
Digikam::DTagListDrag	
Provides a drag object for a list of tags	1284
Digikam::DTextBrowser	1285
Digikam::DTextEdit	
A text edit widget based on QTextEdit with spell checker capabilities based on Sonnet (optional)	1286
Digikam::DTextLabelName	1290
Digikam::DTextLabelValue	1291
Digikam::DTextList	1292
Digikam::DToolTipStyleSheet	1292
Digikam::DTrash	1293
Digikam::DTrashItemInfo	1294
Digikam::DTrashItemModel	1295
Digikam::DTrashItemsListingJob	1299
Digikam::DuplicatesFinder	1301
Digikam::DuplicatesProgressObserver	1304
Digikam::DVBox	
A Vertical widget to host children widgets	1305
Digikam::DWItemDelegate	
This class allows to create item delegates embedding simple widgets to interact with items	1306
Digikam::DWItemDelegatePool	1310
Digikam::DWItemDelegatePoolPrivate	1311
Digikam::DWizardDlg	1312
Digikam::DWizardPage	1312
Digikam::DWorkingPixmap	
A widget to draw progress wheel indicator over thumbnails	1313
Digikam::DXmlGuiWindow	
Generic class to use with all main window	1314
Digikam::DynamicLayout	1318
Digikam::DynamicThread	1319
Digikam::DZoomBar	1322
Digikam::EditableSearchTreeView	
This tree view for searches adds basic editing functionality via the context menu	1324
Digikam::EditorCore	1331
Digikam::EditorStackView	1334
Digikam::EditorTool	1336
Digikam::EditorToolface	1338
Digikam::EditorToolSettings	1339
Digikam::EditorToolThreaded	1341
Digikam::EditorWindow	1345
Digikam::EffectMgr	1351
Digikam::EffectPreview	1352
Digikam::Ellipsoid	
Geometric figure that can be used to describe the approximate shape of the earth	1352
Digikam::EmbossFilter	1359
Digikam::EmptyDTrashItemsJob	1364
Digikam::EmptyImageListProvider	1366
Digikam::EqualizeFilter	1368

Digikam::ExifMetaEngineMergeHelper	1372
Digikam::ExifToolBinary	1374
Digikam::ExifToolConfPanel	1376
Digikam::ExifToolErrorView	1377
Digikam::ExifToolListView	1378
Digikam::ExifToolListViewGroup	1379
Digikam::ExifToolListViewItem	1380
Digikam::ExifToolLoadingView	1381
Digikam::ExifToolParser	1382
Digikam::ExifToolProcess	1388
Digikam::ExifToolProcess::Result	1392
Digikam::ExifToolThread	1392
Digikam::ExifToolWidget	1393
Digikam::ExifWidget	1395
Digikam::ExposureDetector	1398
Digikam::ExposureSettingsContainer	1399
Digikam::FaceClassifier	1400
Digikam::FaceClassifierBase	1402
Digikam::FaceDb	1403
Digikam::FaceDbAccess	1405
Digikam::FaceDbAccessUnlock	1406
Digikam::FaceDbBackend	1407
Digikam::FaceDbOperationGroup	
When you intend to execute a number of write operations to the database, group them while holding a FaceDbOperationGroup	1411
Digikam::FaceDbSchemaUpdater	1412
Digikam::FaceDetector	1412
Digikam::FaceGroup	1414
Digikam::FaceItem	1417
Digikam::FaceItemRetriever	1420
Digikam::FacePipeline	1421
Digikam::FacePipelineBase	1426
Digikam::FacePipelineDetect	1430
Digikam::FacePipelineDetectRecognize	1434
Digikam::FacePipelineEdit	1438
Digikam::FacePipelineExtendedPackage	1443
Digikam::FacePipelineFaceTagsIface	1445
Digikam::FacePipelineFaceTagsIfaceList	1448
Digikam::FacePipelinePackage	1449
Digikam::FacePipelinePackageBase	1450
Digikam::FacePipelineRecognize	1452
Digikam::FacePipelineReset	1456
Digikam::FacePipelineRetrain	1460
Digikam::FacePreprocessor	1464
Digikam::FacePreviewLoader	1465
Digikam::FaceRejectionOverlay	1471
Digikam::FaceRejectionOverlayButton	1476
Digikam::FaceScanSettings	1478
Digikam::FaceScanWidget	1480
Digikam::FacesDetector	1483
Digikam::FacesEngine	1486
Digikam::FaceTags	1489
Digikam::FaceTagsEditor	1491
Digikam::FaceTagsIface	1496
Digikam::FaceUtils	1499
Digikam::FacialRecognitionWrapper	1503
Digikam::FFmpegBinary	1506
Digikam::FFmpegConfigHelper	1508

Digikam::FFmpegLauncher	1510
Digikam::FieldQueryBuilder	1512
Digikam::FileActionItemInfoList	1513
Digikam::FileActionMngr	1515
Digikam::FileActionMngrDatabaseWorker	1517
Digikam::FileActionMngrFileWorker	1521
Digikam::FileActionProgress	1524
Digikam::FileActionProgressItemContainer	1527
Digikam::FileActionProgressItemCreator	1528
Digikam::FilePropertiesOption	1529
Digikam::FileReadLocker	1531
Digikam::FileReadWriteLockKey	1531
Digikam::FileSaveConflictBox	1532
Digikam::FileSaveOptionsBox	1533
Digikam::FileSaveOptionsDlg	1535
Digikam::FilesDownloader	1536
Digikam::FileWorkerInterface	1537
Digikam::FileWriteLocker	1539
Digikam::FilmContainer	1539
Digikam::FilmContainer::ListItem	1540
Digikam::FilmFilter	1541
Digikam::FilmGrainContainer	1545
Digikam::FilmGrainFilter	1546
Digikam::FilmGrainSettings	1550
Digikam::Filter	1551
Digikam::FilterAction	1552
Digikam::FilterActionFilter	1557
Digikam::FiltersHistoryWidget	1563
Digikam::FilterSideBarWidget	
Sidebar widget containing the all filter widgets	1564
Digikam::FilterStatusBar	1568
Digikam::FindDuplicatesAlbum	
The FindDuplicatesAlbum class Widgets used to show all reference images	1568
Digikam::FindDuplicatesAlbumItem	1570
Digikam::FindDuplicatesView	1571
Digikam::FingerPrintsGenerator	1572
Digikam::FingerprintsTask	1576
Digikam::FirstRunDlg	1578
Digikam::FocusPoint	1578
Digikam::FocusPointGroup	1580
Digikam::FocusPointItem	1582
Digikam::FocusPointsExtractor	1585
Digikam::FocusPointsWriter	1586
Digikam::FrameOsd	1586
Digikam::FrameOsdSettings	1587
Digikam::FrameOsdWidget	1588
Digikam::FrameUtils	1588
Digikam::FreeRotationContainer	1588
Digikam::FreeRotationFilter	1590
Digikam::FreeRotationSettings	1594
Digikam::FreeSpaceToolTip	1596
Digikam::FreeSpaceWidget	1598
Digikam::FullObjectDetection	1599
Digikam::FullScreenSettings	1600
Digikam::FuzzySearchSideBarWidget	1601
Digikam::FuzzySearchView	1605
Digikam::GeoCoordinates	1607
Digikam::GeodeticCalculator	1609

Digikam::GeoDragDropHandler	1616
Digikam::GeofaceCluster	1616
Digikam::GeofaceGlobalObject	
Global object for geolocation interface to hold items common to all geolocation interface Widget instances	1617
Digikam::GeofaceInternalWidgetInfo	
Class to hold information about map widgets stored in the <a href="#">GeofaceGlobalObject</a>	1619
Digikam::GeofaceSharedData	1620
Digikam::GeolocationFilter	1622
Digikam::GeolocationSettings	1623
Digikam::GeolocationSettingsContainer	
The class <a href="#">GeolocationSettingsContainer</a> encapsulates all Marble related settings	1624
Digikam::GeoModelHelper	
Helper class to access data in models	1625
Digikam::GeoPluginAboutDlg	1628
Digikam::GPCamera	
Gphoto2 camera Implementation of abstract type <a href="#">DKCamera</a>	1629
Digikam::GPSBookmarkModelHelper	1635
Digikam::GPSBookmarkOwner	1638
Digikam::GPSCorrelatorWidget	1639
Digikam::GPSDataContainer	1640
Digikam::GPSDBJobInfo	1641
Digikam::GPSDBJobsThread	1643
Digikam::GPSGeofaceModelHelper	1646
Digikam::GPSItemContainer	1649
Digikam::GPSItemDelegate	1652
Digikam::GPSItemInfo	1652
Digikam::GPSItemInfoSorter	1653
Digikam::GPSItemList	1654
Digikam::GPSItemListContextMenu	1656
Digikam::GPSItemListDragDropHandler	1657
Digikam::GPSItemModel	1658
Digikam::GPSItemSortProxyModel	1660
Digikam::GPSJob	1661
Digikam::GPSLinkItemSelectionModel	
Makes it possible to share a selection in multiple views which do not have the same source model	1663
Digikam::GPSMarkerTiler	
Marker model for storing data needed to display markers on the map	1664
Digikam::GPSModelIndexProxyMapper	
This class facilitates easy mapping of indexes and selections through proxy models	1671
Digikam::GPSSearchSideBarWidget	1673
Digikam::GPSSearchView	1677
Digikam::GPSUndoCommand	1680
Digikam::GPSUndoCommand::UndoInfo	1681
Digikam::Graph< VertexProperties, EdgeProperties >	
The graph base class template	1681
Digikam::Graph< VertexProperties, EdgeProperties >::DominatorTree	1689
Digikam::Graph< VertexProperties, EdgeProperties >::Edge	1689
Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch	1690
Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch::BreadthFirstSearchVisitor	1691
Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch::CommonVisitor	1692
Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch::DepthFirstSearchVisitor	1693
Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch::lessThanMapEdgeToTarget< GraphType, VertexLessThan	1694
Digikam::Graph< VertexProperties, EdgeProperties >::Path	
Helper class to find paths through the graph	1694



Digikam::Graph< VertexProperties, EdgeProperties >::Vertex	
These two classes provide source-compatible wrappers for the vertex and edge descriptors, providing default construction to null and the isNull() method	1695
Digikam::GraphicsDImgItem	1696
Digikam::GraphicsDImgView	1698
Digikam::GreycstorationContainer	1700
Digikam::GreycstorationFilter	1701
Digikam::GreycstorationSettings	1706
Digikam::GroupedImagesFinder	1707
Digikam::GroupIndicatorOverlay	1708
Digikam::GroupIndicatorOverlayWidget	1712
Digikam::GroupingViewImplementation	1713
Digikam::GroupItemFilterSettings	1714
Digikam::GroupStateComputer	1714
Digikam::Haar::Calculator	1715
Digikam::Haar::ImageData	1715
Digikam::Haar::SignatureData	1716
Digikam::Haar::SignatureMap	
This class provides very fast lookup if a certain pixel is set (positive or negative) in the loaded coefficient set	1716
Digikam::Haar::WeightBin	1716
Digikam::Haar::Weights	1717
Digikam::Haarface	1717
Digikam::HaarProgressObserver	1721
Digikam::HidingStateChanger	1722
Digikam::Highlighter	1725
Digikam::HistogramBox	1726
Digikam::HistogramPainter	
A class that paints a histogram on a QPixmap	1727
Digikam::HistogramWidget	1731
Digikam::HistoryEdgeProperties	
Every edge has one associated object of this class	1733
Digikam::HistoryImageld	1733
Digikam::HistoryVertexProperties	
Every vertex has one associated object of this class	1735
Digikam::HotPixelContainer	1736
Digikam::HotPixelFixer	1737
Digikam::HotPixelProps	1741
Digikam::HotPixelSettings	1742
Digikam::HotPixelsWeights	1743
Digikam::HoverButtonDelegateOverlay	1744
Digikam::HSLContainer	1748
Digikam::HSLFilter	1749
Digikam::HSLSettings	1753
Digikam::HSPreviewWidget	1754
Digikam::HTMLWidget	1755
Digikam::HTMLWidgetPage	1756
Digikam::lccManager	1757
Digikam::lccPostLoadingManager	1760
Digikam::lccPreviewWidget	1762
Digikam::lccProfile	1763
Digikam::lccProfileInfoDlg	1766
Digikam::lccProfilesComboBox	1767
Digikam::lccProfilesMenuAction	1770
Digikam::lccProfilesSettings	1772
Digikam::lccProfileWidget	1774
Digikam::lccRenderingIntentComboBox	1777
Digikam::lccSettings	1778

Digikam::ICCSettingsContainer	1780
Digikam::IccTransform	1781
Digikam::IccTransformFilter	1784
Digikam::Identity	1789
Digikam::IdentityProvider	1790
Digikam::ImageChangeset	1791
Digikam::ImageCommonContainer	1792
Digikam::ImageCurves	1792
Digikam::ImageDialog	1795
Digikam::ImageDialogIconProvider	1796
Digikam::ImageDialogPreview	1797
Digikam::ImageDialogToolTip	1798
Digikam::ImageGuideWidget	1800
Digikam::ImageHistogram	1802
Digikam::ImageHistoryEntry	1804
Digikam::ImageIface	1805
Digikam::ImageLevels	1808
Digikam::ImageListProvider	
This class provides access to a list of unspecified entities, where for each entry a QImage can be provided	1808
Digikam::ImageMetadataContainer	1810
Digikam::ImagePreviewItem	1811
Digikam::ImageQualityCalculator	1812
Digikam::ImageQualityCalculator::ResultDetection	1813
Digikam::ImageQualityConfSelector	1813
Digikam::ImageQualityContainer	1814
Digikam::ImageQualityParser	1815
Digikam::ImageQualitySettings	1816
Digikam::ImageQualitySorter	1817
Digikam::ImageQualityTask	1821
Digikam::ImageQualityThread	1823
Digikam::ImageQualityThreadPool	1824
Digikam::ImageRegionItem	1825
Digikam::ImageRegionWidget	1827
Digikam::ImageRelation	1829
Digikam::ImageSortFilterModel	1830
Digikam::ImageTagChangeset	1833
Digikam::ImageTagProperty	1834
Digikam::ImageTagPropertyName	1834
Digikam::ImageWindow	1835
Digikam::ImageZoomSettings	1841
Digikam::ImportCategorizedView	1844
Digikam::ImportCategoryDrawer	1852
Digikam::ImportContextMenuHelper	1855
Digikam::ImportCoordinatesOverlay	1861
Digikam::ImportDelegate	1865
Digikam::ImportDownloadOverlay	1872
Digikam::ImportDragDropHandler	1876
Digikam::ImportFilterComboBox	1878
Digikam::ImportFilterDlg	1879
Digikam::ImportFilterModel	1881
Digikam::ImportIconView	1887
Digikam::ImportItemModel	1895
Digikam::ImportItemPropertiesSideBarImport	1902
Digikam::ImportItemPropertiesTab	1907
Digikam::ImportLockOverlay	1909
Digikam::ImportNormalDelegate	1913
Digikam::ImportOverlayWidget	1918

Digikam::ImportPreviewView	1919
Digikam::ImportRatingOverlay	1922
Digikam::ImportRenameParser	1927
Digikam::ImportRotateOverlay	1928
Digikam::ImportRotateOverlayButton	1933
Digikam::ImportSettings	1936
Digikam::ImportSortFilterModel	1939
Digikam::ImportStackedView	1942
Digikam::ImportThumbnailBar	1944
Digikam::ImportThumbnailDelegate	1951
Digikam::ImportThumbnailModel	1957
Digikam::ImportUI	1962
Digikam::ImportView	1966
Digikam::InfoDlg	1968
Digikam::InfraredContainer	1969
Digikam::InfraredFilter	1970
Digikam::InitializationObserver	1975
Digikam::InsertBookmarksCommand	1976
Digikam::InternalTagName	1977
Digikam::InvertFilter	1978
Digikam::IOFileSettings	1982
Digikam::IOJob	1983
Digikam::IOJobData	1984
Digikam::IOJobsManager	1986
Digikam::IOJobsThread	1988
Digikam::IptcCoreContactInfo	1992
Digikam::IptcCoreLocationInfo	1992
Digikam::IptcMetaEngineMergeHelper	1993
Digikam::IptcWidget	1994
Digikam::ItemAlbumFilterModel	1997
Digikam::ItemAlbumModel	2002
Digikam::ItemAttributesWatch	2008
Digikam::ItemCategorizedView	2010
Digikam::ItemCategoryDrawer	2018
Digikam::ItemChangeHint	2020
Digikam::ItemComments	2021
Digikam::ItemCoordinatesOverlay	2026
Digikam::ItemCopyMoveHint	2029
Digikam::ItemCopyright	2030
Digikam::ItemDelegate	2035
Digikam::ItemDelegateOverlay	2042
Digikam::ItemDelegateOverlayContainer	2045
Digikam::ItemDescEditTab	2047
Digikam::ItemDragDropHandler	2050
Digikam::ItemExtendedProperties	2052
Digikam::ItemFaceDelegate	2055
Digikam::ItemFilterModel	2061
Digikam::ItemFilterModelFilterer	2069
Digikam::ItemFilterModelPrepareHook	2071
Digikam::ItemFilterModelPreparer	2072
Digikam::ItemFilterModelWorker	2075
Digikam::ItemFilterSettings	2077
Digikam::ItemFiltersHistoryItemDelegate	2079
Digikam::ItemFiltersHistoryModel	2080
Digikam::ItemFiltersHistoryTreeItem	2081
Digikam::ItemFullScreenOverlay	2082
Digikam::ItemFullScreenOverlayButton	2087
Digikam::ItemGPS	2090

Digikam::ItemGPSModelHelper	2094
Digikam::ItemHistoryGraph	2096
Digikam::ItemHistoryGraphData	2100
Digikam::ItemHistoryGraphModel	2105
Digikam::ItemIconView	2108
Digikam::ItemInfo	
Provides access to the database for a single image	2113
Digikam::ItemInfoAlbumsJob	2123
Digikam::ItemInfoCache	2124
Digikam::ItemInfoData	2126
Digikam::ItemInfoJob	2128
Digikam::ItemInfoList	2129
Digikam::ItemInfoReadLocker	2130
Digikam::ItemInfoSet	
A container of associated <a href="#">ItemInfo</a> and queue id	2130
Digikam::ItemInfoStatic	2130
Digikam::ItemInfoTaskSplitter	2131
Digikam::ItemInfoWriteLocker	2133
Digikam::ItemListDragDropHandler	2134
Digikam::ItemLister	2134
Digikam::ItemListerJobGrowingPartsSendingReceiver	2137
Digikam::ItemListerJobPartsSendingReceiver	2139
Digikam::ItemListerJobReceiver	2141
Digikam::ItemListerReceiver	2143
Digikam::ItemListerRecord	2144
Digikam::ItemListerValueListReceiver	2145
Digikam::ItemListModel	2147
Digikam::ItemMarkerTiler	2153
Digikam::ItemMetadataAdjustmentHint	2158
Digikam::ItemModel	2160
Digikam::ItemPosition	2168
Digikam::ItemPreviewCanvas	2172
Digikam::ItemPreviewView	2175
Digikam::ItemPropertiesColorsTab	2178
Digikam::ItemPropertiesGPSTab	2179
Digikam::ItemPropertiesHistoryTab	2180
Digikam::ItemPropertiesMetadataTab	2181
Digikam::ItemPropertiesSideBar	2182
Digikam::ItemPropertiesSideBarDB	2187
Digikam::ItemPropertiesTab	2193
Digikam::ItemPropertiesVersionsTab	2197
Digikam::ItemQueryBuilder	2198
Digikam::ItemQueryPostHook	2198
Digikam::ItemQueryPostHooks	2199
Digikam::ItemRatingOverlay	2200
Digikam::ItemRotateOverlay	2205
Digikam::ItemRotateOverlayButton	2210
Digikam::ItemScanInfo	2212
Digikam::ItemScanner	2212
Digikam::ItemSelectionOverlay	2219
Digikam::ItemSelectionOverlayButton	2223
Digikam::ItemSelectionPropertiesTab	2226
Digikam::ItemShortInfo	2228
Digikam::ItemSortCollator	2228
Digikam::ItemSortSettings	2229
Digikam::ItemTagPair	2232
Digikam::ItemThumbnailBar	2234
Digikam::ItemThumbnailDelegate	2242

Digikam::ItemThumbnailModel	2248
Digikam::ItemVersionsModel	2255
Digikam::ItemViewCategorized	2256
Digikam::ItemViewDelegate	2263
Digikam::ItemViewHoverButton	2268
Digikam::ItemViewImportDelegate	2271
Digikam::ItemViewToolTip	2277
Digikam::ItemViewUtilities	2279
Digikam::ItemVisibilityController	2281
Digikam::ItemVisibilityControllerPropertyObject	2285
Digikam::JPEGUtils::digikam_source_mgr	2286
Digikam::JPEGUtils::JpegRotator	2286
Digikam::KDNNodeBase	2289
Digikam::KDNNodeBase::NodeCompareResult	2290
Digikam::KDNNodeOpenFace	2291
Digikam::KDNNodeSFace	2293
Digikam::KDTreeBase	2295
Digikam::KDTreeOpenFace	2297
Digikam::KDTreeSFace	2298
Digikam::KeywordSearchReader	2299
Digikam::KeywordSearchWriter	2301
Digikam::LabelsSideBarWidget	2303
Digikam::LabelsTreeView	2307
Digikam::LanguagesList	2311
Digikam::LcmsLock	2311
Digikam::LensDistortionFilter	2312
Digikam::LensDistortionPixelAccess	
LensDistortionPixelAccess class: solving the eternal problem: random, cubic-interpolated, sub-pixel coordinate access to an image	2316
Digikam::LensFunCameraSelector	2317
Digikam::LensFunContainer	2318
Digikam::LensFunFilter	2319
Digikam::LensFunIface	2323
Digikam::LensFunSettings	2324
Digikam::LevelsContainer	2325
Digikam::LevelsFilter	2326
Digikam::LibsInfoDlg	2331
Digikam::LightTablePreview	2333
Digikam::LightTableThumbBar	2337
Digikam::LightTableView	2345
Digikam::LightTableWindow	2347
Digikam::ListItem	2351
Digikam::ListViewComboBox	2353
Digikam::LoadingCache	2356
Digikam::LoadingCache::CacheLock	2359
Digikam::LoadingCacheFileWatch	2360
Digikam::LoadingCacheInterface	2361
Digikam::LoadingDescription	2362
Digikam::LoadingDescription::PostProcessingParameters	2364
Digikam::LoadingDescription::PreviewParameters	2365
Digikam::LoadingProcess	2366
Digikam::LoadingProcessListener	2367
Digikam::LoadingTask	2368
Digikam::LoadSaveFileInfoProvider	2370
Digikam::LoadSaveNotifier	2372
Digikam::LoadSaveTask	2374
Digikam::LoadSaveThread	2376
Digikam::LocalContrastContainer	2382

Digikam::LocalContrastFilter	2383
Digikam::LocalContrastSettings	2387
Digikam::LocalizeConfig	2388
Digikam::LocalizeContainer	
The class <a href="#">LocalizeContainer</a> encapsulates all spell-check and localize related settings	2389
Digikam::LocalizeSelector	2390
Digikam::LocalizeSelectorList	2391
Digikam::LocalizeSettings	2392
Digikam::LookupAltitude	2394
Digikam::LookupAltitude::Request	2395
Digikam::LookupAltitudeGeonames	2396
Digikam::LookupFactory	2398
Digikam::MaintenanceData	2399
Digikam::MaintenanceDlg	2399
Digikam::MaintenanceMgr	2400
Digikam::MaintenanceSettings	2400
Digikam::MaintenanceThread	2403
Digikam::MaintenanceTool	2405
Digikam::MakerNoteWidget	2408
Digikam::ManagedLoadSaveThread	2411
Digikam::MapBackend	2418
Digikam::MapDragData	2420
Digikam::MapDragDropHandler	2421
Digikam::MapViewModelHelper	2423
Digikam::MapWidget	
The central map view class of geolocation interface	2426
Digikam::MapWidgetView	
Class containing digiKam's central map view	2433
Digikam::Mat	
Mat:	2437
Digikam::MdKeyListViewItem	2437
Digikam::MediaPlayerView	2438
Digikam::MetadataHub	2439
Digikam::MetadataHubMgr	2445
Digikam::MetadataKeys	2446
Digikam::MetadataListView	2448
Digikam::MetadataListViewItem	2449
Digikam::MetadataOption	2450
Digikam::MetadataOptionDialog	2453
Digikam::MetadataPage	2454
Digikam::MetadataPanel	2455
Digikam::MetadataRemove	2457
Digikam::MetadataRemoveTask	2461
Digikam::MetadataSelector	2463
Digikam::MetadataSelectorItem	2464
Digikam::MetadataSelectorView	2465
Digikam::MetadataStatusBar	2466
Digikam::MetadataSynchronizer	2467
Digikam::MetadataSyncTask	2471
Digikam::MetadataWidget	2473
Digikam::MetaEngine	2475
Digikam::MetaEngineData	2505
Digikam::MetaEngineMergeHelper< Data, Key, KeyString, KeyStringList >	2506
Digikam::MetaEnginePreviews	2507
Digikam::MetaEngineRotation	2508
Digikam::MetaEngineSettings	2510
Digikam::MetaEngineSettingsContainer	
The class <a href="#">MetaEngineSettingsContainer</a> encapsulates all metadata related settings	2511

Digikam::MigrateFromDigikam4Page	2513
Digikam::MimeFilter	2514
Digikam::MixerContainer	2515
Digikam::MixerFilter	2516
Digikam::MixerSettings	2520
Digikam::MLClassifierFoundation	2522
Digikam::MLClassifierFoundation::VotingGroups	2523
Digikam::MLClassifierFoundation::VotingGroups::VoteTally	2523
Digikam::MLPipelineFoundation	2524
Digikam::MLPipelineFoundation::_MLPipelinePerformanceProfile	2527
Digikam::MLPipelinePackageFoundation	2528
Digikam::MLPipelinePackageNotify	2529
Digikam::ModelCompleter	2530
Digikam::ModelIndexedBasedComboBox	2532
Digikam::ModelMenu	
A QMenu that is dynamically populated from a QAbstractItemModel	2533
Digikam::Modifier	2535
Digikam::MonthWidget	2538
Digikam::MysqlAdminBinary	2539
Digikam::MysqlInitBinary	2542
Digikam::MysqlServerBinary	2545
Digikam::MysqlUpgradeBinary	2548
Digikam::NamespaceEditDlg	2550
Digikam::NamespaceEntry	
Provide a simple container for dmetadata namespaces variables, such as names, what types of data expects and extra xml tags	2551
Digikam::NamespaceListView	2552
Digikam::NetworkManager	2553
Digikam::NewItemFinder	2555
Digikam::NoDuplicatesImportFilterModel	2559
Digikam::NoDuplicatesItemFilterModel	2562
Digikam::NoiseDetector	2564
Digikam::NonDeterministicRandomData	2565
Digikam::NormalizeFilter	2567
Digikam::NormalSearchTreeView	
Tree view for all saved "normal" searches	2571
Digikam::NRContainer	2578
Digikam::NREstimate	2580
Digikam::NRFilter	2585
Digikam::NRSettings	2589
Digikam::OilPaintFilter	2591
Digikam::OnlineVersionChecker	2595
Digikam::OnlineVersionDlg	2597
Digikam::OnlineVersionDwnl	2598
Digikam::OpenCVDNNFaceDetector	2598
Digikam::OpenCVDNNFaceRecognizer	2600
Digikam::OpenfacePreprocessor	2601
Digikam::OpenFilePage	2602
Digikam::Option	2603
Digikam::OverlayWidget	
This is a widget that can align itself with another one, without using a layout, so that it can actually be on top of other widgets	2605
Digikam::PackageLoadingDescriptionList	2607
Digikam::PAlbum	
A Physical Album representation	2608
Digikam::PanIconFrame	
Frame with popup menu behavior to host <a href="#">PanIconWidget</a>	2611
Digikam::PanIconWidget	2613

Digikam::ParallelAdapter< A >	2615
Digikam::ParallelPipes	2618
Digikam::ParallelWorkers	2620
Digikam::Parser	2623
Digikam::ParseResults	2624
Digikam::ParseSettings	2625
Digikam::PeopleSideBarWidget	2626
Digikam::PersistentWidgetDelegateOverlay	2630
Digikam::PhotoInfoContainer	2635
Digikam::PickLabelFilter	2636
Digikam::PickLabelMenuAction	2638
Digikam::PickLabelSelector	2639
Digikam::PickLabelWidget	2640
Digikam::PlaceholderWidget	2642
Digikam::PointTransformAffine	2642
Digikam::PositionKeys	2643
Digikam::PreviewList	2645
Digikam::PreviewListItem	2646
Digikam::PreviewLoadingTask	2647
Digikam::PreviewLoadThread	2650
Digikam::PreviewPage	2657
Digikam::PreviewSettings	2658
Digikam::PreviewThreadWrapper	2659
Digikam::PreviewToolBar	2660
Digikam::ProcessLauncher	2662
Digikam::ProgressItem	2663
Digikam::ProgressManager	
The <a href="#">ProgressManager</a> singleton keeps track of all ongoing transactions and notifies observers (progress dialogs) when their progress percent value changes, when they are completed (by their owner), and when they are canceled	2670
Digikam::ProgressView	2678
Digikam::ProxyClickLineEdit	2680
Digikam::ProxyLineEdit	2683
Digikam::QListImageListProvider	
A wrapper implementation for <a href="#">ImageListProvider</a> if you have a QList of QImages	2685
Digikam::QMapForAdaptors< Key, Value >	
Adds the necessary typedefs so that <code>associative_property_map</code> accepts a QMap, and it can be used as a Boost Property Map	2687
Digikam::QueueListView	2688
Digikam::QueueListViewItem	2690
Digikam::QueueMgrWindow	2691
Digikam::QueuePool	2695
Digikam::QueuePoolBar	2697
Digikam::QueueSettings	
This container host all common settings used by a queue, not including assigned batch tools	2697
Digikam::QueueSettingsView	2698
Digikam::QueueToolTip	2699
Digikam::RainDropFilter	2701
Digikam::RandomNumberGenerator	
This class differs from standard pseudo random number generators ( <code>rand()</code> ) in these points:	2705
Digikam::RangeDialog	2707
Digikam::RangeModifier	2709
Digikam::RatingBox	2712
Digikam::RatingComboBox	2714
Digikam::RatingComboBoxDelegate	2716
Digikam::RatingComboBoxModel	2717
Digikam::RatingComboBoxWidget	2718
Digikam::RatingFilter	2721



Digikam::RatingFilterWidget	2723
Digikam::RatingMenuAction	2725
Digikam::RatingStarDrawer	2726
Digikam::RatingWidget	2727
Digikam::RawCameraDlg	2729
Digikam::RawPage	2730
Digikam::RawProcessingFilter	
This is a special filter	2731
Digikam::RecognitionBenchmark	2738
Digikam::RecognitionBenchmark::Statistics	2740
Digikam::RecognitionPreprocessor	2741
Digikam::RecognitionTrainingProvider	
A simple QImage training data container used by RecognitionDatabase::train(Identity, QImage, QString)	2741
Digikam::RecognitionTrainingUpdateQueue	2743
Digikam::RecognitionWorker	2744
Digikam::RedEye::RegressionTree	2747
Digikam::RedEye::ShapePredictor	2747
Digikam::RedEye::SplitFeature	2748
Digikam::RedEyeCorrectionContainer	2748
Digikam::RedEyeCorrectionFilter	2749
Digikam::RedEyeCorrectionSettings	2753
Digikam::RefocusFilter	2755
Digikam::RefocusMatrix	2759
Digikam::RegionFrameItem	2760
Digikam::RemoveBookmarksCommand	2764
Digikam::RemoveDoublesModifier	2765
Digikam::RemoveFilterAction	2768
Digikam::RenameCustomizer	2769
Digikam::RenameFileJob	2770
Digikam::ReplaceDialog	2772
Digikam::ReplaceModifier	2773
Digikam::RestoreDTrashItemsJob	2776
Digikam::RGBBackend	
This class is a base class for Open Street Map and Geonames backends	2778
Digikam::RGInfo	
This class contains data needed in reverse geocoding process	2779
Digikam::RGTagModel	
The model that holds data for the tag tree displayed in ReverseGeocodingWidget	2780
Digikam::RGWidget	
Main widget for reverse geocoding	2787
Digikam::RubberItem	2791
Digikam::Rule	2794
Digikam::RuleDialog	2798
Digikam::SafeTemporaryFile	2799
Digikam::SAlbum	
A Search Album representation	2799
Digikam::SaveProperties	2803
Digikam::SavingContext	2804
Digikam::SavingTask	2805
Digikam::ScanController	2807
Digikam::ScanController::FileMetadataWrite	
When writing metadata to the file, the file content on disk changes, but the information is taken from the database; therefore, the resulting scanning process can be optimized	2812
Digikam::ScanStateFilter	2814
Digikam::ScriptingSettings	2817
Digikam::SearchChangeset	2817
Digikam::SearchesDBJobInfo	2818

Digikam::SearchesDBJobsThread	2820
Digikam::SearchesJob	2823
Digikam::SearchField	2825
Digikam::SearchFieldAlbum	2827
Digikam::SearchFieldCheckBox	2831
Digikam::SearchFieldChoice	2835
Digikam::SearchFieldColorDepth	2839
Digikam::SearchFieldComboBox	2842
Digikam::SearchFieldGroup	2845
Digikam::SearchFieldGroupLabel	2846
Digikam::SearchFieldKeyword	2848
Digikam::SearchFieldLabels	2851
Digikam::SearchFieldMonthDay	2855
Digikam::SearchFieldPageOrientation	2859
Digikam::SearchFieldRangeDate	2862
Digikam::SearchFieldRangeDouble	2866
Digikam::SearchFieldRangeInt	2870
Digikam::SearchFieldRangeTime	2874
Digikam::SearchFieldRating	2878
Digikam::SearchFieldText	2882
Digikam::SearchFilterModel	
Filter model for searches that can filter by search type	2885
Digikam::SearchGroup	2890
Digikam::SearchGroupLabel	2893
Digikam::SearchInfo	
A container class for transporting search information from the database to <a href="#">AlbumManager</a>	2894
Digikam::SearchModel	2895
Digikam::SearchModificationHelper	
Utility class providing methods to modify search albums ( <a href="#">SAAlbum</a> ) in a way useful to implement views	2901
Digikam::SearchSideBarWidget	2907
Digikam::SearchTabHeader	2911
Digikam::SearchTextBar	
A text input for searching entries with visual feedback	2912
Digikam::SearchTextBarDb	
A text input for searching entries with visual feedback	2915
Digikam::SearchTextFilterSettings	2919
Digikam::SearchTextSettings	2920
Digikam::SearchTreeView	2921
Digikam::SearchView	2927
Digikam::SearchViewBottomBar	2930
Digikam::SearchViewThemedPartsCache	2931
Digikam::SearchWindow	2932
Digikam::SearchXmlCachingReader	2934
Digikam::SearchXmlReader	2937
Digikam::SearchXmlWriter	2940
Digikam::SequenceNumberDialog	2943
Digikam::SequenceNumberOption	2945
Digikam::Setup	2948
Digikam::SetupAlbumView	2951
Digikam::SetupCamera	2952
Digikam::SetupCategory	2953
Digikam::SetupCollectionDelegate	2954
Digikam::SetupCollectionModel	2958
Digikam::SetupCollectionModel::Item	2961
Digikam::SetupCollections	2962
Digikam::SetupCollectionTreeView	2963
Digikam::SetupDatabase	2964

Digikam::SetupEditor	2965
Digikam::SetupEditorIface	2966
Digikam::SetupGeolocation	2967
Digikam::SetupICC	2968
Digikam::SetupImageQualitySorter	2969
Digikam::SetupIOFiles	2970
Digikam::SetupLightTable	2970
Digikam::SetupMetadata	2971
Digikam::SetupMime	2972
Digikam::SetupMisc	2973
Digikam::SetupPlugins	2974
Digikam::SetupRaw	2975
Digikam::SetupTemplate	2976
Digikam::SetupToolTip	2977
Digikam::SetupVersioning	2978
Digikam::SharedLoadingTask	2979
Digikam::SharedLoadSaveThread	2983
Digikam::SharedQueue< T >	2987
Digikam::SharpContainer	2987
Digikam::SharpenFilter	2989
Digikam::SharpSettings	2993
Digikam::ShearFilter	2995
Digikam::ShowHideVersionsOverlay	3000
Digikam::Sidebar	
This class handles a sidebar view	3004
Digikam::SidebarSplitter	3009
Digikam::SidebarWidget	
Abstract base class for widgets that are use in one of digikams's sidebars	3011
Digikam::SidecarFinder	3014
Digikam::SimilarityDb	3014
Digikam::SimilarityDbAccess	3020
Digikam::SimilarityDbBackend	3022
Digikam::SimilarityDbSchemaUpdater	3026
Digikam::SimpleTreeModel	3027
Digikam::SimpleTreeModel::Item	3028
Digikam::SinglePhotoPreviewLayout	3029
Digikam::SketchWidget	3031
Digikam::SlideVideo	3033
Digikam::SoftProofDialog	3034
Digikam::SolidHardwareDlg	3035
Digikam::SpellCheckConfig	3036
Digikam::SqueezedComboBox	
This widget is a QComboBox, but then a little bit different	3036
Digikam::StackedView	3041
Digikam::StartScanPage	3043
Digikam::StateSavingObject	
An interface-like class with utility methods and a general public interface to support state saving and restoring for objects via KConfig	3044
Digikam::StatusbarProgressWidget	3049
Digikam::StatusProgressBar	3050
Digikam::StayPoppedUpComboBox	3052
Digikam::StretchFilter	3055
Digikam::StyleSheetDebugger	3059
Digikam::SubjectData	3060
Digikam::SubjectEdit	3061
Digikam::SubjectWidget	3063
Digikam::SyncJob	3064
Digikam::SystemSettings	3065

Digikam::SystemSettingsWidget	3066
Digikam::TableView	3067
Digikam::TableViewColumn	3071
Digikam::TableViewColumnConfiguration	3073
Digikam::TableViewColumnConfigurationWidget	3074
Digikam::TableViewColumnDescription	3074
Digikam::TableViewColumnFactory	3075
Digikam::TableViewColumnProfile	3076
Digikam::TableViewColumns::ColumnAudioVideoProperties	3077
Digikam::TableViewColumns::ColumnDigikamProperties	3081
Digikam::TableViewColumns::ColumnFileConfigurationWidget	3084
Digikam::TableViewColumns::ColumnFileProperties	3086
Digikam::TableViewColumns::ColumnGeoConfigurationWidget	3089
Digikam::TableViewColumns::ColumnGeoProperties	3091
Digikam::TableViewColumns::ColumnItemProperties	3095
Digikam::TableViewColumns::ColumnPhotoConfigurationWidget	3098
Digikam::TableViewColumns::ColumnPhotoProperties	3100
Digikam::TableViewColumns::ColumnThumbnail	3104
Digikam::TableViewConfigurationDialog	3107
Digikam::TableViewItemDelegate	3108
Digikam::TableViewModel	3109
Digikam::TableViewModel::Item	3112
Digikam::TableViewSelectionModeSyncer	3112
Digikam::TableViewShared	3113
Digikam::TableViewTreeView	3114
Digikam::TagChangeset	3116
Digikam::TagCheckView	3117
Digikam::TagCompleter	3125
Digikam::TagData	3126
Digikam::TagDragDropHandler	3126
Digikam::TagEditDlg	3128
Digikam::TagFilterView	
A view to filter the currently displayed album by tags	3129
Digikam::TagFolderView	3138
Digikam::TaggingAction	3146
Digikam::TaggingActionFactory	3146
Digikam::TaggingActionFactory::ConstraintInterface	3148
Digikam::TagInfo	
A container class for transporting tag information from the database to <a href="#">AlbumManager</a>	3149
Digikam::TagList	3149
Digikam::TagMgrListModel	3150
Digikam::TagMgrListView	3152
Digikam::TagMgrTreeView	3153
Digikam::TagModel	3160
Digikam::TagModificationHelper	
Utility class providing methods to modify tag albums ( <a href="#">TAlbum</a> ) in a way useful to implement views	3167
Digikam::TagProperties	3172
Digikam::TagPropertiesFilterModel	
Filter model for tags that can filter by tag property	3174
Digikam::TagProperty	3178
Digikam::TagPropertyName	3178
Digikam::TagPropWidget	3178
Digikam::TagRegion	3179
Digikam::TagsActionMgr	3182
Digikam::TagsCache	3184
Digikam::TagsDBJobInfo	3191
Digikam::TagsDBJobsThread	3192
Digikam::TagsEdit	3195

Digikam::TagShortInfo	3196
Digikam::TagsJob	3196
Digikam::TagsLineEditOverlay	3198
Digikam::TagsManager	3202
Digikam::TagsManagerFilterModel	3205
Digikam::TagsPopupMenu	3209
Digikam::TagTreeView	3210
Digikam::TagTreeViewSelectComboBox	3216
Digikam::TagViewSideBarWidget	3220
Digikam::TAlbum	
A Tag Album representation	3223
Digikam::Template	3227
Digikam::TemplateList	3228
Digikam::TemplateListItem	3229
Digikam::TemplateManager	3230
Digikam::TemplatePanel	3231
Digikam::TemplateSelector	3232
Digikam::TemplateViewer	3234
Digikam::TextFilter	3236
Digikam::TextureContainer	3237
Digikam::TextureFilter	3238
Digikam::TextureSettings	3242
Digikam::ThemeManager	3243
Digikam::ThreadManager	3244
Digikam::ThumbBarDock	
A dock widget specifically designed for thumbnail bars (class ThumbnailView or one of its descendants)	3245
Digikam::ThumbnailAligningDelegate	3247
Digikam::ThumbnailCreator	3247
Digikam::ThumbnailIdentifier	3250
Digikam::ThumbnailImageCatcher	3251
Digikam::ThumbnailInfo	3253
Digikam::ThumbnailInfoProvider	3255
Digikam::ThumbnailLoadingTask	3256
Digikam::ThumbnailLoadThread	3259
Digikam::ThumbnailSize	3269
Digikam::ThumbsDb	3270
Digikam::ThumbsDbAccess	3271
Digikam::ThumbsDbBackend	3272
Digikam::ThumbsDbInfo	3276
Digikam::ThumbsDbInfoProvider	3276
Digikam::ThumbsDbSchemaUpdater	3277
Digikam::ThumbsGenerator	3278
Digikam::ThumbsTask	3281
Digikam::TileGrouper	3283
Digikam::TileIndex	3283
Digikam::TimeAdjustContainer	
Container that store all timestamp adjustments	3284
Digikam::TimeAdjustSettings	3285
Digikam::TimelineSideBarWidget	3287
Digikam::TimeLineWidget	3291
Digikam::TimeZoneComboBox	3293
Digikam::Token	
Token is the smallest parsing unit in AdvancedRename utility	3293
Digikam::TonalityContainer	3295
Digikam::TonalityFilter	3296
Digikam::ToolListViewGroup	3300
Digikam::ToolListViewItem	3301

Digikam::ToolSettingsView	3302
Digikam::ToolsListView	3303
Digikam::ToolsView	3304
Digikam::TooltipCreator	3305
Digikam::TooltipDialog	3305
Digikam::TooltipsPage	3306
Digikam::TrackCorrelator	3307
Digikam::TrackCorrelator::Correlation	3308
Digikam::TrackCorrelator::CorrelationOptions	3308
Digikam::TrackCorrelatorThread	3309
Digikam::TrackListModel	3310
Digikam::TrackManager	3311
Digikam::TrackManager::Track	3313
Digikam::TrackManager::TrackPoint	3313
Digikam::TrackReader	3314
Digikam::TrackReader::TrackReadResult	3314
Digikam::TrainerWorker	3315
Digikam::TrainingDataProvider	
A <a href="#">TrainingDataProvider</a> provides a call-back interface for the training process to retrieve the necessary information	3318
Digikam::TransactionItem	3320
Digikam::TransactionItemView	3322
Digikam::TransitionMngr	3323
Digikam::TransitionPreview	3323
Digikam::TrashView	3324
Digikam::TreeBranch	3326
Digikam::TreeProxyModel	3326
Digikam::TreeViewComboBox	3327
Digikam::TreeViewLineEditComboBox	3330
Digikam::TrimmedModifier	3333
Digikam::TwoProgressItemsContainer	3336
Digikam::UMSCamera	
USB Mass Storage camera Implementation of abstract type <a href="#">DKCamera</a>	3337
Digikam::UndoAction	3343
Digikam::UndoActionIrreversible	3344
Digikam::UndoActionReversible	3345
Digikam::UndoCache	3346
Digikam::UndoManager	3346
Digikam::UndoMetadataContainer	3346
Digikam::UndoState	3347
Digikam::UniqueModifier	3348
Digikam::UnsharpMaskFilter	3351
Digikam::VersionFileInfo	3355
Digikam::VersionFileOperation	3355
Digikam::VersioningPromptUserSaveDialog	3357
Digikam::VersionItemFilterSettings	3357
Digikam::VersionManager	3358
Digikam::VersionManagerSettings	3358
Digikam::VersionNamingScheme	3359
Digikam::VersionsDelegate	3362
Digikam::VersionsTreeView	3365
Digikam::VersionsWidget	3368
Digikam::VideoFrame	3369
Digikam::VideoInfoContainer	3369
Digikam::VideoMetadataContainer	3369
Digikam::VideoStripFilter	3370
Digikam::VideoThumbDecoder	3370
Digikam::VideoThumbnailer	3370

Digikam::VideoThumbWriter	3370
Digikam::VidPlayerDlg	3371
Digikam::VidSlideSettings	3371
Digikam::VidSlideTask	3377
Digikam::VidSlideThread	3379
Digikam::VisibilityController	3381
Digikam::VisibilityObject	3382
Digikam::WBContainer	3383
Digikam::WBFilter	3384
Digikam::WBSettings	3389
Digikam::WebBrowserDlg	3390
Digikam::WebWidget	3391
Digikam::WelcomePage	3392
Digikam::WelcomePageView	3393
Digikam::WelcomePageViewPage	3394
Digikam::WorkerObject	3395
Digikam::Workflow	
This container group all queue common settings plus all assigned batch tools	3398
Digikam::WorkflowDlg	3398
Digikam::WorkflowItem	3399
Digikam::WorkflowList	3400
Digikam::WorkflowManager	3401
Digikam::WorkingWidget	3402
Digikam::WSAlbum	3403
Digikam::WSComboBoxIntermediate	3403
Digikam::WSLoginDialog	3404
Digikam::WSNewAlbumDialog	3405
Digikam::WSSelectUserDlg	3406
Digikam::WSSettings	3407
Digikam::WSSettingsWidget	3409
Digikam::WSToolDialog	3411
Digikam::WSToolUtils	3412
Digikam::XbelReader	3412
Digikam::XbelWriter	3413
Digikam::XmpMetaEngineMergeHelper	3414
Digikam::XmpWidget	3415
ShowFoto::NoDuplicatesShowfotoFilterModel	3418
ShowFoto::Showfoto	3421
ShowFoto::ShowfotoCategorizedView	3427
ShowFoto::ShowfotoCoordinatesOverlay	3435
ShowFoto::ShowfotoCoordinatesOverlayWidget	3438
ShowFoto::ShowfotoDelegate	3440
ShowFoto::ShowfotoDragDropHandler	3446
ShowFoto::ShowfotoFilterModel	3449
ShowFoto::ShowfotoFolderViewBar	3455
ShowFoto::ShowfotoFolderViewBookmarkDlg	3457
ShowFoto::ShowfotoFolderViewBookmarkItem	3458
ShowFoto::ShowfotoFolderViewBookmarkList	3459
ShowFoto::ShowfotoFolderViewBookmarks	3460
ShowFoto::ShowfotoFolderViewList	3461
ShowFoto::ShowfotoFolderViewModel	3462
ShowFoto::ShowfotoFolderViewSideBar	3463
ShowFoto::ShowfotoFolderViewToolTip	3466
ShowFoto::ShowfotoFolderViewUndo	3467
ShowFoto::ShowfotoInfoface	3468
ShowFoto::ShowfotoItemInfo	3470
ShowFoto::ShowfotoItemModel	3472
ShowFoto::ShowfotoItemSortSettings	3478

ShowFoto::ShowfotoItemViewDelegate	3481
ShowFoto::ShowfotoKineticScroller	
Vertical kinetic scroller implementation without overshoot and bouncing	3486
ShowFoto::ShowfotoNormalDelegate	3487
ShowFoto::ShowfotoSettings	3492
ShowFoto::ShowfotoSetup	3494
ShowFoto::ShowfotoSetupMetadata	3497
ShowFoto::ShowfotoSetupMisc	3498
ShowFoto::ShowfotoSetupPlugins	3499
ShowFoto::ShowfotoSetupRaw	3500
ShowFoto::ShowfotoSetupToolTip	3501
ShowFoto::ShowfotoSortFilterModel	3502
ShowFoto::ShowfotoStackViewFavoriteItem	3505
ShowFoto::ShowfotoStackViewFavoriteItemDlg	3507
ShowFoto::ShowfotoStackViewFavoriteList	3508
ShowFoto::ShowfotoStackViewFavorites	3510
ShowFoto::ShowfotoStackViewItem	3511
ShowFoto::ShowfotoStackViewList	3512
ShowFoto::ShowfotoStackViewSideBar	3514
ShowFoto::ShowfotoStackViewToolTip	3517
ShowFoto::ShowfotoThumbnailBar	3519
ShowFoto::ShowfotoThumbnailDelegate	3526
ShowFoto::ShowfotoThumbnailModel	3531



# Chapter 8

## Namespace Documentation

### 8.1 Digikam Namespace Reference

NOTE: This is because of the [CollectionManager](#) private slot.

#### Namespaces

- namespace [Matrix](#)

*If the picture is displayed according to the exif orientation tag, the user will request rotating operations relative to what he sees, and that is the picture rotated according to the EXIF tag.*

#### Classes

- class [AbstractAlbumModel](#)
- class [AbstractAlbumTreeView](#)
  - Base class for all tree views that display Album-based content provided by an [AbstractSpecificAlbumModel](#).*
- class [AbstractAlbumTreeViewSelectComboBox](#)
- class [AbstractCheckableAlbumModel](#)
- class [AbstractCheckableAlbumTreeView](#)
- class [AbstractCountingAlbumModel](#)
- class [AbstractCountingAlbumTreeView](#)
- class [AbstractDetector](#)
- class [AbstractItemDragDropHandler](#)
- class [AbstractMarkerTiler](#)
- class [AbstractSearchGroupContainer](#)
- class [AbstractSpecificAlbumModel](#)
- class [AbstractWidgetDelegateOverlay](#)
- class [ActionCategorizedView](#)
- class [ActionData](#)
- class [ActionItemModel](#)
- class [ActionJob](#)
- class [ActionSortFilterProxyModel](#)
- class [ActionTask](#)
- class [ActionThread](#)
- class [ActionThreadBase](#)
- class [ActionVersionsOverlay](#)

- class [AddBookmarkDialog](#)
- class [AddBookmarkProxyModel](#)
  - Proxy model that filters out the bookmarks so only the folders are left behind.*
- class [AddTagsComboBox](#)
- class [AddTagsLineEdit](#)
- class [AdvancedMetadataTab](#)
- class [AdvancedRenameDialog](#)
- class [AdvancedRenameInput](#)
- class [AdvancedRenameLineEdit](#)
- class [AdvancedRenameListItem](#)
- class [AdvancedRenameManager](#)
- class [AdvancedRenameProcessDialog](#)
- class [AdvancedRenameWidget](#)
- class [AdvancedSettings](#)
- class [AestheticDetector](#)
- class [Akonadiface](#)
- class [Album](#)
  - Abstract base class for all album types.*
- class [AlbumChangeset](#)
- class [AlbumCopyMoveHint](#)
- class [AlbumCustomizer](#)
- class [AlbumDragDropHandler](#)
- class [AlbumFilterModel](#)
- class [AlbumFolderViewSideBarWidget](#)
- class [AlbumHistory](#)
  - Manages the history of the last visited albums.*
- class [AlbumInfo](#)
  - A container class for transporting album information from the database to [AlbumManager](#).*
- class [AlbumIterator](#)
  - Iterate over all children of this [Album](#).*
- class [AlbumLabelsSearchHandler](#)
- class [AlbumManager](#)
  - [AlbumManager](#) manages albums: does listing of albums and controls the lifetime of it.*
- class [AlbumModel](#)
- class [AlbumModelDragDropHandler](#)
- class [AlbumModificationHelper](#)
  - Utility class providing methods to modify physical albums ([PAlbum](#)) in a way useful to implement views.*
- class [AlbumParser](#)
- class [AlbumPointer](#)
  - You can use [AlbumPointer](#) to store a guarded pointer to [Album](#) or one of the subclasses (use template parameter).*
- class [AlbumPointerList](#)
- class [AlbumPropsEdit](#)
- class [AlbumRootChangeset](#)
- class [AlbumRootInfo](#)
- class [AlbumsDBJobInfo](#)
- class [AlbumsDBJobsThread](#)
- class [AlbumSelectComboBox](#)
- class [AlbumSelectDialog](#)
- class [AlbumSelectionTreeView](#)
  - [Album](#) tree view used in the left sidebar to select [PAlbums](#) and perform operations on them via a context menu.*
- class [AlbumSelectors](#)
- class [AlbumSelectTabs](#)
- class [AlbumSelectTreeView](#)

*Enables a simple context menu only for creating a new album.*

- class [AlbumSelectWidget](#)
- class [AlbumShortInfo](#)
- class [AlbumSimplified](#)

*This class is used when parsing response of listAlbums().*

- class [AlbumsJob](#)
- class [AlbumThumbnailLoader](#)
- class [AlbumTreeView](#)
- class [AlbumTreeViewSelectComboBox](#)
- class [AlbumWatch](#)
- class [AltLangStrEdit](#)
- class [AnimatedClearButton](#)
- class [AnimatedVisibility](#)
- class [AntiVignettingContainer](#)
- class [AntiVignettingFilter](#)
- class [AntiVignettingSettings](#)
- class [ApplicationSettings](#)
- class [AssignedBatchTools](#)

*Container to assign Batch tools and settings to an item by Url.*

- class [AssignedListView](#)
- class [AssignedListViewItem](#)
- class [AssignNameOverlay](#)
- class [AssignNameWidget](#)
- class [AssignNameWidgetStates](#)
- class [AudPlayerWdg](#)
- class [AutoCrop](#)
- class [AutoExpoFilter](#)
- class [AutoLevelsFilter](#)
- class [AutoTagsAssign](#)
- class [AutotagsAssignment](#)
- class [AutotagsAssignmentTask](#)
- class [AutoTagsScanSettings](#)
- class [AutoTagsScanWidget](#)
- class [BackendGeonamesRG](#)

*This class calls Geonames' reverse geocoding service.*

- class [BackendGeonamesUSRG](#)

*This class calls Geonames' get address service available only for USA locations.*

- class [BackendGoogleMaps](#)
- class [BackendMarble](#)
- class [BackendMarbleLayer](#)
- class [BackendOsmRG](#)

*This class calls Open Street Map's reverse geocoding service.*

- class [BalooInfo](#)
- class [BalooWrap](#)

*The [BalooWrap](#) class is a singleton class which offer functionality for reading and writing image comment, tags and rating from Baloo to digiKam and from digiKam to Baloo.*

- class [BasicDImgFilterGenerator](#)
- class [BatchTool](#)
- class [BatchToolSet](#)

*A container of associated batch tool and settings.*

- class [BatchToolsFactory](#)
- class [BCGContainer](#)
- class [BCGFilter](#)
- class [BCGSettings](#)

- class [BdEngineBackend](#)
- class [BlackFrameListView](#)
- class [BlackFrameListViewItem](#)
- class [BlackFrameParser](#)
- class [BlackFrameToolTip](#)
- class [BlurDetector](#)
- class [BlurFilter](#)
- class [BlurFXFilter](#)
- class [BookmarkNode](#)
- class [BookmarksDialog](#)
- class [BookmarksManager](#)

*Bookmark manager, owner of the bookmarks, loads, saves and basic tasks.*

- class [BookmarksMenu](#)

*Menu that is dynamically populated from the bookmarks.*

- class [BookmarksModel](#)

*BookmarksModel is a QAbstractItemModel wrapper around the BookmarkManager.*

- class [BorderContainer](#)
- class [BorderFilter](#)
- class [BorderSettings](#)
- class [BqmlInfoface](#)
- class [BuildTrashCountersJob](#)
- class [BWSepiaContainer](#)
- class [BWSepiaFilter](#)
- class [BWSepiaSettings](#)
- class [CameraAutoDetectThread](#)
- class [CameraController](#)
- class [CameraFolderDialog](#)
- class [CameraFolderItem](#)
- class [CameraFolderView](#)
- class [CameraHistoryUpdater](#)
- class [CameraInfoDialog](#)
- class [CameraItem](#)
- class [CameraItemList](#)
- class [CameraList](#)
- class [CameraMessageBox](#)
- class [CameraNameHelper](#)
- class [CameraNameOption](#)
- class [CameraSelection](#)
- class [CameraThumbsCtrl](#)
- class [CameraType](#)
- class [CamlItemInfo](#)
- class [CamlItemSortSettings](#)
- class [Canvas](#)
- class [CaptionEdit](#)
- class [CaptionsMap](#)

*A map used to store a list of Alternative Language values + author and date properties The map key is the language code following RFC3066 notation (like "fr-FR" for French), and the [CaptionsMap](#) value all caption properties.*

- class [CaptionValues](#)
- class [CaptureDlg](#)
- class [CaptureWidget](#)
- class [CaseModifier](#)
- class [CategorizedItemModel](#)
- class [CBContainer](#)
- class [CBFilter](#)

- class [CBSettings](#)
- class [ChangeBookmarkCommand](#)
- class [ChangeFaceRecognitionModelDlg](#)
- class [CharcoalFilter](#)
- class [CheckableAlbumFilterModel](#)

*Filter model for checkable album models that allows more filtering options based on check state.*

- class [ChoiceSearchComboBox](#)
- class [ChoiceSearchModel](#)
- class [CIETongueWidget](#)
- class [ClickDragReleaseItem](#)
- class [ClockPhotoDialog](#)
- struct [CMat](#)

*CMat:*

- class [CollectionImageChangeset](#)
- class [CollectionLocation](#)
- class [CollectionManager](#)
- class [CollectionPage](#)
- class [CollectionScanner](#)
- class [CollectionScannerHintContainer](#)
- class [CollectionScannerObserver](#)
- class [ColorCorrectionDlg](#)
- class [ColorFXContainer](#)
- class [ColorFXFilter](#)
- class [ColorFXSettings](#)
- class [ColorGradientWidget](#)
- class [ColorLabelFilter](#)
- class [ColorLabelMenuAction](#)
- class [ColorLabelSelector](#)
- class [ColorLabelWidget](#)
- class [ComboBoxDelegate](#)
- class [CommentInfo](#)
- class [CommonKeys](#)
- class [CompressionDetector](#)
- class [ContentAwareContainer](#)
- class [ContentAwareFilter](#)
- class [ContextMenuHelper](#)

*A helper class to add actions and special menus to the context menu.*

- class [CoordinatesOverlayWidget](#)
- class [CopyOrMoveJob](#)
- class [CopyrightInfo](#)
- class [CoreDB](#)
- class [CoreDbAccess](#)

*The [CoreDbAccess](#) provides access to the database: Create an instance of this class on the stack to retrieve a pointer to the database.*

- class [CoreDbAccessUnlock](#)
- class [CoreDbBackend](#)
- class [CoreDbCopyManager](#)
- class [CoreDbDownloadHistory](#)
- class [CoreDbNameFilter](#)
- class [CoreDbOperationGroup](#)

*When you intend to execute a number of write operations to the database, group them while holding a [CoreDbOperationGroup](#).*

- class [CoreDbPrivilegesChecker](#)
- class [CoreDbSchemaUpdater](#)

- class [CoreDbTransaction](#)
  - Convenience class: You can create a [CoreDbTransaction](#) object for a scope for which you want to declare a database commit.*
- class [CoreDbUrl](#)
- class [CoreDbWatch](#)
- class [CountrySelector](#)
- class [CurvesBox](#)
- class [CurvesContainer](#)
- class [CurvesFilter](#)
- class [CurvesSettings](#)
- class [CurvesWidget](#)
- class [CustomStepsDoubleSpinBox](#)
- class [CustomStepsIntSpinBox](#)
- class [DAboutData](#)
- class [DAbstractSliderSpinBox](#)
- class [DActiveLabel](#)
  - A widget to host an image into a label with an active url which can be open to default web browser using simple mouse click.*
- class [DAdjustableLabel](#)
  - A label to show text adjusted to widget size.*
- class [DAlbum](#)
  - A Date Album representation.*
- class [DAlbumDrag](#)
  - Provides a drag object for an album.*
- class [DAlbumInfo](#)
- class [DArrowClickLabel](#)
- class [DatabaseCopyThread](#)
- class [DatabaseLoadSaveFileInfoProvider](#)
- class [DatabaseMigrationDialog](#)
- class [DatabaseOption](#)
- class [DatabaseOptionDialog](#)
- class [DatabasePage](#)
- class [DatabaseServer](#)
- class [DatabaseServerError](#)
- class [DatabaseServerStarter](#)
- class [DatabaseSettingsWidget](#)
- class [DatabaseTask](#)
- class [DatabaseWorkerInterface](#)
- class [DatabaseWriter](#)
- class [DateAlbumModel](#)
  - A model for date based albums.*
- class [DateFolderView](#)
- class [DateFolderViewSideBarWidget](#)
- class [DateFormat](#)
- class [DateOption](#)
- class [DateOptionDialog](#)
- class [DatesDBJobInfo](#)
- class [DatesDBJobsThread](#)
- class [DatesJob](#)
- class [DateTreeView](#)
- class [DbCleaner](#)
- class [DbEngineAccess](#)
  - The [DbEngineAccess](#) class provides access to the database: Create an instance of this class on the stack to retrieve a pointer to the database.*

- class [DbEngineAction](#)
- class [DbEngineActionElement](#)
- class [DbEngineActionType](#)

*The [DbEngineActionType](#) is used by the [BdEngineBackend](#) to wrap another data object within an sql statement and controls whether it should be used as field entry or as value (prepared to an sql statement with positional binding).*

- class [DbEngineConfig](#)
- class [DbEngineConfigSettings](#)
- class [DbEngineConfigSettingsLoader](#)
- class [DbEngineConnectionChecker](#)
- class [DbEngineErrorAnswer](#)
- class [DbEngineErrorHandler](#)
- class [DbEngineGuiErrorHandler](#)
- class [DbEngineLocking](#)
- class [DbEngineParameters](#)

*This class encapsulates all parameters needed to establish a connection to a database (inspired by the API of Qt::← Sql).*

- class [DbEngineSqlQuery](#)
- class [DbHeaderListItem](#)
- class [DBinaryIface](#)
- class [DBinarySearch](#)

*This class has nothing to do with a binary search, it is a widget to search for binaries.*

- class [DBInfolface](#)
- class [DBJob](#)
- class [DBJobInfo](#)
- class [DBJobsManager](#)
- class [DBJobsThread](#)
- class [DbKeysCollection](#)

*A class for managing / grouping database keys.*

- class [DbKeySelector](#)
- class [DbKeySelectorItem](#)
- class [DbKeySelectorView](#)
- class [DbShrinkDialog](#)
- class [DBStatDlg](#)
- class [DBusSignalListenerThread](#)
- class [DBusyDlg](#)
- class [DBusyThread](#)
- class [DCameraDragObject](#)

*Provides a drag object for a camera object.*

- class [DCameraltemListDrag](#)
- class [DCategorizedSortFilterProxyModel](#)

*Provides a drag object for a list of camera items.*

*This class lets you categorize a view.*

- class [DCategorizedView](#)

*Item view for listing items.*

- class [DCategoryDrawer](#)

*The category drawing is performed by this class.*

- class [DClickLabel](#)
- class [DColor](#)
- class [DColorComposer](#)
- class [DColorSelector](#)

*A widget to choose a color from a palette.*

- class [DColorValueSelector](#)
- class [DComboBox](#)

- class [DConfigDlg](#)  
*A dialog base class which can handle multiple pages.*
- class [DConfigDlgMgr](#)  
*The [DConfigDlgMgr](#) class provides a means of automatically retrieving, saving and resetting basic settings.*
- class [DConfigDlgModel](#)  
*A base class for a model used by [DConfigDlgView](#).*
- class [DConfigDlgTitle](#)  
*This class provides a widget often used for [DConfigDlg](#) titles.*
- class [DConfigDlgView](#)  
*A base class which can handle multiple pages.*
- class [DConfigDlgWdg](#)  
*Page widget with many layouts (faces).*
- class [DConfigDlgWdgItem](#)  
*[DConfigDlgWdgItem](#) is used by [DConfigDlgWdg](#) and represents a page.*
- class [DConfigDlgWdgModel](#)  
*This page model is used by.*
- class [DCursorTracker](#)  
*This class implements a window which looks like a tool tip.*
- class [DDateEdit](#)  
*A date editing widget that consists of an editable combo box.*
- class [DDatePicker](#)  
*Provides a widget for calendar date input.*
- class [DDatePickerPopup](#)  
*This menu helps the user to select a date quickly.*
- class [DDateTable](#)  
*This is a support class for the [DDatePicker](#) class.*
- class [DDateTimeEdit](#)  
*This class is basically the same as the KDE Date Time widget with the exception that a [QTimeEdit](#) is placed directly besides it.*
- class [DDoubleNumInput](#)
- class [DDoubleSliderSpinBox](#)
- class [DefaultRenameParser](#)
- class [DefaultValueDialog](#)
- class [DefaultValueModifier](#)
- class [DefaultVersionNamingScheme](#)
- class [DeleteDialog](#)
- class [DeleteItem](#)
- class [DeleteItemList](#)
- class [DeleteJob](#)
- class [DeleteWidget](#)
- class [DeltaTime](#)  
*Container that hold the time difference for clock photo dialog.*
- class [DetByClockPhotoButton](#)
- class [DetectionBenchmark](#)
- class [DetectionWorker](#)
- class [DExpanderBox](#)
- class [DExpanderBoxExclusive](#)
- class [DFileDialog](#)
- class [DFileOperations](#)
- class [DFileSelector](#)  
*A widget to choose a single local file or path.*
- class [DFontProperties](#)



- class [DFontSelect](#)
- class [DGradientSlider](#)
- class [DHBox](#)

*An Horizontal widget to host children widgets.*

- class [DHistoryView](#)
- class [DHueSaturationSelector](#)
- class [DigikamApp](#)
- class [DigikamItemDelegate](#)
- class [DigikamItemView](#)
- class [DImageHistory](#)
- class [DImg](#)
- class [DImgBuiltinFilter](#)
- class [DImgChildItem](#)
- class [DImgFilterGenerator](#)
- class [DImgFilterManager](#)
- class [DImgLoader](#)
- class [DImgLoaderObserver](#)
- class [DImgLoaderSettings](#)
- class [DImgPreviewItem](#)
- class [DImgThreadedAnalyser](#)
- class [DImgThreadedFilter](#)
- class [DInfoInterface](#)
- class [DIntNumInput](#)
- class [DIntRangeBox](#)
- class [DIO](#)
- class [DirectoryNameOption](#)
- class [DisjointMetadata](#)
- class [DisjointMetadataDataFields](#)

*This class was split from `DisjointMetadata::Private` to allow to use the automatic C++ copy constructor (`DisjointMetadata::Private` contains a `QMutex` and is thus non-copyable)*

- class [DistortionFXFilter](#)
- class [DItemDelegate](#)
- class [DItemDrag](#)

*Provides a drag object with additional information for internal drag&drop.*

- class [DItemInfo](#)

*`DItemInfo` is a class to get item information from host application (`Showfoto` or `digiKam`) The interface is re-implemented in host and depend how item information must be retrieved (from a database or by file metadata).*

- class [DItemsList](#)
- class [DItemsListView](#)
- class [DItemsListViewItem](#)
- class [DItemToolTip](#)
- class [DKCamera](#)
- class [DLabelExpander](#)
- class [DLineWidget](#)

*A widget to show an horizontal or vertical line separator.*

- class [DLogoAction](#)
- class [DMessageBox](#)
- class [DMetadata](#)
- class [DMetadataSettings](#)
- class [DMetadataSettingsContainer](#)

*The class `DMetadataSettingsContainer` is designed to dynamically add namespaces.*

- class [DMetaInfoface](#)
- class [DModelFactory](#)

*This class is simply a factory of all models that build the core of the digikam application.*

- class [DMultiTabBar](#)
  - A Widget for horizontal and vertical tabs.*
- class [DMultiTabBarButton](#)
- class [DMultiTabBarFrame](#)
- class [DMultiTabBarTab](#)
- class [DNGConvertSettings](#)
- class [DNGSettings](#)
- class [DNGWriter](#)
- class [DNGWriterHost](#)
- class [DNNBaseDetectorModel](#)
- class [DNNFaceDetectorBase](#)
- class [DNNFaceDetectorSSD](#)
- class [DNNFaceDetectorYOLO](#)
- class [DNNFaceDetectorYuNet](#)
- class [DNNFaceExtractorBase](#)
- class [DNNModelBase](#)
- class [DNNModelConfig](#)
- class [DNNModelInfoContainer](#)
- class [DNNModelManager](#)
- class [DNNModelNet](#)
- class [DNNModelSFace](#)
- class [DNNModelYuNet](#)
- class [DNNOpenFaceExtractor](#)
- class [DNNResnetDetector](#)
- class [DNNSFaceExtractor](#)
- class [DNNYoloDetector](#)
- class [DNotificationPopup](#)
  - A dialog-like popup that displays messages without interrupting the user.*
- class [DNotificationWidget](#)
  - This widget can be used to provide inline positive or negative feedback, or to implement opportunistic interactions.*
- class [DOnlineTranslator](#)
  - Provides translation data.*
- struct [DOnlineTranslatorOption](#)
  - Contains translation options for a single word.*
- class [DOnlineTts](#)
  - Provides TTS URL generation.*
- class [DownloadInfo](#)
- class [DownloadSettings](#)
- class [DPixelsAliasFilter](#)
- class [DPlainTextEdit](#)
  - A text edit widget based on QPlainTextEdit with spell checker capabilities based on Sonnet (optional).*
- class [DPlugin](#)
  - A digiKam external plugin abstract class.*
- class [DPluginAboutDlg](#)
- class [DPluginAction](#)
- class [DPluginAuthor](#)
- class [DPluginBqm](#)
- class [DPluginConfView](#)
- class [DPluginConfViewBqm](#)
- class [DPluginConfViewDImg](#)
- class [DPluginConfViewEditor](#)
- class [DPluginConfViewGeneric](#)
- class [DPluginDialog](#)

- class [DPluginDImg](#)
- class [DPluginEditor](#)
- class [DPluginGeneric](#)
- class [DPluginLoader](#)

*The class that handles digiKam's external plugins.*

- class [DPluginRawImport](#)
- class [DPluginSetup](#)
- class [DPointSelect](#)
- class [DPopupFrame](#)
- class [DPreviewImage](#)
- class [DPreviewManager](#)
- class [DProgressDlg](#)
- class [DProgressWdg](#)
- class [DragDropModelImplementation](#)
- class [DragDropViewImplementation](#)
- class [DragHandle](#)

*An alternative handle for QDockWidget's that looks like a toolbar handle.*

- class [DRawDecoder](#)
- class [DRawDecoderSettings](#)
- class [DRawDecoderWidget](#)
- class [DRawDecoding](#)
- class [DRawInfo](#)
- class [DSaveSettingsWidget](#)
- class [DSelectedItem](#)
- class [DSelector](#)

*DSelector is the base class for other widgets which provides the ability to choose from a one-dimensional range of values.*

- class [DServiceInfo](#)
- class [DServiceMenu](#)
- class [DSliderSpinBox](#)
- class [DSplashScreen](#)
- class [DSqueezedClickLabel](#)
- class [DTagListDrag](#)

*Provides a drag object for a list of tags.*

- class [DTextBrowser](#)
- class [DTextEdit](#)

*A text edit widget based on QTextEdit with spell checker capabilities based on Sonnet (optional).*

- class [DTextLabelName](#)
- class [DTextLabelValue](#)
- class [DTextList](#)
- class [DToolTipStyleSheet](#)
- class [DTrash](#)
- class [DTrashItemInfo](#)
- class [DTrashItemModel](#)
- class [DTrashItemsListingJob](#)
- class [DuplicatesFinder](#)
- class [DuplicatesProgressObserver](#)
- class [DVBox](#)

*A Vertical widget to host children widgets.*

- class [DWItemDelegate](#)

*This class allows to create item delegates embedding simple widgets to interact with items.*

- class [DWItemDelegatePool](#)
- class [DWItemDelegatePoolPrivate](#)
- class [DWizardDlg](#)

- class [DWizardPage](#)
- class [DWorkingPixmap](#)
  - A widget to draw progress wheel indicator over thumbnails.*
- class [DXmlGuiWindow](#)
  - Generic class to use with all main window.*
- class [DynamicLayout](#)
- class [DynamicThread](#)
- class [DZoomBar](#)
- class [EditableSearchTreeView](#)
  - This tree view for searches adds basic editing functionality via the context menu.*
- class [EditorCore](#)
- class [EditorStackView](#)
- class [EditorTool](#)
- class [EditorTooliface](#)
- class [EditorToolSettings](#)
- class [EditorToolThreaded](#)
- class [EditorWindow](#)
- class [EffectMngr](#)
- class [EffectPreview](#)
- class [Ellipsoid](#)
  - Geometric figure that can be used to describe the approximate shape of the earth.*
- class [EmbossFilter](#)
- class [EmptyDTrashItemsJob](#)
- class [EmptyImageListProvider](#)
- class [EqualizeFilter](#)
- class [ExifMetaEngineMergeHelper](#)
- class [ExifToolBinary](#)
- class [ExifToolConfPanel](#)
- class [ExifToolErrorView](#)
- class [ExifToolListView](#)
- class [ExifToolListViewGroup](#)
- class [ExifToolListViewItem](#)
- class [ExifToolLoadingView](#)
- class [ExifToolParser](#)
- class [ExifToolProcess](#)
- class [ExifToolThread](#)
- class [ExifToolWidget](#)
- class [ExifWidget](#)
- class [ExposureDetector](#)
- class [ExposureSettingsContainer](#)
- class [FaceClassifier](#)
- class [FaceClassifierBase](#)
- class [FaceDb](#)
- class [FaceDbAccess](#)
- class [FaceDbAccessUnlock](#)
- class [FaceDbBackend](#)
- class [FaceDbOperationGroup](#)
  - When you intend to execute a number of write operations to the database, group them while holding a [FaceDbOperationGroup](#).*
- class [FaceDbSchemaUpdater](#)
- class [FaceDetector](#)
- class [FaceGroup](#)
- class [FaceItem](#)
- class [FaceItemRetriever](#)

- class [FacePipeline](#)
- class [FacePipelineBase](#)
- class [FacePipelineDetect](#)
- class [FacePipelineDetectRecognize](#)
- class [FacePipelineEdit](#)
- class [FacePipelineExtendedPackage](#)
- class [FacePipelineFaceTagsIface](#)
- class [FacePipelineFaceTagsIfaceList](#)
- class [FacePipelinePackage](#)
- class [FacePipelinePackageBase](#)
- class [FacePipelineRecognize](#)
- class [FacePipelineReset](#)
- class [FacePipelineRetrain](#)
- class [FacePreprocessor](#)
- class [FacePreviewLoader](#)
- class [FaceRejectionOverlay](#)
- class [FaceRejectionOverlayButton](#)
- class [FaceScanSettings](#)
- class [FaceScanWidget](#)
- class [FacesDetector](#)
- class [FacesEngine](#)
- class [FaceTags](#)
- class [FaceTagsEditor](#)
- class [FaceTagsIface](#)
- class [FaceUtils](#)
- class [FacialRecognitionWrapper](#)
- class [FFmpegBinary](#)
- class [FFmpegConfigHelper](#)
- class [FFmpegLauncher](#)
- class [FieldQueryBuilder](#)
- class [FileActionItemInfoList](#)
- class [FileActionMngr](#)
- class [FileActionMngrDatabaseWorker](#)
- class [FileActionMngrFileWorker](#)
- class [FileActionProgress](#)
- class [FileActionProgressItemContainer](#)
- class [FileActionProgressItemCreator](#)
- class [FilePropertiesOption](#)
- class [FileReadLocker](#)
- class [FileReadWriteLockKey](#)
- class [FileSaveConflictBox](#)
- class [FileSaveOptionsBox](#)
- class [FileSaveOptionsDlg](#)
- class [FilesDownloader](#)
- class [FileWorkerInterface](#)
- class [FileWriteLocker](#)
- class [FilmContainer](#)
- class [FilmFilter](#)
- class [FilmGrainContainer](#)
- class [FilmGrainFilter](#)
- class [FilmGrainSettings](#)
- class [Filter](#)
- class [FilterAction](#)
- class [FilterActionFilter](#)
- class [FiltersHistoryWidget](#)

- class [FilterSideBarWidget](#)  
*Sidebar widget containing the all filter widgets.*
- class [FilterStatusBar](#)
- class [FindDuplicatesAlbum](#)  
*The [FindDuplicatesAlbum](#) class Widgets used to show all reference images.*
- class [FindDuplicatesAlbumItem](#)
- class [FindDuplicatesView](#)
- class [FingerPrintsGenerator](#)
- class [FingerprintsTask](#)
- class [FirstRunDlg](#)
- class [FocusPoint](#)
- class [FocusPointGroup](#)
- class [FocusPointItem](#)
- class [FocusPointsExtractor](#)
- class [FocusPointsWriter](#)
- class [FrameOsd](#)
- class [FrameOsdSettings](#)
- class [FrameOsdWidget](#)
- class [FrameUtils](#)
- class [FreeRotationContainer](#)
- class [FreeRotationFilter](#)
- class [FreeRotationSettings](#)
- class [FreeSpaceToolTip](#)
- class [FreeSpaceWidget](#)
- class [FullObjectDetection](#)
- class [FullScreenSettings](#)
- class [FuzzySearchSideBarWidget](#)
- class [FuzzySearchView](#)
- class [GeoCoordinates](#)
- class [GeodeticCalculator](#)
- class [GeoDragDropHandler](#)
- class [GeofaceCluster](#)
- class [GeofaceGlobalObject](#)  
*Global object for geolocation interface to hold items common to all geolocation interface Widget instances.*
- class [GeofaceInternalWidgetInfo](#)  
*Class to hold information about map widgets stored in the [GeofaceGlobalObject](#).*
- class [GeofaceSharedData](#)
- class [GeolocationFilter](#)
- class [GeolocationSettings](#)
- class [GeolocationSettingsContainer](#)  
*The class [GeolocationSettingsContainer](#) encapsulates all Marble related settings.*
- class [GeoModelHelper](#)  
*Helper class to access data in models.*
- class [GeoPluginAboutDlg](#)
- class [GPCamera](#)  
*Gphoto2 camera Implementation of abstract type [DKCamera](#).*
- class [GPSBookmarkModelHelper](#)
- class [GPSBookmarkOwner](#)
- class [GPSCorrelatorWidget](#)
- class [GPSDataContainer](#)
- class [GPSDBJobInfo](#)
- class [GPSDBJobsThread](#)
- class [GPSGeofaceModelHelper](#)

- class [GPSItemContainer](#)
- class [GPSItemDelegate](#)
- class [GPSItemInfo](#)
- class [GPSItemInfoSorter](#)
- class [GPSItemList](#)
- class [GPSItemListContextMenu](#)
- class [GPSItemListDragDropHandler](#)
- class [GPSItemModel](#)
- class [GPSItemSortProxyModel](#)
- class [GPSJob](#)
- class [GPSLinkItemSelectionModel](#)

*Makes it possible to share a selection in multiple views which do not have the same source model.*

- class [GPSMarkerTiler](#)

*Marker model for storing data needed to display markers on the map.*

- class [GPSModelIndexProxyMapper](#)

*This class facilitates easy mapping of indexes and selections through proxy models.*

- class [GPSSearchSideBarWidget](#)
- class [GPSSearchView](#)
- class [GPSUndoCommand](#)
- class [Graph](#)

*The graph base class template.*

- class [GraphicsDImgItem](#)
- class [GraphicsDImgView](#)
- class [GreycstorationContainer](#)
- class [GreycstorationFilter](#)
- class [GreycstorationSettings](#)
- class [GroupedImagesFinder](#)
- class [GroupIndicatorOverlay](#)
- class [GroupIndicatorOverlayWidget](#)
- class [GroupingViewImplementation](#)
- class [GroupItemFilterSettings](#)
- class [GroupStateComputer](#)
- class [Haarface](#)
- class [HaarProgressObserver](#)
- class [HidingStateChanger](#)
- class [Highlighter](#)
- class [HistogramBox](#)
- class [HistogramPainter](#)

*A class that paints a histogram on a QPixmap.*

- class [HistogramWidget](#)
- class [HistoryEdgeProperties](#)

*Every edge has one associated object of this class.*

- class [HistoryImageld](#)
- class [HistoryVertexProperties](#)

*Every vertex has one associated object of this class.*

- class [HotPixelContainer](#)
- class [HotPixelFixer](#)
- class [HotPixelProps](#)
- class [HotPixelSettings](#)
- class [HotPixelsWeights](#)
- class [HoverButtonDelegateOverlay](#)
- class [HSLContainer](#)
- class [HSLFilter](#)

- class [HSLSettings](#)
- class [HSPreviewWidget](#)
- class [HTMLWidget](#)
- class [HTMLWidgetPage](#)
- class [IccManager](#)
- class [IccPostLoadingManager](#)
- class [ICCPreviewWidget](#)
- class [IccProfile](#)
- class [ICCProfileInfoDlg](#)
- class [IccProfilesComboBox](#)
- class [IccProfilesMenuAction](#)
- class [IccProfilesSettings](#)
- class [ICCProfileWidget](#)
- class [IccRenderingIntentComboBox](#)
- class [IccSettings](#)
- class [ICCSettingsContainer](#)
- class [IccTransform](#)
- class [IccTransformFilter](#)
- class [Identity](#)
- class [IdentityProvider](#)
- class [ImageChangeset](#)
- class [ImageCommonContainer](#)
- class [ImageCurves](#)
- class [ImageDialog](#)
- class [ImageDialogIconProvider](#)
- class [ImageDialogPreview](#)
- class [ImageDialogToolTip](#)
- class [ImageGuideWidget](#)
- class [ImageHistogram](#)
- class [ImageHistoryEntry](#)
- class [ImageIface](#)
- class [ImageLevels](#)
- class [ImageListProvider](#)

*This class provides access to a list of unspecified entities, where for each entry a QImage can be provided.*

- class [ImageMetadataContainer](#)
- class [ImagePreviewItem](#)
- class [ImageQualityCalculator](#)
- class [ImageQualityConfSelector](#)
- class [ImageQualityContainer](#)
- class [ImageQualityParser](#)
- class [ImageQualitySettings](#)
- class [ImageQualitySorter](#)
- class [ImageQualityTask](#)
- class [ImageQualityThread](#)
- class [ImageQualityThreadPool](#)
- class [ImageRegionItem](#)
- class [ImageRegionWidget](#)
- class [ImageRelation](#)
- class [ImageSortFilterModel](#)
- class [ImageTagChangeset](#)
- class [ImageTagProperty](#)
- class [ImageTagPropertyName](#)
- class [ImageWindow](#)
- class [ImageZoomSettings](#)



- class [ImportCategorizedView](#)
- class [ImportCategoryDrawer](#)
- class [ImportContextMenuHelper](#)
- class [ImportCoordinatesOverlay](#)
- class [ImportDelegate](#)
- class [ImportDownloadOverlay](#)
- class [ImportDragDropHandler](#)
- class [ImportFilterComboBox](#)
- class [ImportFilterDlg](#)
- class [ImportFilterModel](#)
- class [ImportIconView](#)
- class [ImportItemModel](#)
- class [ImportItemPropertiesSideBarImport](#)
- class [ImportItemPropertiesTab](#)
- class [ImportLockOverlay](#)
- class [ImportNormalDelegate](#)
- class [ImportOverlayWidget](#)
- class [ImportPreviewView](#)
- class [ImportRatingOverlay](#)
- class [ImportRenameParser](#)
- class [ImportRotateOverlay](#)
- class [ImportRotateOverlayButton](#)
- class [ImportSettings](#)
- class [ImportSortFilterModel](#)
- class [ImportStackedView](#)
- class [ImportThumbnailBar](#)
- class [ImportThumbnailDelegate](#)
- class [ImportThumbnailModel](#)
- class [ImportUI](#)
- class [ImportView](#)
- class [InfoDlg](#)
- class [InfraredContainer](#)
- class [InfraredFilter](#)
- class [InitializationObserver](#)
- class [InsertBookmarksCommand](#)
- class [InternalTagName](#)
- class [InvertFilter](#)
- class [IOFileSettings](#)
- class [IOJob](#)
- class [IOJobData](#)
- class [IOJobsManager](#)
- class [IOJobsThread](#)
- class [IptcCoreContactInfo](#)
- class [IptcCoreLocationInfo](#)
- class [IptcMetaEngineMergeHelper](#)
- class [IptcWidget](#)
- class [ItemAlbumFilterModel](#)
- class [ItemAlbumModel](#)
- class [ItemAttributesWatch](#)
- class [ItemCategorizedView](#)
- class [ItemCategoryDrawer](#)
- class [ItemChangeHint](#)
- class [ItemComments](#)
- class [ItemCoordinatesOverlay](#)
- class [ItemCopyMoveHint](#)

- class [ItemCopyright](#)
- class [ItemDelegate](#)
- class [ItemDelegateOverlay](#)
- class [ItemDelegateOverlayContainer](#)
- class [ItemDescEditTab](#)
- class [ItemDragDropHandler](#)
- class [ItemExtendedProperties](#)
- class [ItemFaceDelegate](#)
- class [ItemFilterModel](#)
- class [ItemFilterModelFilterer](#)
- class [ItemFilterModelPrepareHook](#)
- class [ItemFilterModelPreparer](#)
- class [ItemFilterModelWorker](#)
- class [ItemFilterSettings](#)
- class [ItemFiltersHistoryItemDelegate](#)
- class [ItemFiltersHistoryModel](#)
- class [ItemFiltersHistoryTreeItem](#)
- class [ItemFullScreenOverlay](#)
- class [ItemFullScreenOverlayButton](#)
- class [ItemGPS](#)
- class [ItemGPSModelHelper](#)
- class [ItemHistoryGraph](#)
- class [ItemHistoryGraphData](#)
- class [ItemHistoryGraphModel](#)
- class [ItemIconView](#)
- class [ItemInfo](#)

*The [ItemInfo](#) class contains provides access to the database for a single image.*

- class [ItemInfoAlbumsJob](#)
- class [ItemInfoCache](#)
- class [ItemInfoData](#)
- class [ItemInfoJob](#)
- class [ItemInfoList](#)
- class [ItemInfoReadLocker](#)
- class [ItemInfoSet](#)

*A container of associated [ItemInfo](#) and queue id.*

- class [ItemInfoStatic](#)
- class [ItemInfoTaskSplitter](#)
- class [ItemInfoWriteLocker](#)
- class [ItemListDragDropHandler](#)
- class [ItemLister](#)
- class [ItemListerJobGrowingPartsSendingReceiver](#)
- class [ItemListerJobPartsSendingReceiver](#)
- class [ItemListerJobReceiver](#)
- class [ItemListerReceiver](#)
- class [ItemListerRecord](#)
- class [ItemListerValueListReceiver](#)
- class [ItemListModel](#)
- class [ItemMarkerTiler](#)
- class [ItemMetadataAdjustmentHint](#)
- class [ItemModel](#)
- class [ItemPosition](#)
- class [ItemPreviewCanvas](#)
- class [ItemPreviewView](#)
- class [ItemPropertiesColorsTab](#)

- class [ItemPropertiesGPSTab](#)
- class [ItemPropertiesHistoryTab](#)
- class [ItemPropertiesMetadataTab](#)
- class [ItemPropertiesSideBar](#)
- class [ItemPropertiesSideBarDB](#)
- class [ItemPropertiesTab](#)
- class [ItemPropertiesVersionsTab](#)
- class [ItemQueryBuilder](#)
- class [ItemQueryPostHook](#)
- class [ItemQueryPostHooks](#)
- class [ItemRatingOverlay](#)
- class [ItemRotateOverlay](#)
- class [ItemRotateOverlayButton](#)
- class [ItemScanInfo](#)
- class [ItemScanner](#)
- class [ItemSelectionOverlay](#)
- class [ItemSelectionOverlayButton](#)
- class [ItemSelectionPropertiesTab](#)
- class [ItemShortInfo](#)
- class [ItemSortCollator](#)
- class [ItemSortSettings](#)
- class [ItemTagPair](#)
- class [ItemThumbnailBar](#)
- class [ItemThumbnailDelegate](#)
- class [ItemThumbnailModel](#)
- class [ItemVersionsModel](#)
- class [ItemViewCategorized](#)
- class [ItemViewDelegate](#)
- class [ItemViewHoverButton](#)
- class [ItemViewImportDelegate](#)
- class [ItemViewToolTip](#)
- class [ItemViewUtilities](#)
- class [ItemVisibilityController](#)
- class [ItemVisibilityControllerPropertyObject](#)
- class [KDNodeBase](#)
- class [KDNodeOpenFace](#)
- class [KDNodeSFace](#)
- class [KDTreeBase](#)
- class [KDTreeOpenFace](#)
- class [KDTreeSFace](#)
- class [KeywordSearchReader](#)
- class [KeywordSearchWriter](#)
- class [LabelsSideBarWidget](#)
- class [LabelsTreeView](#)
- class [LanguagesList](#)
- class [LcmsLock](#)
- class [LensDistortionFilter](#)
- class [LensDistortionPixelAccess](#)
  - *[LensDistortionPixelAccess](#) class: solving the eternal problem: random, cubic-interpolated, sub-pixel coordinate access to an image.*
- class [LensFunCameraSelector](#)
- class [LensFunContainer](#)
- class [LensFunFilter](#)
- class [LensFunIface](#)

- class [LensFunSettings](#)
- class [LevelsContainer](#)
- class [LevelsFilter](#)
- class [LibsInfoDlg](#)
- class [LightTablePreview](#)
- class [LightTableThumbBar](#)
- class [LightTableView](#)
- class [LightTableWindow](#)
- class [ListItem](#)
- class [ListViewComboBox](#)
- class [LoadingCache](#)
- class [LoadingCacheFileWatch](#)
- class [LoadingCacheInterface](#)
- class [LoadingDescription](#)
- class [LoadingProcess](#)
- class [LoadingProcessListener](#)
- class [LoadingTask](#)
- class [LoadSaveFileInfoProvider](#)
- class [LoadSaveNotifier](#)
- class [LoadSaveTask](#)
- class [LoadSaveThread](#)
- class [LocalContrastContainer](#)
- class [LocalContrastFilter](#)
- class [LocalContrastSettings](#)
- class [LocalizeConfig](#)
- class [LocalizeContainer](#)

*The class [LocalizeContainer](#) encapsulates all spell-check and localize related settings.*

- class [LocalizeSelector](#)
- class [LocalizeSelectorList](#)
- class [LocalizeSettings](#)
- class [LookupAltitude](#)
- class [LookupAltitudeGeonames](#)
- class [LookupFactory](#)
- class [MaintenanceData](#)
- class [MaintenanceDlg](#)
- class [MaintenanceMgr](#)
- class [MaintenanceSettings](#)
- class [MaintenanceThread](#)
- class [MaintenanceTool](#)
- class [MakerNoteWidget](#)
- class [ManagedLoadSaveThread](#)
- class [MapBackend](#)
- class [MapDragData](#)
- class [MapDragDropHandler](#)
- class [MapViewModelHelper](#)
- class [MapWidget](#)

*The central map view class of geolocation interface.*

- class [MapWidgetView](#)

*Class containing digiKam's central map view.*

- struct [Mat](#)

*Mat:*

- class [MdKeyListViewItem](#)
- class [MediaPlayerView](#)
- class [MetadataHub](#)

- class [MetadataHubMngr](#)
- class [MetadataKeys](#)
- class [MetadataListView](#)
- class [MetadataListViewItem](#)
- class [MetadataOption](#)
- class [MetadataOptionDialog](#)
- class [MetadataPage](#)
- class [MetadataPanel](#)
- class [MetadataRemover](#)
- class [MetadataRemoveTask](#)
- class [MetadataSelector](#)
- class [MetadataSelectorItem](#)
- class [MetadataSelectorView](#)
- class [MetadataStatusBar](#)
- class [MetadataSynchronizer](#)
- class [MetadataSyncTask](#)
- class [MetadataWidget](#)
- class [MetaEngine](#)
- class [MetaEngineData](#)
- class [MetaEngineMergeHelper](#)
- class [MetaEnginePreviews](#)
- class [MetaEngineRotation](#)
- class [MetaEngineSettings](#)
- class [MetaEngineSettingsContainer](#)

*The class [MetaEngineSettingsContainer](#) encapsulates all metadata related settings.*

- class [MigrateFromDigikam4Page](#)
- class [MimeFilter](#)
- class [MixerContainer](#)
- class [MixerFilter](#)
- class [MixerSettings](#)
- class [MLClassifierFoundation](#)
- class [MLPipelineFoundation](#)
- class [MLPipelinePackageFoundation](#)
- class [MLPipelinePackageNotify](#)
- class [ModelCompleter](#)
- class [ModelIndexedBasedComboBox](#)
- class [ModelMenu](#)

*A [QMenu](#) that is dynamically populated from a [QAbstractItemModel](#).*

- class [Modifier](#)
- class [MonthWidget](#)
- class [MysqlAdminBinary](#)
- class [MysqlInitBinary](#)
- class [MysqlServerBinary](#)
- class [MysqlUpgradeBinary](#)
- class [NamespaceEditDlg](#)
- class [NamespaceEntry](#)

*The [NamespaceEntry](#) class provide a simple container for dmetadata namespaces variables, such as names, what types of data expects and extra xml tags.*

- class [NamespaceListView](#)
- class [NetworkManager](#)
- class [NewItemsFinder](#)
- class [NoDuplicatesImportFilterModel](#)
- class [NoDuplicatesItemFilterModel](#)
- class [NoiseDetector](#)

- class [NonDeterministicRandomData](#)
- class [NormalizeFilter](#)
- class [NormalSearchTreeView](#)

*Tree view for all saved "normal" searches.*

- class [NRContainer](#)
- class [NREstimate](#)
- class [NRFilter](#)
- class [NRSettings](#)
- class [OilPaintFilter](#)
- class [OnlineVersionChecker](#)
- class [OnlineVersionDlg](#)
- class [OnlineVersionDwnl](#)
- class [OpenCVDNNFaceDetector](#)
- class [OpenCVDNNFaceRecognizer](#)
- class [OpenfacePreprocessor](#)
- class [OpenFilePage](#)
- class [Option](#)
- class [OverlayWidget](#)

*This is a widget that can align itself with another one, without using a layout, so that it can actually be on top of other widgets.*

- class [PackageLoadingDescriptionList](#)
- class [PAlbum](#)

*A Physical Album representation.*

- class [PanIconFrame](#)

*Frame with popup menu behavior to host [PanIconWidget](#).*

- class [PanIconWidget](#)
- class [ParallelAdapter](#)
- class [ParallelPipes](#)
- class [ParallelWorkers](#)
- class [Parser](#)
- class [ParseResults](#)
- class [ParseSettings](#)
- class [PeopleSideBarWidget](#)
- class [PersistentWidgetDelegateOverlay](#)
- class [PhotoInfoContainer](#)
- class [PickLabelFilter](#)
- class [PickLabelMenuAction](#)
- class [PickLabelSelector](#)
- class [PickLabelWidget](#)
- class [PlaceholderWidget](#)
- class [PointTransformAffine](#)
- class [PositionKeys](#)
- class [PreviewList](#)
- class [PreviewListItem](#)
- class [PreviewLoadingTask](#)
- class [PreviewLoadThread](#)
- class [PreviewPage](#)
- class [PreviewSettings](#)
- class [PreviewThreadWrapper](#)
- class [PreviewToolBar](#)
- class [ProcessLauncher](#)
- class [ProgressItem](#)
- class [ProgressManager](#)

The *ProgressManager* singleton keeps track of all ongoing transactions and notifies observers (progress dialogs) when their progress percent value changes, when they are completed (by their owner), and when they are canceled.

- class [ProgressView](#)
- class [ProxyClickLineEdit](#)
- class [ProxyLineEdit](#)
- class [QListImageListProvider](#)

A wrapper implementation for *ImageListProvider* if you have a *QList* of *QImages*.

- class [QMapForAdaptors](#)

Adds the necessary typedefs so that *associative\_property\_map* accepts a *QMap*, and it can be used as a *Boost Property Map*.

- class [QueueListView](#)
- class [QueueListViewItem](#)
- class [QueueMgrWindow](#)
- class [QueuePool](#)
- class [QueuePoolBar](#)
- class [QueueSettings](#)

This container host all common settings used by a queue, not including assigned batch tools.

- class [QueueSettingsView](#)
- class [QueueToolTip](#)
- class [RainDropFilter](#)
- class [RandomNumberGenerator](#)

This class differs from standard pseudo random number generators (*rand()*) in these points:

- class [RangeDialog](#)
- class [RangeModifier](#)
- class [RatingBox](#)
- class [RatingComboBox](#)
- class [RatingComboBoxDelegate](#)
- class [RatingComboBoxModel](#)
- class [RatingComboBoxWidget](#)
- class [RatingFilter](#)
- class [RatingFilterWidget](#)
- class [RatingMenuAction](#)
- class [RatingStarDrawer](#)
- class [RatingWidget](#)
- class [RawCameraDlg](#)
- class [RawPage](#)
- class [RawProcessingFilter](#)

This is a special filter.

- class [RecognitionBenchmarker](#)
- class [RecognitionPreprocessor](#)
- class [RecognitionTrainingProvider](#)

A simple *QImage* training data container used by *RecognitionDatabase::train(Identity, QImage, QString)*.

- class [RecognitionTrainingUpdateQueue](#)
- class [RecognitionWorker](#)
- class [RedEyeCorrectionContainer](#)
- class [RedEyeCorrectionFilter](#)
- class [RedEyeCorrectionSettings](#)
- class [RefocusFilter](#)
- class [RefocusMatrix](#)
- class [RegionFrameItem](#)
- class [RemoveBookmarksCommand](#)
- class [RemoveDoublesModifier](#)
- class [RemoveFilterAction](#)
- class [RenameCustomizer](#)

- class [RenameFileJob](#)
- class [ReplaceDialog](#)
- class [ReplaceModifier](#)
- class [RestoreDTrashItemsJob](#)
- class [RGBackend](#)

*This class is a base class for Open Street Map and Geonames backends.*

- class [RGInfo](#)

*This class contains data needed in reverse geocoding process.*

- class [RGTagModel](#)

*The model that holds data for the tag tree displayed in [ReverseGeocodingWidget](#).*

- class [RGWidget](#)

*The [RGWidget](#) class represents the main widget for reverse geocoding.*

- class [RubberItem](#)
- class [Rule](#)
- class [RuleDialog](#)
- class [SafeTemporaryFile](#)
- class [SAlbum](#)

*A [Search Album](#) representation.*

- class [SaveProperties](#)
- class [SavingContext](#)
- class [SavingTask](#)
- class [ScanController](#)
- class [ScanStateFilter](#)
- class [ScriptingSettings](#)
- class [SearchChangeset](#)
- class [SearchesDBJobInfo](#)
- class [SearchesDBJobsThread](#)
- class [SearchesJob](#)
- class [SearchField](#)
- class [SearchFieldAlbum](#)
- class [SearchFieldCheckBox](#)
- class [SearchFieldChoice](#)
- class [SearchFieldColorDepth](#)
- class [SearchFieldComboBox](#)
- class [SearchFieldGroup](#)
- class [SearchFieldGroupLabel](#)
- class [SearchFieldKeyword](#)
- class [SearchFieldLabels](#)
- class [SearchFieldMonthDay](#)
- class [SearchFieldPageOrientation](#)
- class [SearchFieldRangeDate](#)
- class [SearchFieldRangeDouble](#)
- class [SearchFieldRangeInt](#)
- class [SearchFieldRangeTime](#)
- class [SearchFieldRating](#)
- class [SearchFieldText](#)
- class [SearchFilterModel](#)

*Filter model for searches that can filter by search type.*

- class [SearchGroup](#)
- class [SearchGroupLabel](#)
- class [SearchInfo](#)

*A container class for transporting search information from the database to [AlbumManager](#).*

- class [SearchModel](#)



- class [SearchModificationHelper](#)  
*Utility class providing methods to modify search albums (SAlbum) in a way useful to implement views.*
- class [SearchSideBarWidget](#)
- class [SearchTabHeader](#)
- class [SearchTextBar](#)  
*A text input for searching entries with visual feedback.*
- class [SearchTextBarDb](#)  
*A text input for searching entries with visual feedback.*
- class [SearchTextFilterSettings](#)
- class [SearchTextSettings](#)
- class [SearchTreeView](#)
- class [SearchView](#)
- class [SearchViewBottomBar](#)
- class [SearchViewThemedPartsCache](#)
- class [SearchWindow](#)
- class [SearchXmlCachingReader](#)
- class [SearchXmlReader](#)
- class [SearchXmlWriter](#)
- class [SequenceNumberDialog](#)
- class [SequenceNumberOption](#)
- class [Setup](#)
- class [SetupAlbumView](#)
- class [SetupCamera](#)
- class [SetupCategory](#)
- class [SetupCollectionDelegate](#)
- class [SetupCollectionModel](#)
- class [SetupCollections](#)
- class [SetupCollectionTreeView](#)
- class [SetupDatabase](#)
- class [SetupEditor](#)
- class [SetupEditorInterface](#)
- class [SetupGeolocation](#)
- class [SetupICC](#)
- class [SetupImageQualitySorter](#)
- class [SetupIOFiles](#)
- class [SetupLightTable](#)
- class [SetupMetadata](#)
- class [SetupMime](#)
- class [SetupMisc](#)
- class [SetupPlugins](#)
- class [SetupRaw](#)
- class [SetupTemplate](#)
- class [SetupToolTip](#)
- class [SetupVersioning](#)
- class [SharedLoadingTask](#)
- class [SharedLoadSaveThread](#)
- class [SharedQueue](#)
- class [SharpContainer](#)
- class [SharpenFilter](#)
- class [SharpSettings](#)
- class [ShearFilter](#)
- class [ShowHideVersionsOverlay](#)
- class [Sidebar](#)  
*This class handles a sidebar view.*

- class [SidebarSplitter](#)
- class [SidebarWidget](#)

*Abstract base class for widgets that are use in one of digikams's sidebars.*

- class [SidecarFinder](#)
- class [SimilarityDb](#)
- class [SimilarityDbAccess](#)
- class [SimilarityDbBackend](#)
- class [SimilarityDbSchemaUpdater](#)
- class [SimpleTreeModel](#)
- class [SinglePhotoPreviewLayout](#)
- class [SketchWidget](#)
- class [SlideVideo](#)
- class [SoftProofDialog](#)
- class [SolidHardwareDlg](#)
- class [SpellCheckConfig](#)
- class [SqueezedComboBox](#)

*This widget is a QComboBox, but then a little bit different.*

- class [StackedView](#)
- class [StartScanPage](#)
- class [StateSavingObject](#)

*An interface-like class with utility methods and a general public interface to support state saving and restoring for objects via KConfig.*

- class [StatusbarProgressWidget](#)
- class [StatusProgressBar](#)
- class [StayPoppedUpComboBox](#)
- class [StretchFilter](#)
- class [StyleSheetDebugger](#)
- class [SubjectData](#)
- class [SubjectEdit](#)
- class [SubjectWidget](#)
- class [SyncJob](#)
- class [SystemSettings](#)
- class [SystemSettingsWidget](#)
- class [TableView](#)
- class [TableViewColumn](#)
- class [TableViewColumnConfiguration](#)
- class [TableViewColumnConfigurationWidget](#)
- class [TableViewColumnDescription](#)
- class [TableViewColumnFactory](#)
- class [TableViewColumnProfile](#)
- class [TableViewConfigurationDialog](#)
- class [TableViewItemDelegate](#)
- class [TableViewModel](#)
- class [TableViewSelectionModelSyncer](#)
- class [TableViewShared](#)
- class [TableViewTreeView](#)
- class [TagChangeset](#)
- class [TagCheckView](#)
- class [TagCompleter](#)
- struct [TagData](#)
- class [TagDragDropHandler](#)
- class [TagEditDlg](#)
- class [TagFilterView](#)

*A view to filter the currently displayed album by tags.*

- class [TagFolderView](#)
- class [TaggingAction](#)
- class [TaggingActionFactory](#)
- class [TagInfo](#)

*A container class for transporting tag information from the database to [AlbumManager](#).*

- class [TagList](#)
- class [TagMngrListModel](#)
- class [TagMngrListView](#)
- class [TagMngrTreeView](#)
- class [TagModel](#)
- class [TagModificationHelper](#)

*Utility class providing methods to modify tag albums ([TAlbum](#)) in a way useful to implement views.*

- class [TagProperties](#)
- class [TagPropertiesFilterModel](#)

*Filter model for tags that can filter by tag property.*

- class [TagProperty](#)
- class [TagPropertyName](#)
- class [TagPropWidget](#)
- class [TagRegion](#)
- class [TagsActionMngr](#)
- class [TagsCache](#)
- class [TagsDBJobInfo](#)
- class [TagsDBJobsThread](#)
- class [TagsEdit](#)
- class [TagShortInfo](#)
- class [TagsJob](#)
- class [TagsLineEditOverlay](#)
- class [TagsManager](#)
- class [TagsManagerFilterModel](#)
- class [TagsPopupMenu](#)
- class [TagTreeView](#)
- class [TagTreeViewSelectComboBox](#)
- class [TagViewSideBarWidget](#)
- class [TAlbum](#)

*A Tag Album representation.*

- class [Template](#)
- class [TemplateList](#)
- class [TemplateListItem](#)
- class [TemplateManager](#)
- class [TemplatePanel](#)
- class [TemplateSelector](#)
- class [TemplateViewer](#)
- class [TextFilter](#)
- class [TextureContainer](#)
- class [TextureFilter](#)
- class [TextureSettings](#)
- class [ThemeManager](#)
- class [ThreadManager](#)
- class [ThumbBarDock](#)

*A dock widget specifically designed for thumbnail bars (class [ThumbNailView](#) or one of its descendants).*

- class [ThumbnailAligningDelegate](#)
- class [ThumbnailCreator](#)
- class [ThumbnailIdentifier](#)
- class [ThumbnailImageCatcher](#)

- class [ThumbnailInfo](#)
- class [ThumbnailInfoProvider](#)
- class [ThumbnailLoadingTask](#)
- class [ThumbnailLoadThread](#)
- class [ThumbnailSize](#)
- class [ThumbsDb](#)
- class [ThumbsDbAccess](#)
- class [ThumbsDbBackend](#)
- class [ThumbsDbInfo](#)
- class [ThumbsDbInfoProvider](#)
- class [ThumbsDbSchemaUpdater](#)
- class [ThumbsGenerator](#)
- class [ThumbsTask](#)
- class [TileGroupier](#)
- class [TileIndex](#)
- class [TimeAdjustContainer](#)

*Container that store all timestamp adjustments.*

- class [TimeAdjustSettings](#)
- class [TimelineSideBarWidget](#)
- class [TimeLineWidget](#)
- class [TimeZoneComboBox](#)
- class [Token](#)

*Token is the smallest parsing unit in AdvancedRename utility*

- class [TonalityContainer](#)
- class [TonalityFilter](#)
- class [ToolListViewGroup](#)
- class [ToolListViewItem](#)
- class [ToolSettingsView](#)
- class [ToolsListView](#)
- class [ToolsView](#)
- class [TooltipCreator](#)
- class [TooltipDialog](#)
- class [TooltipsPage](#)
- class [TrackCorrelator](#)
- class [TrackCorrelatorThread](#)
- class [TrackListModel](#)
- class [TrackManager](#)
- class [TrackReader](#)
- class [TrainerWorker](#)
- class [TrainingDataProvider](#)

*A [TrainingDataProvider](#) provides a call-back interface for the training process to retrieve the necessary information.*

- class [TransactionItem](#)
- class [TransactionItemView](#)
- class [TransitionMngr](#)
- class [TransitionPreview](#)
- class [TrashView](#)
- class [TreeBranch](#)
- class [TreeProxyModel](#)
- class [TreeViewComboBox](#)
- class [TreeViewLineEditComboBox](#)
- class [TrimmedModifier](#)
- class [TwoProgressItemsContainer](#)
- class [UMSCamera](#)

*USB Mass Storage camera Implementation of abstract type [DKCamera](#).*

- class [UndoAction](#)
- class [UndoActionIrreversible](#)
- class [UndoActionReversible](#)
- class [UndoCache](#)
- class [UndoManager](#)
- class [UndoMetadataContainer](#)
- class [UndoState](#)
- class [UniqueModifier](#)
- class [UnsharpMaskFilter](#)
- class [VersionFileInfo](#)
- class [VersionFileOperation](#)
- class [VersioningPromptUserSaveDialog](#)
- class [VersionItemFilterSettings](#)
- class [VersionManager](#)
- class [VersionManagerSettings](#)
- class [VersionNamingScheme](#)
- class [VersionsDelegate](#)
- class [VersionsTreeView](#)
- class [VersionsWidget](#)
- class [VideoFrame](#)
- class [VideoInfoContainer](#)
- class [VideoMetadataContainer](#)
- class [VideoStripFilter](#)
- class [VideoThumbDecoder](#)
- class [VideoThumbnailer](#)
- class [VideoThumbWriter](#)
- class [VidPlayerDlg](#)
- class [VidSlideSettings](#)
- class [VidSlideTask](#)
- class [VidSlideThread](#)
- class [VisibilityController](#)
- class [VisibilityObject](#)
- class [WBContainer](#)
- class [WBFilter](#)
- class [WBSettings](#)
- class [WebBrowserDlg](#)
- class [WebWidget](#)
- class [WelcomePage](#)
- class [WelcomePageView](#)
- class [WelcomePageViewPage](#)
- class [WorkerObject](#)
- class [Workflow](#)

*This container group all queue common settings plus all assigned batch tools.*

- class [WorkflowDlg](#)
- class [WorkflowItem](#)
- class [WorkflowList](#)
- class [WorkflowManager](#)
- class [WorkingWidget](#)
- class [WSAlbum](#)
- class [WSComboBoxIntermediate](#)
- class [WSLoginDialog](#)
- class [WSNewAlbumDialog](#)
- class [WSSelectUserDlg](#)
- class [WSSettings](#)

- class [WSSettingsWidget](#)
- class [WSToolDialog](#)
- class [WSToolUtils](#)
- class [XbelReader](#)
- class [XbelWriter](#)
- class [XmpMetaEngineMergeHelper](#)
- class [XmpWidget](#)

## Typedefs

- typedef `QHash< ActionJob *, int >` **ActionJobCollection**  
*Define a QHash of job/priority to process by [ActionThreadBase](#) manager.*
- typedef `QList< Album * >` **AlbumList**
- typedef `QMap< int, QPixmap >` **AlbumThumbnailMap**
- typedef `QList< BatchToolSet >` **BatchSetList**  
*An indexed map of batch tools with settings.*
- typedef `QMap< QString, QVariant >` **BatchToolSettings**  
*A map of batch tool settings (setting key, setting value).*
- typedef `QList< BatchTool * >` **BatchToolsList**  
*A list of batch tool instances.*
- typedef `QPair< CamItemInfo, QPixmap >` **CachedItem**
- typedef `QList< CamItemInfo >` **CamItemInfoList**
- typedef `QPair< QByteArray, CHUpdateItemMap >` **CHUpdateItem**
- typedef `QMultiMap< QDateTime, CamItemInfo >` **CHUpdateItemMap**
- typedef `QFlags< CropHandleFlag >` **CropHandle**
- typedef `QPair< QDateTime, QDateTime >` **DateRange**  
*Range of a contiguous dates selection <start date, end date>.*
- typedef `QList< DateRange >` **DateRangeList**  
*List of dates range selected.*
- typedef `QMap< QString, QString >` **DbKeyIdsMap**
- typedef `QMap< QString, DbKeysCollection * >` **DbOptionKeysMap**
- typedef `QMap< QString, QVariant >` **DImgLoaderPrms**  
*Map container of widget parameter name/value.*
- typedef `bool(* DItemsListsLessThanHandler) (const QTreeWidgetItem *current, const QTreeWidgetItem &other)`  
*Type of static fonction used to customize sort items in list.*
- typedef `enum Digikam::_DNNLoaderType` **DNNLoaderType**
- typedef `enum Digikam::_DNNModelUsage` **DNNModelUsage**
- typedef `QList< DNNModelUsage >` **DNNModelUsageList**
- typedef `QList< DownloadSettings >` **DownloadSettingsList**
- typedef `QList< DTrashItemInfo >` **DTrashItemInfoList**
- typedef `FileReadWriteLockPriv` **Entry**
- typedef `QMap< QString, QStringList >` **FFMpegProperties**
- typedef `QList< Filter * >` **FilterList**
- typedef `QFlags< GeoExtraAction >` **GeoExtraActions**
- typedef `QFlags< GeoGroupStateEnum >` **GeoGroupState**
- typedef `QFlags< GeoMouseMoveMode >` **GeoMouseMoveModes**
- typedef `Graph < HistoryVertexProperties, HistoryEdgeProperties >` **HistoryGraph**
- typedef `QMap< QString, ICCTagInfo >` **ICCTagInfoMap**
- typedef `QMap< QPair< qlonglong, QString >, QList< int > >` **IdAlbumMap**
- typedef `QSharedPointer< DImgFilterGenerator >` **ImgFilterPtr**
- typedef `QPair< int, int >` **IntPair**

- typedef QList< IntPair > **IntPairList**
- typedef ItemInfoList::iterator **ItemInfoListIterator**
- typedef QExplicitlySharedDataPointer< ItemTagPairPriv > **ItemTagPairPrivSharedPointer**
- typedef QList< MetadataInfo::Field > **MetadataFields**
- typedef QHash< QString, QByteArray > **MyHash**
- typedef QPair< QUrl, QString > **NewNameInfo**
- typedef QList< NewNameInfo > **NewNamesList**
- typedef QPair< QPointF, [HudSide](#) > **OptimalPosition**
- typedef QPair< QString, QVariant > **PathValuePair**
- typedef QList< int > **QIntList**
- typedef QList< [ItemInfoSet](#) > **QueuePoolItemsList**  
*A list of all queued items from the pool.*
- typedef QList< [Rule](#) \* > **RulesList**
- typedef struct [Digikam::TagData](#) **TagData**
- typedef QList< [TagProperty](#) >::const\_iterator **TagPropertiesConstIterator**
- typedef QExplicitlySharedDataPointer< TagProperties::TagPropertiesPriv > **TagPropertiesPrivSharedPointer**
- typedef QPair< TagPropertiesConstIterator, TagPropertiesConstIterator > **TagPropertiesRange**
- typedef QList< [Token](#) \* > **TokenList**
- typedef QPair< int, int > **YearMonth**

## Enumerations

- enum { **TaggingActionRole** = Qt::UserRole + 1 , **CompletionRole** = Qt::UserRole + 2 }
- enum **\_DNNLoaderType** { **DNNLoaderNet** , **DNNLoaderConfig** , **DNNLoaderYuNet** , **DNNLoaderSFace** }
- enum **\_DNNModelUsage** { **DNNUsageFaceDetection** , **DNNUsageFaceRecognition** , **DNNUsageRedeyeDetection** , **DNNUsageObjectDetection** , **DNNUsageAesthetics** }
- enum **ChannelType** { **LuminosityChannel** = 0 , **RedChannel** , **GreenChannel** , **BlueChannel** , **AlphaChannel** , **ColorChannels** }
- enum **ClickDragState** { **HoverState** , **PressedState** , **PressDragState** , **ClickedMoveState** }
- enum **ColorLabel** { **NoColorLabel** = 0 , **RedLabel** , **OrangeLabel** , **YellowLabel** , **GreenLabel** , **BlueLabel** , **MagentaLabel** , **GrayLabel** , **BlackLabel** , **WhiteLabel** , **FirstColorLabel** = NoColorLabel , **LastColorLabel** = WhiteLabel , **NumberOfColorLabels** = LastColorLabel + 1 }
- enum **CropHandleFlag** { **CH\_None** , **CH\_Top** = 1 , **CH\_Left** = 2 , **CH\_Right** = 4 , **CH\_Bottom** = 8 , **CH\_TopLeft** = CH\_Top | CH\_Left , **CH\_BottomLeft** = CH\_Bottom | CH\_Left , **CH\_TopRight** = CH\_Top | CH\_Right , **CH\_BottomRight** = CH\_Bottom | CH\_Right , **CH\_Content** = 16 }
- enum **DColorChooserMode** { **ChooserClassic** = 0x0000 , **ChooserHue** = 0x0001 , **ChooserSaturation** = 0x0002 , **ChooserValue** = 0x0003 , **ChooserRed** = 0x0004 , **ChooserGreen** = 0x0005 , **ChooserBlue** = 0x0006 }
- enum [DetectorNNModel](#) { **DNNDetectorSSD** = 0 , **DNNDetectorYOLOv3** , **DNNDetectorYuNet** }
- enum **DropAction** { **NoAction** , **CopyAction** , **MoveAction** , **GroupAction** , **SortAction** , **GroupAndMoveAction** , **AssignTagAction** }
- enum **FilterType** { **TEXT** = 0 , **MIME** , **GEOLOCATION** , **TAGS** , **LABELS** }

- enum [FullScreenOptions](#) {  
**FS\_TOOLBARS** = 0x00000001 , **FS\_THUMBBAR** = 0x00000002 , **FS\_SIDEBARS** = 0x00000004 ,  
**FS\_STATUSBAR** = 0x00000008 ,  
**FS\_NONE** = 0x00000010 , **FS\_ALBUMGUI** = **FS\_TOOLBARS** | **FS\_THUMBBAR** | **FS\_SIDEBARS** | **FS\_←**  
**\_STATUSBAR** , **FS\_EDITOR** = **FS\_TOOLBARS** | **FS\_THUMBBAR** | **FS\_SIDEBARS** | **FS\_STATUSBAR** ,  
**FS\_LIGHTTABLE** = **FS\_TOOLBARS** | **FS\_SIDEBARS** | **FS\_STATUSBAR** ,  
**FS\_IMPORTUI** = **FS\_TOOLBARS** | **FS\_THUMBBAR** | **FS\_SIDEBARS** | **FS\_STATUSBAR** }  
*Optional parts which can be hidden or not from managed window configuration panel.*
- enum class **FuzzyAlgorithm** { **Unknown** = 0 , **Haar** = 1 , **TlIdf** = 2 }
- enum **GeoExtraAction** { **ExtraActionSticky** = 1 , **ExtraLoadTracks** = 2 }
- enum [GeoGroupStateEnum](#) {  
**SelectedMask** = 0x03 << 0 , **SelectedNone** = 0x00 << 0 , **SelectedSome** = 0x03 << 0 , **SelectedAll** =  
0x02 << 0 ,  
**FilteredPositiveMask** = 0x03 << 2 , **FilteredPositiveNone** = 0x00 << 2 , **FilteredPositiveSome** = 0x03  
<< 2 , **FilteredPositiveAll** = 0x02 << 2 ,  
**RegionSelectedMask** = 0x03 << 4 , **RegionSelectedNone** = 0x00 << 4 , **RegionSelectedSome** = 0x03  
<< 4 , **RegionSelectedAll** = 0x02 << 4 }  
*Representation of possible tile or cluster states.*
- enum **GeoMouseMode** {  
**MouseModePan** = 1 , **MouseModeRegionSelection** = 2 , **MouseModeRegionSelectionFromIcon** = 4 ,  
**MouseModeFilter** = 8 ,  
**MouseModeSelectThumbnail** = 16 , **MouseModeZoomIntoGroup** = 32 , **MouseModeLast** = 32 }
- enum **HistogramBoxType** {  
**RGB** = 0 , **RGBA** , **LRGB** , **LRGBA** ,  
**LRGBC** , **LRGBAC** }
- enum [HistogramRenderingType](#) { **FullImageHistogram** = 0 , **ImageSelectionHistogram** }
- enum [HistogramScale](#) { **LinScaleHistogram** = 0 , **LogScaleHistogram** }
- enum [HudSide](#) {  
**HS\_None** = 0 , **HS\_Top** = 1 , **HS\_Bottom** = 2 , **HS\_Inside** = 4 ,  
**HS\_TopInside** = **HS\_Top** | **HS\_Inside** , **HS\_BottomInside** = **HS\_Bottom** | **HS\_Inside** }
- enum **ImportRotateOverlayDirection** { **ImportRotateOverlayLeft** , **ImportRotateOverlayRight** }
- enum **ItemRotateOverlayDirection** { **ItemRotateOverlayLeft** , **ItemRotateOverlayRight** }
- enum **MapLayout** { **MapLayoutOne** = 0 , **MapLayoutHorizontal** = 1 , **MapLayoutVertical** = 2 }
- enum **MatColorOrder** {  
**MCO\_BGR** , **MCO\_RGB** , **MCO\_BGRA** = **MCO\_BGR** , **MCO\_RGBA** = **MCO\_RGB** ,  
**MCO\_ARGB** , **MCO\_INVALID** }
- enum [MeaningOfDirection](#) { [ParentToChild](#) , [ChildToParent](#) }  
*Each edge is directed: "vertex1 -> vertex2".*
- enum [OperationType](#) {  
**MetadataOps** = 0 , **ImportExportOps** , **BQMOps** , **LightTableOps** ,  
**SlideshowOps** , **RenameOps** , **ToolsOps** , **UnspecifiedOps** }  
*Types of operations for ApplicationSettings.*
- enum **PickLabel** {  
**NoPickLabel** = 0 , **RejectedLabel** , **PendingLabel** , **AcceptedLabel** ,  
**FirstPickLabel** = **NoPickLabel** , **LastPickLabel** = **AcceptedLabel** , **NumberOfPickLabels** = **LastPickLabel** +  
1 }
- enum **PreprocessorSelection** { **OPENFACE** = 0 }
- enum **StdActionType** {  
**StdCopyAction** = 0 , **StdPasteAction** , **StdCutAction** , **StdQuitAction** ,  
**StdCloseAction** , **StdZoomInAction** , **StdZoomOutAction** , **StdOpenAction** ,  
**StdSaveAction** , **StdSaveAsAction** , **StdRevertAction** , **StdBackAction** ,  
**StdForwardAction** }
- enum **TrackColumns** { **ColumnVisible** = 0 , **ColumnNPoints** = 1 , **ColumnFilename** = 2 , **ColumnCount** =  
3 }
- enum **Type** { **TypeChild** = 1 , **TypeSpacer** = 2 , **TypeNewChild** = 4 }
- enum class [YoloVersions](#) { **YOLOV5NANO** = 0 , **YOLOV5XLARGE** , **RESNET50** }



## Functions

- static QAction \* **addCancelAction** (QMenu \*const menu)
- static QAction \* **addGroupAction** (QMenu \*const menu)
- static QAction \* **addGroupAndMoveAction** (QMenu \*const menu)
- const QString **additionalInformation** ()
- static QAction \* **addSortAction** (QMenu \*const menu)
- static QString **adjustedActionText** (const QAction \*const action)
- DIGIKAM\_EXPORT QProcessEnvironment **adjustedEnvironmentForApplmage** ()
  - If digiKam run into Applmage, return a cleaned environment for QProcess to execute a program outside the bundle without broken run-time dependencies.*
- static bool **approximates** (const QSizeF &s1, const QSizeF &s2)
- QDateTime **asDateTimeLocal** (const QDateTime &dt)
  - This method returns QDateTime with Local timespec.*
- QDateTime **asDateTimeUTC** (const QDateTime &dt)
  - This method returns QDateTime with UTC timespec.*
- static const QPointF **boundMargin** (selMargin, selMargin)
- bool **checkSidecarSettings** ()
- void **checkTree** (TreeBranch \*const checkBranch, int level)
- static int **clamp** (int from, int maxVal)
- static QStringList **cleanUserFilterString** (const QString &filterString)
- QStringList **cleanUserFilterString** (QString filterString, const bool caseSensitive, const bool useSemicolon)
- static QPointF **closestPointOfRect** (const QPointF &p, const QRectF &r)
- static QString **colorToString** (const QColor &c)
- template<class ContainerA , class ContainerB >
  - bool **containsAnyOf** (const ContainerA &listA, const ContainerB &listB)
- template<class ContainerA , typename Value , class ContainerB >
  - bool **containsNoneOfExcept** (const ContainerA &list, const ContainerB &noneOfList, const Value &exception)
- void **coordinatesToClipboard** (const GeoCoordinates &coordinates, const QUrl &url, const QString &title)
- static DropAction **copyOrMove** (const QDropEvent \*const e, QWidget \*const view, bool allowMove=true, bool askForGrouping=false)
- static QColor **darkShade** (QColor c)
- QShortcut \* **defineShortcut** (QWidget \*const w, const QKeySequence &key, const QObject \*receiver, const char \*slot)
  - Convenience method for creating keyboard shortcuts.*
- static const quint8 \* **determineFilmStrip** (quint32 videoWidth, quint32 &videoStripWidth, quint32 &videoStripHeight)
- const QDateTime **digiKamBuildDate** ()
- int **digiKamMakeIntegerVersion** (int major, int minor, int patch)
- const QString **digiKamVersion** ()
- void **DNotificationWrapper** (const QString &eventId, const QString &message, QWidget \*const parent, const QString &>windowTitle, const QPixmap &pixmap=QPixmap())
  - Show a notification using KNotify, or KPassivePopup if KNotify is unavailable.*
- static QString **fastNumberToString** (int id)
- static QString **fastNumberToString** (int id)
- static QString **fastNumberToString** (qlonglong id)
  - NOTE: Feel free to optimize.*
- template<typename T >
  - [PointTransformAffine](#) **findAffineTransform** (const std::vector< std::vector< T > > &fromPoints, const std::vector< std::vector< T > > &toPoints)
  - [PointTransformAffine](#) **findSimilarityTransform** (const std::vector< std::vector< float > > &fromPoints, const std::vector< std::vector< float > > &toPoints)
- static QString **formatFontSize** (qreal size)

- static void **formatProfiles** (const QList< [IccProfile](#) > &givenProfiles, QList< [IccProfile](#) > \*const returnedProfiles, QStringList \*const userText)
  - NOTE: if needed outside this class, make it a public static method in a namespace.*
- void **Geoface\_assert** (const char \*const condition, const char \*const filename, const int lineNumber)
- GeoCoordinates::PairList **GeofaceHelperNormalizeBounds** (const GeoCoordinates::Pair &boundsPair)
  - Split bounds crossing the dateline into parts which do not cross the dateline.*
- bool **GeofaceHelperParseBoundsString** (const QString &boundsString, QPair< [GeoCoordinates](#), [GeoCoordinates](#) > \*const boundsCoordinates)
  - Parses a '((lat1, lon1), (lat2, lon2))' bounds string as returned by the JavaScript parts.*
- bool **GeofaceHelperParseLatLonString** (const QString &latLonString, [GeoCoordinates](#) \*const coordinates)
  - Parse a 'lat,lon' string a returned by the JavaScript parts.*
- bool **GeofaceHelperParseXYStringToPoint** (const QString &xyString, QPoint \*const point)
  - Parse a 'X.xxx,Y.yyy' string as returned by the JavaScript parts.*
- qreal **getComponentValue** (const QColor &color, DColorChooserMode chooserMode)
- static QString **getDateFormatLinkText** ()
- std::vector< cv::Rect > **getEyes** (const [FullObjectDetection](#) &shape)
- static int **getOffset** (int Width, int X, int Y, int bytesDepth)
- QString **getUserAgentName** ()
- int **getWarningLevelFromGPSDataContainer** (const [GPSDataContainer](#) &data)
- static int64\_t **heifQIODeviceMetaGetPosition** (void \*userdata)
- static int **heifQIODeviceMetaRead** (void \*data, size\_t size, void \*userdata)
- static int **heifQIODeviceMetaSeek** (int64\_t position, void \*userdata)
- static heif\_reader\_grow\_status **heifQIODeviceMetaWait** (int64\_t target\_size, void \*userdata)
- bool **iccProfileLessThan** ([IccProfile](#) a, [IccProfile](#) b)
- cv::Mat **image2Mat** (const QImage &img, int requiredMatType, MatColorOrder requiredOrder)
  - Convert QImage to cv::Mat.*
- cv::Mat **image2Mat\_shared** (const QImage &img, MatColorOrder \*const order)
  - Convert QImage to cv::Mat without data copy.*
- NoiseDetector::Mat3D **initFiltersHaar** ()
- DIGIKAM\_EXPORT void **installQtTranslationFiles** (QApplication &app)
  - For bundles only, main function to manage all Qt translation files at run-time in application instance.*
- static int **interp** (const quint16 \*src, int p, const int \*off, float dr, float dg, float db)
  - TODO: using liblcms would be fancier... Tetrahedral interpolation, taken from AOSP Gallery2 app.*
- static int **intMult16** (uint a, uint b)
- static int **intMult8** (uint a, uint b)
  - This method is based on the Simulate Texture Film tutorial from GimpGuru.org web site available at this url : [www.gimpguru.org/Tutorials/SimulatedTexture/](http://www.gimpguru.org/Tutorials/SimulatedTexture/).*
- [PointTransformAffine](#) **inv** (const [PointTransformAffine](#) &trans)
- static bool **is7BitAscii** (const QString &s)
- bool **isCursorClicked** (const QPoint &pos, double cursorPos, int width, int height, int gradientWidth)
- bool **isReadableImageFile** (const QString &filePath)
  - Return true if filePath is an image readable by application for thumbnail, preview, or edit.*
- DIGIKAM\_EXPORT bool **isRunningInApplImageBundle** ()
  - Return true if application run in ApplImage bundle.*
- bool **isRunningOnMacOS** ()
- DIGIKAM\_EXPORT bool **isRunningOnNativeKDE** ()
  - Return true if application run on native KDE desktop.*
- template<typename T >
  - static void **ItemFilterFx** (const quint16 \*lutrgb, int lutTableSize, T \*rgb, uint start, uint end, int maxVal, int intensity)
- static QStringList **joinMainAndUserFilterString** (const QChar &sep, const QString &filter, const QString &userFilter)
  - helper method*

- int **layoutMargin** ()  
*Default margin to use in layout.*
- int **layoutSpacing** ()  
*Default spacing to use in layout.*
- template<class T >  
T **length\_squared** (const std::vector< T > &diff)
- bool **lessThanByTitle** (const Album \*first, const Album \*second)  
*for qSort*
- static bool **lessThanForAlbumShortInfo** (const AlbumShortInfo &first, const AlbumShortInfo &second)
- static bool **lessThanForTagProperty** (const TagProperty &first, const TagProperty &second)
- static bool **lessThanForTagShortInfo** (const TagShortInfo &first, const TagShortInfo &second)
- static bool **lessThanLimitedPrecision** (double a, double b)
- static int **Lim\_Max** (int Now, int Up, int Max)
- DIGIKAM\_EXPORT void **loadEcmQtTranslationFiles** (QApplication &app)  
*For bundles only, load ECM Qt translation files at run-time in application instance.*
- DIGIKAM\_EXPORT void **loadStdQtTranslationFiles** (QApplication &app)  
*For bundles only, load standard Qt translation files at run-time in application instance.*
- static bool **localeLessThan** (const QString &a, const QString &b)
- DIGIKAM\_EXPORT QString **macOSBundlePrefix** ()  
*Prefix of macOS Bundle to access to internal Unix hierarchy.*
- QStringList **makeTagString** (const RInfo &info, const QString &inputFormat, const QString &backend←Name)
- QImage **mat2Image** (const cv::Mat &mat, MatColorOrder order, QImage::Format formatHint)  
*Convert cv::Mat to QImage.*
- QImage **mat2Image\_shared** (const cv::Mat &mat, QImage::Format formatHint)  
*Convert cv::Mat to QImage without data copy.*
- static QList< qlonglong > **mergedIdLists** (const HistoryImageId &referenceId, const QList< qlonglong > &uuidList, const QList< qlonglong > &candidates)
- static const QString **mimeTypeCutSelection** (QLatin1String("application/x-kde-cutselection"))
- static int **minimumListHeight** (const QListWidget \*list, int numVisibleEntry)
- static int **minimumListWidth** (const QListWidget \*list)
- static bool **naturalLessThan** (const PathValuePair &a, const PathValuePair &b)
- static bool **newestInfoFirst** (const ItemInfo &a, const ItemInfo &b)
- static int **normalizeAndClamp** (int norm, int sum, int max)
- static bool **oldestInfoFirst** (const ItemInfo &a, const ItemInfo &b)
- void **openOnlineDocumentation** (const QString &section=QString(), const QString &chapter=QString(), const QString &reference=QString())  
*Open online handbook at the section/chapter/reference page.*
- FocusPoint::TypePoint **operator&** (FocusPoint::TypePoint type1, FocusPoint::TypePoint type2)
- FocusPoint::TypePoint & **operator&=** (FocusPoint::TypePoint &type1, FocusPoint::TypePoint type2)
- PointTransformAffine **operator\*** (const PointTransformAffine &lhs, const PointTransformAffine &rhs)
- template<class T >  
std::vector< std::vector< T > > **operator\*** (const std::vector< std::vector< T > > &v1, const std::vector< std::vector< T > > &v2)
- template<class T >  
std::vector< T > **operator\*** (const std::vector< std::vector< T > > &v1, const std::vector< T > &v2)
- template<class T >  
std::vector< std::vector< T > > **operator\*** (const std::vector< T > &v1, const std::vector< T > &v2)
- template<class T >  
std::vector< T > **operator\*** (const std::vector< T > &v1, float d)
- template<class T >  
std::vector< std::vector< T > > **operator+** (const std::vector< std::vector< T > > &v1, const std::vector< std::vector< T > > &v2)

- `template<class T >`  
`std::vector< std::vector< T > > operator+` (const `std::vector< std::vector< T > >` &v1, float d)
- `template<class T >`  
`std::vector< T > operator+` (const `std::vector< T >` &v1, const `std::vector< T >` &v2)
- `template<class T >`  
`std::vector< T > operator-` (const `std::vector< T >` &v1)
- `template<class T >`  
`std::vector< T > operator-` (const `std::vector< T >` &v1, const `std::vector< T >` &v2)
- `template<class T >`  
`std::vector< std::vector< T > > operator/` (const `std::vector< std::vector< T > >` &v1, int divisor)
- `template<class T >`  
`std::vector< T > operator/` (const `std::vector< T >` &v1, int divisor)
- `bool operator<` (const [ThumbnailIdentifier](#) &a, const [ThumbnailIdentifier](#) &b)
- `QDataStream & operator<<` (`QDataStream &ds`, const [CamItemInfo](#) &info)
- `QDataStream & operator<<` (`QDataStream &ds`, const [PhotoInfoContainer](#) &info)
- `QDataStream & operator<<` (`QDataStream &ds`, const [VideoInfoContainer](#) &info)
- `QDebug operator<<` (`QDebug dbg`, const [BatchToolSet](#) &s)  
*QDebug() stream operator. Writes property t to the debug output in a nicely formatted way.*
- `QDebug operator<<` (`QDebug dbg`, const [CamItemInfo](#) &info)  
*QDebug() stream operator. Writes property info to the debug output in a nicely formatted way.*
- `QDebug operator<<` (`QDebug dbg`, const [CaptionValues](#) &val)  
*QDebug() stream operator. Writes values val to the debug output in a nicely formatted way.*
- `QDebug operator<<` (`QDebug dbg`, const [DbEngineParameters](#) &p)
- `QDebug operator<<` (`QDebug dbg`, const [DMetadataSettingsContainer](#) &inf)  
*QDebug() stream operator. Writes property inf to the debug output in a nicely formatted way.*
- `QDebug operator<<` (`QDebug dbg`, const [DRawDecoderSettings](#) &s)  
*QDebug() stream operator. Writes settings s to the debug output in a nicely formatted way.*
- `QDebug operator<<` (`QDebug dbg`, const [DRawInfo](#) &c)  
*QDebug() stream operator. Writes container c to the debug output in a nicely formatted way.*
- `QDebug operator<<` (`QDebug dbg`, const [DTrashItemInfo](#) &info)  
*QDebug() stream operator. Writes property info to the debug output in a nicely formatted way.*
- `QDebug operator<<` (`QDebug dbg`, const [FaceTagsIface](#) &f)
- `QDebug operator<<` (`QDebug dbg`, const [FocusPoint](#) &fp)  
*QDebug() stream operator. Writes FocusPoint to the debug output in a nicely formatted way.*
- `QDebug operator<<` (`QDebug dbg`, const [GeolocationSettingsContainer](#) &inf)  
*QDebug(DIGIKAM\_GEOENGINE\_LOG) << QString::fromUtf8() stream operator. Writes property inf to the debug output in a nicely formatted way.*
- `QDebug operator<<` (`QDebug dbg`, const [HistoryImageId](#) &id)
- `QDebug operator<<` (`QDebug dbg`, const [HistoryVertexProperties](#) &props)
- `QDebug operator<<` (`QDebug dbg`, const [ImageQualityContainer](#) &s)  
*QDebug() stream operator. Writes property s to the debug output in a nicely formatted way.*
- `QDebug operator<<` (`QDebug dbg`, const [IptcCoreContactInfo](#) &inf)  
*QDebug() stream operator. Writes property inf to the debug output in a nicely formatted way.*
- `QDebug operator<<` (`QDebug dbg`, const [IptcCoreLocationInfo](#) &inf)  
*QDebug() stream operator. Writes property inf to the debug output in a nicely formatted way.*
- `QDebug operator<<` (`QDebug dbg`, const [ItemHistoryGraph](#) &g)
- `QDebug operator<<` (`QDebug dbg`, const [LocalizeContainer](#) &inf)  
*QDebug() stream operator. Writes property inf to the debug output in a nicely formatted way.*
- `QDebug operator<<` (`QDebug dbg`, const [MaintenanceSettings](#) &s)  
*QDebug(DIGIKAM\_GENERAL\_LOG) stream operator.*
- `QDebug operator<<` (`QDebug dbg`, const [MetaEngineSettingsContainer](#) &inf)  
*QDebug() stream operator. Writes property inf to the debug output in a nicely formatted way.*
- `QDebug operator<<` (`QDebug dbg`, const [NamespaceEntry](#) &inf)

- QDebug()* stream operator. Writes property *inf* to the debug output in a nicely formatted way.
- QDebug **operator**<< (QDebug dbg, const [NRContainer](#) &inf)
- QDebug()* stream operator. Writes property *inf* to the debug output in a nicely formatted way.
- QDebug **operator**<< (QDebug dbg, const [PhotoInfoContainer](#) &t)
- QDebug()* stream operator. Writes property *t* to the debug output in a nicely formatted way.
- QDebug **operator**<< (QDebug dbg, const [TagRegion](#) &r)
- QDebug **operator**<< (QDebug dbg, const [Template](#) &t)
- QDebug()* stream operator. Writes *Template* to the debug output in a nicely formatted way.
- QDebug **operator**<< (QDebug dbg, const [VideoInfoContainer](#) &t)
- QDebug()* stream operator. Writes property *t* to the debug output in a nicely formatted way.
- QDebug **operator**<< (QDebug stream, const [ItemInfo](#) &info)
- QDebug()* stream operator. Writes property *info* to the debug output in a nicely formatted way.
- static bool **operator**== (const [DImageHistory::Entry](#) &e1, const [DImageHistory::Entry](#) &e2)
- bool **operator**== (const [SearchTextSettings](#) &a, const [SearchTextSettings](#) &b)
- QDataStream & **operator**>> (QDataStream &ds, [CamItemInfo](#) &info)
- QDataStream & **operator**>> (QDataStream &ds, [PhotoInfoContainer](#) &info)
- QDataStream & **operator**>> (QDataStream &ds, [VideoInfoContainer](#) &info)
- [FocusPoint::TypePoint](#) **operator**| ([FocusPoint::TypePoint](#) type1, [FocusPoint::TypePoint](#) type2)
- Boolean Operators over TypePoint type.*
- [FocusPoint::TypePoint](#) & **operator**|= ([FocusPoint::TypePoint](#) &type1, [FocusPoint::TypePoint](#) type2)
- static QString **profileUserString** (const [IccProfile](#) &p)
- NOTE: if needed outside this class, make it a public static method in a namespace.*
- **Q\_GLOBAL\_STATIC\_WITH\_ARGS** ([DbEngineConfigSettingsLoader](#), dbcloader,(QStandardPaths::locate(QStandardPaths::GenericDataLocation, QLatin1String("digikam/database/dbconfig.xml")), dbcconfig←\_xml\_version)) [DbEngineConfigSettings](#) [DbEngineConfig](#)
- size\_t **qHash** (const [CollectionLocation](#) &loc)
- QT\_HASH\_TYPE **qHash** (const [Digikam::AlbumCopyMoveHint](#) &hint)
- size\_t **qHash** (const [ItemInfo](#) &info)
- int **QPointSquareDistance** (const QPoint &a, const QPoint &b)
- Helper function, returns the square of the distance between two points.*
- template<typename T, class Container >  
void **removeAnyInterval** (Container &list, const T &begin, const T &end)
- static int **renderingIntentToLcmsIntent** ([IccTransform::RenderingIntent](#) intent)
- LqrRetVal **s\_carverProgressEnd** (const gchar \*end\_message)
- LqrRetVal **s\_carverProgressInit** (const gchar \*init\_message)
- LqrRetVal **s\_carverProgressUpdate** (gdouble percentage)
- bool **s\_checkSolidCamera** (const [Solid::Device](#) &cameraDevice)
- static const QString **s\_configUseLargeThumbsEntry** (QLatin1String("Use Large Thumbs"))
- bool **s\_dmcompare** (const [NamespaceEntry](#) &e1, const [NamespaceEntry](#) &e2)
- static DropAction **s\_groupAction** (const QDropEvent \*const, QWidget \*const view)
- bool **s\_inlineTranslateString** (const QString &text, const QString &trCode, QString &tr, QString &error)
- Helper re-entrant static method to translate a string with online translator.*
- bool **s\_isHeifSuccess** (const struct heif\_error \*const error)
- QStringList **s\_keywordsSeparation** (const QString &data)
- QString **s\_labelForSolidCamera** (const [Solid::Device](#) &cameraDevice)
- QString **s\_rawFileExtensions** ()
- QMap< QString, QString > **s\_rawFileExtensionsdWithDesc** ()
- int **s\_rawFileExtensionsVersion** ()
- void **s\_readHEICMetadata** (struct heif\_context \*const heif\_context, heif\_item\_id image\_id, QByteArray &exif, QByteArray &iptc, QByteArray &xmp)
- qint64 **s\_secondsSinceJanuary1904** (const QDateTime &dt)
- QString **s\_setXmpTagStringFromEntry** (const [DMetadata](#) \*const meta, const QStringList &lst, const [DMetadata::MetaDataMap](#) &map, const QStringList &xmpTags=QStringList())



- Search first occurrence of string in 'map' with keys given by 'lst'.*
- static QVariant **safeToVariant** (const QString &s)
- void **setComponentValue** (QColor &color, DColorChooserMode chooserMode, qreal value)
- bool **setExifXmpTagDataVariant** (DMetadata \*const meta, const char \*const exifTagName, const char \*const xmpTagName, const QVariant &value)
- DIGIKAM\_EXPORT void **setMacOSEnvironment** ()
  - For MacOS bundles only, set necessary MacOS environment variables.*
- void **setOpenCLEnvironment** (bool b)
  - For OpenCV framework, set necessary environment variables to use OpenCL features.*
- DIGIKAM\_EXPORT void **setWindowsEnvironment** (QApplication &app)
  - For Windows only, set necessary Windows environment variables.*
- void **showDigikamComponentsInfo** ()
- void **showDigikamDatabaseStat** ()
- void **showRawCameraList** ()
  - Show a dialog with all RAW camera supported by digiKam, through libraw.*
- QDateTime **startOfDay** (const QDate &date)
  - This method returns QDateTime from with date set to parameter date and time set to start of the day.*
- static QColor **stringToColor** (const QString &s)
- QStringList **supportedImageMimeTypes** (QIODevice::OpenModeFlag mode, QString &allTypes)
  - Return list of supported image formats by Qt for reading or writing operations if suitable container used by QFileDialog.*
- static DropAction **tagAction** (const QDropEvent \*const, QWidget \*const view, bool askForGrouping)
- static LqrEnergyFuncBuiltinType **toLqrEnergy** (ContentAwareContainer::EnergyFunction func)
- static LqrResizeOrder **toLqrOrder** (Qt::Orientation direction)
- QString **toolButtonStyleSheet** ()
  - Style sheet for transparent QToolButtons over image and video preview.*
- static QString **toString** (const HistoryVertexProperties &props)
- bool **TrackCorrelationLessThan** (const TrackCorrelator::Correlation &a, const TrackCorrelator::Correlation &b)
- DIGIKAM\_EXPORT void **unloadQtTranslationFiles** (QApplication &app)
  - For bundles only, unload all Qt translation files at run-time in application instance.*
- static bool **uuidDoesNotDiffer** (const HistoryImageId &referenceId, qlonglong id)

## Variables

- auto **accessCol**
- auto **accessRow**
- static const int **AUTOEXPANDEDELAY** = 800
  - Delay in milliseconds to automatically expands album tree-view with D&D See bug #286263 for details.*
- ImageCurves::CRMatrix **CR\_basis**
- static const int **DEFAULT\_POPUP\_TIME** = 6 \* 1000
- static const int **DEFAULT\_POPUP\_TYPE** = DNotificationPopup::Boxed
- const int **DNN\_MODEL\_THRESHOLD\_NOT\_SET** = 1000
- static const char \* **ExifHumanList** []
- static float **FACE\_TEMPLATE** [3][2]
  - Template for face landmark to perform alignment with open face This variable must be declared as static so that it is allocated as long as digiKam is still running.*
- const std::map< FaceScanSettings::FaceDetectionSize, int > **faceenum2size**
- const int **GeofaceMinMarkerGroupingRadius** = 1
- const int **GeofaceMinThumbnailGroupingRadius** = 15
- const int **GeofaceMinThumbnailSize** = GeofaceMinThumbnailGroupingRadius \* 2
- static const char \* **lptcHumanList** []
- static const char \* **MakerNoteHumanList** []

- const int **MAX\_MATRIX\_SIZE** = 25
- static const int **MAX\_SEGMENT\_16BIT** = [NUM\\_SEGMENTS\\_16BIT](#) - 1
- static const int **MAX\_SEGMENT\_8BIT** = [NUM\\_SEGMENTS\\_8BIT](#) - 1
- static const struct Digikam::NameSpaceDefinition [namespaceTitleDefinitions](#) []
- static const int **NoRating** = -1
- static const int **NUM\_SEGMENTS\_16BIT** = 65536
  - Segments for histograms and curves.*
- static const int **NUM\_SEGMENTS\_8BIT** = 256
- static const Qt::WindowFlags **POPUP\_FLAGS** = Qt::Tool | Qt::WindowStaysOnTopHint | Qt::FramelessWindowHint
- static const int **RatingMax** = 5
- static const int **RatingMin** = 0
  - Field value limits for all digiKam-specific fields (not EXIF/IPTC fields)*
- const float **RATIO\_POINT\_IMAGE** = 1 / 120
- const int **RoleGPSItemInfo** = Qt::UserRole + 1
- bool **s\_hResize** = false
- static bool **s\_imageSmoothScale** = true
- [ContentAwareFilter](#) \* **s\_resiser** = nullptr
- bool **s\_stage** = false
  - Resizement is decomposed in 2 stages: horizontal and vertical.*
- static bool **s\_useLargeThumbs** = false
- bool **s\_wResize** = false
- static const qreal **selMargin** = 8.0
- const int **SIZE\_FILTER** = 4
- static const quint32 **SMALLEST\_FILM\_STRIP\_WIDTH** = 4
- static const double [spectral\\_chromaticity](#) [81][3]
  - The following table gives the CIE color matching functions  $\bar{x}(\lambda)$ ,  $\bar{y}(\lambda)$ , and  $\bar{z}(\lambda)$ , for wavelengths  $\lambda$  at 5 nanometer increments from 380 nm through 780 nm.*
- static const quint8 [videoStrip16](#) [16 \* 16 \* 3]
- static const quint8 **videoStrip32** [32 \* 32 \* 3]
- static const quint8 [videoStrip4](#) [4 \* 4 \* 3]
- static const quint8 **videoStrip64** [64 \* 64 \* 3]
- static const quint8 [videoStrip8](#) [8 \* 8 \* 3]
- static const char \* [XmpHumanList](#) []

### 8.1.1 Detailed Description

NOTE: Good explanations about GPS (in French) can be found at this url : [www.gpspassion.com/forumsen/topic.asp?TOPIC\\_ID=16593](http://www.gpspassion.com/forumsen/topic.asp?TOPIC_ID=16593).

References about DNG: DNG SDK tutorial: [www.adobeforums.com/webx/.3bc2944e](http://www.adobeforums.com/webx/.3bc2944e) [www.adobeforums.com/webx/.3c054bde](http://www.adobeforums.com/webx/.3c054bde) DNG review: [www.barrypearson.co.uk/articles/dng/index.htm](http://www.barrypearson.co.uk/articles/dng/index.htm) DNG intro: [www.adobe.com/digitalimag/pdfs/dng\\_primer.pdf](http://www.adobe.com/digitalimag/pdfs/dng_primer.pdf) [www.adobe.com/products/dng/pdfs/DNG\\_primer\\_manufacturers.pdf](http://www.adobe.com/products/dng/pdfs/DNG_primer_manufacturers.pdf) DNG Specification: [www.images.adobe.com/content/dam/Adobe/en/products/photoshop/pdfs/dng\\_spec\\_1.5.0.0.pdf](http://www.images.adobe.com/content/dam/Adobe/en/products/photoshop/pdfs/dng_spec_1.5.0.0.pdf) TIFF/EP Spec.: [www.map.tu.chiba-u.ac.jp/IEC/100/TA2/recdoc/N4378.pdf](http://www.map.tu.chiba-u.ac.jp/IEC/100/TA2/recdoc/N4378.pdf) DNG SDK reference: [www.thomasdideriksen.dk/misc/File%20Formats/dng\\_sdk\\_refman.pdf](http://www.thomasdideriksen.dk/misc/File%20Formats/dng_sdk_refman.pdf) DNG SDK tarball: [helpx.adobe.com/photoshop/digital-negative.html::dng\\_sdk\\_download](http://helpx.adobe.com/photoshop/digital-negative.html::dng_sdk_download) DNG users forum: [www.adobeforums.com/webx/.3bb5f0ec](http://www.adobeforums.com/webx/.3bb5f0ec).

Applications using DNG SDK: DNG4PS2: [dng4ps2.chat.ru/index\\_en.html](http://dng4ps2.chat.ru/index_en.html) CORNERFIX: [sourceforge.net/projects/cornerfix](http://sourceforge.net/projects/cornerfix) ADOBE DNG CONVERTER: [helpx.adobe.com/photoshop/using/adobe-dng-converter.html](http://helpx.adobe.com/photoshop/using/adobe-dng-converter.html) DNGCONVERT: [github.com/jmue/dngconvert](https://github.com/jmue/dngconvert) MOVIE2DNG: [elphel.svn.sourceforge.net/svnroot/elphel/tools/](http://elphel.svn.sourceforge.net/svnroot/elphel/tools/) Movie2DNG RAW2DNG : [github.com/Fimagena/raw2dng](https://github.com/Fimagena/raw2dng)

## 8.1.2 Typedef Documentation

### 8.1.2.1 ActionJobCollection

```
typedef QHash<ActionJob*, int> Digikam::ActionJobCollection
```

Priority value can be used to control the run queue's order of execution. Zero priority want mean to process job with higher priority.

### 8.1.2.2 DItemsListIsLessThanHandler

```
typedef bool(* Digikam::DItemsListIsLessThanHandler) (const QTreeWidgetItem *current, const QTreeWidgetItem &other)
```

Sort items call this method in DItemsListViewItem::operator<. To setup this method, uses DItemLIst::setIsLessThanHandler().

## 8.1.3 Enumeration Type Documentation

### 8.1.3.1 DetectorNNModel

```
enum Digikam::DetectorNNModel
```

Enumerator

DNNDetectorSSD	SSD MobileNet neural network inference [ <a href="https://github.com/arunponnusamy/cvlib">https://github.com/arunponnusamy/cvlib</a> ].
DNNDetectorYOLOv3	YOLO neural network inference [ <a href="https://github.com/sthanhng/yoloface">https://github.com/sthanhng/yoloface</a> ].
DNNDetectorYuNet	YuNet neural network inference [ <a href="https://github.com/opencv/opencv_zoo/tree/main">https://github.com/opencv/opencv_zoo/tree/main</a> ].

### 8.1.3.2 FullScreenOptions

```
enum Digikam::FullScreenOptions
```

Enumerator

FS_TOOLBARS	Manage Tools bar in full-screen mode.
FS_THUMBBAR	Manage Thumb bar in full-screen mode.
FS_SIDEBARS	Manage Side bars in full-screen mode.
FS_STATUSBAR	Manage Status bar in full-screen mode.
FS_NONE	No full-screen options.
FS_ALBUMGUI	<a href="#">Album</a> GUI Config.
FS_EDITOR	Image Editor Config.
FS_LIGHTTABLE	Light Table Config.
FS_IMPORTUI	Import UI Config.



### 8.1.3.3 GeoGroupStateEnum

```
enum Digikam::GeoGroupStateEnum
```

The idea is that a group consists of more than one object. Thus the resulting state is that either none of the objects, some or all of them have a certain state. The constants for each state are set up such that they can be logically or'ed: If a group has the state `___All`, and another the state `___Some`, the bit representing `___Some` is always propagated along. You only have to make sure that once you reach an object with `___None`, and the computed state is `___All`, to set the `___Some` bit.

Selected`___`: An object is selected. FilteredPositive`___`: An object was highlighted by a filter. This usually means that not-positively-filtered objects should be hidden. RegionSelected`___`: An object is inside a region of interest on the map.

### 8.1.3.4 HistogramRenderingType

```
enum Digikam::HistogramRenderingType
```

#### Enumerator

FullImageHistogram	Full image histogram rendering.
ImageSelectionHistogram	Image selection histogram rendering.

### 8.1.3.5 HistogramScale

```
enum Digikam::HistogramScale
```

#### Enumerator

LinScaleHistogram	Linear scale.
LogScaleHistogram	Logarithmic scale.

### 8.1.3.6 HudSide

```
enum Digikam::HudSide
```

#### Enumerator

HS_None	Special value used to avoid initial animation.
---------	--

### 8.1.3.7 MeaningOfDirection

```
enum Digikam::MeaningOfDirection
```

This direction has a meaning with methods such as `roots()` or `leaves()`.

**Enumerator**

ParentToChild	Edges are directed from a parent to its child.
ChildToParent	Edges are direct from a child to its parent.

**8.1.3.8 OperationType**

```
enum Digikam::OperationType
```

Originally introduced for grouping to configure whether an operation should be done on all group members or only it's head.

**Enumerator**

UnspecifiedOps	This element must always come last.
----------------	-------------------------------------

**8.1.3.9 YoloVersions**

```
enum class Digikam::YoloVersions [strong]
```

**Enumerator**

YOLOV5NANO	yolov5n_batch_16_s320.onnx
YOLOV5XLARGE	yolov5x_batch_16_s320.onnx
RESNET50	resnet50.onnx

**8.1.4 Function Documentation****8.1.4.1 adjustedEnvironmentForAppImage()**

```
QProcessEnvironment Digikam::adjustedEnvironmentForAppImage ( )
```

Use case : system based Hugin CLI tools called by Panorama wizard. If digiKam do not run as AppImage bundle, this method return a QProcessEnvironment instance based on system environment.

**8.1.4.2 coordinatesToClipboard()**

```
void DIGIKAM_EXPORT Digikam::coordinatesToClipboard (
    const GeoCoordinates & coordinates,
    const QUrl & url,
    const QString & title )
```

NOTE: importing this representation into Marble does not show anything, but Merkaartor shows the point

importing this data into Marble and Merkaartor works

### 8.1.4.3 DNotificationWrapper()

```
void DIGIKAM_EXPORT Digikam::DNotificationWrapper (
    const QString & eventId,
    const QString & message,
    QWidget *const parent,
    const QString & windowTitle,
    const QPixmap & pixmap = QPixmap() )
```

#### Parameters

<i>eventId</i>	Event id for this notification, KNotification::Notification is used if this is empty. Events have to be configured in digikam.notifyrc
<i>message</i>	Message to display
<i>parent</i>	Widget which owns the notification
<i>windowTitle</i>	Title of the notification window (only used for KPassivePopup)
<i>pixmap</i>	Pixmap to show in the notification, in addition to the digikam logo.

### 8.1.4.4 fastNumberToString()

```
static QString Digikam::fastNumberToString (
    qulonglong id ) [inline], [static]
```

QString::number is 3x slower.

### 8.1.4.5 GeofaceHelperParseLatLonString()

```
DIGIKAM_EXPORT bool Digikam::GeoIfaceHelperParseLatLonString (
    const QString & latLonString,
    GeoCoordinates *const coordinates )
```

helper functions

#### Returns

true if the string could be parsed successfully

### 8.1.4.6 image2Mat()

```
cv::Mat Digikam::image2Mat (
    const QImage & img,
    int requiredMatType = CV_8UC(0),
    MatColorOrder requiredOrder = MCO_BGR )
```

Convert QImage to/from cv::Mat.

- cv::Mat
  - Supported channels

- \* 1 channel
- \* 3 channels (B G R), (R G B)
- \* 4 channels (B G R A), (R G B A), (A R G B)
- Supported depth
  - \* CV\_8U [0, 255]
  - \* CV\_16U [0, 65535]
  - \* CV\_32F [0, 1.0]
- QImage
  - All of the formats of QImage are supported.

#### 8.1.4.7 image2Mat\_shared()

```
cv::Mat Digikam::image2Mat_shared (
    const QImage & img,
    MatColorOrder *const order = nullptr )
```

Convert QImage to/from cv::Mat without data copy.

- Supported QImage formats and cv::Mat types are:
  - QImage::Format\_Indexed8 <==> CV\_8UC1
  - QImage::Format\_Alpha8 <==> CV\_8UC1
  - QImage::Format\_Grayscale8 <==> CV\_8UC1
  - QImage::Format\_RGB888 <==> CV\_8UC3 (R G B)
  - QImage::Format\_RGB32 <==> CV\_8UC4 (A R G B or B G R A)
  - QImage::Format\_ARGB32 <==> CV\_8UC4 (A R G B or B G R A)
  - QImage::Format\_ARGB32\_Premultiplied <==> CV\_8UC4 (A R G B or B G R A)
  - QImage::Format\_RGBX8888 <==> CV\_8UC4 (R G B A)
  - QImage::Format\_RGBA8888 <==> CV\_8UC4 (R G B A)
  - QImage::Format\_RGBA8888\_Premultiplied <==> CV\_8UC4 (R G B A)
- For QImage::Format\_RGB32 ,QImage::Format\_ARGB32 and QImage::Format\_ARGB32\_Premultiplied, the color channel order of cv::Mat will be (B G R A) in little endian system or (A R G B) in big endian system.
- User must make sure that the color channels order is the same as the color channels order required by QImage.

#### 8.1.4.8 openOnlineDocumentation()

```
DIGIKAM_EXPORT void Digikam::openOnlineDocumentation (
    const QString & section = QString(),
    const QString & chapter = QString(),
    const QString & reference = QString() )
```

if section and chapter and reference are empty, fromt page is open. ( [https://en.wikipedia.org/wiki/Matrix\\_\(protocol\)#Bridges](https://en.wikipedia.org/wiki/Matrix_(protocol)#Bridges)) if only chapter and reference are empty, section page is open. (as: [https://docs.digikam.org/en/main\\_window.html](https://docs.digikam.org/en/main_window.html)) if only reference is empty, chapter from section page is open. (as: [https://docs.digikam.org/en/main\\_window/people\\_view.html](https://docs.digikam.org/en/main_window/people_view.html)) else reference at chapter from section page is open. (as: [https://docs.digikam.org/en/main\\_window/people\\_view.html#face-recognition](https://docs.digikam.org/en/main_window/people_view.html#face-recognition))

#### 8.1.4.9 operator<<()

```
QDebug Digikam::operator<< (
    QDebug dbg,
    const MaintenanceSettings & s )
```

QDebug() stream operator. Writes property *s* to the debug output in a nicely formatted way.

Writes property *s* to the debug output in a nicely formatted way.

#### 8.1.4.10 QPointSquareDistance()

```
DIGIKAM_EXPORT int Digikam::QPointSquareDistance (
    const QPoint & a,
    const QPoint & b )
```

##### Parameters

<i>a</i>	Point a
<i>b</i>	Point b

##### Returns

Square of the distance between *a* and *b*

#### 8.1.4.11 s\_inlineTranslateString()

```
bool DIGIKAM_EXPORT Digikam::s_inlineTranslateString (
    const QString & text,
    const QString & trCode,
    QString & tr,
    QString & error )
```

Language from string is auto-detected, and target language is specified to 'trCode'. If string can be processed, translation is returned to 'tr' and function return true, else false is returned with a dysfunction description in 'error'.

#### 8.1.4.12 s\_rawFileExtensionsdWithDesc()

```
DIGIKAM_EXPORT QMap< QString, QString > Digikam::s_rawFileExtensionsdWithDesc ( )
```

NOTE: extension list Version 1 and 2 are taken from [www.cybercom.net/~dcoffin/dcraw/rawphoto.c](http://www.cybercom.net/~dcoffin/dcraw/rawphoto.c)

Ext	Descriptions From
	<a href="http://www.file-extensions.org">www.file-extensions.org</a>
	<a href="http://en.wikipedia.org/wiki/RAW_file_format">en.wikipedia.org/wiki/RAW_file_format</a>
	<a href="http://filext.com">filext.com</a>

NOTE: VERSION 1

These images are based on the TIFF image standard.

For these models: Kodak DSC Pro SLR/c, Kodak DSC Pro SLR/n, Kodak DSC Pro 14N, Kodak DSC PRO 14nx.

DNG is publicly available archival format for the raw files generated by digital cameras. By addressing the lack of an open standard for the raw files created by individual camera models, DNG helps ensure that photographers will be able to access their files in the future.

For DSC-F828 8 megapixel digital camera or Sony DSC-R1.

For devices based on Foveon X3 direct image sensor.

For Alpha devices.

NOTE: VERSION 2

NOTE: VERSION 3

NOTE: VERSION 4

NOTE: VERSION 5

NOTE: VERSION 6

NOTE: VERSION 7

NOTE: VERSION 8

#### 8.1.4.13 `s_rawFileExtensionsVersion()`

```
DIGIKAM_EXPORT int Digikam::s_rawFileExtensionsVersion ( )
```

NOTE: increment this number whenever you change the above strings

#### 8.1.4.14 `s_setXmpTagStringFromEntry()`

```
QString Digikam::s_setXmpTagStringFromEntry (
    const DMetadata *const meta,
    const QStringList & lst,
    const DMetadata::MetaDataMap & map,
    const QStringList & xmpTags = QStringList() )
```

Return the string match. If 'xmpTags' is not empty, register XMP tags value with string.

#### 8.1.4.15 `setExifXmpTagDataVariant()`

```
bool Digikam::setExifXmpTagDataVariant (
    DMetadata *const meta,
    const char *const exifTagName,
    const char *const xmpTagName,
    const QVariant & value )
```

### 8.1.4.16 supportedImageMimeTypes()

```
DIGIKAM_EXPORT QStringList Digikam::supportedImageMimeTypes (
    QIODevice::OpenModeFlag mode,
    QString & allTypes )
```

For simple container of type mime, use 'allTypes' string. Supported modes are QIODevice::ReadOnly, QIODevice::WriteOnly, and QIODevice::ReadWrite.

## 8.1.5 Variable Documentation

### 8.1.5.1 accessCol

```
auto Digikam::accessCol
```

#### Initial value:

```
= [] (const cv::Mat& mat)
{
    return [mat](int index)
    {
        return mat.col(index);
    };
}
```

### 8.1.5.2 accessRow

```
auto Digikam::accessRow
```

#### Initial value:

```
= [] (const cv::Mat& mat)
{
    return [mat](int index)
    {
        return mat.row(index);
    };
}
```

### 8.1.5.3 CR\_basis

```
ImageCurves::CRMatrix Digikam::CR_basis
```

#### Initial value:

```
=
{
    { -0.5, 1.5, -1.5, 0.5 },
    { 1.0, -2.5, 2.0, -0.5 },
    { -0.5, 0.0, 0.5, 0.0 },
    { 0.0, 1.0, 0.0, 0.0 },
}
```

### 8.1.5.4 ExifHumanList

```
const char* Digikam::ExifHumanList[] [static]
```

#### Initial value:

```
=
{
    "Make",
    "Model",
    "DateTime",
    "ImageDescription",
    "Copyright",
    "ShutterSpeedValue",
    "ApertureValue",
    "ExposureProgram",
    "ExposureMode",
    "ExposureBiasValue",
    "ExposureTime",
    "WhiteBalance",
    "ISOSpeedRatings",
    "FocalLength",
    "SubjectDistance",
    "MeteringMode",
    "Contrast",
    "Saturation",
    "Sharpness",
    "LightSource",
    "Flash",
    "FNumber",
    "GPSLatitude",
    "GPSLongitude",
    "GPSAltitude",
    "-1"
}
```

### 8.1.5.5 FACE\_TEMPLATE

```
float Digikam::FACE_TEMPLATE[3][2] [static]
```

#### Initial value:

```
=
{
    { 18.639072F, 16.249624F },
    { 75.73048F, 15.18443F },
    { 47.515285F, 49.38637F }
}
```

We need that because this variable is the internal data for matrix faceTemplate below.

### 8.1.5.6 faceenum2size

```
const std::map<FaceScanSettings::FaceDetectionSize, int> Digikam::faceenum2size
```

#### Initial value:

```
{
    { FaceScanSettings::FaceDetectionSize::ExtraLarge, 420 },
    { FaceScanSettings::FaceDetectionSize::Large, 620 },
    { FaceScanSettings::FaceDetectionSize::Medium, 800 },
    { FaceScanSettings::FaceDetectionSize::Small, 1200 },
    { FaceScanSettings::FaceDetectionSize::ExtraSmall, 2000 }
}
```

### 8.1.5.7 GeofaceMinMarkerGroupingRadius

```
const int Digikam::GeoIfaceMinMarkerGroupingRadius = 1
```



### 8.1.5.8 IptcHumanList

```
const char* Digikam::IptcHumanList[] [static]
```

#### Initial value:

```
=
{
    "Caption",
    "City",
    "Contact",
    "Copyright",
    "Credit",
    "DateCreated",
    "Headline",
    "Keywords",
    "ProvinceState",
    "Source",
    "Urgency",
    "Writer",
    "-1"
}
```

### 8.1.5.9 namespaceTitleDefinitions

```
const struct Digikam::NameSpaceDefinition Digikam::namespaceTitleDefinitions[] [static]
```

#### Initial value:

```
=
{
    {
        NamespaceEntry::TAGS,
    },
    {
        NamespaceEntry::TITLE,
    },
    {
        NamespaceEntry::RATING,
    },
    {
        NamespaceEntry::COMMENT,
    },
    {
        NamespaceEntry::PICKLABEL,
    },
    {
        NamespaceEntry::COLORLABEL,
    },
}
```

### 8.1.5.10 spectral\_chromaticity

```
const double Digikam::spectral_chromaticity[81][3] [static]
```

This table is used in conjunction with Planck's law for the energy spectrum of a black body at a given temperature to plot the black body curve on the CIE chart.

The following table gives the spectral chromaticity co-ordinates  $x(\lambda)$  and  $y(\lambda)$  for wavelengths in 5 nanometer increments from 380 nm through 780 nm. These coordinates represent the position in the CIE x-y space of pure spectral colors of the given wavelength, and thus define the outline of the CIE "tongue" diagram.



### 8.1.5.14 XmpHumanList

```
const char* Digikam::XmpHumanList[] [static]
```

#### Initial value:

```
=
{
    "Description",
    "City",
    "Relation",
    "Rights",
    "Publisher",
    "CreateDate",
    "Title",
    "Identifier",
    "State",
    "Source",
    "Rating",
    "Advisory",
    "-1"
}
```

## 8.2 Digikam::Matrix Namespace Reference

If the picture is displayed according to the exif orientation tag, the user will request rotating operations relative to what he sees, and that is the picture rotated according to the EXIF tag.

### Functions

- [MetaEngineRotation matrix](#) ([MetaEngine::ImageOrientation](#) exifOrientation)
- [MetaEngineRotation matrix](#) ([MetaEngineRotation::TransformationAction](#) action)

### Variables

- static const [MetaEngineRotation flipHorizontal](#) (-1, 0, 0, 1)
- static const [MetaEngineRotation flipVertical](#) (1, 0, 0, -1)
- static const [MetaEngineRotation identity](#) (1, 0, 0, 1)
- static const [MetaEngineRotation rotate180](#) (-1, 0, 0, -1)
- static const [MetaEngineRotation rotate270](#) (0, -1, 1, 0)
- static const [MetaEngineRotation rotate90](#) (0, 1, -1, 0)
- static const [MetaEngineRotation rotate90flipHorizontal](#) (0, 1, 1, 0)  
*first rotate, then flip*
- static const [MetaEngineRotation rotate90flipVertical](#) (0, -1, -1, 0)  
*first rotate, then flip*

### 8.2.1 Detailed Description

So the operation requested and the given EXIF angle must be combined. E.g. if orientation is "6" (rotate 90 clockwise to show correctly) and the user selects 180 clockwise, the operation is 270. If the user selected 270, the operation would be None (and clearing the exif tag).

This requires to describe the transformations in a model which cares for both composing (180+90=270) and eliminating (180+180=no action), as well as the non-commutative nature of the operations (vflip+90 is not 90+vflip)

All 2D transformations can be described by a 2x3 matrix, see [QWMetaEngineRotation](#). All transformations needed here - rotate 90, 180, 270, flipV, flipH - can be described in a 2x2 matrix with the values 0,1,-1 (because flipping is expressed by changing the sign only, and sine and cosine of 90, 180 and 270 are either 0,1 or -1).

$$x' = m_{11} x + m_{12} y \quad y' = m_{21} x + m_{22} y$$

Moreover, all combinations of these rotate/flip operations result in one of the eight matrices defined below. This did not prove that mathematically, but empirically.

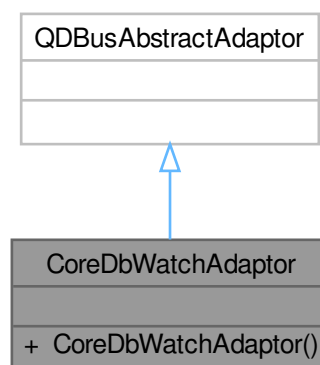


# Chapter 9

## Class Documentation

### 9.1 CoreDbWatchAdaptor Class Reference

Inheritance diagram for CoreDbWatchAdaptor:



#### Signals

- void **signalAlbumChangeDBus** (const QString &databaseldentifier, const QString &applicationIdentifier, const [Digikam::AlbumChangeset](#) &changeset)
- void **signalAlbumRootChangeDBus** (const QString &databaseldentifier, const QString &applicationIdentifier, const [Digikam::AlbumRootChangeset](#) &changeset)
- void **signalCollectionImageChangeDBus** (const QString &databaseldentifier, const QString &applicationIdentifier, const [Digikam::CollectionImageChangeset](#) &changeset)
- void **signalImageChangeDBus** (const QString &databaseldentifier, const QString &applicationIdentifier, const [Digikam::ImageChangeset](#) &changeset)
- void **signalImageTagChangeDBus** (const QString &databaseldentifier, const QString &applicationIdentifier, const [Digikam::ImageTagChangeset](#) &changeset)
- void **signalSearchChangeDBus** (const QString &databaseldentifier, const QString &applicationIdentifier, const [Digikam::SearchChangeset](#) &changeset)
- void **signalTagChangeDBus** (const QString &databaseldentifier, const QString &applicationIdentifier, const [Digikam::TagChangeset](#) &changeset)

Public Member Functions

- CoreDbWatchAdaptor (Digikam::CoreDbWatch \*const watch)

## 9.2 Digikam::AbstractAlbumModel Class Reference

Inheritance diagram for Digikam::AbstractAlbumModel:



## Public Types

- enum [AlbumDataRole](#) {  
[AlbumTitleRole](#) = Qt::UserRole , [AlbumTypeRole](#) = Qt::UserRole + 1 , [AlbumPointerRole](#) = Qt::UserRole + 2  
, [AlbumIdRole](#) = Qt::UserRole + 3 ,  
[AlbumGlobalIdRole](#) = Qt::UserRole + 4 , [AlbumSortRole](#) = Qt::UserRole + 5 }
- enum [RootAlbumBehavior](#) { [IncludeRootAlbum](#) , [IgnoreRootAlbum](#) }  
*AbstractAlbumModel is the abstract base class for all models that present Album objects as managed by AlbumManager.*

## Signals

- void [rootAlbumAvailable](#) ()  
*This is initialized once after creation, if the root album becomes available, if it was not already available at time of construction.*

## Public Member Functions

- [AbstractAlbumModel](#) ([Album::Type](#) albumType, [Album](#) \*const rootAlbum, [RootAlbumBehavior](#) rootBehavior=[IncludeRootAlbum](#), [QObject](#) \*const parent=nullptr)  
*Create an AbstractAlbumModel object for albums with the given type.*
- [Album](#) \* [albumForIndex](#) (const [QModelIndex](#) &index) const  
*Returns the album object associated with the given model index.*
- [Album::Type](#) [albumType](#) () const  
*Returns the Album::Type of the contained albums.*
- int [columnCount](#) (const [QModelIndex](#) &parent=[QModelIndex](#)()) const override
- [QVariant](#) [data](#) (const [QModelIndex](#) &index, int role=[Qt::DisplayRole](#)) const override
- [AlbumModelDragDropHandler](#) \* [dragDropHandler](#) () const  
*Returns the drag drop handler, or 0 if none is installed.*
- bool [dropMimeData](#) (const [QMimeData](#) \*data, [Qt::DropAction](#) action, int row, int column, const [QModelIndex](#) &parent) override
- [Qt::ItemFlags](#) [flags](#) (const [QModelIndex](#) &index) const override
- bool [hasChildren](#) (const [QModelIndex](#) &parent=[QModelIndex](#)()) const override
- [QVariant](#) [headerData](#) (int section, [Qt::Orientation](#) orientation, int role=[Qt::DisplayRole](#)) const override
- [QModelIndex](#) [index](#) (int row, int column, const [QModelIndex](#) &parent=[QModelIndex](#)()) const override
- [QModelIndex](#) [indexForAlbum](#) ([Album](#) \*album) const  
*Return the QModelIndex for the given album, or an invalid index if the album is not contained in this model.*
- bool [isFaceTagModel](#) () const  
*Returns true if the album model a face tag model.*
- [QMimeData](#) \* [mimeData](#) (const [QModelIndexList](#) &indexes) const override
- [QStringList](#) [mimeTypes](#) () const override
- [QModelIndex](#) [parent](#) (const [QModelIndex](#) &index) const override
- [Album](#) \* [rootAlbum](#) () const
- [RootAlbumBehavior](#) [rootAlbumBehavior](#) () const  
*Returns the root album behavior set for this model.*
- [QModelIndex](#) [rootAlbumIndex](#) () const  
*Return the index corresponding to the root album.*
- int [rowCount](#) (const [QModelIndex](#) &parent=[QModelIndex](#)()) const override
- void [setDragDropHandler](#) ([AlbumModelDragDropHandler](#) \*handler)  
*Set a drag drop handler.*
- void [setDropIndex](#) (const [QModelIndex](#) &index)  
*Set current index from QDragMoveEvent.*
- [Qt::DropActions](#) [supportedDropActions](#) () const override

## Static Public Member Functions

- static [Album](#) \* [retrieveAlbum](#) (const QModelIndex &index)  
*Returns the album represented by the index.*

## Protected Slots

- void [slotAlbumAboutToBeAdded](#) ([Album](#) \*album, [Album](#) \*parent, [Album](#) \*prev)
- void [slotAlbumAboutToBeDeleted](#) ([Album](#) \*album)
- void [slotAlbumAdded](#) ([Album](#) \*)
- void [slotAlbumHasBeenDeleted](#) ([Album](#) \*album)
- void [slotAlbumIconChanged](#) ([Album](#) \*album)
- void [slotAlbumRenamed](#) ([Album](#) \*album)
- void [slotAlbumsCleared](#) ()

## Protected Member Functions

- virtual void [albumCleared](#) ([Album](#) \*)  
*Notification when an entry is removed.*
- virtual QVariant [albumData](#) ([Album](#) \*a, int role) const  
*For subclassing convenience: A part of the implementation of data()*
- virtual void [allAlbumsCleared](#) ()  
*Notification when all entries are removed.*
- virtual QString [columnHeader](#) () const  
*For subclassing convenience: A part of the implementation of headerData()*
- virtual QVariant [decorationRoleData](#) ([Album](#) \*a) const  
*For subclassing convenience: A part of the implementation of data()*
- virtual bool [filterAlbum](#) ([Album](#) \*album) const  
*Returns true for those and only those albums that shall be contained in this model.*
- virtual QVariant [fontRoleData](#) ([Album](#) \*a) const  
*For subclassing convenience: A part of the implementation of data()*
- virtual Qt::ItemFlags [itemFlags](#) ([Album](#) \*album) const  
*For subclassing convenience: A part of the implementation of itemFlags()*
- void [setEnableDrag](#) (bool enable)  
*Switch on drag and drop globally for all items.*
- void [setEnableDrop](#) (bool enable)
- void [setFaceTagModel](#) (bool enable)
- virtual QVariant [sortRoleData](#) ([Album](#) \*a) const  
*For subclassing convenience: A part of the implementation of data()*

## 9.2.1 Member Enumeration Documentation

### 9.2.1.1 AlbumDataRole

enum [Digikam::AbstractAlbumModel::AlbumDataRole](#)

#### Enumerator

<a href="#">AlbumTitleRole</a>	Returns the album title. Principally the same as display role, but without any additions.
<a href="#">AlbumTypeRole</a>	Returns the <a href="#">Album::Type</a> of the associated album.
<a href="#">AlbumPointerRole</a>	Returns a pointer to the associated <a href="#">Album</a> object.
<a href="#">AlbumIdRole</a>	Returns the id of the associated <a href="#">Album</a> object.
<a href="#">AlbumGlobalIdRole</a>	Returns the global id (unique across all album types)
<a href="#">AlbumSortRole</a>	Returns the data to sort on.



### 9.2.1.2 RootAlbumBehavior

```
enum Digikam::AbstractAlbumModel::RootAlbumBehavior
```

You will want to create an instance of the base classes.

Enumerator

IncludeRootAlbum	The root album will be included as a single parent item with all top-level album as children.
IgnoreRootAlbum	The root album will not be included, but all top-level album are represented as top-level items in this view.

## 9.2.2 Constructor & Destructor Documentation

### 9.2.2.1 AbstractAlbumModel()

```
Digikam::AbstractAlbumModel::AbstractAlbumModel (
    Album::Type albumType,
    Album *const rootAlbum,
    RootAlbumBehavior rootBehavior = IncludeRootAlbum,
    QObject *const parent = nullptr ) [explicit]
```

Pass the root album if it is already available. Do not use this class directly, but one of the subclasses.

## 9.2.3 Member Function Documentation

### 9.2.3.1 albumCleared()

```
virtual void Digikam::AbstractAlbumModel::albumCleared (
    Album * ) [inline], [protected], [virtual]
```

Reimplemented in [Digikam::AbstractCountingAlbumModel](#), and [Digikam::AbstractCheckableAlbumModel](#).

### 9.2.3.2 albumData()

```
QVariant Digikam::AbstractAlbumModel::albumData (
    Album * a,
    int role ) const [protected], [virtual]
```

Note

these can be reimplemented in a subclass

Reimplemented in [Digikam::AbstractCountingAlbumModel](#), [Digikam::AbstractCheckableAlbumModel](#), [Digikam::AlbumModel](#), [Digikam::TagModel](#), and [Digikam::SearchModel](#).

### 9.2.3.3 allAlbumsCleared()

```
virtual void Digikam::AbstractAlbumModel::allAlbumsCleared ( ) [inline], [protected], [virtual]
```

Reimplemented in [Digikam::AbstractCountingAlbumModel](#), and [Digikam::AbstractCheckableAlbumModel](#).

### 9.2.3.4 columnHeader()

```
QString Digikam::AbstractAlbumModel::columnHeader ( ) const [protected], [virtual]
```

Reimplemented in [Digikam::AbstractSpecificAlbumModel](#).

### 9.2.3.5 decorationRoleData()

```
QVariant Digikam::AbstractAlbumModel::decorationRoleData (
    Album * a ) const [protected], [virtual]
```

Reimplemented in [Digikam::AlbumModel](#), [Digikam::TagModel](#), and [Digikam::DateAlbumModel](#).

### 9.2.3.6 filterAlbum()

```
bool Digikam::AbstractAlbumModel::filterAlbum (
    Album * album ) const [protected], [virtual]
```

They must have a common root album, which is set in the constructor.

### 9.2.3.7 fontRoleData()

```
QVariant Digikam::AbstractAlbumModel::fontRoleData (
    Album * a ) const [protected], [virtual]
```

Reimplemented in [Digikam::TagModel](#).

### 9.2.3.8 retrieveAlbum()

```
Album * Digikam::AbstractAlbumModel::retrieveAlbum (
    const QModelIndex & index ) [static]
```

In contrast to [albumForIndex\(\)](#), the index can be from any proxy model, as long as an [AbstractAlbumModel](#) is at the end.

### 9.2.3.9 rootAlbumAvailable

```
void Digikam::AbstractAlbumModel::rootAlbumAvailable ( ) [signal]
```

This is emitted regardless of root album policy.

#### 9.2.3.10 rootAlbumIndex()

```
QModelIndex Digikam::AbstractAlbumModel::rootAlbumIndex ( ) const
```

If the policy is IgnoreRootAlbum, this is an invalid index.

#### 9.2.3.11 setEnableDrag()

```
void Digikam::AbstractAlbumModel::setEnableDrag (
    bool enable ) [protected]
```

Default is true. For per-item cases reimplement [itemFlags\(\)](#).

#### 9.2.3.12 sortRoleData()

```
QVariant Digikam::AbstractAlbumModel::sortRoleData (
    Album * a ) const [protected], [virtual]
```

Reimplemented in [Digikam::DateAlbumModel](#).

## 9.3 Digikam::AbstractAlbumTreeView Class Reference

Base class for all tree views that display Album-based content provided by an [AbstractSpecificAlbumModel](#).



## Public Types

- enum [Flag](#) { [CreateDefaultModel](#) , [CreateDefaultFilterModel](#) , [CreateDefaultDelegate](#) , [ShowCountAccordingToSettings](#) , [AlwaysShowInclusiveCounts](#) , **DefaultFlags** = [CreateDefaultFilterModel](#) | [CreateDefaultDelegate](#) | [ShowCountAccordingToSettings](#) }
- typedef QFlags< [Flag](#) > **Flags**

## Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }  
*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

## Public Slots

- void **adaptColumnsToContent** ()  
*Adapt the column sizes to the contents of the tree view.*
- void [expandEverything](#) (const QModelIndex &index)  
*Expands the complete tree under the given index.*
- void **scrollToSelectedAlbum** ()  
*Scrolls to the first selected album if there is one.*
- void [setCurrentAlbums](#) (const QList< Album \* > &albums, bool selectInAlbumManager=true)  
*Selects the given album.*
- void **setSearchTextSettings** (const [SearchTextSettings](#) &settings)
- void **slotCollapseAllNodes** ()  
*slotCollapseAllNodes - collapse all nodes without root node*
- void **slotCollapseNode** ()  
*slotCollapseNode - collapse recursively selected nodes*
- void **slotExpandNode** ()  
*slotExpandNode - expands recursively selected nodes*

## Signals

- void **currentAlbumChanged** (Album \*currentAlbum)  
*Emitted when the currently selected album changes.*
- void [selectedAlbumsChanged](#) (const QList< Album \* > &selectedAlbums)  
*Emitted when the current selection changes.*

## Public Member Functions

- [AbstractAlbumTreeView](#) (QWidget \*const parent, Flags flags)  
*Constructs an album tree view.*
- void **addContextMenuElement** ([ContextMenuElement](#) \*const element)
- [AlbumFilterModel](#) \* **albumFilterModel** () const
- [AbstractSpecificAlbumModel](#) \* **albumModel** () const
- QList< [ContextMenuElement](#) \* > **contextMenuElements** () const
- template<class A >  
QList< A \* > **currentAlbums** ()
- void [doLoadState](#) () override

Implements state loading for the album tree view in a somewhat clumsy procedure because the model may not be fully loaded when this method is called.

- void **doSaveState** () override  
*Implement this hook method for state saving.*
- bool **expandMatches** (const QModelIndex &index)  
*Ensures that every current match is visible by expanding all parent entries.*
- QModelIndex **indexVisuallyAt** (const QPoint &p)  
*This is a combination of indexAt() checked with visualRect().*
- void **removeContextMenuElement** (ContextMenuElement \*const element)
- QList< Album \* > **selectedItems** ()  
*selectedItems() -*
- void **setAlbumManagerCurrentAlbum** (const bool setCurrentAlbum)  
*Some treeviews shall control the global current album kept by AlbumManager.*
- void **setContextMenuIcon** (const QPixmap &pixmap)  
*Set the context menu title and icon.*
- void **setContextMenuTitle** (const QString &title)
- void **setEnabledContextMenu** (const bool enable)  
*Determines the global decision to show a popup menu or not.*
- void **setExpandNewCurrentItem** (const bool doThat)  
*Expand an item when making it the new current item.*
- void **setExpandOnSingleClick** (const bool doThat)  
*Enable expanding of tree items on single click on the item (default: off)*
- void **setSelectAlbumOnClick** (const bool selectOnClick)  
*Sets whether to select an album on click via the album manager or not.*
- void **setSelectOnContextMenu** (const bool select)  
*Sets whether to select the album under the mouse cursor on a context menu request (so that the album is shown using the album manager) or not.*
- bool **viewportEvent** (QEvent \*event) override  
*For internal use only.*

## Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (QObject \*const host)  
*Constructor.*
- virtual ~[StateSavingObject](#) ()  
*Destructor.*
- [StateSavingDepth](#) **getStateSavingDepth** () const  
*Returns the depth used for state saving or loading.*
- void **loadState** ()  
*Invokes loading the class' state.*
- void **saveState** ()  
*Invokes saving the class' state.*
- virtual void **setConfigGroup** (const KConfigGroup &group)  
*Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void **setEntryPrefix** (const QString &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void **setStateSavingDepth** (const [StateSavingDepth](#) depth)  
*Sets the depth used for state saving or loading.*

### Protected Slots

- void **albumSettingsChanged** ()
- void **slotCurrentChanged** ()
- virtual void **slotRootAlbumAvailable** ()  
*override if implemented behavior is not as intended*
- void **slotSearchTextSettingsAboutToChange** (bool searched, bool willSearch)
- void **slotSearchTextSettingsChanged** (bool wasSearching, bool searching)
- void **slotSelectionChanged** ()

### Protected Member Functions

- virtual void **addCustomContextMenuActions** (ContextMenuHelper &cmh, Album \*album)  
*Hook method to add custom actions to the generated context menu.*
- virtual QPixmap **contextMenuIcon** () const  
*Hook method that can be implemented to return a special icon used for the context menu.*
- virtual QString **contextMenuTitle** () const  
*Hook method to implement that returns the title for the context menu.*
- void **dragEnterEvent** (QDragEnterEvent \*e) override
- void **dragLeaveEvent** (QDragLeaveEvent \*e) override
- void **dragMoveEvent** (QDragMoveEvent \*e) override
- void **dropEvent** (QDropEvent \*e) override
- virtual void **handleCustomContextMenuAction** (QAction \*action, const AlbumPointer< Album > &album)  
*Hook method to handle the custom context menu actions that were added with addCustomContextMenuActions.*
- virtual void **middleButtonPressed** (Album \*a)
- void **mousePressEvent** (QMouseEvent \*e) override  
*Other helper methods.*
- virtual QPixmap  **pixmapForDrag** (const QStyleOptionViewItem &option, QList< QModelIndex > indexes)  
*TODO: Move to delegate, when we have one.*
- void **rowsAboutToBeRemoved** (const QModelIndex &parent, int start, int end) override
- void **rowsInserted** (const QModelIndex &index, int start, int end) override
- void **setAlbumFilterModel** (AlbumFilterModel \*const filterModel)
- void **setAlbumModel** (AbstractSpecificAlbumModel \*const model)
- virtual bool **showContextMenuAt** (QContextMenuEvent \*event, Album \*albumForEvent)  
*Hook method to implement that determines if a context menu shall be displayed for the given event at the position coded in the event.*
- void **startDrag** (Qt::DropActions supportedActions) override

### Protected Member Functions inherited from Digikam::StateSavingObject

- QString **entryName** (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup **getConfigGroup** () const  
*Returns the config group that must be used for state saving and loading.*

### Protected Attributes

- AlbumFilterModel \* **m\_albumFilterModel** = nullptr
- AbstractSpecificAlbumModel \* **m\_albumModel** = nullptr
- bool **m\_checkOnMiddleClick** = false
- AlbumModelDragDropHandler \* **m\_dragDropHandler** = nullptr
- Flags **m\_flags** = DefaultFlags
- int **m\_lastScrollBarValue** = 0
- bool **m\_restoreCheckState** = false

### 9.3.1 Detailed Description

This class enables various utility functions like selecting albums on mouse actions or providing an infrastructure for displaying a context menu for albums.

Context menu handling is implemented as template methods with hook methods that can be implemented by subclasses to provide a custom behavior. In default mode no context menu is shown at all. It must be enabled via a call to `setEnabledContextMenu`.

### 9.3.2 Member Enumeration Documentation

#### 9.3.2.1 Flag

```
enum Digikam::AbstractAlbumTreeView::Flag
```

Enumerator

CreateDefaultModel	Create a default model. Not supported by abstract classes. Not part of default flags!
CreateDefaultFilterModel	Create a default filter model.
CreateDefaultDelegate	Create a delegate which paints according to settings. If not set, the Qt default delegate of the view is used.
ShowCountAccordingToSettings	Show the count according to the settings. If not set, call <code>setShowCount()</code> on the model yourself.
AlwaysShowInclusiveCounts	Always show the inclusive counts. Not part of default flags!

### 9.3.3 Constructor & Destructor Documentation

#### 9.3.3.1 AbstractAlbumTreeView()

```
Digikam::AbstractAlbumTreeView::AbstractAlbumTreeView (
    QWidget *const parent,
    Flags flags )
```

If you give 0 for model, call `setAlbumModel` afterwards. If you supply 0 for filterModel, call `setAlbumFilterModel` afterwards.

### 9.3.4 Member Function Documentation

#### 9.3.4.1 addCustomContextMenuActions()

```
void Digikam::AbstractAlbumTreeView::addCustomContextMenuActions (
    ContextMenuHelper & cmh,
    Album * album ) [protected], [virtual]
```

Parameters

<i>cmh</i>	helper object to create the context menu
<i>album</i>	tag on which the context menu will be created. May be null if it is requested on no tag entry



Reimplemented in [Digikam::TagFilterView](#), [Digikam::AlbumSelectTreeView](#), [Digikam::TagCheckView](#), [Digikam::TagFolderView](#), [Digikam::EditableSearchTreeView](#), and [Digikam::NormalSearchTreeView](#).

#### 9.3.4.2 contextMenuIcon()

```
QPixmap Digikam::AbstractAlbumTreeView::contextMenuIcon ( ) const [protected], [virtual]
```

##### Returns

the icon for the context menu

#### 9.3.4.3 contextMenuTitle()

```
QString Digikam::AbstractAlbumTreeView::contextMenuTitle ( ) const [protected], [virtual]
```

##### Returns

title for the context menu

Reimplemented in [Digikam::TagFolderView](#), and [Digikam::EditableSearchTreeView](#).

#### 9.3.4.4 doLoadState()

```
void Digikam::AbstractAlbumTreeView::doLoadState ( ) [override], [virtual]
```

Therefore the config is first parsed into `d->statesByAlbumId` which holds the state of all tree view entries indexed by album id. Afterwards an initial sync run is done restoring the state of all model entries that are already present at this time. Every processed entry is removed from `d->statesByAlbumId`. If there are still entries left in this map we assume that the model is not fully loaded at the moment. Therefore the `rowsInserted` signal is connected to a slot that restores the state of new rows based on the remaining entries in `d->statesByAlbumId`.

Implements [Digikam::StateSavingObject](#).

Reimplemented in [Digikam::AbstractCheckableAlbumTreeView](#), and [Digikam::TagCheckView](#).

#### 9.3.4.5 doSaveState()

```
void Digikam::AbstractAlbumTreeView::doSaveState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

Reimplemented in [Digikam::AbstractCheckableAlbumTreeView](#), and [Digikam::TagCheckView](#).

#### 9.3.4.6 expandEverything

```
void Digikam::AbstractAlbumTreeView::expandEverything (
    const QModelIndex & index ) [slot]
```

## Parameters

<i>index</i>	the index to start expanding everything
--------------	---

**9.3.4.7 expandMatches()**

```
bool Digikam::AbstractAlbumTreeView::expandMatches (
    const QModelIndex & index )
```

## Parameters

<i>index</i>	the index to start ensuring expansion state
--------------	---

## Returns

`true` if there was a match under `index`. This return value can normally be ignored by the caller because it is only used for an internal recursion.

**9.3.4.8 handleCustomContextMenuAction()**

```
void Digikam::AbstractAlbumTreeView::handleCustomContextMenuAction (
    QAction * action,
    const AlbumPointer< Album > & album ) [protected], [virtual]
```

## Parameters

<i>action</i>	the action that was chosen by the user, may be null if none of the custom actions were selected
<i>album</i>	the tag on which the context menu was requested. May be null if there was no

Reimplemented in [Digikam::TagFilterView](#), [Digikam::AlbumSelectTreeView](#), [Digikam::TagFolderView](#), [Digikam::EditableSearchTreeView](#) and [Digikam::NormalSearchTreeView](#).

**9.3.4.9 indexVisuallyAt()**

```
QModelIndex Digikam::AbstractAlbumTreeView::indexVisuallyAt (
    const QPoint & p )
```

`p` must be in the viewport currently. Decoration will not be included. Suitable for mouse click positions.

**9.3.4.10 pixmapForDrag()**

```
QPixmap Digikam::AbstractAlbumTreeView::pixmapForDrag (
    const QStyleOptionViewItem & option,
    QList< QModelIndex > indexes ) [protected], [virtual]
```

Copy code from image delegate for creating icons when dragging multiple items

#### 9.3.4.11 selectedAlbumsChanged

```
void Digikam::AbstractAlbumTreeView::selectedAlbumsChanged (
    const QList< Album * > & selectedAlbums ) [signal]
```

Use currentChanged unless in multi-selection mode.

#### 9.3.4.12 setAlbumManagerCurrentAlbum()

```
void Digikam::AbstractAlbumTreeView::setAlbumManagerCurrentAlbum (
    const bool setCurrentAlbum )
```

Other treeview are self-contained and shall not change the current album. Default: false

#### 9.3.4.13 setContextMenuIcon()

```
void Digikam::AbstractAlbumTreeView::setContextMenuIcon (
    const QPixmap & pixmap )
```

This is used by the default implementation of [contextMenuIcon\(\)](#) and [contextMenuTitle\(\)](#). You can alternatively reimplement these methods.

#### 9.3.4.14 setCurrentAlbums

```
void Digikam::AbstractAlbumTreeView::setCurrentAlbums (
    const QList< Album * > & albums,
    bool selectInAlbumManager = true ) [slot]
```

##### Parameters

<i>albums</i>	the albums to select
<i>selectInAlbumManager</i>	the album will be set as current album, if both this parameter is true and <a href="#">setAlbumManagerCurrentAlbum()</a> was set to true.

#### 9.3.4.15 setEnableContextMenu()

```
void Digikam::AbstractAlbumTreeView::setEnableContextMenu (
    const bool enable )
```

More detailed decision at which position a menu can be shown and where not can be made by implementing [showContextMenuAt](#).

##### Parameters

<i>enable</i>	if true, a context menu can be shown
---------------	--------------------------------------

### 9.3.4.16 setSelectAlbumOnClick()

```
void Digikam::AbstractAlbumTreeView::setSelectAlbumOnClick (
    const bool selectOnClick )
```

#### Parameters

<i>selectOnClick</i>	if true, a click on an item automatically sets this item as the current album in the album manager
----------------------	--

### 9.3.4.17 setSelectOnContextMenu()

```
void Digikam::AbstractAlbumTreeView::setSelectOnContextMenu (
    const bool select )
```

Defaults to true.

#### Parameters

<i>select</i>	true if a context menu request shall select the album
---------------	---

### 9.3.4.18 showContextMenuAt()

```
bool Digikam::AbstractAlbumTreeView::showContextMenuAt (
    QContextMenuEvent * event,
    Album * albumForEvent ) [protected], [virtual]
```

#### Parameters

<i>event</i>	context menu event to react on
<i>albumForEvent</i>	the album at the mouse position or null if there is no album at that position

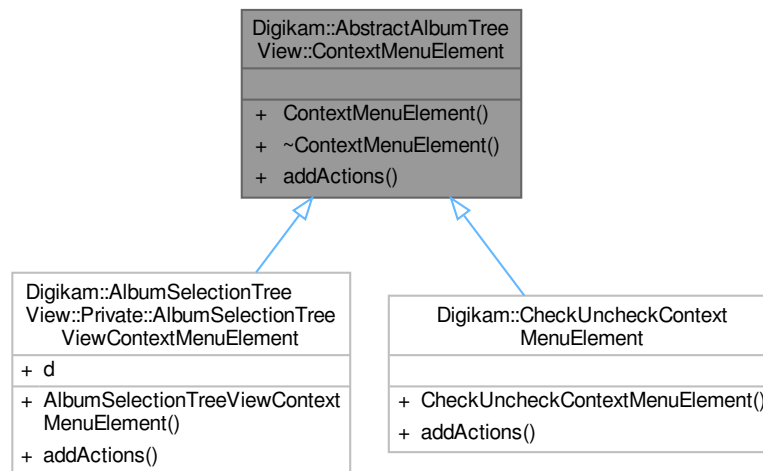
#### Returns

true if a context menu shall be displayed at the event coordinates, else false

## 9.4 Digikam::AbstractAlbumTreeView::ContextMenuElement Class Reference

Add a context menu element which can add actions to the context menu when the menu is generated.

Inheritance diagram for Digikam::AbstractAlbumTreeView::ContextMenuElement:



## Public Member Functions

- virtual void `addActions` (`AbstractAlbumTreeView *view`, `ContextMenuHelper &cmh`, `Album *album`)=0  
*Add actions to the context menu being generated.*

### 9.4.1 Detailed Description

First, `addCustomContextMenuActions` is called, then all elements' `addActions` method is called in order of addition.

### 9.4.2 Member Function Documentation

#### 9.4.2.1 addActions()

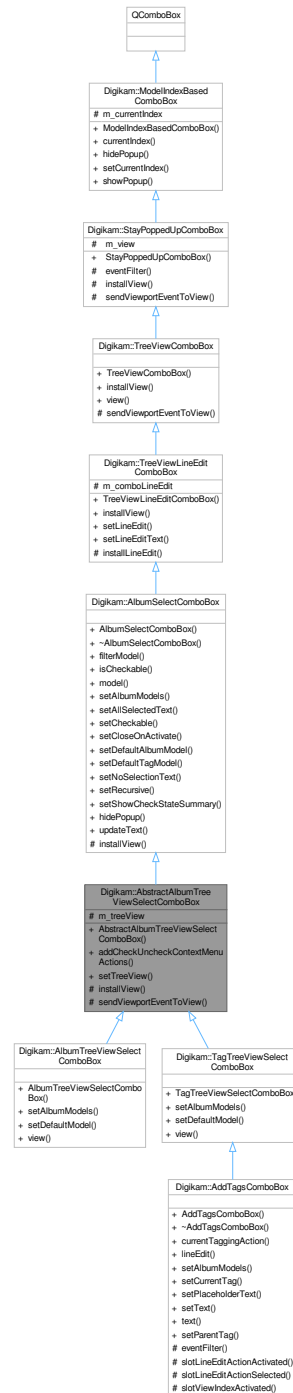
```
virtual void Digikam::AbstractAlbumTreeView::ContextMenuElement::addActions (
    AbstractAlbumTreeView * view,
    ContextMenuHelper & cmh,
    Album * album ) [pure virtual]
```

#### Parameters

<code>view</code>	The <code>AbstractAlbumTreeView</code> which generates the menu
<code>cmh</code>	helper object to create the context menu
<code>album</code>	the album on which the context menu will be created. May be null if it is requested on no tag entry

## 9.5 Digikam::AbstractAlbumTreeViewSelectComboBox Class Reference

Inheritance diagram for Digikam::AbstractAlbumTreeViewSelectComboBox:



### Public Member Functions

- [AbstractAlbumTreeViewSelectComboBox](#) (QWidget \*const parent=nullptr)

*Abstract class.*

- void [addCheckUncheckContextMenuActions](#) ()  
*Enables a context menu which contains options to check or uncheck groups of albums, given you have a checkable model.*
- void [setTreeView](#) ([AbstractAlbumTreeView](#) \*const treeView)  
*Set a tree view created by you instead of creating a default view (in the subclasses).*

### Public Member Functions inherited from [Digikam::AlbumSelectComboBox](#)

- [AlbumSelectComboBox](#) (QWidget \*const parent=nullptr)
- QSortFilterProxyModel \* [filterModel](#) () const  
*Return the filter model in use.*
- bool [isCheckedable](#) () const
- [AbstractCheckableAlbumModel](#) \* [model](#) () const  
*Returns the source model.*
- void [setAlbumModels](#) ([AbstractCheckableAlbumModel](#) \*model, [AlbumFilterModel](#) \*filterModel=nullptr)
- void [setAllSelectedText](#) (bool all)  
*Enable or disable the text used to describe the status when all album is selected.*
- void [setCheckable](#) (bool checkable)  
*Enable checkboxes next to the items.*
- void [setCloseOnActivate](#) (bool close)  
*Enable closing when an item was activated (clicked).*
- void [setDefaultAlbumModel](#) ()  
*Once after creation, call one of these three methods.*
- void [setDefaultTagModel](#) ()
- void [setNoSelectionText](#) (const QString &text)  
*Sets the text that is used to describe the state when no album is selected.*
- void [setRecursive](#) (bool recursive)  
*If all subalbums shall be selected when parent will be selected.*
- void [setShowCheckStateSummary](#) (bool show)  
*If the box is checkable, enable showing a resume a la "3 Albums checked" in the combo box text.*

### Public Member Functions inherited from [Digikam::TreeViewLineEditComboBox](#)

- [TreeViewLineEditComboBox](#) (QWidget \*const parent=nullptr)  
*This class provides a [TreeViewComboBox](#) with a read-only line edit.*
- void [installView](#) (QAbstractItemView \*view=nullptr) override  
*Replace the standard combo box list view with a QTreeView.*
- void [setLineEdit](#) (QLineEdit \*edit)
- void [setLineEditText](#) (const QString &text)  
*Set the text of the line edit (the text that is visible if the popup is not opened).*

### Public Member Functions inherited from [Digikam::TreeViewComboBox](#)

- [TreeViewComboBox](#) (QWidget \*parent=nullptr)  
*This class provides a [QComboBox](#) with a [QTreeView](#) instead of the usual [QListView](#).*
- QTreeView \* [view](#) () const  
*Returns the QTreeView of this class.*

## Public Member Functions inherited from [Digikam::StayPoppedUpComboBox](#)

- [StayPoppedUpComboBox](#) (QWidget \*const parent=nullptr)

*This class provides an abstract QComboBox with a custom view (which is created by implementing subclasses) instead of the usual QListView.*

## Public Member Functions inherited from [Digikam::ModelIndexBasedComboBox](#)

- [ModelIndexBasedComboBox](#) (QWidget \*const parent=nullptr)  
*QComboBox has a current index based on a single integer.*
- QModelIndex **currentIndex** () const
- void **hidePopup** () override
- void **setCurrentIndex** (const QModelIndex &index)
- void **showPopup** () override

## Protected Member Functions

- void [installView](#) (QAbstractItemView \*view=nullptr) override  
*Replace the standard combo box list view with a QTreeView.*
- void [sendViewportEventToView](#) (QEvent \*e) override  
*Implement in subclass: Send the given event to the viewportEvent() method of m\_view.*

## Protected Member Functions inherited from [Digikam::AlbumSelectComboBox](#)

- void [installView](#) (QAbstractItemView \*view=nullptr) override  
*Replace the standard combo box list view with a QTreeView.*

## Protected Member Functions inherited from [Digikam::TreeViewLineEditComboBox](#)

- virtual void [installLineEdit](#) ()  
*Sets a line edit.*

## Protected Member Functions inherited from [Digikam::TreeViewComboBox](#)

- void [sendViewportEventToView](#) (QEvent \*e) override  
*Implement in subclass: Send the given event to the viewportEvent() method of m\_view.*

## Protected Member Functions inherited from [Digikam::StayPoppedUpComboBox](#)

- bool **eventFilter** (QObject \*watched, QEvent \*event) override
- void [installView](#) (QAbstractItemView \*view)  
*Replace the standard combo box list view with the given view.*

## Protected Attributes

- [AbstractAlbumTreeView](#) \* **m\_treeView** = nullptr



**Protected Attributes inherited from [Digikam::TreeViewLineEditComboBox](#)**

- `QLineEdit * m_comboLineEdit = nullptr`

**Protected Attributes inherited from [Digikam::StayPoppedUpComboBox](#)**

- `QAbstractItemView * m_view = nullptr`

**Protected Attributes inherited from [Digikam::ModelIndexBasedComboBox](#)**

- `QPersistentModelIndex m_currentIndex`

**Additional Inherited Members****Public Slots inherited from [Digikam::AlbumSelectComboBox](#)**

- `void hidePopup ()` override
- `virtual void updateText ()`  
*Updates the text describing the selection ("3 Albums selected").*

**9.5.1 Constructor & Destructor Documentation****9.5.1.1 AbstractAlbumTreeViewSelectComboBox()**

```
Digikam::AbstractAlbumTreeViewSelectComboBox::AbstractAlbumTreeViewSelectComboBox (
    QWidget *const parent = nullptr ) [explicit]
```

This is an [AlbumSelectComboBox](#) which installs an [AlbumTreeView](#), not a plain `QTreeView`, as view.

**9.5.2 Member Function Documentation****9.5.2.1 addCheckUncheckContextMenuActions()**

```
void Digikam::AbstractAlbumTreeViewSelectComboBox::addCheckUncheckContextMenuActions ( )
```

Call this method after `setModel()`.

**9.5.2.2 installView()**

```
void Digikam::AbstractAlbumTreeViewSelectComboBox::installView (
    QAbstractItemView * view = nullptr ) [override], [protected], [virtual]
```

Call this after installing an appropriate model.

Reimplemented from [Digikam::TreeViewComboBox](#).

### 9.5.2.3 `sendViewportEventToView()`

```
void Digikam::AbstractAlbumTreeViewSelectComboBox::sendViewportEventToView (
    QEvent * e ) [override], [protected], [virtual]
```

This method is protected for a usual `QAbstractItemView`. You can override, pass a view, and call parent implementation. The existing view will be used. You must then also reimplement `sendViewportEventToView`.

Implements [Digikam::StayPoppedUpComboBox](#).

### 9.5.2.4 `setTreeView()`

```
void Digikam::AbstractAlbumTreeViewSelectComboBox::setTreeView (
    AbstractAlbumTreeView *const treeView )
```

Only takes effect before calling `setModel`.

## 9.6 Digikam::AbstractCheckableAlbumModel Class Reference

Inheritance diagram for Digikam::AbstractCheckableAlbumModel:



### Public Slots

- void **checkAllAlbums** (const QModelIndex &parent=QModelIndex())  
Checks all albums beneath the given parent.

- void **checkAllParentAlbums** (const QModelIndex &child)  
*Checks all parent albums starting at the child, including it.*
- void **invertCheckedAlbums** (const QModelIndex &parent=QModelIndex())  
*Inverts the checked state of all albums under the given parent.*
- void **resetAllCheckedAlbums** ()  
*Resets the checked state of all albums to Qt::Unchecked.*
- void **resetCheckedAlbums** (const QModelIndex &parent=QModelIndex())  
*Resets the checked state of all albums under the given parent.*
- void **resetCheckedParentAlbums** (const QModelIndex &child)  
*Resets the checked state of all parents of the child including it.*
- void **setChecked** (Album \*album, bool isChecked)  
*Sets the check state of album to Checked or Unchecked.*
- void **setCheckState** (Album \*album, Qt::CheckState state)  
*Sets the check state of the album.*
- void **setCheckStateForChildren** (Album \*album, Qt::CheckState state)  
*Sets the checked state recursively for all children of but not for the given album.*
- void **setCheckStateForParents** (Album \*album, Qt::CheckState state)  
*Sets the checked state recursively for all parents of but not for the given album.*
- void **toggleChecked** (Album \*album)  
*Toggles the check state of album between Checked or Unchecked.*

### Public Slots inherited from [Digikam::AbstractCountingAlbumModel](#)

- void **excludeChildrenCount** (const QModelIndex &index)  
*Displays only the count of the album, without adding child albums' counts.*
- void **includeChildrenCount** (const QModelIndex &index)  
*Displays sum of the count of the album and child albums' counts.*
- void **setCountHash** (const QHash< int, int > &idCountHash)  
*Enable displaying the count.*
- void **setShowCount** (bool show)  
*Call to enable or disable showing the count. Default is false.*

### Signals

- void **checkStateChanged** (Album \*album, Qt::CheckState checkState)  
*Emitted when the check state of an album changes.*

### Signals inherited from [Digikam::AbstractCountingAlbumModel](#)

- void **signalUpdateAlbumCount** (Album \*album)

### Signals inherited from [Digikam::AbstractAlbumModel](#)

- void **rootAlbumAvailable** ()  
*This is initialized once after creation, if the root album becomes available, if it was not already available at time of construction.*

**Public Member Functions**

- **AbstractCheckableAlbumModel** ([Album::Type](#) albumType, [Album](#) \*const rootAlbum, [RootAlbumBehavior](#) rootBehavior=[IncludeRootAlbum](#), [QObject](#) \*const parent=nullptr)
 

*Abstract base class that manages the check state of Albums.*
- [QList](#)< [Album](#) \* > **checkedAlbums** () const
 

*Returns a list of album with check state Checked.*
- [Qt::CheckState](#) **checkState** ([Album](#) \*album) const
 

*Returns the check state of the album.*
- bool **isAddExcludeTristate** () const
- bool **isCheckable** () const
- bool **isChecked** ([Album](#) \*album) const
 

*Returns if the given album has the check state Checked.*
- bool **isTristate** () const
- [QList](#)< [Album](#) \* > **partiallyCheckedAlbums** () const
 

*Returns a list of album with partially check state Checked.*
- bool **rootIsCheckable** () const
- void **setAddExcludeTristate** (bool b)
 

*Sets a special tristate mode, which offers the three modes "unchecked", "added" and "excluded", where "excluded" corresponds to partially checked internally, but is reflected in the treeview through the decoration only.*
- void **setCheckable** (bool isCheckable)
 

*Triggers if the albums in this model are checkable.*
- void **setRecursive** (bool recursive)
 

*If an item gets checked, all childs get checked as well, If an item gets unchecked, all childs get unchecked as well.*
- void **setRootCheckable** (bool rootIsCheckable)
 

*Triggers if the root album is checkable.*
- void **setTristate** (bool isTristate)
 

*Triggers if the albums in this model are tristate.*

**Public Member Functions inherited from [Digikam::AbstractCountingAlbumModel](#)**

- **AbstractCountingAlbumModel** ([Album::Type](#) albumType, [Album](#) \*const rootAlbum, [RootAlbumBehavior](#) rootBehavior=[IncludeRootAlbum](#), [QObject](#) \*const parent=nullptr)
 

*Supports displaying a count alongside the album name in DisplayRole.*
- virtual int **albumCount** ([Album](#) \*album) const
 

*Returns the number of included items for this album.*
- bool **showCount** () const

**Public Member Functions inherited from [Digikam::AbstractSpecificAlbumModel](#)**

- **AbstractSpecificAlbumModel** ([Album::Type](#) albumType, [Album](#) \*const rootAlbum, [RootAlbumBehavior](#) rootBehavior=[IncludeRootAlbum](#), [QObject](#) \*const parent=nullptr)
 

*Abstract base class, do not instantiate.*

## Public Member Functions inherited from Digikam::AbstractAlbumModel

- [AbstractAlbumModel](#) ([Album::Type](#) albumType, [Album](#) \*const rootAlbum, [RootAlbumBehavior](#) rootBehavior=[IncludeRootAlbum](#), [QObject](#) \*const parent=nullptr)
  - Create an [AbstractAlbumModel](#) object for albums with the given type.*
- [Album](#) \* **albumForIndex** (const [QModelIndex](#) &index) const
  - Returns the album object associated with the given model index.*
- [Album::Type](#) **albumType** () const
  - Returns the [Album::Type](#) of the contained albums.*
- int **columnCount** (const [QModelIndex](#) &parent=[QModelIndex](#)()) const override
- [QVariant](#) **data** (const [QModelIndex](#) &index, int role=[Qt::DisplayRole](#)) const override
- [AlbumModelDragDropHandler](#) \* **dragDropHandler** () const
  - Returns the drag drop handler, or 0 if none is installed.*
- bool **dropMimeData** (const [QMimeData](#) \*data, [Qt::DropAction](#) action, int row, int column, const [QModelIndex](#) &parent) override
- [Qt::ItemFlags](#) **flags** (const [QModelIndex](#) &index) const override
- bool **hasChildren** (const [QModelIndex](#) &parent=[QModelIndex](#)()) const override
- [QVariant](#) **headerData** (int section, [Qt::Orientation](#) orientation, int role=[Qt::DisplayRole](#)) const override
- [QModelIndex](#) **index** (int row, int column, const [QModelIndex](#) &parent=[QModelIndex](#)()) const override
- [QModelIndex](#) **indexForAlbum** ([Album](#) \*album) const
  - Return the [QModelIndex](#) for the given album, or an invalid index if the album is not contained in this model.*
- bool **isFaceTagModel** () const
  - Returns true if the album model a face tag model.*
- [QMimeData](#) \* **mimeData** (const [QModelIndexList](#) &indexes) const override
- [QStringList](#) **mimeTypes** () const override
- [QModelIndex](#) **parent** (const [QModelIndex](#) &index) const override
- [Album](#) \* **rootAlbum** () const
- [RootAlbumBehavior](#) **rootAlbumBehavior** () const
  - Returns the root album behavior set for this model.*
- [QModelIndex](#) **rootAlbumIndex** () const
  - Return the index corresponding to the root album.*
- int **rowCount** (const [QModelIndex](#) &parent=[QModelIndex](#)()) const override
- void **setDragDropHandler** ([AlbumModelDragDropHandler](#) \*handler)
  - Set a drag drop handler.*
- void **setDropIndex** (const [QModelIndex](#) &index)
  - Set current index from [QDragMoveEvent](#).*
- [Qt::DropActions](#) **supportedDropActions** () const override

## Protected Member Functions

- void **albumCleared** ([Album](#) \*album) override
  - Notification when an entry is removed.*
- [QVariant](#) **albumData** ([Album](#) \*a, int role) const override
  - For subclassing convenience: A part of the implementation of data()*
- void **allAlbumsCleared** () override
  - Notification when all entries are removed.*
- [Qt::ItemFlags](#) **flags** (const [QModelIndex](#) &index) const override
- void **prepareAddExcludeDecoration** ([Album](#) \*a, [QPixmap](#) &icon) const
  - If in [AddExcludeTristate](#) mode, changes the icon as to indicate the state.*
- bool **setData** (const [QModelIndex](#) &index, const [QVariant](#) &value, int role, bool recursive)
- bool **setData** (const [QModelIndex](#) &index, const [QVariant](#) &value, int role=[Qt::EditRole](#)) override

### Protected Member Functions inherited from Digikam::AbstractCountingAlbumModel

- void `albumCleared` (`Album *album`) override  
*Notification when an entry is removed.*
- QVariant `albumData` (`Album *a`, int role) const override  
*Reimplemented from parent classes.*
- virtual `Album * albumForId` (int id) const =0  
*need to implement in subclass*
- virtual QString `albumName` (`Album *a`) const  
*Can reimplement in subclass.*
- void `allAlbumsCleared` () override  
*Notification when all entries are removed.*
- void `setCount` (`Album *album`, int count)  
*If you do not use `setCountHash`, `excludeChildrenCount` and `includeChildrenCount`, you can set a count here.*
- void `setup` ()  
*Call this method in children class constructors to init signal/slots connections.*

### Protected Member Functions inherited from Digikam::AbstractSpecificAlbumModel

- QString `columnHeader` () const override  
*For subclassing convenience: A part of the implementation of `headerData()`*
- void `emitDataChangedForChildren` (`Album *album`)
- void `setColumnHeader` (const QString &header)
- void `setupThumbnailLoading` ()  
*You need to call this from your constructor if you intend to load the thumbnail facilities of this class.*

### Protected Member Functions inherited from Digikam::AbstractAlbumModel

- virtual QVariant `decorationRoleData` (`Album *a`) const  
*For subclassing convenience: A part of the implementation of `data()`*
- virtual bool `filterAlbum` (`Album *album`) const  
*Returns true for those and only those albums that shall be contained in this model.*
- virtual QVariant `fontRoleData` (`Album *a`) const  
*For subclassing convenience: A part of the implementation of `data()`*
- virtual Qt::ItemFlags `itemFlags` (`Album *album`) const  
*For subclassing convenience: A part of the implementation of `itemFlags()`*
- void `setEnableDrag` (bool enable)  
*Switch on drag and drop globally for all items.*
- void `setEnableDrop` (bool enable)
- void `setFaceTagModel` (bool enable)
- virtual QVariant `sortRoleData` (`Album *a`) const  
*For subclassing convenience: A part of the implementation of `data()`*

### Additional Inherited Members

### Public Types inherited from Digikam::AbstractAlbumModel

- enum `AlbumDataRole` {  
`AlbumTitleRole` = Qt::UserRole , `AlbumTypeRole` = Qt::UserRole + 1 , `AlbumPointerRole` = Qt::UserRole + 2  
, `AlbumIdRole` = Qt::UserRole + 3 ,  
`AlbumGlobalIdRole` = Qt::UserRole + 4 , `AlbumSortRole` = Qt::UserRole + 5 }
- enum `RootAlbumBehavior` { `IncludeRootAlbum` , `IgnoreRootAlbum` }  
*`AbstractAlbumModel` is the abstract base class for all models that present `Album` objects as managed by `AlbumManager`.*

## Static Public Member Functions inherited from [Digikam::AbstractAlbumModel](#)

- static [Album](#) \* [retrieveAlbum](#) (const [QModelIndex](#) &index)  
*Returns the album represented by the index.*

## Protected Slots inherited from [Digikam::AbstractCountingAlbumModel](#)

- void [slotAlbumMoved](#) ([Album](#) \*album)

## Protected Slots inherited from [Digikam::AbstractSpecificAlbumModel](#)

- void [slotGotThumbnailFromIcon](#) ([Album](#) \*album, const [QPixmap](#) &thumbnail)
- void [slotReloadThumbnails](#) ()
- void [slotThumbnailLost](#) ([Album](#) \*album)

## Protected Slots inherited from [Digikam::AbstractAlbumModel](#)

- void [slotAlbumAboutToBeAdded](#) ([Album](#) \*album, [Album](#) \*parent, [Album](#) \*prev)
- void [slotAlbumAboutToBeDeleted](#) ([Album](#) \*album)
- void [slotAlbumAdded](#) ([Album](#) \*)
- void [slotAlbumHasBeenDeleted](#) ([Album](#) \*album)
- void [slotAlbumIconChanged](#) ([Album](#) \*album)
- void [slotAlbumRenamed](#) ([Album](#) \*album)
- void [slotAlbumsCleared](#) ()

## Protected Attributes inherited from [Digikam::AbstractSpecificAlbumModel](#)

- [QString](#) [m\\_columnHeader](#)

## 9.6.1 Constructor & Destructor Documentation

### 9.6.1.1 [AbstractCheckableAlbumModel\(\)](#)

```
Digikam::AbstractCheckableAlbumModel::AbstractCheckableAlbumModel (
    Album::Type albumType,
    Album *const rootAlbum,
    RootAlbumBehavior rootBehavior = IncludeRootAlbum,
    QObject *const parent = nullptr ) [explicit]
```

Call [setCheckable\(true\)](#) to enable checkable albums.

## 9.6.2 Member Function Documentation

### 9.6.2.1 [albumCleared\(\)](#)

```
void Digikam::AbstractCheckableAlbumModel::albumCleared (
    Album * ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::AbstractAlbumModel](#).



### 9.6.2.2 albumData()

```
QVariant Digikam::AbstractCheckableAlbumModel::albumData (
    Album * a,
    int role ) const [override], [protected], [virtual]
```

#### Note

these can be reimplemented in a subclass

Reimplemented from [Digikam::AbstractAlbumModel](#).

Reimplemented in [Digikam::AlbumModel](#), [Digikam::TagModel](#), and [Digikam::searchModel](#).

### 9.6.2.3 allAlbumsCleared()

```
void Digikam::AbstractCheckableAlbumModel::allAlbumsCleared ( ) [override], [protected],
[virtual]
```

Reimplemented from [Digikam::AbstractAlbumModel](#).

### 9.6.2.4 checkStateChanged

```
void Digikam::AbstractCheckableAlbumModel::checkStateChanged (
    Album * album,
    Qt::CheckState checkState ) [signal]
```

checkState contains the new Qt::CheckState of album

### 9.6.2.5 setData()

```
bool Digikam::AbstractCheckableAlbumModel::setData (
    const QModelIndex & index,
    const QVariant & value,
    int role = Qt::EditRole ) [override], [protected]
```

#### Note

Do not call this function directly, use the setData(..., bool recursive)

### 9.6.2.6 setRootCheckable()

```
void Digikam::AbstractCheckableAlbumModel::setRootCheckable (
    bool rootIsCheckable )
```

Only applicable if the root album is contained at all, and if isCheckable() is true.

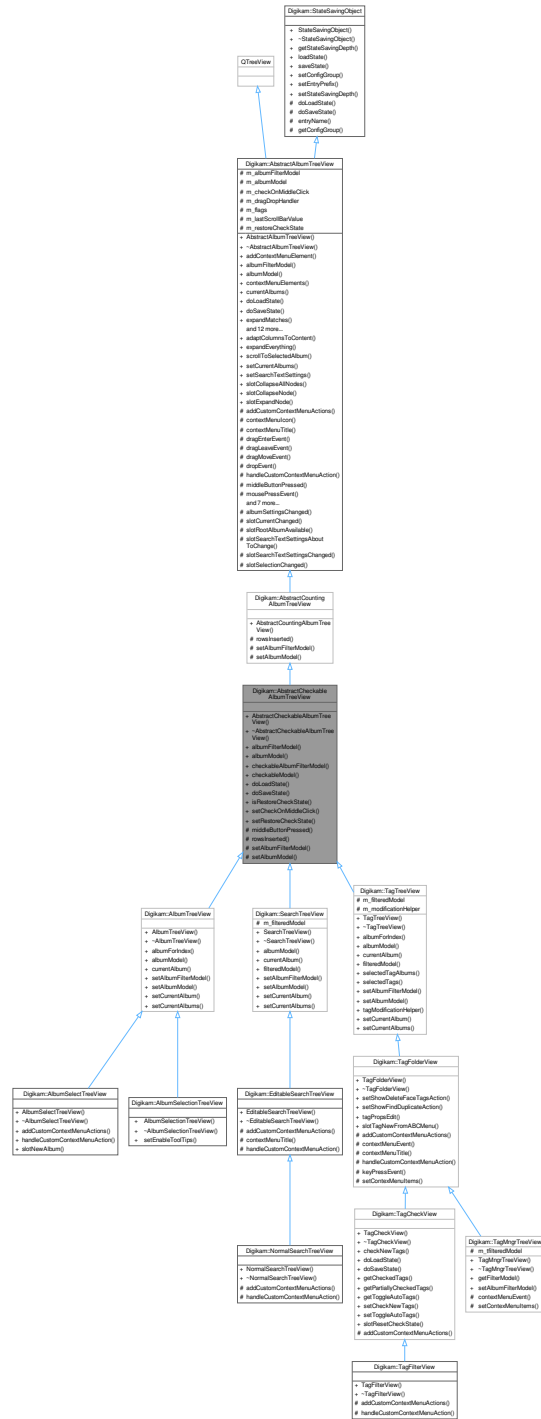
### 9.6.2.7 setTristate()

```
void Digikam::AbstractCheckableAlbumModel::setTristate (  
    bool isTristate )
```

Used to allow the user to actively set a third state, don't use if you only want to display a third state. Note that you want to set `setCheckable(true)` before.

# 9.7 Digikam::AbstractCheckableAlbumTreeView Class Reference

Inheritance diagram for Digikam::AbstractCheckableAlbumTreeView:



## Public Member Functions

- [AbstractCheckableAlbumTreeView](#) (QWidget \*const parent, Flags flags)  
*Models of these view can be checkable, they need not.*

- [CheckableAlbumFilterModel](#) \* **albumFilterModel** () const
- [AbstractCheckableAlbumModel](#) \* **albumModel** () const
  - Manage check state through the model directly.*
- [CheckableAlbumFilterModel](#) \* **checkableAlbumFilterModel** () const
- [AbstractCheckableAlbumModel](#) \* **checkableModel** () const
- void **doLoadState** () override
  - Implements state loading for the album tree view in a somewhat clumsy procedure because the model may not be fully loaded when this method is called.*
- void **doSaveState** () override
  - Implement this hook method for state saving.*
- bool **isRestoreCheckState** () const
  - Tells if the check state is restored while loading / saving state.*
- void **setCheckOnMiddleClick** (bool doThat)
  - Enable checking on middle mouse button click (default: on).*
- void **setRestoreCheckState** (bool restore)
  - Set whether to restore check state or not.*

## Public Member Functions inherited from [Digikam::AbstractCountingAlbumTreeView](#)

- **AbstractCountingAlbumTreeView** (QWidget \*const parent, Flags flags)

## Public Member Functions inherited from [Digikam::AbstractAlbumTreeView](#)

- [AbstractAlbumTreeView](#) (QWidget \*const parent, Flags flags)
  - Constructs an album tree view.*
- void **addContextMenuElement** ([ContextMenuElement](#) \*const element)
- [AlbumFilterModel](#) \* **albumFilterModel** () const
- [AbstractSpecificAlbumModel](#) \* **albumModel** () const
- QList< [ContextMenuElement](#) \* > **contextMenuElements** () const
- template<class A >
  - QList< A \* > **currentAlbums** ()
- bool **expandMatches** (const QModelIndex &index)
  - Ensures that every current match is visible by expanding all parent entries.*
- QModelIndex **indexVisuallyAt** (const QPoint &p)
  - This is a combination of indexAt() checked with visualRect().*
- void **removeContextMenuElement** ([ContextMenuElement](#) \*const element)
- QList< [Album](#) \* > **selectedItems** ()
  - selectedItems() -*
- void **setAlbumManagerCurrentAlbum** (const bool setCurrentAlbum)
  - Some treeviews shall control the global current album kept by [AlbumManager](#).*
- void **setContextMenuIcon** (const QPixmap &pixmap)
  - Set the context menu title and icon.*
- void **setContextMenuTitle** (const QString &title)
- void **setEnabledContextMenu** (const bool enable)
  - Determines the global decision to show a popup menu or not.*
- void **setExpandNewCurrentItem** (const bool doThat)
  - Expand an item when making it the new current item.*
- void **setExpandOnSingleClick** (const bool doThat)
  - Enable expanding of tree items on single click on the item (default: off)*
- void **setSelectAlbumOnClick** (const bool selectOnClick)

- Sets whether to select an album on click via the album manager or not.*
- void [setSelectOnContextMenu](#) (const bool select)
  - Sets whether to select the album under the mouse cursor on a context menu request (so that the album is shown using the album manager) or not.*
- bool **viewportEvent** (QEvent \*event) override
  - For internal use only.*

## Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (QObject \*const host)
  - Constructor.*
- virtual [~StateSavingObject](#) ()
  - Destructor.*
- [StateSavingDepth](#) [getStateSavingDepth](#) () const
  - Returns the depth used for state saving or loading.*
- void **loadState** ()
  - Invokes loading the class' state.*
- void **saveState** ()
  - Invokes saving the class' state.*
- virtual void [setConfigGroup](#) (const KConfigGroup &group)
  - Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void [setEntryPrefix](#) (const QString &prefix)
  - Define a prefix that will be used for every entry in the config group.*
- void [setStateSavingDepth](#) (const [StateSavingDepth](#) depth)
  - Sets the depth used for state saving or loading.*

## Protected Member Functions

- void [middleButtonPressed](#) (Album \*a) override
- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- void **setAlbumFilterModel** ([CheckableAlbumFilterModel](#) \*const filterModel)
- void **setAlbumModel** ([AbstractCheckableAlbumModel](#) \*const model)

## Protected Member Functions inherited from [Digikam::AbstractCountingAlbumTreeView](#)

- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- void **setAlbumFilterModel** ([AlbumFilterModel](#) \*const filterModel)
- void **setAlbumModel** ([AbstractCountingAlbumModel](#) \*const model)

## Protected Member Functions inherited from [Digikam::AbstractAlbumTreeView](#)

- virtual void [addCustomContextMenuActions](#) ([ContextMenuHelper](#) &cmh, [Album](#) \*album)
  - Hook method to add custom actions to the generated context menu.*
- virtual QPixmap [contextMenuIcon](#) () const
  - Hook method that can be implemented to return a special icon used for the context menu.*
- virtual QString [contextMenuTitle](#) () const
  - Hook method to implement that returns the title for the context menu.*
- void **dragEnterEvent** ([QDragEnterEvent](#) \*e) override
- void **dragLeaveEvent** ([QDragLeaveEvent](#) \*e) override
- void **dragMoveEvent** ([QDragMoveEvent](#) \*e) override
- void **dropEvent** ([QDropEvent](#) \*e) override
- virtual void [handleCustomContextMenuAction](#) ([QAction](#) \*action, const [AlbumPointer](#)< [Album](#) > &album)
  - Hook method to handle the custom context menu actions that were added with [addCustomContextMenuActions](#).*
- void **mousePressEvent** ([QMouseEvent](#) \*e) override
  - Other helper methods.*
- virtual QPixmap [pixmapForDrag](#) (const [QStyleOptionViewItem](#) &option, [QList](#)< [QModelIndex](#) > indexes)
  - TODO: Move to delegate, when we have one.*
- void **rowsAboutToBeRemoved** (const [QModelIndex](#) &parent, int start, int end) override
- void **rowsInserted** (const [QModelIndex](#) &index, int start, int end) override
- void **setAlbumFilterModel** ([AlbumFilterModel](#) \*const filterModel)
- void **setAlbumModel** ([AbstractSpecificAlbumModel](#) \*const model)
- virtual bool [showContextMenuAt](#) ([QContextMenuEvent](#) \*event, [Album](#) \*albumForEvent)
  - Hook method to implement that determines if a context menu shall be displayed for the given event at the position coded in the event.*
- void **startDrag** ([Qt::DropActions](#) supportedActions) override

## Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- QString [entryName](#) (const QString &base) const
  - Always use this method to create config group entry names.*
- [KConfigGroup](#) [getConfigGroup](#) () const
  - Returns the config group that must be used for state saving and loading.*

## Additional Inherited Members

## Public Types inherited from [Digikam::AbstractAlbumTreeView](#)

- enum [Flag](#) {
  - [CreateDefaultModel](#) , [CreateDefaultFilterModel](#) , [CreateDefaultDelegate](#) , [ShowCountAccordingToSettings](#) , [AlwaysShowInclusiveCounts](#) , **DefaultFlags** = [CreateDefaultFilterModel](#) | [CreateDefaultDelegate](#) | [ShowCountAccordingToSettings](#) }
- typedef [QFlags](#)< [Flag](#) > **Flags**

## Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }
  - This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

## Public Slots inherited from [Digikam::AbstractAlbumTreeView](#)

- void **adaptColumnsToContent** ()  
*Adapt the column sizes to the contents of the tree view.*
- void **expandEverything** (const QModelIndex &index)  
*Expands the complete tree under the given index.*
- void **scrollToSelectedAlbum** ()  
*Scrolls to the first selected album if there is one.*
- void **setCurrentAlbums** (const QList< Album \* > &albums, bool selectInAlbumManager=true)  
*Selects the given album.*
- void **setSearchTextSettings** (const SearchTextSettings &settings)
- void **slotCollapseAllNodes** ()  
*slotCollapseAllNodes - collapse all nodes without root node*
- void **slotCollapseNode** ()  
*slotCollapseNode - collapse recursively selected nodes*
- void **slotExpandNode** ()  
*slotExpandNode - expands recursively selected nodes*

## Signals inherited from [Digikam::AbstractAlbumTreeView](#)

- void **currentAlbumChanged** (Album \*currentAlbum)  
*Emitted when the currently selected album changes.*
- void **selectedAlbumsChanged** (const QList< Album \* > &selectedAlbums)  
*Emitted when the current selection changes.*

## Protected Slots inherited from [Digikam::AbstractAlbumTreeView](#)

- void **albumSettingsChanged** ()
- void **slotCurrentChanged** ()
- virtual void **slotRootAlbumAvailable** ()  
*override if implemented behavior is not as intended*
- void **slotSearchTextSettingsAboutToChange** (bool searched, bool willSearch)
- void **slotSearchTextSettingsChanged** (bool wasSearching, bool searching)
- void **slotSelectionChanged** ()

## Protected Attributes inherited from [Digikam::AbstractAlbumTreeView](#)

- AlbumFilterModel \* **m\_albumFilterModel** = nullptr
- AbstractSpecificAlbumModel \* **m\_albumModel** = nullptr
- bool **m\_checkOnMiddleClick** = false
- AlbumModelDragDropHandler \* **m\_dragDropHandler** = nullptr
- Flags **m\_flags** = DefaultFlags
- int **m\_lastScrollBarValue** = 0
- bool **m\_restoreCheckState** = false

## 9.7.1 Constructor & Destructor Documentation

### 9.7.1.1 AbstractCheckableAlbumTreeView()

```
Digikam::AbstractCheckableAlbumTreeView::AbstractCheckableAlbumTreeView (
    QWidget *const parent,
    Flags flags ) [explicit]
```

You need to enable it on the model.

## 9.7.2 Member Function Documentation

### 9.7.2.1 doLoadState()

```
void Digikam::AbstractCheckableAlbumTreeView::doLoadState ( ) [override], [virtual]
```

Therefore the config is first parsed into `d->statesByAlbumId` which holds the state of all tree view entries indexed by album id. Afterwards an initial sync run is done restoring the state of all model entries that are already present at this time. Every processed entry is removed from `d->statesByAlbumId`. If there are still entries left in this map we assume that the model is not fully loaded at the moment. Therefore the `rowsInserted` signal is connected to a slot that restores the state of new rows based on the remaining entries in `d->statesByAlbumId`.

Reimplemented from [Digikam::AbstractAlbumTreeView](#).

Reimplemented in [Digikam::TagCheckView](#).

### 9.7.2.2 doSaveState()

```
void Digikam::AbstractCheckableAlbumTreeView::doSaveState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Reimplemented from [Digikam::AbstractAlbumTreeView](#).

Reimplemented in [Digikam::TagCheckView](#).

### 9.7.2.3 isRestoreCheckState()

```
bool Digikam::AbstractCheckableAlbumTreeView::isRestoreCheckState ( ) const
```

#### Returns

true if restoring check state is active

### 9.7.2.4 middleButtonPressed()

```
void Digikam::AbstractCheckableAlbumTreeView::middleButtonPressed (
    Album * a ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::AbstractAlbumTreeView](#).

### 9.7.2.5 setRestoreCheckState()

```
void Digikam::AbstractCheckableAlbumTreeView::setRestoreCheckState (
    bool restore )
```

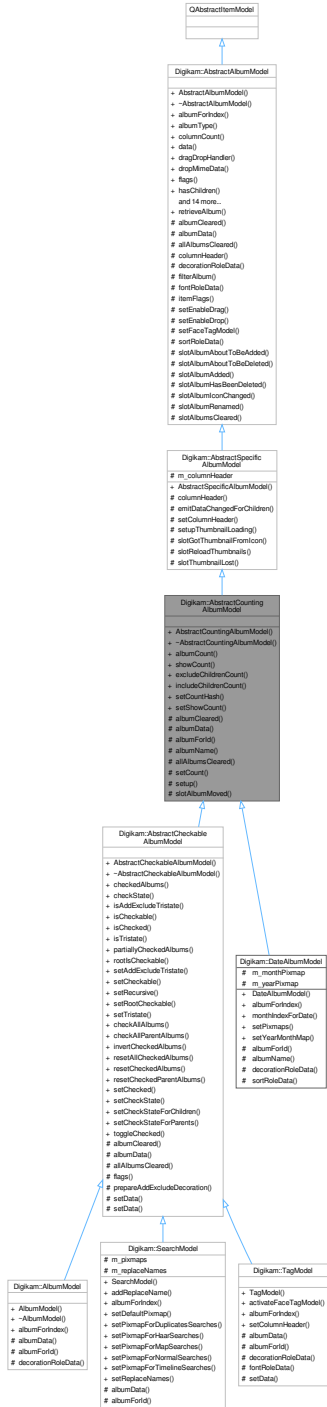


Parameters

<i>restore</i>	if true, restore check state
----------------	------------------------------

## 9.8 Digikam::AbstractCountingAlbumModel Class Reference

Inheritance diagram for Digikam::AbstractCountingAlbumModel:



## Public Slots

- void **excludeChildrenCount** (const QModelIndex &index)  
*Displays only the count of the album, without adding child albums' counts.*
- void **includeChildrenCount** (const QModelIndex &index)  
*Displays sum of the count of the album and child albums' counts.*
- void **setCountHash** (const QHash< int, int > &idCountHash)  
*Enable displaying the count.*
- void **setShowCount** (bool show)  
*Call to enable or disable showing the count. Default is false.*

## Signals

- void **signalUpdateAlbumCount** (Album \*album)

## Signals inherited from [Digikam::AbstractAlbumModel](#)

- void **rootAlbumAvailable** ()  
*This is initialized once after creation, if the root album becomes available, if it was not already available at time of construction.*

## Public Member Functions

- **AbstractCountingAlbumModel** (Album::Type albumType, Album \*const rootAlbum, RootAlbumBehavior rootBehavior=[IncludeRootAlbum](#), QObject \*const parent=nullptr)  
*Supports displaying a count alongside the album name in DisplayRole.*
- virtual int **albumCount** (Album \*album) const  
*Returns the number of included items for this album.*
- bool **showCount** () const

## Public Member Functions inherited from [Digikam::AbstractSpecificAlbumModel](#)

- **AbstractSpecificAlbumModel** (Album::Type albumType, Album \*const rootAlbum, RootAlbumBehavior rootBehavior=[IncludeRootAlbum](#), QObject \*const parent=nullptr)  
*Abstract base class, do not instantiate.*

## Public Member Functions inherited from [Digikam::AbstractAlbumModel](#)

- **AbstractAlbumModel** (Album::Type albumType, Album \*const rootAlbum, RootAlbumBehavior rootBehavior=[IncludeRootAlbum](#), QObject \*const parent=nullptr)  
*Create an [AbstractAlbumModel](#) object for albums with the given type.*
- Album \* **albumForIndex** (const QModelIndex &index) const  
*Returns the album object associated with the given model index.*
- Album::Type **albumType** () const  
*Returns the [Album::Type](#) of the contained albums.*
- int **columnCount** (const QModelIndex &parent=QModelIndex()) const override
- QVariant **data** (const QModelIndex &index, int role=Qt::DisplayRole) const override
- AlbumModelDragDropHandler \* **dragDropHandler** () const  
*Returns the drag drop handler, or 0 if none is installed.*

- bool **dropMimeData** (const QMimeData \*data, Qt::DropAction action, int row, int column, const QModelIndex &parent) override
- Qt::ItemFlags **flags** (const QModelIndex &index) const override
- bool **hasChildren** (const QModelIndex &parent=QModelIndex()) const override
- QVariant **headerData** (int section, Qt::Orientation orientation, int role=Qt::DisplayRole) const override
- QModelIndex **index** (int row, int column, const QModelIndex &parent=QModelIndex()) const override
- QModelIndex **indexForAlbum** (Album \*album) const
  - Return the QModelIndex for the given album, or an invalid index if the album is not contained in this model.*
- bool **isFaceTagModel** () const
  - Returns true if the album model a face tag model.*
- QMimeData \* **mimeData** (const QModelIndexList &indexes) const override
- QStringList **mimeTypes** () const override
- QModelIndex **parent** (const QModelIndex &index) const override
- Album \* **rootAlbum** () const
- RootAlbumBehavior **rootAlbumBehavior** () const
  - Returns the root album behavior set for this model.*
- QModelIndex **rootAlbumIndex** () const
  - Return the index corresponding to the root album.*
- int **rowCount** (const QModelIndex &parent=QModelIndex()) const override
- void **setDragDropHandler** (AlbumModelDragDropHandler \*handler)
  - Set a drag drop handler.*
- void **setDropIndex** (const QModelIndex &index)
  - Set current index from QDragMoveEvent.*
- Qt::DropActions **supportedDropActions** () const override

#### Protected Slots

- void **slotAlbumMoved** (Album \*album)

#### Protected Slots inherited from [Digikam::AbstractSpecificAlbumModel](#)

- void **slotGotThumbnailFromIcon** (Album \*album, const QPixmap &thumbnail)
- void **slotReloadThumbnails** ()
- void **slotThumbnailLost** (Album \*album)

#### Protected Slots inherited from [Digikam::AbstractAlbumModel](#)

- void **slotAlbumAboutToBeAdded** (Album \*album, Album \*parent, Album \*prev)
- void **slotAlbumAboutToBeDeleted** (Album \*album)
- void **slotAlbumAdded** (Album \*)
- void **slotAlbumHasBeenDeleted** (Album \*album)
- void **slotAlbumIconChanged** (Album \*album)
- void **slotAlbumRenamed** (Album \*album)
- void **slotAlbumsCleared** ()

### Protected Member Functions

- void `albumCleared (Album *album)` override  
*Notification when an entry is removed.*
- QVariant `albumData (Album *a, int role)` const override  
*Reimplemented from parent classes.*
- virtual `Album * albumForId (int id)` const =0  
*need to implement in subclass*
- virtual QString `albumName (Album *a)` const  
*Can reimplement in subclass.*
- void `allAlbumsCleared ()` override  
*Notification when all entries are removed.*
- void `setCount (Album *album, int count)`  
*If you do not use `setCountHash`, `excludeChildrenCount` and `includeChildrenCount`, you can set a count here.*
- void `setup ()`  
*Call this method in children class constructors to init signal/slots connections.*

### Protected Member Functions inherited from [Digikam::AbstractSpecificAlbumModel](#)

- QString `columnHeader ()` const override  
*For subclassing convenience: A part of the implementation of `headerData()`*
- void `emitDataChangedForChildren (Album *album)`
- void `setColumnHeader (const QString &header)`
- void `setupThumbnailLoading ()`  
*You need to call this from your constructor if you intend to load the thumbnail facilities of this class.*

### Protected Member Functions inherited from [Digikam::AbstractAlbumModel](#)

- virtual QVariant `decorationRoleData (Album *a)` const  
*For subclassing convenience: A part of the implementation of `data()`*
- virtual bool `filterAlbum (Album *album)` const  
*Returns true for those and only those albums that shall be contained in this model.*
- virtual QVariant `fontRoleData (Album *a)` const  
*For subclassing convenience: A part of the implementation of `data()`*
- virtual Qt::ItemFlags `itemFlags (Album *album)` const  
*For subclassing convenience: A part of the implementation of `itemFlags()`*
- void `setEnableDrag (bool enable)`  
*Switch on drag and drop globally for all items.*
- void `setEnableDrop (bool enable)`
- void `setFaceTagModel (bool enable)`
- virtual QVariant `sortRoleData (Album *a)` const  
*For subclassing convenience: A part of the implementation of `data()`*

### Additional Inherited Members

### Public Types inherited from [Digikam::AbstractAlbumModel](#)

- enum `AlbumDataRole` {  
`AlbumTitleRole` = Qt::UserRole , `AlbumTypeRole` = Qt::UserRole + 1 , `AlbumPointerRole` = Qt::UserRole + 2  
, `AlbumIdRole` = Qt::UserRole + 3 ,  
`AlbumGlobalIdRole` = Qt::UserRole + 4 , `AlbumSortRole` = Qt::UserRole + 5 }
- enum `RootAlbumBehavior` { `IncludeRootAlbum` , `IgnoreRootAlbum` }  
*[AbstractAlbumModel](#) is the abstract base class for all models that present [Album](#) objects as managed by [AlbumManager](#).*

## Static Public Member Functions inherited from [Digikam::AbstractAlbumModel](#)

- static [Album](#) \* [retrieveAlbum](#) (const QModelIndex &index)  
*Returns the album represented by the index.*

## Protected Attributes inherited from [Digikam::AbstractSpecificAlbumModel](#)

- QString [m\\_columnHeader](#)

### 9.8.1 Member Function Documentation

#### 9.8.1.1 albumCleared()

```
void Digikam::AbstractCountingAlbumModel::albumCleared (
    Album * ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::AbstractAlbumModel](#).

#### 9.8.1.2 albumCount()

```
int Digikam::AbstractCountingAlbumModel::albumCount (
    Album * album ) const [virtual]
```

##### Returns

positive value or -1 if unknown

#### 9.8.1.3 albumData()

```
QVariant Digikam::AbstractCountingAlbumModel::albumData (
    Album * a,
    int role ) const [override], [protected], [virtual]
```

Reimplemented from [Digikam::AbstractAlbumModel](#).

Reimplemented in [Digikam::AlbumModel](#), [Digikam::TagModel](#), and [Digikam::SearchModel](#).

#### 9.8.1.4 albumForId()

```
virtual Album * Digikam::AbstractCountingAlbumModel::albumForId (
    int id ) const [protected], [pure virtual]
```

Implemented in [Digikam::AlbumModel](#), [Digikam::TagModel](#), [Digikam::SearchModel](#), and [Digikam::DateAlbumModel](#).

### 9.8.1.5 albumName()

```
QString Digikam::AbstractCountingAlbumModel::albumName (
    Album * a ) const [protected], [virtual]
```

Reimplemented in [Digikam::DateAlbumModel](#).

### 9.8.1.6 allAlbumsCleared()

```
void Digikam::AbstractCountingAlbumModel::allAlbumsCleared ( ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::AbstractAlbumModel](#).

### 9.8.1.7 excludeChildrenCount

```
void Digikam::AbstractCountingAlbumModel::excludeChildrenCount (
    const QModelIndex & index ) [slot]
```

This is the default. Can connect to QTreeView's expanded() signal.

### 9.8.1.8 includeChildrenCount

```
void Digikam::AbstractCountingAlbumModel::includeChildrenCount (
    const QModelIndex & index ) [slot]
```

Can connect to QTreeView's collapsed() signal.

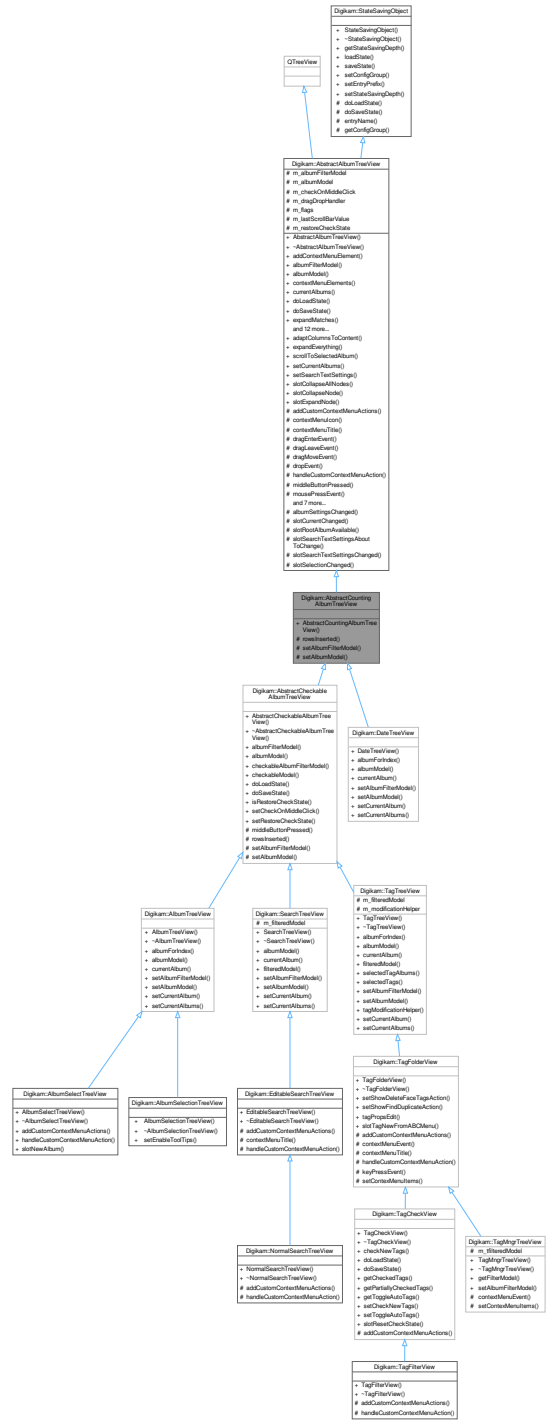
### 9.8.1.9 setCountHash

```
void Digikam::AbstractCountingAlbumModel::setCountHash (
    const QHash< int, int > & idCountHash ) [slot]
```

Set a map of album id -> count (excluding children). If an album is not contained, no count is displayed. To display a count of 0, there must be an entry album id -> 0.

# 9.9 Digikam::AbstractCountingAlbumTreeView Class Reference

Inheritance diagram for Digikam::AbstractCountingAlbumTreeView:



### Public Member Functions

- **AbstractCountingAlbumTreeView** (QWidget \*const parent, Flags flags)

## Public Member Functions inherited from [Digikam::AbstractAlbumTreeView](#)

- [AbstractAlbumTreeView](#) (QWidget \*const parent, Flags flags)  
*Constructs an album tree view.*
- void **addContextMenuElement** ([ContextMenuElement](#) \*const element)
- [AlbumFilterModel](#) \* **albumFilterModel** () const
- [AbstractSpecificAlbumModel](#) \* **albumModel** () const
- QList< [ContextMenuElement](#) \* > **contextMenuElements** () const
- template<class A >  
QList< A \* > **currentAlbums** ()
- void **doLoadState** () override  
*Implements state loading for the album tree view in a somewhat clumsy procedure because the model may not be fully loaded when this method is called.*
- void **doSaveState** () override  
*Implement this hook method for state saving.*
- bool **expandMatches** (const QModelIndex &index)  
*Ensures that every current match is visible by expanding all parent entries.*
- QModelIndex **indexVisuallyAt** (const QPoint &p)  
*This is a combination of indexAt() checked with visualRect().*
- void **removeContextMenuElement** ([ContextMenuElement](#) \*const element)
- QList< [Album](#) \* > **selectedItems** ()  
*selectedItems()* -
- void **setAlbumManagerCurrentAlbum** (const bool setCurrentAlbum)  
*Some treeviews shall control the global current album kept by [AlbumManager](#).*
- void **setContextMenuIcon** (const QPixmap &pixmap)  
*Set the context menu title and icon.*
- void **setContextMenuTitle** (const QString &title)
- void **setEnabledContextMenu** (const bool enable)  
*Determines the global decision to show a popup menu or not.*
- void **setExpandNewCurrentItem** (const bool doThat)  
*Expand an item when making it the new current item.*
- void **setExpandOnSingleClick** (const bool doThat)  
*Enable expanding of tree items on single click on the item (default: off)*
- void **setSelectAlbumOnClick** (const bool selectOnClick)  
*Sets whether to select an album on click via the album manager or not.*
- void **setSelectOnContextMenu** (const bool select)  
*Sets whether to select the album under the mouse cursor on a context menu request (so that the album is shown using the album manager) or not.*
- bool **viewportEvent** (QEvent \*event) override  
*For internal use only.*

## Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (QObject \*const host)  
*Constructor.*
- virtual ~[StateSavingObject](#) ()  
*Destructor.*
- [StateSavingDepth](#) **getStateSavingDepth** () const  
*Returns the depth used for state saving or loading.*
- void **loadState** ()  
*Invokes loading the class' state.*



- void **saveState** ()  
*Invokes saving the class' state.*
- virtual void **setConfigGroup** (const KConfigGroup &group)  
*Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void **setEntryPrefix** (const QString &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void **setStateSavingDepth** (const StateSavingDepth depth)  
*Sets the depth used for state saving or loading.*

### Protected Member Functions

- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- void **setAlbumFilterModel** (AlbumFilterModel \*const filterModel)
- void **setAlbumModel** (AbstractCountingAlbumModel \*const model)

### Protected Member Functions inherited from Digikam::AbstractAlbumTreeView

- virtual void **addCustomContextMenuActions** (ContextMenuHelper &cmh, Album \*album)  
*Hook method to add custom actions to the generated context menu.*
- virtual QPixmap **contextMenuIcon** () const  
*Hook method that can be implemented to return a special icon used for the context menu.*
- virtual QString **contextMenuTitle** () const  
*Hook method to implement that returns the title for the context menu.*
- void **dragEnterEvent** (QDragEnterEvent \*e) override
- void **dragLeaveEvent** (QDragLeaveEvent \*e) override
- void **dragMoveEvent** (QDragMoveEvent \*e) override
- void **dropEvent** (QDropEvent \*e) override
- virtual void **handleCustomContextMenuAction** (QAction \*action, const AlbumPointer< Album > &album)  
*Hook method to handle the custom context menu actions that were added with addCustomContextMenuActions.*
- virtual void **middleButtonPressed** (Album \*a)
- void **mousePressEvent** (QMouseEvent \*e) override  
*Other helper methods.*
- virtual QPixmap **pixmapForDrag** (const QStyleOptionViewItem &option, QList< QModelIndex > indexes)  
*TODO: Move to delegate, when we have one.*
- void **rowsAboutToBeRemoved** (const QModelIndex &parent, int start, int end) override
- void **rowsInserted** (const QModelIndex &index, int start, int end) override
- void **setAlbumFilterModel** (AlbumFilterModel \*const filterModel)
- void **setAlbumModel** (AbstractSpecificAlbumModel \*const model)
- virtual bool **showContextMenuAt** (QContextMenuEvent \*event, Album \*albumForEvent)  
*Hook method to implement that determines if a context menu shall be displayed for the given event at the position coded in the event.*
- void **startDrag** (Qt::DropActions supportedActions) override

### Protected Member Functions inherited from Digikam::StateSavingObject

- QString **entryName** (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup **getConfigGroup** () const  
*Returns the config group that must be used for state saving and loading.*

## Additional Inherited Members

### Public Types inherited from [Digikam::AbstractAlbumTreeView](#)

- enum [Flag](#) { [CreateDefaultModel](#) , [CreateDefaultFilterModel](#) , [CreateDefaultDelegate](#) , [ShowCountAccordingToSettings](#) , [AlwaysShowInclusiveCounts](#) , **DefaultFlags** = [CreateDefaultFilterModel](#) | [CreateDefaultDelegate](#) | [ShowCountAccordingToSettings](#) }
- typedef QFlags< [Flag](#) > **Flags**

### Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }  
*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

### Public Slots inherited from [Digikam::AbstractAlbumTreeView](#)

- void **adaptColumnsToContent** ()  
*Adapt the column sizes to the contents of the tree view.*
- void **expandEverything** (const QModelIndex &index)  
*Expands the complete tree under the given index.*
- void **scrollToSelectedAlbum** ()  
*Scrolls to the first selected album if there is one.*
- void **setCurrentAlbums** (const QList< [Album](#) \* > &albums, bool selectInAlbumManager=true)  
*Selects the given album.*
- void **setSearchTextSettings** (const [SearchTextSettings](#) &settings)
- void **slotCollapseAllNodes** ()  
*slotCollapseAllNodes - collapse all nodes without root node*
- void **slotCollapseNode** ()  
*slotCollapseNode - collapse recursively selected nodes*
- void **slotExpandNode** ()  
*slotExpandNode - expands recursively selected nodes*

### Signals inherited from [Digikam::AbstractAlbumTreeView](#)

- void **currentAlbumChanged** ([Album](#) \*currentAlbum)  
*Emitted when the currently selected album changes.*
- void **selectedAlbumsChanged** (const QList< [Album](#) \* > &selectedAlbums)  
*Emitted when the current selection changes.*

### Protected Slots inherited from [Digikam::AbstractAlbumTreeView](#)

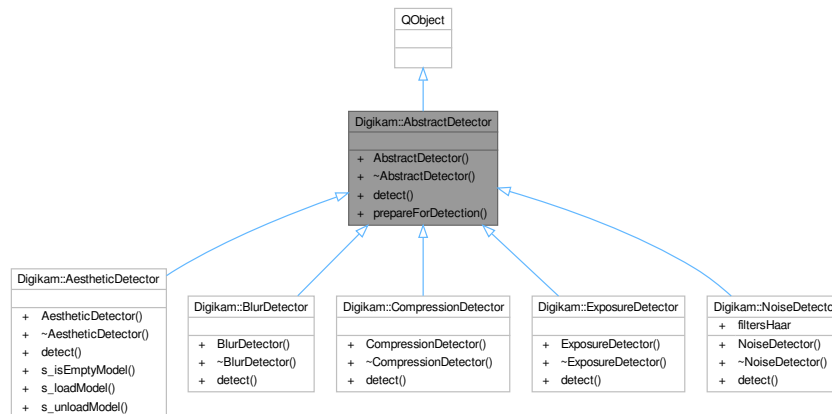
- void **albumSettingsChanged** ()
- void **slotCurrentChanged** ()
- virtual void **slotRootAlbumAvailable** ()  
*override if implemented behavior is not as intended*
- void **slotSearchTextSettingsAboutToChange** (bool searched, bool willSearch)
- void **slotSearchTextSettingsChanged** (bool wasSearching, bool searching)
- void **slotSelectionChanged** ()

## Protected Attributes inherited from [Digikam::AbstractAlbumTreeView](#)

- [AlbumFilterModel](#) \* **m\_albumFilterModel** = nullptr
- [AbstractSpecificAlbumModel](#) \* **m\_albumModel** = nullptr
- bool **m\_checkOnMiddleClick** = false
- [AlbumModelDragDropHandler](#) \* **m\_dragDropHandler** = nullptr
- Flags **m\_flags** = DefaultFlags
- int **m\_lastScrollBarValue** = 0
- bool **m\_restoreCheckState** = false

## 9.10 Digikam::AbstractDetector Class Reference

Inheritance diagram for Digikam::AbstractDetector:



### Public Member Functions

- **AbstractDetector** (QObject \*const parent=nullptr)
- virtual float **detect** (const cv::Mat &image) const =0

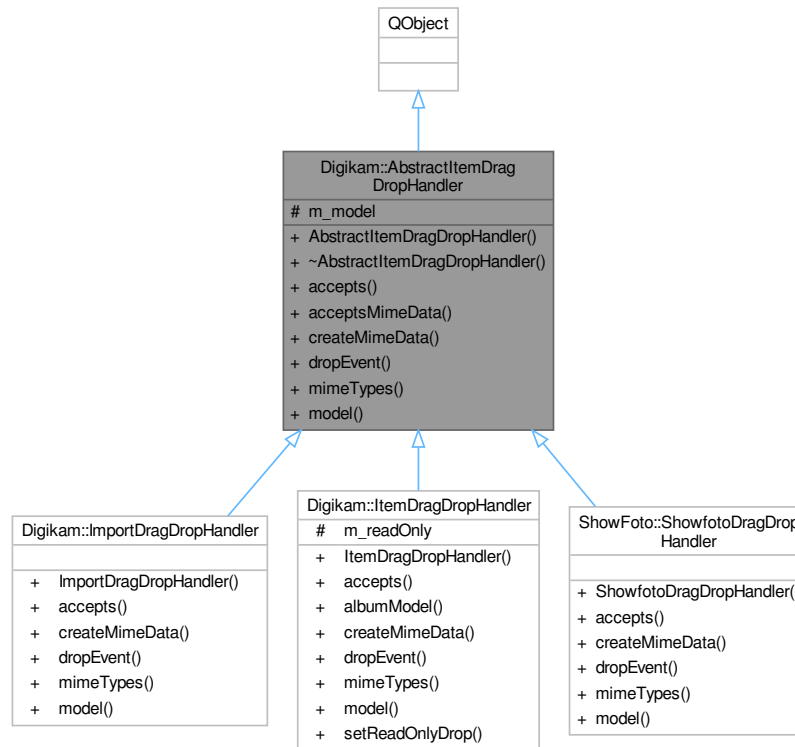
### Static Public Member Functions

- static cv::Mat **prepareForDetection** (const [DImg](#) &inputImage)

*NOTE: Maybe this function will move to read\_image() of imagequalityparser in case all detectors of IQS use cv::Mat.*

## 9.11 Digikam::AbstractItemDragDropHandler Class Reference

Inheritance diagram for Digikam::AbstractItemDragDropHandler:



### Public Member Functions

- **AbstractItemDragDropHandler** (QAbstractItemModel \*const model)
- virtual Qt::DropAction **accepts** (const QDropEvent \*e, const QModelIndex &dropIndex)  
*Returns if the given mime data is accepted for drop on dropIndex.*
- virtual bool **acceptsMimeData** (const QMimeData \*data)  
*Returns if the given mime data can be handled.*
- virtual QMimeData \* **createMimeData** (const QList< QModelIndex > &)  
*Create a mime data object for starting a drag from the given Albums.*
- virtual bool **dropEvent** (QAbstractItemView \*view, const QDropEvent \*e, const QModelIndex &droppedOn)  
*Gives the view and the occurring drop event.*
- virtual QStringList **mimeTypes** () const  
*Returns the supported mime types.*
- QAbstractItemModel \* **model** () const

### Protected Attributes

- QAbstractItemModel \* **m\_model** = nullptr

## 9.11.1 Member Function Documentation

### 9.11.1.1 accepts()

```
Qt::DropAction Digikam::AbstractItemDragDropHandler::accepts (
    const QDropEvent * e,
    const QModelIndex & dropIndex ) [virtual]
```

Returns the proposed action, or Qt::IgnoreAction if not accepted.

Reimplemented in [Digikam::ImportDragDropHandler](#), [Digikam::ItemDragDropHandler](#), and [ShowFoto::ShowfotoDragDropHandler](#).

### 9.11.1.2 acceptsMimeData()

```
bool Digikam::AbstractItemDragDropHandler::acceptsMimeData (
    const QMimeData * data ) [virtual]
```

acceptsMimeData shall return true if a drop of the given mime data will be accepted on any index or place at all. If this returns false, the more specific method [accepts\(\)](#) will not be called for this drag. The default implementation uses [mimeTypes\(\)](#) to check for supported mime types. There is usually no need to reimplement this.

### 9.11.1.3 createMimeData()

```
QMimeData * Digikam::AbstractItemDragDropHandler::createMimeData (
    const QList< QModelIndex > & ) [virtual]
```

Reimplemented in [Digikam::ImportDragDropHandler](#), [Digikam::ItemDragDropHandler](#), and [ShowFoto::ShowfotoDragDropHandler](#).

### 9.11.1.4 dropEvent()

```
bool Digikam::AbstractItemDragDropHandler::dropEvent (
    QAbstractItemView * view,
    const QDropEvent * e,
    const QModelIndex & droppedOn ) [virtual]
```

The index is the index where the drop was dropped on. It may be invalid (dropped on decoration, viewport) Returns true if the event is to be accepted.

Reimplemented in [Digikam::ImportDragDropHandler](#), [Digikam::ItemDragDropHandler](#), and [ShowFoto::ShowfotoDragDropHandler](#).

### 9.11.1.5 mimeTypes()

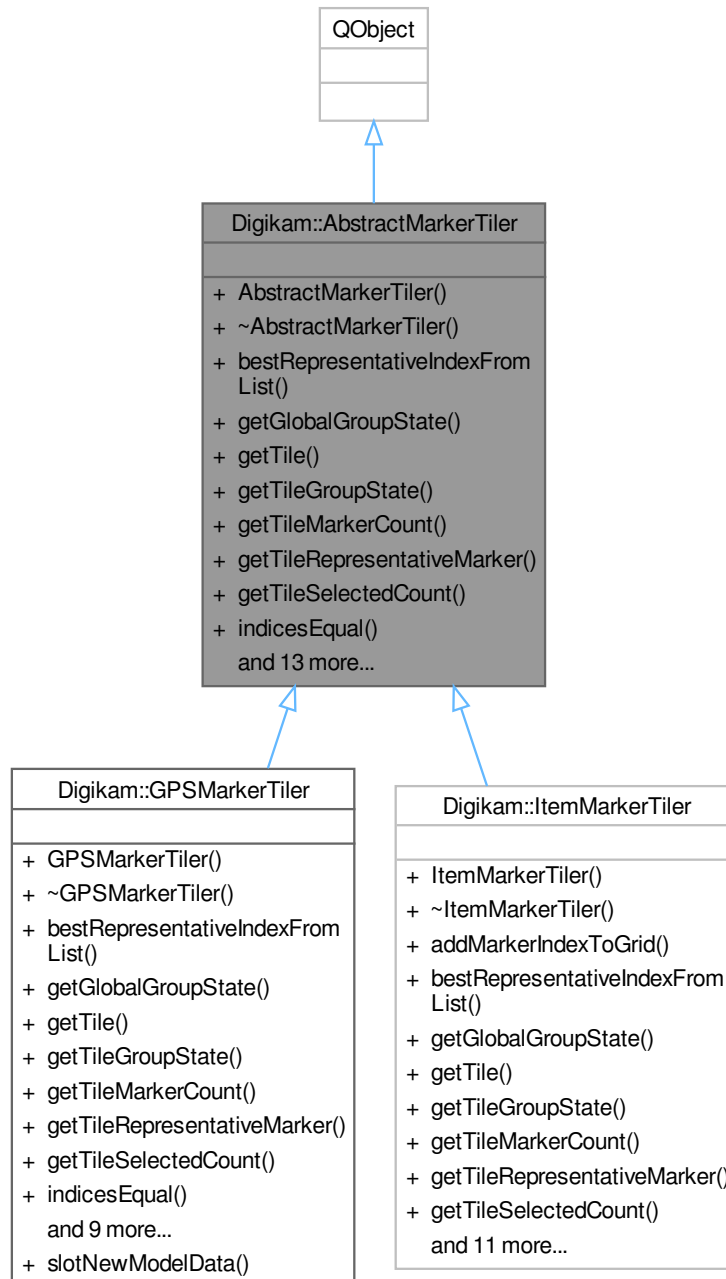
```
QStringList Digikam::AbstractItemDragDropHandler::mimeTypes ( ) const [virtual]
```

Called by the default implementation of model's [mimeTypes\(\)](#).

Reimplemented in [Digikam::ImportDragDropHandler](#), [Digikam::ItemDragDropHandler](#), and [ShowFoto::ShowfotoDragDropHandler](#).

## 9.12 Digikam::AbstractMarkerTiler Class Reference

Inheritance diagram for Digikam::AbstractMarkerTiler:



### Classes

- class [ClickInfo](#)
- class [NonEmptyIterator](#)
- class [Tile](#)

## Public Types

- enum **TilerFlag** { **FlagNull** = 0 , **FlagMovable** = 1 }
- typedef QFlags< TilerFlag > **TilerFlags**

## Signals

- void **signalThumbnailAvailableForIndex** (const QVariant &index, const QPixmap &pixmap)
- void **signalTilesOrSelectionChanged** ()

## Public Member Functions

- **AbstractMarkerTiler** (QObject \*const parent=nullptr)
- virtual QVariant **bestRepresentativeIndexFromList** (const QList< QVariant > &indices, const int sortKey)=0
- virtual GeoGroupState **getGlobalGroupState** ()=0
- virtual **Tile** \* **getTile** (const **TileIndex** &tileIndex, const bool stopIfEmpty)=0
- virtual GeoGroupState **getTileGroupState** (const **TileIndex** &tileIndex)=0
- virtual int **getTileMarkerCount** (const **TileIndex** &tileIndex)=0
- virtual QVariant **getTileRepresentativeMarker** (const **TileIndex** &tileIndex, const int sortKey)=0

*These should be implemented for thumbnail handling.*

- virtual int **getTileSelectedCount** (const **TileIndex** &tileIndex)=0
- bool **indicesEqual** (const QList &a, const QList &b, const int upToLevel) const
- virtual bool **indicesEqual** (const QVariant &a, const QVariant &b) const =0
- bool **isDirty** () const
- virtual void **onIndicesClicked** (const **ClickInfo** &clickInfo)

*These can be implemented if you want to react to actions in geolocation interface.*

- virtual void **onIndicesMoved** (const **TileIndex::List** &tileIndicesList, const **GeoCoordinates** &target←Coordinates, const QPersistentModelIndex &targetSnapIndex)
- virtual QPixmap **pixmapFromRepresentativeIndex** (const QVariant &index, const QSize &size)=0
- virtual void **prepareTiles** (const **GeoCoordinates** &upperLeft, const **GeoCoordinates** &lowerRight, int level)=0
- virtual void **regenerateTiles** ()=0
- void **resetRootTile** ()
- **Tile** \* **rootTile** ()
- virtual void **setActive** (const bool state)=0
- void **setDirty** (const bool state=true)
- virtual **Tile** \* **tileNew** ()=0
- virtual TilerFlags **tilerFlags** () const

*These have to be implemented.*

## 9.12.1 Member Function Documentation

### 9.12.1.1 bestRepresentativeIndexFromList()

```
virtual QVariant Digikam::AbstractMarkerTiler::bestRepresentativeIndexFromList (
    const QList< QVariant > & indices,
    const int sortKey ) [pure virtual]
```

Implemented in [Digikam::GPSMarkerTiler](#).

### 9.12.1.2 `getTile()`

```
virtual Tile * Digikam::AbstractMarkerTiler::getTile (
    const TileIndex & tileIndex,
    const bool stopIfEmpty ) [pure virtual]
```

Implemented in [Digikam::GPSMarkerTiler](#).

### 9.12.1.3 `getTileGroupState()`

```
virtual GeoGroupState Digikam::AbstractMarkerTiler::getTileGroupState (
    const TileIndex & tileIndex ) [pure virtual]
```

Implemented in [Digikam::GPSMarkerTiler](#).

### 9.12.1.4 `getTileRepresentativeMarker()`

```
virtual QVariant Digikam::AbstractMarkerTiler::getTileRepresentativeMarker (
    const TileIndex & tileIndex,
    const int sortKey ) [pure virtual]
```

Implemented in [Digikam::ItemMarkerTiler](#), and [Digikam::GPSMarkerTiler](#).

### 9.12.1.5 `indicesEqual()`

```
virtual bool Digikam::AbstractMarkerTiler::indicesEqual (
    const QVariant & a,
    const QVariant & b ) const [pure virtual]
```

Implemented in [Digikam::GPSMarkerTiler](#).

### 9.12.1.6 `onIndicesClicked()`

```
void Digikam::AbstractMarkerTiler::onIndicesClicked (
    const ClickInfo & clickInfo ) [virtual]
```

Reimplemented in [Digikam::ItemMarkerTiler](#), and [Digikam::GPSMarkerTiler](#).

### 9.12.1.7 `pixmapFromRepresentativeIndex()`

```
virtual QPixmap Digikam::AbstractMarkerTiler::pixmapFromRepresentativeIndex (
    const QVariant & index,
    const QSize & size ) [pure virtual]
```

Implemented in [Digikam::GPSMarkerTiler](#).



### 9.12.1.8 prepareTiles()

```
virtual void Digikam::AbstractMarkerTiler::prepareTiles (
    const GeoCoordinates & upperLeft,
    const GeoCoordinates & lowerRight,
    int level ) [pure virtual]
```

Implemented in [Digikam::GPSMarkerTiler](#).

### 9.12.1.9 setActive()

```
virtual void Digikam::AbstractMarkerTiler::setActive (
    const bool state ) [pure virtual]
```

Implemented in [Digikam::GPSMarkerTiler](#).

### 9.12.1.10 tilerFlags()

```
AbstractMarkerTiler::TilerFlags Digikam::AbstractMarkerTiler::tilerFlags ( ) const [virtual]
```

Reimplemented in [Digikam::ItemMarkerTiler](#).

## 9.13 Digikam::AbstractMarkerTiler::ClickInfo Class Reference

### Public Attributes

- GeoMouseModes **currentMouseMode**
- GeoGroupState **groupSelectionMode**
- QVariant **representativeIndex**
- TileIndex::List **tileIndicesList**

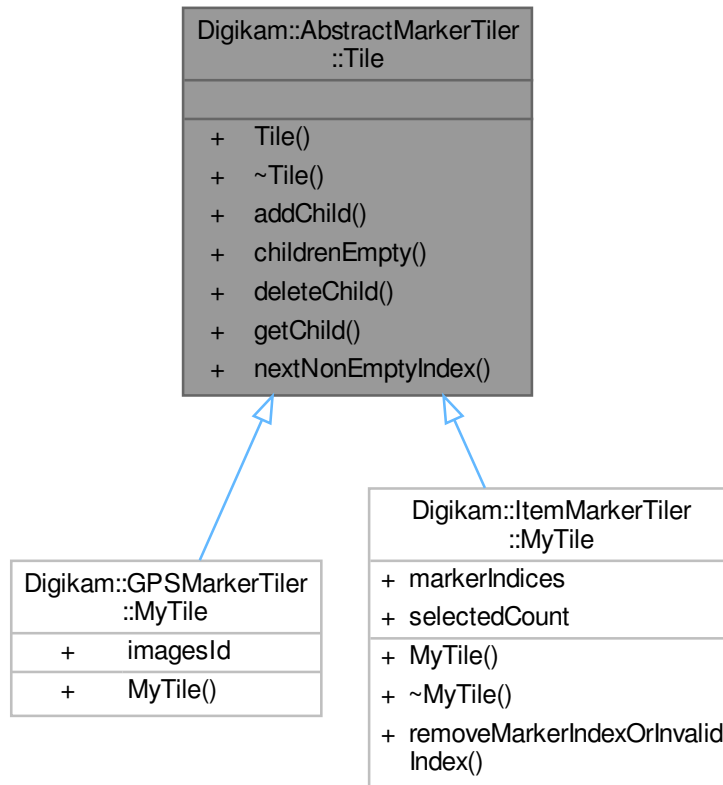
## 9.14 Digikam::AbstractMarkerTiler::NonEmptyIterator Class Reference

### Public Member Functions

- **NonEmptyIterator** ([AbstractMarkerTiler](#) \*const model, const int level)
- **NonEmptyIterator** ([AbstractMarkerTiler](#) \*const model, const int level, const GeoCoordinates::PairList &normalizedMapBounds)
- **NonEmptyIterator** ([AbstractMarkerTiler](#) \*const model, const int level, const [TileIndex](#) &startIndex, const [TileIndex](#) &endIndex)
- bool **atEnd** () const
- [TileIndex](#) **currentIndex** () const
- [AbstractMarkerTiler](#) \* **model** () const
- [TileIndex](#) **nextIndex** ()

## 9.15 Digikam::AbstractMarkerTiler::Tile Class Reference

Inheritance diagram for Digikam::AbstractMarkerTiler::Tile:

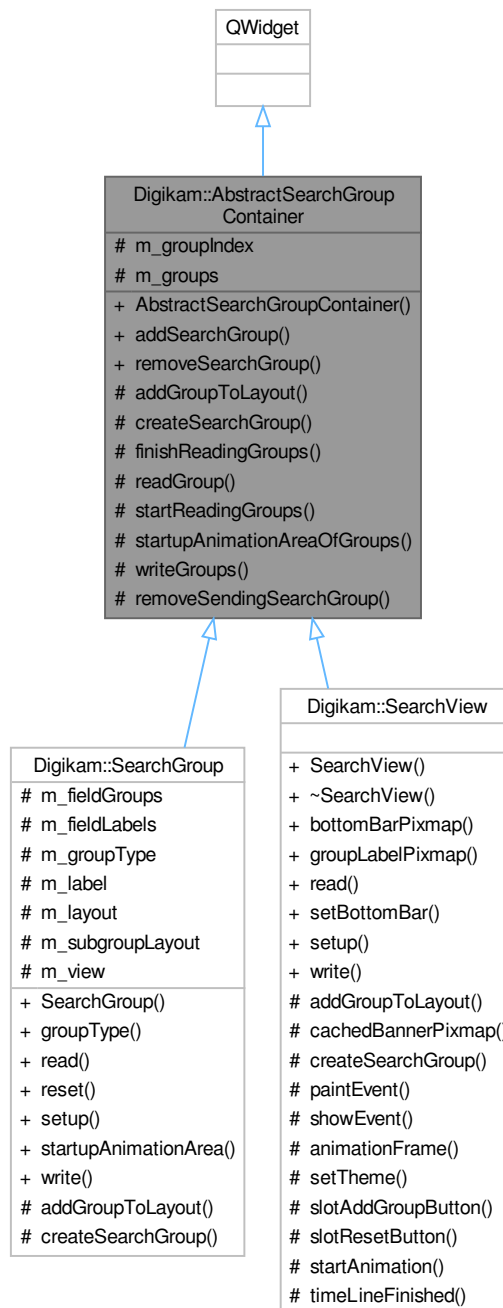


### Public Member Functions

- **Tile \* addChild** (const int linearIndex, **Tile** \*tilePointer)
- bool **childrenEmpty** () const
- void **deleteChild** (**Tile** \*const childTile, const int knownLinearIndex=-1)  
*Sets the pointer to a child tile to zero and deletes the child.*
- **Tile \* getChild** (const int linearIndex)
- int **nextNonEmptyIndex** (int linearIndex) const  
*returns the next non empty child index or -1.*

## 9.16 Digikam::AbstractSearchGroupContainer Class Reference

Inheritance diagram for Digikam::AbstractSearchGroupContainer:



### Public Slots

- `SearchGroup * addSearchGroup ()`
- `void removeSearchGroup (SearchGroup *group)`

## Public Member Functions

- **AbstractSearchGroupContainer** (QWidget \*const parent=nullptr)  
*Abstract base class for classes that contain SearchGroups To contain common code of [SearchView](#) and [SearchGroup](#), as SearchGroups can have subgroups.*

## Protected Slots

- void **removeSendingSearchGroup** ()

## Protected Member Functions

- virtual void **addGroupToLayout** ([SearchGroup](#) \*group)=0  
*Re-implement: Adds a newly created group to the layout structures.*
- virtual [SearchGroup](#) \* **createSearchGroup** ()=0  
*Re-implement: create and setup a search group.*
- void **finishReadingGroups** ()  
*Call when the XML part is finished.*
- void **readGroup** ([SearchXmlCachingReader](#) &reader)  
*Call when a group element is the current element.*
- void **startReadingGroups** ([SearchXmlCachingReader](#) &reader)  
*Call before reading the XML part that could contain group elements.*
- QList< QRect > **startupAnimationAreaOfGroups** () const  
*Collects the data from the same method of all contained groups (position relative to this widget)*
- void **writeGroups** ([SearchXmlWriter](#) &writer) const  
*Write contained groups to writer.*

## Protected Attributes

- int **m\_groupIndex** = 0
- QList< [SearchGroup](#) \* > **m\_groups**

## 9.16.1 Member Function Documentation

### 9.16.1.1 addGroupToLayout()

```
virtual void Digikam::AbstractSearchGroupContainer::addGroupToLayout (
    SearchGroup * group ) [protected], [pure virtual]
```

Implemented in [Digikam::SearchGroup](#), and [Digikam::SearchView](#).

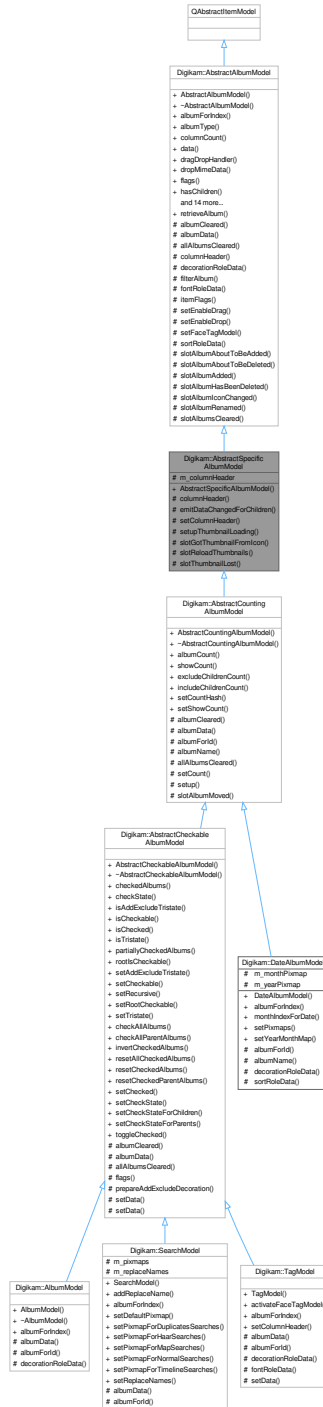
### 9.16.1.2 createSearchGroup()

```
virtual SearchGroup * Digikam::AbstractSearchGroupContainer::createSearchGroup ( ) [protected],
[pure virtual]
```

Implemented in [Digikam::SearchGroup](#), and [Digikam::SearchView](#).

## 9.17 Digikam::AbstractSpecificAlbumModel Class Reference

Inheritance diagram for Digikam::AbstractSpecificAlbumModel:



### Public Member Functions

- **AbstractSpecificAlbumModel** (**Album::Type** albumType, **Album** \*const rootAlbum, **RootAlbumBehavior** rootBehavior=**IncludeRootAlbum**, **QObject** \*const parent=nullptr)

*Abstract base class, do not instantiate.*

## Public Member Functions inherited from [Digikam::AbstractAlbumModel](#)

- [AbstractAlbumModel](#) ([Album::Type](#) albumType, [Album](#) \*const rootAlbum, [RootAlbumBehavior](#) rootBehavior=[IncludeRootAlbum](#), [QObject](#) \*const parent=nullptr)
  - Create an [AbstractAlbumModel](#) object for albums with the given type.
- [Album](#) \* **albumForIndex** (const [QModelIndex](#) &index) const
  - Returns the album object associated with the given model index.
- [Album::Type](#) **albumType** () const
  - Returns the [Album::Type](#) of the contained albums.
- int **columnCount** (const [QModelIndex](#) &parent=[QModelIndex](#)()) const override
- [QVariant](#) **data** (const [QModelIndex](#) &index, int role=[Qt::DisplayRole](#)) const override
- [AlbumModelDragDropHandler](#) \* **dragDropHandler** () const
  - Returns the drag drop handler, or 0 if none is installed.
- bool **dropMimeData** (const [QMimeData](#) \*data, [Qt::DropAction](#) action, int row, int column, const [QModelIndex](#) &parent) override
- [Qt::ItemFlags](#) **flags** (const [QModelIndex](#) &index) const override
- bool **hasChildren** (const [QModelIndex](#) &parent=[QModelIndex](#)()) const override
- [QVariant](#) **headerData** (int section, [Qt::Orientation](#) orientation, int role=[Qt::DisplayRole](#)) const override
- [QModelIndex](#) **index** (int row, int column, const [QModelIndex](#) &parent=[QModelIndex](#)()) const override
- [QModelIndex](#) **indexForAlbum** ([Album](#) \*album) const
  - Return the [QModelIndex](#) for the given album, or an invalid index if the album is not contained in this model.
- bool **isFaceTagModel** () const
  - Returns true if the album model a face tag model.
- [QMimeData](#) \* **mimeData** (const [QModelIndexList](#) &indexes) const override
- [QStringList](#) **mimeTypes** () const override
- [QModelIndex](#) **parent** (const [QModelIndex](#) &index) const override
- [Album](#) \* **rootAlbum** () const
- [RootAlbumBehavior](#) **rootAlbumBehavior** () const
  - Returns the root album behavior set for this model.
- [QModelIndex](#) **rootAlbumIndex** () const
  - Return the index corresponding to the root album.
- int **rowCount** (const [QModelIndex](#) &parent=[QModelIndex](#)()) const override
- void **setDragDropHandler** ([AlbumModelDragDropHandler](#) \*handler)
  - Set a drag drop handler.
- void **setDropIndex** (const [QModelIndex](#) &index)
  - Set current index from [QDragMoveEvent](#).
- [Qt::DropActions](#) **supportedDropActions** () const override

## Protected Slots

- void **slotGotThumbnailFromIcon** ([Album](#) \*album, const [QPixmap](#) &thumbnail)
- void **slotReloadThumbnails** ()
- void **slotThumbnailLost** ([Album](#) \*album)

## Protected Slots inherited from [Digikam::AbstractAlbumModel](#)

- void **slotAlbumAboutToBeAdded** ([Album](#) \*album, [Album](#) \*parent, [Album](#) \*prev)
- void **slotAlbumAboutToBeDeleted** ([Album](#) \*album)
- void **slotAlbumAdded** ([Album](#) \*)
- void **slotAlbumHasBeenDeleted** ([Album](#) \*album)
- void **slotAlbumIconChanged** ([Album](#) \*album)
- void **slotAlbumRenamed** ([Album](#) \*album)
- void **slotAlbumsCleared** ()

### Protected Member Functions

- QString `columnHeader` () const override  
*For subclassing convenience: A part of the implementation of `headerData()`*
- void `emitDataChangedForChildren` (Album \*album)
- void `setColumnHeader` (const QString &header)
- void `setupThumbnailLoading` ()  
*You need to call this from your constructor if you intend to load the thumbnail facilities of this class.*

### Protected Member Functions inherited from Digikam::AbstractAlbumModel

- virtual void `albumCleared` (Album \*)  
*Notification when an entry is removed.*
- virtual QVariant `albumData` (Album \*a, int role) const  
*For subclassing convenience: A part of the implementation of `data()`*
- virtual void `allAlbumsCleared` ()  
*Notification when all entries are removed.*
- virtual QVariant `decorationRoleData` (Album \*a) const  
*For subclassing convenience: A part of the implementation of `data()`*
- virtual bool `filterAlbum` (Album \*album) const  
*Returns true for those and only those albums that shall be contained in this model.*
- virtual QVariant `fontRoleData` (Album \*a) const  
*For subclassing convenience: A part of the implementation of `data()`*
- virtual Qt::ItemFlags `itemFlags` (Album \*album) const  
*For subclassing convenience: A part of the implementation of `itemFlags()`*
- void `setEnableDrag` (bool enable)  
*Switch on drag and drop globally for all items.*
- void `setEnableDrop` (bool enable)
- void `setFaceTagModel` (bool enable)
- virtual QVariant `sortRoleData` (Album \*a) const  
*For subclassing convenience: A part of the implementation of `data()`*

### Protected Attributes

- QString `m_columnHeader`

### Additional Inherited Members

### Public Types inherited from Digikam::AbstractAlbumModel

- enum `AlbumDataRole` {  
`AlbumTitleRole` = Qt::UserRole , `AlbumTypeRole` = Qt::UserRole + 1 , `AlbumPointerRole` = Qt::UserRole + 2  
, `AlbumIdRole` = Qt::UserRole + 3 ,  
`AlbumGlobalIdRole` = Qt::UserRole + 4 , `AlbumSortRole` = Qt::UserRole + 5 }
- enum `RootAlbumBehavior` { `IncludeRootAlbum` , `IgnoreRootAlbum` }  
*AbstractAlbumModel is the abstract base class for all models that present Album objects as managed by AlbumManager.*





**Public Member Functions**

- [AbstractWidgetDelegateOverlay](#) (QObject \*const parent)  
*This class provides functionality for using a widget in an overlay.*
- void [setActive](#) (bool active) override  
*If active is true, this will call [createWidget\(\)](#), initialize the widget for use, and setup connections for the virtual slots.*

**Public Member Functions inherited from [Digikam::ItemDelegateOverlay](#)**

- [ItemDelegateOverlay](#) (QObject \*const parent=nullptr)
- virtual bool [acceptsDelegate](#) (QAbstractItemDelegate \*) const
- QAbstractItemDelegate \* [delegate](#) () const
- virtual void [mouseMoved](#) (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)  
*Only these two methods are implemented as virtual methods.*
- virtual void [paint](#) (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index)
- void [setDelegate](#) (QAbstractItemDelegate \*delegate)
- void [setView](#) (QAbstractItemView \*view)
- QAbstractItemView \* [view](#) () const

**Protected Slots**

- virtual void [slotEntered](#) (const QModelIndex &index)  
*Default implementation shows the widget iff the index is valid and checkIndex returns true.*
- virtual void [slotLayoutChanged](#) ()
- virtual void [slotReset](#) ()  
*Default implementations of these three slots call [hide\(\)](#)*
- virtual void [slotRowsRemoved](#) (const QModelIndex &parent, int start, int end)
- virtual void [slotViewportEntered](#) ()

**Protected Slots inherited from [Digikam::ItemDelegateOverlay](#)**

- virtual void [visualChange](#) ()  
*Called when any change from the delegate occurs - when the overlay is installed, when size hints, styles or fonts change.*

**Protected Member Functions**

- virtual bool [checkIndex](#) (const QModelIndex &index) const  
*Return true here if you want to show the overlay for the given index.*
- bool [checkIndexOnEnter](#) (const QModelIndex &index) const  
*Utility method called from [slotEntered](#).*
- virtual QWidget \* [createWidget](#) ()=0  
*Create your widget here.*
- bool [eventFilter](#) (QObject \*obj, QEvent \*event) override
- virtual void [hide](#) ()  
*Called when the widget shall be hidden (mouse cursor left index, viewport, uninstalled etc.).*
- virtual QString [notifyMultipleMessage](#) (const QModelIndex &, int number)
- QWidget \* [parentWidget](#) () const  
*Returns the widget to be used as parent for your widget created in [createWidget\(\)](#)*
- virtual void [viewportLeaveEvent](#) (QObject \*obj, QEvent \*event)  
*Called when a QEvent::Leave of the viewport is received.*
- virtual void [widgetEnterEvent](#) ()  
*Called when a QEvent::Enter resp.*
- void [widgetEnterNotifyMultiple](#) (const QModelIndex &index)  
*A sample implementation for above methods.*
- virtual void [widgetLeaveEvent](#) ()
- void [widgetLeaveNotifyMultiple](#) ()

## Protected Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- `QList< QModelIndex >` **affectedIndexes** (const QModelIndex &index) const
- `bool` **affectsMultiple** (const QModelIndex &index) const  
*For the context that an overlay can affect multiple items: Assuming the currently overlaid index is given.*
- `int` **numberOfAffectedIndexes** (const QModelIndex &index) const
- `bool` **viewHasMultiSelection** () const  
*Utility method.*

## Protected Attributes

- `bool` **m\_mouseButtonPressedOnWidget** = false
- `QWidget *` **m\_widget** = nullptr

## Protected Attributes inherited from [Digikam::ItemDelegateOverlay](#)

- `QAbstractItemDelegate *` **m\_delegate** = nullptr
- `QAbstractItemView *` **m\_view** = nullptr

## Additional Inherited Members

## Signals inherited from [Digikam::ItemDelegateOverlay](#)

- `void` **hideNotification** ()
- `void` **requestNotification** (const QModelIndex &index, const QString &message)
- `void` **update** (const QModelIndex &index)

## 9.18.1 Constructor & Destructor Documentation

### 9.18.1.1 AbstractWidgetDelegateOverlay()

```
Digikam::AbstractWidgetDelegateOverlay::AbstractWidgetDelegateOverlay (
    QObject *const parent ) [explicit]
```

You must reimplement at least `createWidget` to return your widget. Per default it will be shown when the cursor enters an index and hidden when left. Reimplement `slotEntered()` and `mouseMove()` for more fine grained control.

## 9.18.2 Member Function Documentation

### 9.18.2.1 checkIndex()

```
bool Digikam::AbstractWidgetDelegateOverlay::checkIndex (
    const QModelIndex & index ) const [protected], [virtual]
```

The default implementation returns true.

Reimplemented in [Digikam::AssignNameOverlay](#), [Digikam::FaceRejectionOverlay](#), [Digikam::GroupIndicatorOverlay](#), [Digikam::ItemCoordinatesOverlay](#), [Digikam::ItemFullScreenOverlay](#), [Digikam::ItemRotateOverlay](#), [Digikam::ShowHideVersionsOverlay](#), [Digikam::ActionVersionsOverlay](#), [ShowFoto::ShowfotoCoordinatesOverlay](#), [Digikam::ImportCoordinatesOverlay](#), [Digikam::ImportLockOverlay](#), [Digikam::ImportDownloadOverlay](#), and [Digikam::ImportRotateOverlay](#).

### 9.18.2.2 createWidget()

```
virtual QWidget * Digikam::AbstractWidgetDelegateOverlay::createWidget ( ) [protected], [pure virtual]
```

When creating the object, pass [parentWidget\(\)](#) as parent widget. Ownership of the object is passed. It will be deleted in [setActive\(false\)](#).

Implemented in [Digikam::AssignNameOverlay](#), [Digikam::GroupIndicatorOverlay](#), [Digikam::ItemCoordinatesOverlay](#), [Digikam::ItemRatingOverlay](#), [Digikam::TagsLineEditOverlay](#), [Digikam::HoverButtonDelegateOverlay](#), [ShowFoto::ShowfotoCoordinatesOverlay](#), [Digikam::ImportCoordinatesOverlay](#), [Digikam::ImportLockOverlay](#), [Digikam::ImportDownloadOverlay](#), and [Digikam::ImportRatingOverlay](#).

### 9.18.2.3 hide()

```
void Digikam::AbstractWidgetDelegateOverlay::hide ( ) [protected], [virtual]
```

Default implementation [hide\(\)](#)s [m\\_widget](#).

Reimplemented in [Digikam::ItemRatingOverlay](#), [Digikam::TagsLineEditOverlay](#), [Digikam::PersistentWidgetDelegateOverlay](#), and [Digikam::ImportRatingOverlay](#).

### 9.18.2.4 setActive()

```
void Digikam::AbstractWidgetDelegateOverlay::setActive (
    bool active ) [override], [virtual]
```

If active is false, this will delete the widget and disconnect all signal from model and view to this object (!)

Reimplemented from [Digikam::ItemDelegateOverlay](#).

Reimplemented in [Digikam::FaceRejectionOverlay](#), [Digikam::ItemCoordinatesOverlay](#), [Digikam::ItemFullScreenOverlay](#), [Digikam::ItemRotateOverlay](#), [Digikam::ItemSelectionOverlay](#), [Digikam::ShowHideVersionsOverlay](#), [Digikam::ActionVersionsOverlay](#), [Digikam::HoverButtonDelegateOverlay](#), [Digikam::PersistentWidgetDelegateOverlay](#), [ShowFoto::ShowfotoCoordinatesOverlay](#), [Digikam::ImportCoordinatesOverlay](#), [Digikam::ImportLockOverlay](#), [Digikam::ImportDownloadOverlay](#), [Digikam::ImportRotateOverlay](#), [Digikam::AssignNameOverlay](#), [Digikam::GroupIndicatorOverlay](#), [Digikam::ItemRatingOverlay](#), [Digikam::TagsLineEditOverlay](#), and [Digikam::ImportRatingOverlay](#).

### 9.18.2.5 slotEntered

```
void Digikam::AbstractWidgetDelegateOverlay::slotEntered (
    const QModelIndex & index ) [protected], [virtual], [slot]
```

Reimplemented in [Digikam::GroupIndicatorOverlay](#), [Digikam::ItemCoordinatesOverlay](#), [Digikam::ItemRatingOverlay](#), [Digikam::TagsLineEditOverlay](#), [Digikam::PersistentWidgetDelegateOverlay](#), [ShowFoto::ShowfotoCoordinatesOverlay](#), [Digikam::ImportCoordinatesOverlay](#), [Digikam::ImportLockOverlay](#), [Digikam::ImportDownloadOverlay](#), and [Digikam::ImportRatingOverlay](#).

### 9.18.2.6 slotReset

```
void Digikam::AbstractWidgetDelegateOverlay::slotReset ( ) [protected], [virtual], [slot]
```

Reimplemented in [Digikam::PersistentWidgetDelegateOverlay](#).

### 9.18.2.7 viewportLeaveEvent()

```
void Digikam::AbstractWidgetDelegateOverlay::viewportLeaveEvent (
    QObject * obj,
    QEvent * event ) [protected], [virtual]
```

The default implementation [hide\(\)](#)s.

Reimplemented in [Digikam::AssignNameOverlay](#), and [Digikam::PersistentWidgetDelegateOverlay](#).

### 9.18.2.8 widgetEnterEvent()

```
void Digikam::AbstractWidgetDelegateOverlay::widgetEnterEvent ( ) [protected], [virtual]
```

QEvent::Leave event for the widget is received. The default implementation does nothing.

Reimplemented in [Digikam::AssignNameOverlay](#), [Digikam::FaceRejectionOverlay](#), [Digikam::ItemFullScreenOverlay](#), [Digikam::ItemRatingOverlay](#), [Digikam::ItemRotateOverlay](#), [Digikam::ImportRatingOverlay](#), and [Digikam::ImportRotateOverlay](#).

## 9.19 Digikam::ActionCategorizedView Class Reference

Inheritance diagram for Digikam::ActionCategorizedView:



### Public Member Functions

- **ActionCategorizedView** (`QWidget *const parent=nullptr, bool autoScroll=false`)
- void **adjustGridSize** ()
- void **setupIconMode** ()

## Public Member Functions inherited from [Digikam::DCategorizedView](#)

- **DCategorizedView** (QWidget \*const parent=nullptr)
- virtual QModelIndexList [categorizedIndexesIn](#) (const QRect &rect) const  
*This method will return all indexes whose visual rect intersects rect.*
- virtual QModelIndex [categoryAt](#) (const QPoint &point) const  
*This method will return the first index of the category in the region of which point is found.*
- [DCategoryDrawer](#) \* **categoryDrawer** () const
- virtual QItemSelectionRange [categoryRange](#) (const QModelIndex &index) const  
*This method returns the range of indexes contained in the category in which index is sorted.*
- virtual QRect [categoryVisualRect](#) (const QModelIndex &index) const  
*This method will return the visual rect of the header of the category in which index is sorted.*
- QModelIndex **indexAt** (const QPoint &point) const override
- void **setCategoryDrawer** ([DCategoryDrawer](#) \*categoryDrawer)
- void [setDrawDraggedItems](#) (bool drawDraggedItems)  
*Switch on drawing of dragged items.*
- void **setGridSize** (const QSize &size)
- void **setModel** (QAbstractItemModel \*model) override
- QRect **visualRect** (const QModelIndex &index) const override

## Protected Member Functions

- void **autoScroll** (float relativePos, QScrollBar \*scrollBar, QPropertyAnimation \*animation)
- int **autoScrollDuration** (float relativeDifference, QPropertyAnimation \*animation)
- void **leaveEvent** (QEvent \*e) override
- void **mouseMoveEvent** (QMouseEvent \*e) override

## Protected Member Functions inherited from [Digikam::DCategorizedView](#)

- void **dragLeaveEvent** (QDragLeaveEvent \*event) override
- void **dragMoveEvent** (QDragMoveEvent \*event) override
- void **dropEvent** (QDropEvent \*event) override
- void **leaveEvent** (QEvent \*event) override
- void **mouseMoveEvent** (QMouseEvent \*event) override
- void **mousePressEvent** (QMouseEvent \*event) override
- void **mouseReleaseEvent** (QMouseEvent \*event) override
- QModelIndex **moveCursor** (CursorAction cursorAction, Qt::KeyboardModifiers modifiers) override
- void **paintEvent** (QPaintEvent \*event) override
- void **resizeEvent** (QResizeEvent \*event) override
- void **setSelection** (const QRect &rect, QItemSelectionModel::SelectionFlags flags) override
- void **startDrag** (Qt::DropActions supportedActions) override

## Protected Attributes

- bool **m\_autoScroll** = false
- QPropertyAnimation \* **m\_horizontalScrollAnimation** = nullptr
- QPropertyAnimation \* **m\_verticalScrollAnimation** = nullptr

### Additional Inherited Members

### Public Slots inherited from [Digikam::DCategorizedView](#)

- void **reset** () override

### Protected Slots inherited from [Digikam::DCategorizedView](#)

- void **currentChanged** (const QModelIndex &current, const QModelIndex &previous) override
- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- virtual void **rowsInsertedArtificial** (const QModelIndex &parent, int start, int end)
- virtual void **rowsRemoved** (const QModelIndex &parent, int start, int end)
- virtual void **slotLayoutChanged** ()
- void **updateGeometries** () override

## 9.20 Digikam::ActionData Class Reference

### Public Types

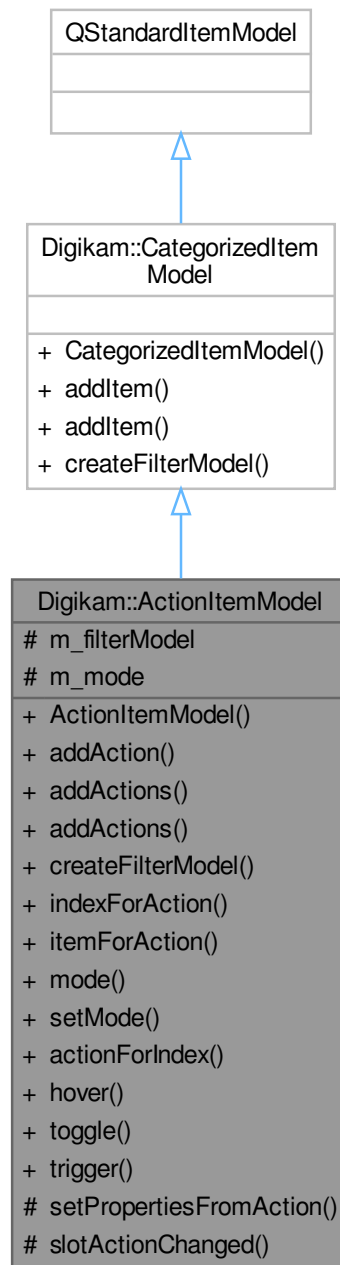
- enum **ActionStatus** {  
    **None** = 0 , **BatchStarted** , **BatchDone** , **BatchFailed** ,  
    **BatchSkipped** , **BatchCanceled** , **TaskDone** , **TaskFailed** ,  
    **TaskCanceled** }

### Public Attributes

- QUrl **destUrl**
- QUrl **fileUrl**
- QString **message**
- bool **noWrite** = false
- ActionStatus **status** = None

## 9.21 Digikam::ActionItemModel Class Reference

Inheritance diagram for Digikam::ActionItemModel:



### Public Types

- enum **ExtraRoles** { **ItemActionRole** = Qt::UserRole + 10 }
- enum **MenuCategoryFlag** { **ToplevelMenuCategory** = 1 << 0 , **ParentMenuCategory** = 1 << 1 , **SortCategoriesAlphabetically** = 1 << 10 , **SortCategoriesByInsertionOrder** = 1 << 11 }
- typedef QFlags< **MenuCategoryFlag** > **MenuCategoryMode**



## Public Types inherited from [Digikam::CategorizedItemModel](#)

- enum [ExtraRoles](#) { [ItemOrderRole](#) = Qt::UserRole + 1 }

## Public Slots

- void [hover](#) (const QModelIndex &index)  
*These three slots will cause the slots of the referred action to be called.*
- void [toggle](#) (const QModelIndex &index)
- void [trigger](#) (const QModelIndex &index)

## Public Member Functions

- [ActionItemModel](#) (QObject \*const parent=nullptr)  
*This class is a [CategorizedItemModel](#) based on QActions, taking an action's text and icon for display and decoration.*
- QStandardItem \* [addAction](#) (QAction \*action, const QString &category, const QVariant &category↔  
Sorting=QVariant())
- void [addActions](#) (QWidget \*widget)
- void [addActions](#) (QWidget \*widget, const QList< QAction \* > &actionWhiteList)
- [DCategorizedSortFilterProxyModel](#) \* [createFilterModel](#) () override
- QModelIndex [indexForAction](#) (QAction \*action) const
- QStandardItem \* [itemForAction](#) (QAction \*action) const  
*Returns the action for the given index.*
- MenuCategoryMode [mode](#) () const
- void [setMode](#) (MenuCategoryMode mode)

## Public Member Functions inherited from [Digikam::CategorizedItemModel](#)

- [CategorizedItemModel](#) (QObject \*const parent=nullptr)
- QStandardItem \* [addItem](#) (const QString &text, const QIcon &decoration, const QVariant &category, const  
QVariant &categorySorting=QVariant())
- QStandardItem \* [addItem](#) (const QString &text, const QVariant &category, const QVariant &category↔  
Sorting=QVariant())

## Static Public Member Functions

- static QAction \* [actionForIndex](#) (const QModelIndex &index)  
*Returns the action for the given index.*

## Protected Slots

- void [slotActionChanged](#) ()

## Protected Member Functions

- void [setPropertiesFromAction](#) (QStandardItem \*item, QAction \*action)

## Protected Attributes

- [DCategorizedSortFilterProxyModel](#) \* [m\\_filterModel](#) = nullptr
- MenuCategoryMode [m\\_mode](#) = MenuCategoryMode([ToplevelMenuCategory](#) | [SortCategoriesAlphabetically](#))

## 9.21.1 Member Enumeration Documentation

### 9.21.1.1 MenuCategoryFlag

```
enum Digikam::ActionItemModel::MenuCategoryFlag
```

## Enumerator

ToplevelMenuCategory	The toplevel menu's text is used as category.
ParentMenuCategory	If the action is in a submenu, this menu's text is taken as category.
SortCategoriesAlphabetically	Sort categories alphabetically by category name.
SortCategoriesByInsertionOrder	Sort categories by the order they are added (found in the scanned menu)

## 9.21.2 Constructor & Destructor Documentation

### 9.21.2.1 ActionItemModel()

```
Digikam::ActionItemModel::ActionItemModel (
    QObject *const parent = nullptr ) [explicit]
```

It is possible to retrieve an action for an index, and to call the action's slots from a given index.

## 9.21.3 Member Function Documentation

### 9.21.3.1 actionForIndex()

```
QAction * Digikam::ActionItemModel::actionForIndex (
    const QModelIndex & index ) [static]
```

The method can also be used for indices from proxy models.

### 9.21.3.2 createFilterModel()

```
DCategorizedSortFilterProxyModel * Digikam::ActionItemModel::createFilterModel ( ) [override],
[virtual]
```

Reimplemented from [Digikam::CategorizedItemModel](#).

### 9.21.3.3 hover

```
void Digikam::ActionItemModel::hover (
    const QModelIndex & index ) [slot]
```

Connect here for example a view's signals. Note that you can also pass indices from proxy models.

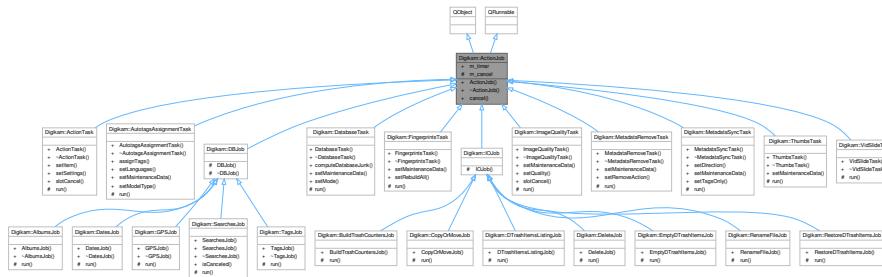
### 9.21.3.4 itemForAction()

```
QStandardItem * Digikam::ActionItemModel::itemForAction (
    QAction * action ) const
```

Note: these methods perform O(n).

## 9.22 Digikam::ActionJob Class Reference

Inheritance diagram for Digikam::ActionJob:



### Public Slots

- void **cancel** ()  
Call this method to cancel job.

### Signals

- void **signalDone** ()  
Use this signal in your implementation to inform [ActionThreadBase](#) manager the job is done.
- void **signalProgress** (int)  
Use this signal in your implementation to inform [ActionThreadBase](#) manager the job progress.
- void **signalStarted** ()  
Use this signal in your implementation to inform [ActionThreadBase](#) manager that job is started.

### Public Member Functions

- **ActionJob** (QObject \*const parent=nullptr)  
Constructor which delegate deletion of [QRunnable](#) instance to [ActionThreadBase](#), not [QThreadPool](#).
- **~ActionJob** () override  
Re-implement destructor in you implementation.

### Public Attributes

- QElapsedTimer **m\_timer**  
Timer to determine the running time of the job.

### Protected Attributes

- bool **m\_cancel** = false  
You can use this boolean in your implementation to know if job must be canceled.

## 9.22.1 Constructor & Destructor Documentation

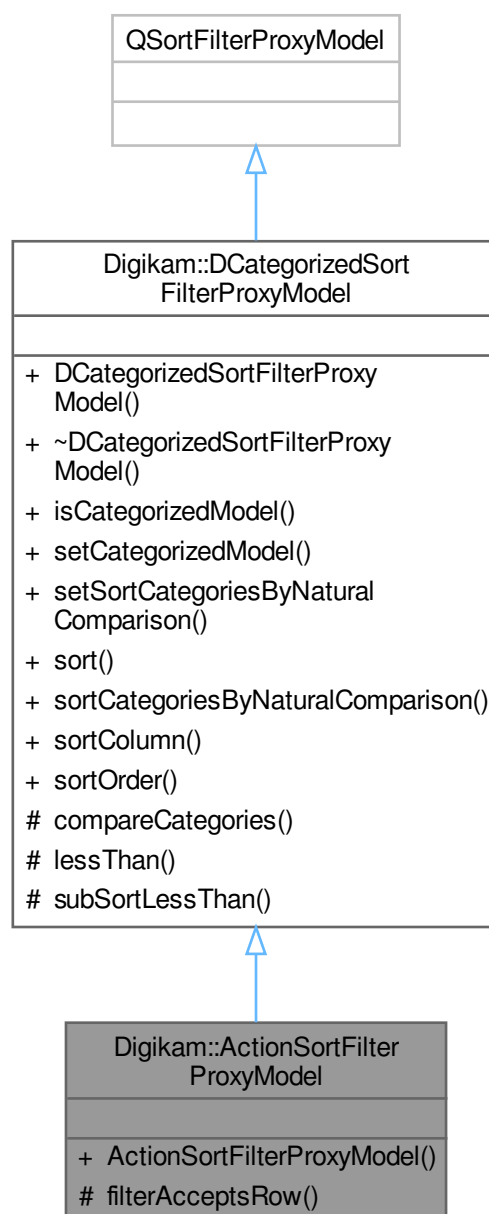
### 9.22.1.1 ~ActionJob()

Digikam::ActionJob::~~ActionJob ( ) [override]

Don't forget to cancel job.

## 9.23 Digikam::ActionSortFilterProxyModel Class Reference

Inheritance diagram for Digikam::ActionSortFilterProxyModel:



**Public Member Functions**

- **ActionSortFilterProxyModel** (QObject \*const parent=nullptr)

**Public Member Functions inherited from [Digikam::DCategorizedSortFilterProxyModel](#)**

- **DCategorizedSortFilterProxyModel** (QObject \*const parent=nullptr)
- bool **isCategorizedModel** () const
- void **setCategorizedModel** (bool categorizedModel)
 

*Enables or disables the categorization feature.*
- void **setSortCategoriesByNaturalComparison** (bool **sortCategoriesByNaturalComparison**)
 

*Set if the sorting using CategorySortRole will use a natural comparison in the case that strings were returned.*
- void **sort** (int column, Qt::SortOrder order=Qt::AscendingOrder) override
 

*Overridden from QSortFilterProxyModel.*
- bool **sortCategoriesByNaturalComparison** () const
- int **sortColumn** () const
- Qt::SortOrder **sortOrder** () const

**Protected Member Functions**

- bool **filterAcceptsRow** (int source\_row, const QModelIndex &source\_parent) const override

**Protected Member Functions inherited from [Digikam::DCategorizedSortFilterProxyModel](#)**

- virtual int **compareCategories** (const QModelIndex &left, const QModelIndex &right) const
 

*This method compares the category of the left index with the category of the right index.*
- bool **lessThan** (const QModelIndex &left, const QModelIndex &right) const override
 

*Overridden from QSortFilterProxyModel.*
- virtual bool **subSortLessThan** (const QModelIndex &left, const QModelIndex &right) const
 

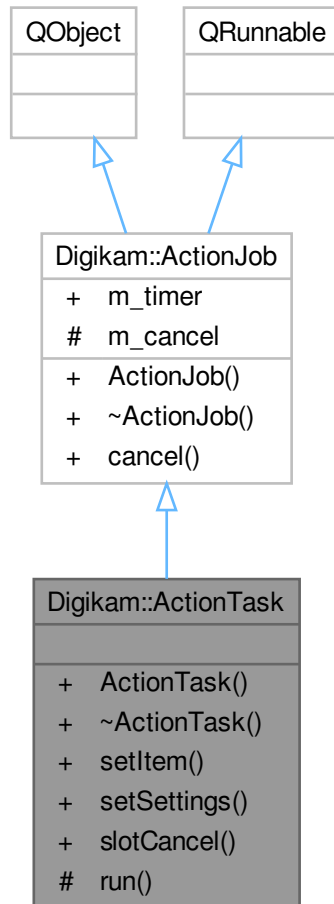
*This method has a similar purpose as lessThan() has on QSortFilterProxyModel.*

**Additional Inherited Members****Public Types inherited from [Digikam::DCategorizedSortFilterProxyModel](#)**

- enum **AdditionalRoles** { **CategoryDisplayRole** = 0x17CE990A , **CategorySortRole** = 0x27857E60 }

## 9.24 Digikam::ActionTask Class Reference

Inheritance diagram for Digikam::ActionTask:



### Public Slots

- void `slotCancel` ()

### Public Slots inherited from [Digikam::ActionJob](#)

- void `cancel` ()  
*Call this method to cancel job.*

### Signals

- void `signalFinished` (const [Digikam::ActionData](#) &ad)
- void `signalStarting` (const [Digikam::ActionData](#) &ad)

## Signals inherited from [Digikam::ActionJob](#)

- void **signalDone** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job is done.*
- void **signalProgress** (int)  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job progress.*
- void **signalStarted** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager that job is started.*

## Public Member Functions

- void **setItem** (const [AssignedBatchTools](#) &tools)
- void **setSettings** (const [QueueSettings](#) &settings)

## Public Member Functions inherited from [Digikam::ActionJob](#)

- **ActionJob** (QObject \*const parent=nullptr)  
*Constructor which delegate deletion of QRunnable instance to [ActionThreadBase](#), not QThreadPool.*
- **~ActionJob** () override  
*Re-implement destructor in you implementation.*

## Protected Member Functions

- void **run** () override

## Additional Inherited Members

## Public Attributes inherited from [Digikam::ActionJob](#)

- QElapsedTimer **m\_timer**  
*Timer to determine the running time of the job.*

## Protected Attributes inherited from [Digikam::ActionJob](#)

- bool **m\_cancel** = false  
*You can use this boolean in your implementation to know if job must be canceled.*

## 9.25 Digikam::ActionThread Class Reference

Inheritance diagram for Digikam::ActionThread:



### Signals

- void **signalCancelActionTask** ()  
*Signal to emit to sub-tasks to cancel processing.*
- void **signalFinished** (const [Digikam::ActionData](#) &ad)  
*Emit when an item from a queue have been processed.*



- void **signalQueueProcessed** ()  
*Emit when a queue have been fully processed (all items from queue are finished).*
- void **signalStarting** (const [Digikam::ActionData](#) &ad)  
*Emit when an item from a queue start to be processed.*

### Public Member Functions

- **ActionThread** (QObject \*const parent)
- void **cancel** ()
- void **processQueueItems** (const QList< [AssignedBatchTools](#) > &items)
- void **setSettings** (const [QueueSettings](#) &settings)

### Public Member Functions inherited from [Digikam::ActionThreadBase](#)

- **ActionThreadBase** (QObject \*const parent=nullptr)
- void **cancel** (bool isCancel=true)  
*Cancel processing of current jobs under progress.*
- int **maximumNumberOfThreads** () const  
*Return the maximum number of threads used to parallelize collection of job processing.*
- void **setDefaultMaximumNumberOfThreads** ()  
*Reset maximum number of threads used to parallelize collection of job processing to max core detected on computer.*
- void **setMaximumNumberOfThreads** (int n)  
*Adjust maximum number of threads used to parallelize collection of job processing.*

### Additional Inherited Members

### Protected Slots inherited from [Digikam::ActionThreadBase](#)

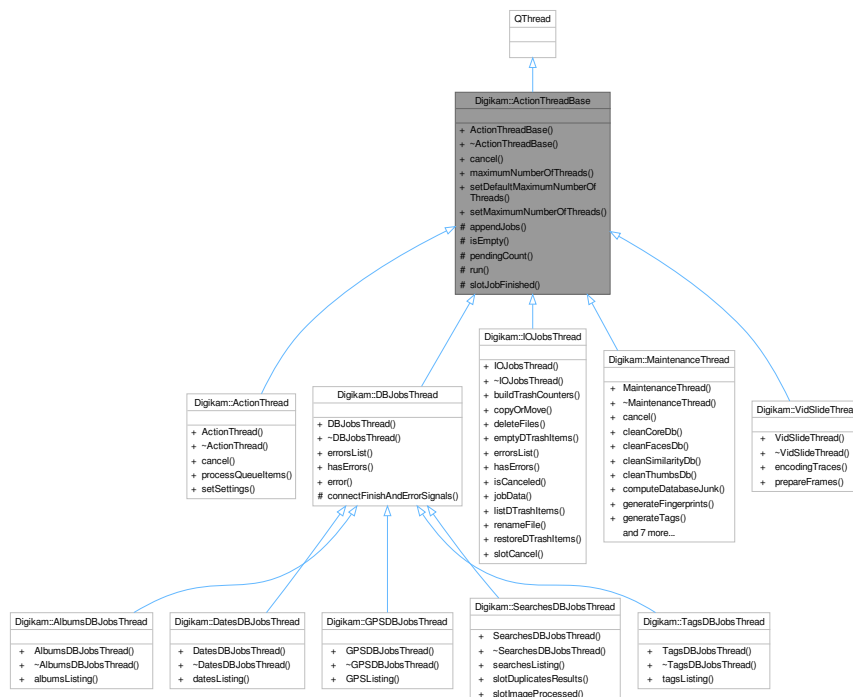
- void **slotJobFinished** ()

### Protected Member Functions inherited from [Digikam::ActionThreadBase](#)

- void **appendJobs** (const [ActionJobCollection](#) &jobs)  
*Append a collection of jobs to process into QThreadPool.*
- bool **isEmpty** () const  
*Return true if list of pending jobs to process is empty.*
- int **pendingCount** () const  
*Return the number of pending jobs to process.*
- void **run** () override  
*Main thread loop used to process jobs in todo list.*

## 9.26 Digikam::ActionThreadBase Class Reference

Inheritance diagram for Digikam::ActionThreadBase:



### Public Member Functions

- **ActionThreadBase** (QObject \*const parent=nullptr)
- void **cancel** (bool isCancel=true)  
*Cancel processing of current jobs under progress.*
- int **maximumNumberOfThreads** () const  
*Return the maximum number of threads used to parallelize collection of job processing.*
- void **setDefaultMaximumNumberOfThreads** ()  
*Reset maximum number of threads used to parallelize collection of job processing to max core detected on computer.*
- void **setMaximumNumberOfThreads** (int n)  
*Adjust maximum number of threads used to parallelize collection of job processing.*

### Protected Slots

- void **slotJobFinished** ()

### Protected Member Functions

- void **appendJobs** (const ActionJobCollection &jobs)  
*Append a collection of jobs to process into QThreadPool.*
- bool **isEmpty** () const  
*Return true if list of pending jobs to process is empty.*
- int **pendingCount** () const  
*Return the number of pending jobs to process.*
- void **run** () override  
*Main thread loop used to process jobs in todo list.*

## 9.26.1 Member Function Documentation

### 9.26.1.1 appendJobs()

```
void Digikam::ActionThreadBase::appendJobs (
    const ActionJobCollection & jobs ) [protected]
```

Jobs are add to pending lists and will be deleted by [ActionThreadBase](#), not QThreadPool.

### 9.26.1.2 setDefaultMaximumNumberOfThreads()

```
void Digikam::ActionThreadBase::setDefaultMaximumNumberOfThreads ( )
```

This method is called in constructor.

## 9.27 Digikam::ActionVersionsOverlay Class Reference

Inheritance diagram for Digikam::ActionVersionsOverlay:



### Signals

- void **activated** (const [ItemInfo](#) &info)

## Signals inherited from [Digikam::ItemDelegateOverlay](#)

- void **hideNotification** ()
- void **requestNotification** (const QModelIndex &index, const QString &message)
- void **update** (const QModelIndex &index)

## Public Member Functions

- **ActionVersionsOverlay** (QObject \*const parent, const QIcon &icon, const QString &text, const QString &tip=QString())
- void **setActive** (bool active) override  
*If active is true, this will call [createWidget\(\)](#), initialize the widget for use, and setup connections for the virtual slots.*
- void **setReferenceModel** (const [ItemModel](#) \*model)

## Public Member Functions inherited from [Digikam::HoverButtonDelegateOverlay](#)

- **HoverButtonDelegateOverlay** (QObject \*const parent)
- [ItemViewHoverButton](#) \* **button** () const
- void **setActive** (bool active) override  
*Will call [createButton\(\)](#).*

## Public Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- [AbstractWidgetDelegateOverlay](#) (QObject \*const parent)  
*This class provides functionality for using a widget in an overlay.*

## Public Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- **ItemDelegateOverlay** (QObject \*const parent=nullptr)
- virtual bool **acceptsDelegate** (QAbstractItemDelegate \*) const
- QAbstractItemDelegate \* **delegate** () const
- virtual void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)  
*Only these two methods are implemented as virtual methods.*
- virtual void **paint** (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index)
- void **setDelegate** (QAbstractItemDelegate \*delegate)
- void **setView** (QAbstractItemView \*view)
- QAbstractItemView \* **view** () const

## Protected Slots

- void **slotClicked** (bool checked)

## Protected Slots inherited from [Digikam::HoverButtonDelegateOverlay](#)

- void **slotEntered** (const QModelIndex &index) override
- void **slotReset** () override

## Protected Slots inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- virtual void [slotEntered](#) (const QModelIndex &index)  
*Default implementation shows the widget iff the index is valid and checkIndex returns true.*
- virtual void [slotLayoutChanged](#) ()
- virtual void [slotReset](#) ()  
*Default implementations of these three slots call [hide\(\)](#)*
- virtual void [slotRowsRemoved](#) (const QModelIndex &parent, int start, int end)
- virtual void [slotViewportEntered](#) ()

## Protected Slots inherited from [Digikam::ItemDelegateOverlay](#)

### Protected Member Functions

- Button \* [button](#) () const
- bool [checkIndex](#) (const QModelIndex &index) const override  
*Return true here if you want to show the overlay for the given index.*
- [ItemViewHoverButton](#) \* [createButton](#) () override  
*Create your widget here.*
- void [updateButton](#) (const QModelIndex &index) override  
*Called when a new index is entered.*

## Protected Member Functions inherited from [Digikam::HoverButtonDelegateOverlay](#)

- QWidget \* [createWidget](#) () override  
*Create your widget here.*
- void [visualChange](#) () override  
*Called when any change from the delegate occurs - when the overlay is installed, when size hints, styles or fonts change.*

## Protected Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- bool [checkIndexOnEnter](#) (const QModelIndex &index) const  
*Utility method called from slotEntered.*
- bool [eventFilter](#) (QObject \*obj, QEvent \*event) override
- virtual void [hide](#) ()  
*Called when the widget shall be hidden (mouse cursor left index, viewport, uninstalled etc.).*
- virtual QString [notifyMultipleMessage](#) (const QModelIndex &, int number)
- QWidget \* [parentWidget](#) () const  
*Returns the widget to be used as parent for your widget created in [createWidget\(\)](#)*
- virtual void [viewportLeaveEvent](#) (QObject \*obj, QEvent \*event)  
*Called when a QEvent::Leave of the viewport is received.*
- virtual void [widgetEnterEvent](#) ()  
*Called when a QEvent::Enter resp.*
- void [widgetEnterNotifyMultiple](#) (const QModelIndex &index)  
*A sample implementation for above methods.*
- virtual void [widgetLeaveEvent](#) ()
- void [widgetLeaveNotifyMultiple](#) ()

## Protected Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- `QList< QModelIndex > affectedIndexes` (const QModelIndex &index) const
- bool `affectsMultiple` (const QModelIndex &index) const  
*For the context that an overlay can affect multiple items: Assuming the currently overlaid index is given.*
- int `numberOfAffectedIndexes` (const QModelIndex &index) const
- bool `viewHasMultiSelection` () const  
*Utility method.*

## Protected Attributes

- QIcon `m_icon`
- const `ItemModel` \* `m_referenceModel` = nullptr
- QString `m_text`
- QString `m_tip`

## Protected Attributes inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- bool `m_mouseButtonPressedOnWidget` = false
- QWidget \* `m_widget` = nullptr

## Protected Attributes inherited from [Digikam::ItemDelegateOverlay](#)

- QAbstractItemDelegate \* `m_delegate` = nullptr
- QAbstractItemView \* `m_view` = nullptr

## 9.27.1 Member Function Documentation

### 9.27.1.1 checkIndex()

```
bool Digikam::ActionVersionsOverlay::checkIndex (
    const QModelIndex & index ) const [override], [protected], [virtual]
```

The default implementation returns true.

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.27.1.2 createButton()

```
ItemViewHoverButton * Digikam::ActionVersionsOverlay::createButton ( ) [override], [protected],
[virtual]
```

Pass view() as parent.

Implements [Digikam::HoverButtonDelegateOverlay](#).

### 9.27.1.3 setActive()

```
void Digikam::ActionVersionsOverlay::setActive (
    bool active ) [override], [virtual]
```

If active is false, this will delete the widget and disconnect all signal from model and view to this object (!)

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.27.1.4 updateButton()

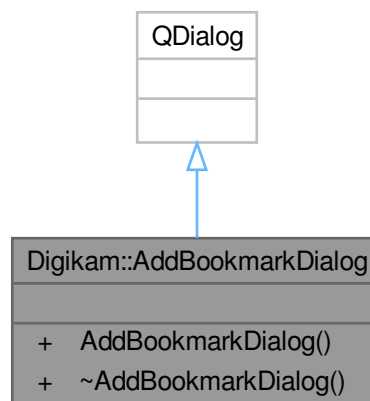
```
void Digikam::ActionVersionsOverlay::updateButton (
    const QModelIndex & index ) [override], [protected], [virtual]
```

Reposition your button here, adjust and store state.

Implements [Digikam::HoverButtonDelegateOverlay](#).

## 9.28 Digikam::AddBookmarkDialog Class Reference

Inheritance diagram for Digikam::AddBookmarkDialog:



### Public Member Functions

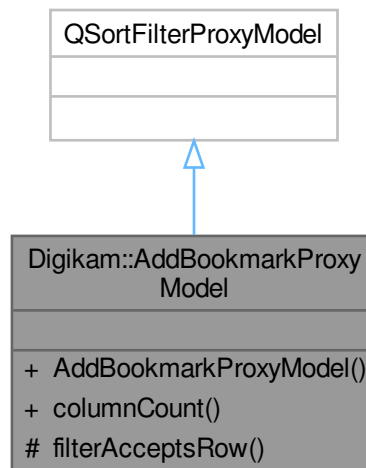
- **AddBookmarkDialog** (const QString &url, const QString &title, QWidget \*const parent=nullptr, [BookmarksManager](#) \*const mngr=nullptr)



## 9.29 Digikam::AddBookmarkProxyModel Class Reference

Proxy model that filters out the bookmarks so only the folders are left behind.

Inheritance diagram for Digikam::AddBookmarkProxyModel:



### Public Member Functions

- **AddBookmarkProxyModel** (QObject \*const parent=nullptr)
- int **columnCount** (const QModelIndex &parent=QModelIndex()) const override

### Protected Member Functions

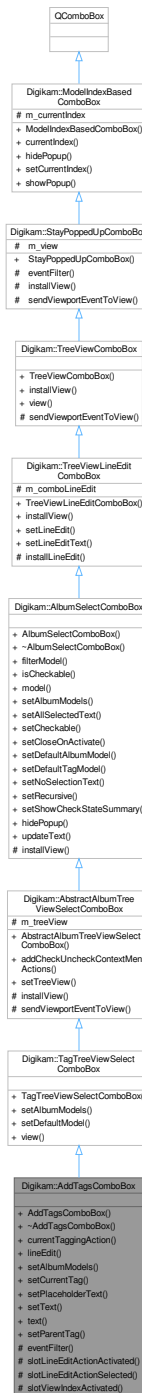
- bool **filterAcceptsRow** (int srow, const QModelIndex &sparent) const override

### 9.29.1 Detailed Description

Used in the add bookmark dialog combobox.

## 9.30 Digikam::AddTagsComboBox Class Reference

Inheritance diagram for Digikam::AddTagsComboBox:



### Public Slots

- void **setParentTag** (TAlbum \*const album)

*Set a parent tag for suggesting a parent tag for a new tag, and a default action.*

## Public Slots inherited from [Digikam::AlbumSelectComboBox](#)

- void **hidePopup** () override
- virtual void **updateText** ()  
*Updates the text describing the selection ("3 Albums selected").*

## Signals

- void **taggingActionActivated** (const [TaggingAction](#) &action)  
*Emitted when the user activates an action (typically, by pressing return)*
- void **taggingActionSelected** (const [TaggingAction](#) &action)  
*Emitted when an action is selected, but not explicitly activated.*

## Public Member Functions

- **AddTagsComboBox** (QWidget \*const parent=nullptr)
- [TaggingAction](#) **currentTaggingAction** ()  
*Returns the currently set tagging action.*
- [AddTagsLineEdit](#) \* **lineEdit** () const
- void **setAlbumModels** ([TagModel](#) \*const model, [TagPropertiesFilterModel](#) \*const filteredModel=nullptr, [CheckableAlbumFilterModel](#) \*const filterModel=nullptr)  
*You must call this after construction.*
- void **setCurrentTag** ([TAlbum](#) \*const album)  
*Sets the currently selected tag.*
- void **setPlaceholderText** (const QString &message)
- void **setText** (const QString &text)
- QString **text** () const

## Public Member Functions inherited from [Digikam::TagTreeViewSelectComboBox](#)

- **TagTreeViewSelectComboBox** (QWidget \*const parent=nullptr)
- void **setAlbumModels** ([TagModel](#) \*model, [TagPropertiesFilterModel](#) \*filteredModel=nullptr, [CheckableAlbumFilterModel](#) \*filterModel=nullptr)
- void **setDefaultModel** ()
- [TagTreeView](#) \* **view** () const

## Public Member Functions inherited from [Digikam::AbstractAlbumTreeViewSelectComboBox](#)

- [AbstractAlbumTreeViewSelectComboBox](#) (QWidget \*const parent=nullptr)  
*Abstract class.*
- void **addCheckUncheckContextMenuActions** ()  
*Enables a context menu which contains options to check or uncheck groups of albums, given you have a checkable model.*
- void **setTreeView** ([AbstractAlbumTreeView](#) \*const treeView)  
*Set a tree view created by you instead of creating a default view (in the subclasses).*

## Public Member Functions inherited from [Digikam::AlbumSelectComboBox](#)

- [AlbumSelectComboBox](#) (QWidget \*const parent=nullptr)
- QSortFilterProxyModel \* [filterModel](#) () const  
*Return the filter model in use.*
- bool [isCheckedable](#) () const
- [AbstractCheckableAlbumModel](#) \* [model](#) () const  
*Returns the source model.*
- void [setAlbumModels](#) ([AbstractCheckableAlbumModel](#) \*model, [AlbumFilterModel](#) \*filterModel=nullptr)
- void [setAllSelectedText](#) (bool all)  
*Enable or disable the text used to describe the status when all album is selected.*
- void [setCheckable](#) (bool checkable)  
*Enable checkboxes next to the items.*
- void [setCloseOnActivate](#) (bool close)  
*Enable closing when an item was activated (clicked).*
- void [setDefaultAlbumModel](#) ()  
*Once after creation, call one of these three methods.*
- void [setDefaultTagModel](#) ()
- void [setNoSelectionText](#) (const QString &text)  
*Sets the text that is used to describe the state when no album is selected.*
- void [setRecursive](#) (bool recursive)  
*If all subalbums shall be selected when parent will be selected.*
- void [setShowCheckStateSummary](#) (bool show)  
*If the box is checkable, enable showing a resume a la "3 Albums checked" in the combo box text.*

## Public Member Functions inherited from [Digikam::TreeViewLineEditComboBox](#)

- [TreeViewLineEditComboBox](#) (QWidget \*const parent=nullptr)  
*This class provides a [TreeViewComboBox](#) with a read-only line edit.*
- void [installView](#) (QAbstractItemView \*view=nullptr) override  
*Replace the standard combo box list view with a QTreeView.*
- void [setLineEdit](#) (QLineEdit \*edit)
- void [setLineEditText](#) (const QString &text)  
*Set the text of the line edit (the text that is visible if the popup is not opened).*

## Public Member Functions inherited from [Digikam::TreeViewComboBox](#)

- [TreeViewComboBox](#) (QWidget \*parent=nullptr)  
*This class provides a [QComboBox](#) with a [QTreeView](#) instead of the usual [QListView](#).*
- QTreeView \* [view](#) () const  
*Returns the QTreeView of this class.*

## Public Member Functions inherited from [Digikam::StayPoppedUpComboBox](#)

- [StayPoppedUpComboBox](#) (QWidget \*const parent=nullptr)  
*This class provides an abstract [QComboBox](#) with a custom view (which is created by implementing subclasses) instead of the usual [QListView](#).*

## Public Member Functions inherited from [Digikam::ModelIndexBasedComboBox](#)

- [ModelIndexBasedComboBox](#) (QWidget \*const parent=nullptr)  
*QComboBox has a current index based on a single integer.*
- QModelIndex **currentIndex** () const
- void **hidePopup** () override
- void **setCurrentIndex** (const QModelIndex &index)
- void **showPopup** () override

## Protected Slots

- void **slotLineEditActionActivated** (const [TaggingAction](#) &action)
- void **slotLineEditActionSelected** (const [TaggingAction](#) &action)
- void **slotViewIndexActivated** (const QModelIndex &)

## Protected Member Functions

- bool **eventFilter** (QObject \*object, QEvent \*event) override

## Protected Member Functions inherited from [Digikam::AbstractAlbumTreeViewSelectComboBox](#)

- void **installView** (QAbstractItemView \*view=nullptr) override  
*Replace the standard combo box list view with a QTreeView.*
- void **sendViewportEventToView** (QEvent \*e) override  
*Implement in subclass: Send the given event to the viewportEvent() method of m\_view.*

## Protected Member Functions inherited from [Digikam::AlbumSelectComboBox](#)

- void **installView** (QAbstractItemView \*view=nullptr) override  
*Replace the standard combo box list view with a QTreeView.*

## Protected Member Functions inherited from [Digikam::TreeViewLineEditComboBox](#)

- virtual void **installLineEdit** ()  
*Sets a line edit.*

## Protected Member Functions inherited from [Digikam::TreeViewComboBox](#)

- void **sendViewportEventToView** (QEvent \*e) override  
*Implement in subclass: Send the given event to the viewportEvent() method of m\_view.*

## Protected Member Functions inherited from [Digikam::StayPoppedUpComboBox](#)

- bool **eventFilter** (QObject \*watched, QEvent \*event) override
- void **installView** (QAbstractItemView \*view)  
*Replace the standard combo box list view with the given view.*

## Additional Inherited Members

### Protected Attributes inherited from [Digikam::AbstractAlbumTreeViewSelectComboBox](#)

- [AbstractAlbumTreeView](#) \* `m_treeView` = nullptr

### Protected Attributes inherited from [Digikam::TreeViewLineEditComboBox](#)

- `QLineEdit` \* `m_comboLineEdit` = nullptr

### Protected Attributes inherited from [Digikam::StayPoppedUpComboBox](#)

- `QAbstractItemView` \* `m_view` = nullptr

### Protected Attributes inherited from [Digikam::ModelIndexBasedComboBox](#)

- `QPersistentModelIndex` `m_currentIndex`

## 9.30.1 Member Function Documentation

### 9.30.1.1 `currentTaggingAction()`

`TaggingAction` `Digikam::AddTagsComboBox::currentTaggingAction ( )`

This is the last action emitted by either `taggingActionActivated()` or `taggingActionSelected()`

### 9.30.1.2 `setAlbumModels()`

```
void Digikam::AddTagsComboBox::setAlbumModels (
    TagModel *const model,
    TagPropertiesFilterModel *const filteredModel = nullptr,
    CheckableAlbumFilterModel *const filterModel = nullptr )
```

If filtered/filterModel is 0, a default one is constructed

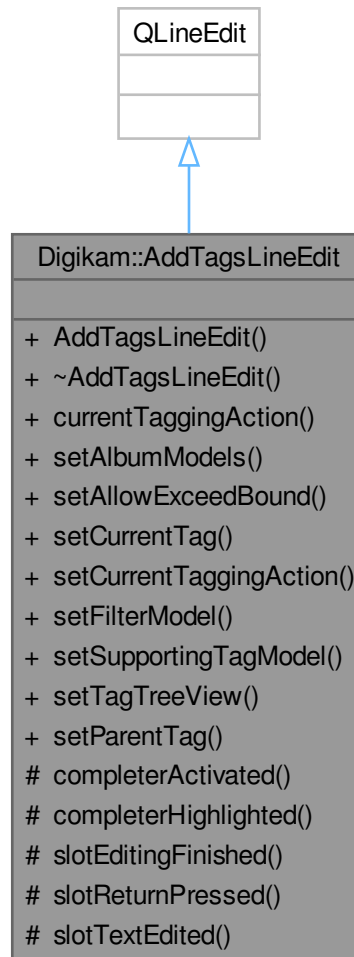
### 9.30.1.3 `taggingActionSelected`

```
void Digikam::AddTagsComboBox::taggingActionSelected (
    const TaggingAction & action ) [signal]
```

(typically by selecting an item in the tree view)

## 9.31 Digikam::AddTagsLineEdit Class Reference

Inheritance diagram for Digikam::AddTagsLineEdit:



### Public Slots

- void `setParentTag` (`Album *const album`)  
*Set a parent tag for suggesting a parent tag for a new tag, and a default action.*

### Signals

- void **taggingActionActivated** (`const TaggingAction &action`)  
*Emitted when the user activates an action (typically, by pressing return)*
- void **taggingActionFinished** ()
- void `taggingActionSelected` (`const TaggingAction &action`)  
*Emitted when an action is selected.*

## Public Member Functions

- **AddTagsLineEdit** (QWidget \*const parent=nullptr)
- **TaggingAction currentTaggingAction** () const
- void **setAlbumModels** (TagModel \*const model, TagPropertiesFilterModel \*const filteredModel, AlbumFilterModel \*const filterModel)
 

*Convenience: Will call `setSupportingTagModel()` and `setFilterModel()`*
- void **setAllowExceedBound** (bool value)
- void **setCurrentTag** (TAlbum \*const tag)
 

*Adjusts the current default tagging action to assign the given tag.*
- void **setCurrentTaggingAction** (const TaggingAction &action)
- void **setFilterModel** (AlbumFilterModel \*const model)
 

*Set a tag filter model.*
- void **setSupportingTagModel** (TagModel \*const model)
 

*Optional: set a model for additional information, like tag icons.*
- void **setTagTreeView** (TagTreeView \*const treeView)
 

*Reads a tag treeview and takes the currently selected tag into account when suggesting a parent tag for a new tag, and a default action.*

## Protected Slots

- void **completerActivated** (const TaggingAction &action)
- void **completerHighlighted** (const TaggingAction &action)
- void **slotEditingFinished** ()
- void **slotReturnPressed** ()
 

*Tagging action is used by facemanagement and assignwidget.*
- void **slotTextEdited** (const QString &text)

## 9.31.1 Member Function Documentation

### 9.31.1.1 setFilterModel()

```
void Digikam::AddTagsLineEdit::setFilterModel (
    AlbumFilterModel *const model )
```

Completion suggestions will be limited to tags contained in the filter model.

### 9.31.1.2 setParentTag

```
void Digikam::AddTagsLineEdit::setParentTag (
    Album *const album ) [slot]
```

If you set a tag tree view, this is taken care for automatically.

### 9.31.1.3 taggingActionSelected

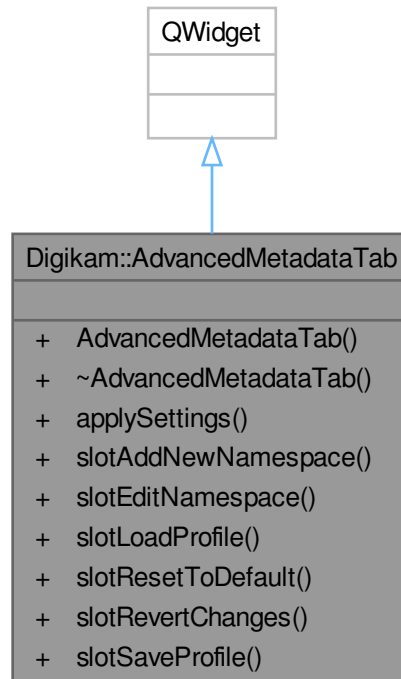
```
void Digikam::AddTagsLineEdit::taggingActionSelected (
    const TaggingAction & action ) [signal]
```

This already happens if anything is typed.



## 9.32 Digikam::AdvancedMetadataTab Class Reference

Inheritance diagram for Digikam::AdvancedMetadataTab:



### Public Slots

- void **slotAddNewNamespace** ()
- void **slotEditNamespace** ()
- void **slotLoadProfile** ()
- void **slotResetToDefault** ()
- void **slotRevertChanges** ()
- void **slotSaveProfile** ()

### Public Member Functions

- [AdvancedMetadataTab](#) (`QWidget *const parent=nullptr`)
- void **applySettings** ()

### 9.32.1 Constructor & Destructor Documentation

#### 9.32.1.1 AdvancedMetadataTab()

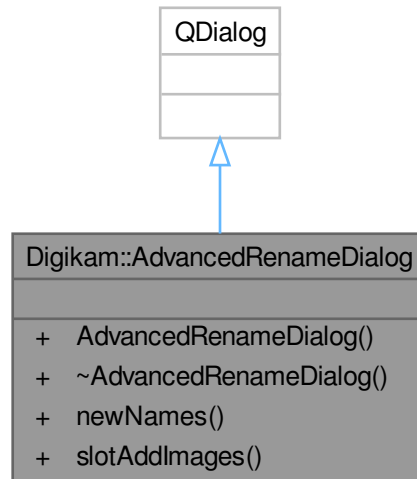
```

Digikam::AdvancedMetadataTab::AdvancedMetadataTab (
    QWidget *const parent = nullptr ) [explicit]
  
```

Connect all actions to `slotRevertAvailable`, which will enable revert to original if an add, edit, delete, or reorder was made

## 9.33 Digikam::AdvancedRenameDialog Class Reference

Inheritance diagram for Digikam::AdvancedRenameDialog:



### Public Slots

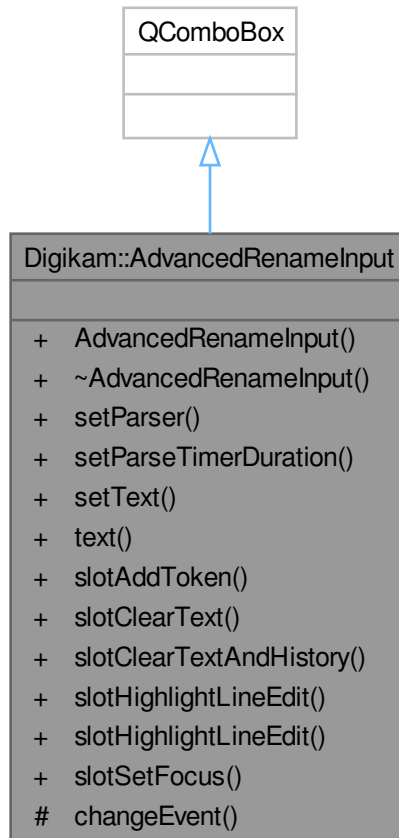
- void **slotAddImages** (const QList< QUrl > &urls)

### Public Member Functions

- **AdvancedRenameDialog** (QWidget \*const parent=nullptr)
- NewNamesList **newNames** () const

## 9.34 Digikam::AdvancedRenameInput Class Reference

Inheritance diagram for Digikam::AdvancedRenameInput:



### Public Slots

- void **slotAddToken** (const QString &)
- void **slotClearText** ()
- void **slotClearTextAndHistory** ()
- void **slotHighlightLineEdit** ()
- void **slotHighlightLineEdit** (const QString &word)
- void **slotSetFocus** ()

### Signals

- void **signalReturnPressed** ()
- void **signalTextChanged** (const QString &)
- void **signalTokenMarked** (bool)

### Public Member Functions

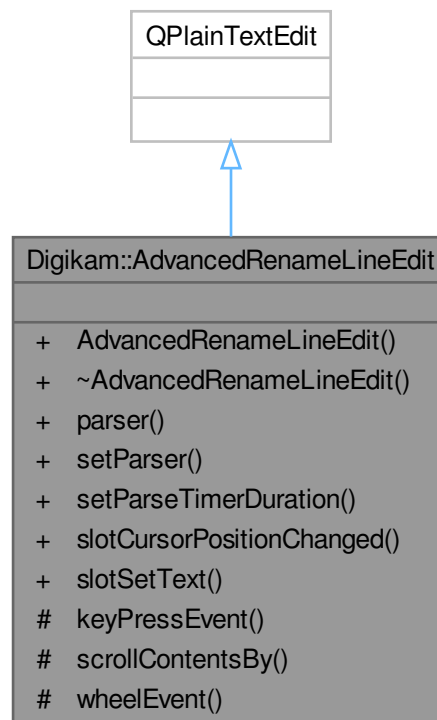
- **AdvancedRenameInput** (QWidget \*const parent=nullptr)
- void **setParser** (Parser \*parser)
- void **setParseTimerDuration** (int milliseconds)
- void **setText** (const QString &text)
- QString **text** () const

### Protected Member Functions

- void **changeEvent** (QEvent \*e) override

## 9.35 Digikam::AdvancedRenameLineEdit Class Reference

Inheritance diagram for Digikam::AdvancedRenameLineEdit:



### Public Slots

- void **slotCursorPositionChanged** ()
- void **slotSetText** (const QString &)

## Signals

- void **signalReturnPressed** ()
- void **signalTextChanged** (const QString &)
- void **signalTokenMarked** (bool)

## Public Member Functions

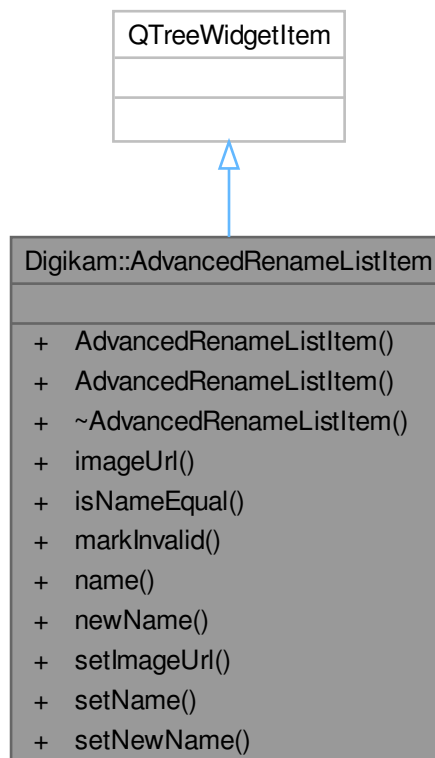
- **AdvancedRenameLineEdit** (QWidget \*const parent=nullptr)
- **Parser** \* **parser** () const
- void **setParser** (**Parser** \*parser)
- void **setParseTimerDuration** (int milliseconds)

## Protected Member Functions

- void **keyPressEvent** (QKeyEvent \*e) override
- void **scrollContentsBy** (int dx, int dy) override
- void **wheelEvent** (QWheelEvent \*e) override

## 9.36 Digikam::AdvancedRenameListItem Class Reference

Inheritance diagram for Digikam::AdvancedRenameListItem:



**Public Types**

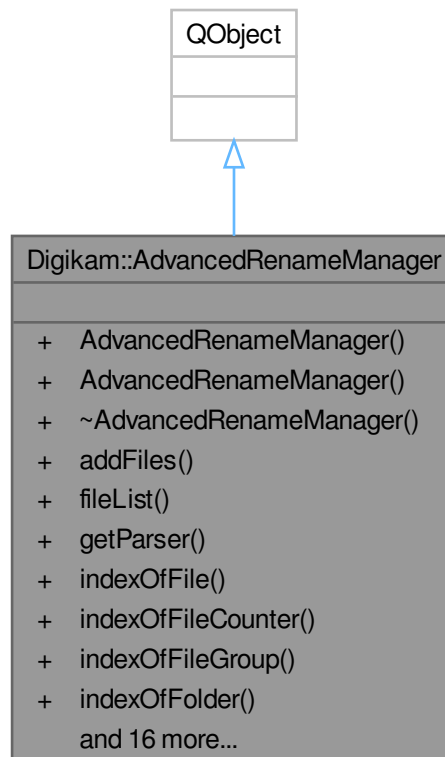
- enum **Column** { **OldName** = 0 , **NewName** }

**Public Member Functions**

- **AdvancedRenameListItem** (QTreeWidgetItem \*const view)
- **AdvancedRenameListItem** (QTreeWidgetItem \*const view, const QUrl &info)
- QUrl **imageUrl** () const
- bool **isNameEqual** () const
- void **markInvalid** (bool invalid)
- QString **name** () const
- QString **newName** () const
- void **setImageUrl** (const QUrl &url)
- void **setName** (const QString &name)
- void **setNewName** (const QString &name)

**9.37 Digikam::AdvancedRenameManager Class Reference**

Inheritance diagram for Digikam::AdvancedRenameManager:



## Public Types

- enum **ParserType** { **DefaultParser** = 0 , **ImportParser** }
- enum **SortAction** { **SortName** = 0 , **SortDate** , **SortSize** , **SortCustom** }
- enum **SortDirection** { **SortAscending** = 0 , **SortDescending** }

## Signals

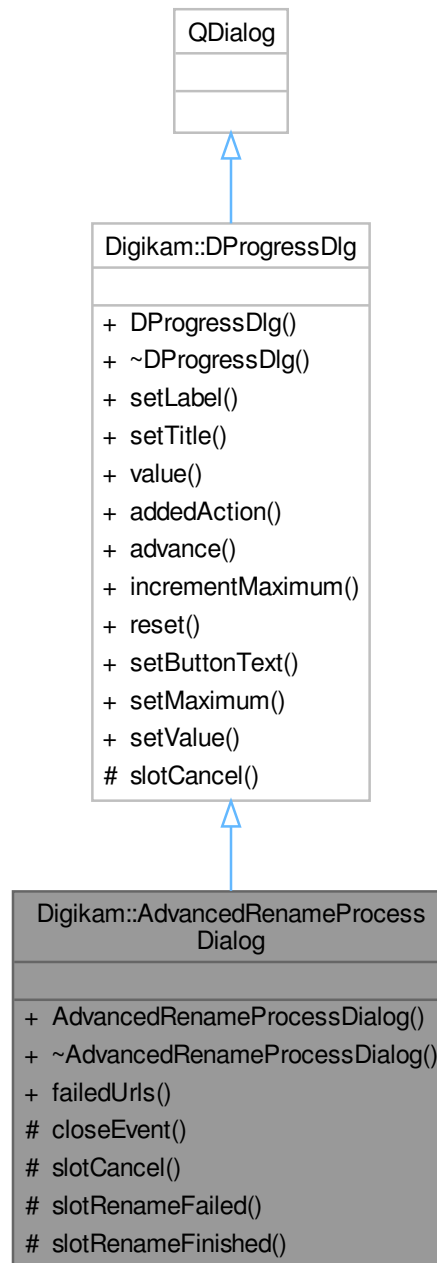
- void **signalSortingChanged** (QList< QUrl >)

## Public Member Functions

- **AdvancedRenameManager** (const QList< [ParseSettings](#) > &files)
- void **addFiles** (const QList< [ParseSettings](#) > &files)
- QStringList **fileList** () const
- [Parser](#) \* **getParser** () const
- int **indexOfFile** (const QString &filename)
- int **indexOfFileCounter** (const QString &filename)
- int **indexOfFileGroup** (const QString &filename)
- int **indexOfFolder** (const QString &filename)
- QMap< QString, QString > **newFileList** (bool checkFileSystem=false) const
- QString **newName** (const QString &filename) const
- void **parseFiles** ()
- void **parseFiles** (const [ParseSettings](#) &settings)
- void **parseFiles** (const QString &parseString)
- void **parseFiles** (const QString &parseString, const [ParseSettings](#) &settings)
- QString **randomStringOfIndex** (int index)
- void **reset** ()
- void **setCutFileName** (int index)
- void **setParserType** (ParserType type)
- void **setSortAction** (SortAction action)
- void **setSortDirection** (SortDirection direction)
- void **setStartIndex** (int index)
- void **setWidget** ([AdvancedRenameWidget](#) \*widget)
- SortAction **sortAction** () const
- SortDirection **sortDirection** () const

## 9.38 Digikam::AdvancedRenameProcessDialog Class Reference

Inheritance diagram for Digikam::AdvancedRenameProcessDialog:



### Public Member Functions

- **AdvancedRenameProcessDialog** (const NewNamesList &list, QWidget \*const parent=nullptr)
- `QList< QUrl > failedUrls ()` const



## Public Member Functions inherited from [Digikam::DProgressDlg](#)

- **DProgressDlg** (QWidget \*const parent=nullptr, const QString &caption=QString())
- void **setLabel** (const QString &text)
- void **setTitle** (const QString &text)
- int **value** () const

## Protected Slots

- void **slotCancel** ()
- void **slotRenameFailed** (const QUrl &url)
- void **slotRenameFinished** ()

## Protected Slots inherited from [Digikam::DProgressDlg](#)

- void **slotCancel** ()

## Protected Member Functions

- void **closeEvent** (QCloseEvent \*e) override

## Additional Inherited Members

## Public Slots inherited from [Digikam::DProgressDlg](#)

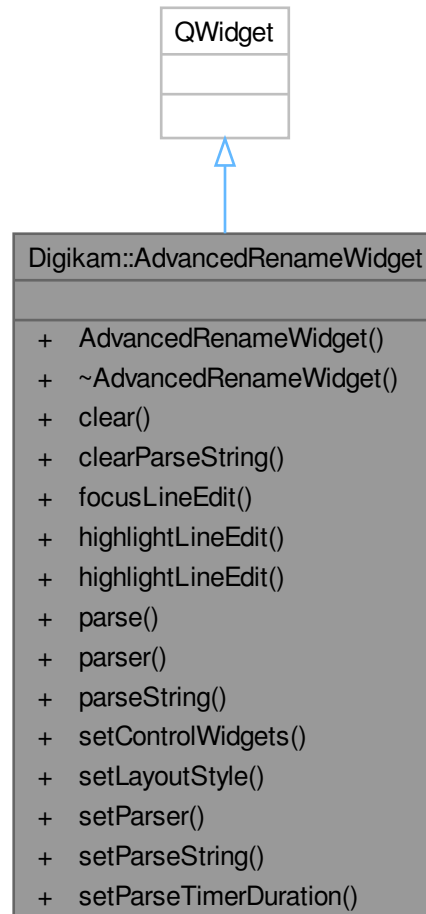
- void **addedAction** (const QPixmap &icon, const QString &text)
- void **advance** (int offset)
- void **incrementMaximum** (int added)
- void **reset** ()
- void **setButtonText** (const QString &text)
- void **setMaximum** (int max)
- void **setValue** (int value)

## Signals inherited from [Digikam::DProgressDlg](#)

- void **signalCancelPressed** ()

## 9.39 Digikam::AdvancedRenameWidget Class Reference

Inheritance diagram for Digikam::AdvancedRenameWidget:



### Public Types

- enum **ControlWidget** {  
**None** = 0x0 , **ToolTipButton** = 0x1 , **TokenButtons** = 0x2 , **ModifierToolButton** = 0x4 ,  
**DefaultControls** = TokenButtons | ToolTipButton | ModifierToolButton }
- typedef QFlags< ControlWidget > **ControlWidgets**
- enum **LayoutStyle** { **LayoutNormal** , **LayoutCompact** }

### Signals

- void **signalReturnPressed** ()
- void **signalTextChanged** (const QString &)

## Public Member Functions

- **AdvancedRenameWidget** (QWidget \*const parent=nullptr)
- void **clear** ()
  - clears the parse string as well as the history*
- void **clearParseString** ()
  - resets the current parse string, the LineEdit widget will be empty*
- void **focusLineEdit** ()
  - set focus for the LineEdit widget*
- void **highlightLineEdit** ()
  - highlight the LineEdit widgets text*
- void **highlightLineEdit** (const QString &word)
  - highlight a word in the LineEdit widgets text*
- QString **parse** (ParseSettings &settings) const
  - evaluates the parse string and executes the parser*
- Parser \* **parser** () const
  - returns a pointer to the currently assigned parser*
- QString **parseString** () const
  - returns the current parse string*
- void **setControlWidgets** (ControlWidgets mask)
  - sets the layout of the control widgets*
- void **setLayoutStyle** (LayoutStyle style)
  - set the layout style of the widget*
- void **setParser** (Parser \*parser)
  - sets the current parser.*
- void **setParseString** (const QString &text)
  - sets the current parse string*
- void **setParseTimerDuration** (int milliseconds)

## 9.39.1 Member Function Documentation

### 9.39.1.1 parse()

```
QString Digikam::AdvancedRenameWidget::parse (
    ParseSettings & settings ) const
```

#### Parameters

<i>settings</i>	information about the file to be renamed
-----------------	--

#### Returns

the new name of the file

### 9.39.1.2 setControlWidgets()

```
void Digikam::AdvancedRenameWidget::setControlWidgets (
    ControlWidgets mask )
```

**See also**

ControlWidget

**Parameters**

<i>mask</i>	a bitmask for setting the control widgets
-------------	---

**9.39.1.3 setLayoutStyle()**

```
void Digikam::AdvancedRenameWidget::setLayoutStyle (
    LayoutStyle style )
```

**Parameters**

<i>style</i>	the style of the layout
--------------	-------------------------

**See also**

LayoutStyle

**9.39.1.4 setParser()**

```
void Digikam::AdvancedRenameWidget::setParser (
    Parser * parser )
```

If a parser has already been assigned, it will be deleted first.

**Parameters**

<i>parser</i>	a pointer to the new parser instance
---------------	--------------------------------------

**9.39.1.5 setParseString()**

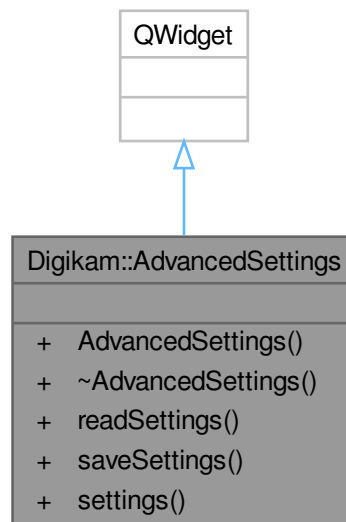
```
void Digikam::AdvancedRenameWidget::setParseString (
    const QString & text )
```

**Parameters**

<i>text</i>	the new parse string
-------------	----------------------

## 9.40 Digikam::AdvancedSettings Class Reference

Inheritance diagram for Digikam::AdvancedSettings:



### Signals

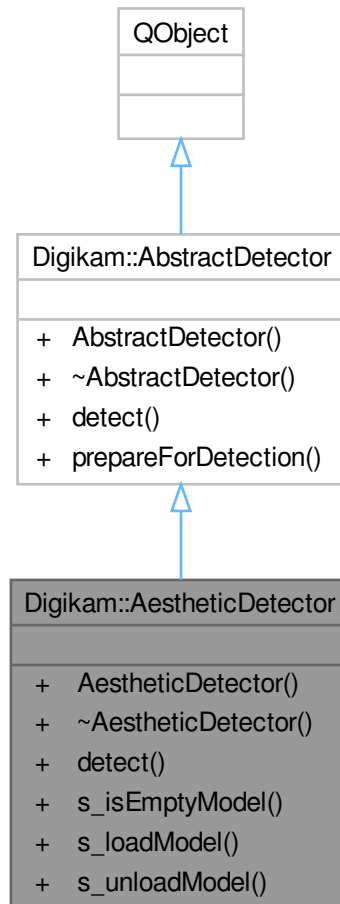
- void **signalDownloadNameChanged** ()

### Public Member Functions

- **AdvancedSettings** (`QWidget *const parent=nullptr`)
- void **readSettings** (`const KConfigGroup &group`)
- void **saveSettings** (`KConfigGroup &group`)
- [DownloadSettings](#) **settings** () const

## 9.41 Digikam::AestheticDetector Class Reference

Inheritance diagram for Digikam::AestheticDetector:



### Public Member Functions

- float [detect](#) (const cv::Mat &image) const override

### Public Member Functions inherited from [Digikam::AbstractDetector](#)

- **AbstractDetector** (QObject \*const parent=nullptr)

### Static Public Member Functions

- static bool **s\_isEmptyModel** ()
- static bool **s\_loadModel** ()
- static void **s\_unloadModel** ()

## Static Public Member Functions inherited from [Digikam::AbstractDetector](#)

- static cv::Mat **prepareForDetection** (const [DImg](#) &inputImage)

*NOTE: Maybe this function will move to `read_image()` of `imagequalityparser` in case all detectors of IQS use `cv::Mat`.*

### 9.41.1 Member Function Documentation

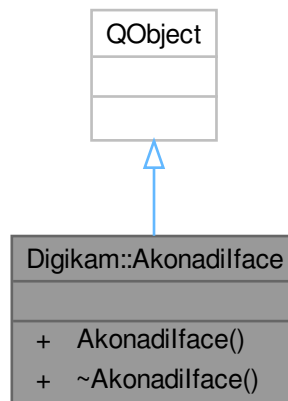
#### 9.41.1.1 detect()

```
float Digikam::AestheticDetector::detect (
    const cv::Mat & image ) const [override], [virtual]
```

Implements [Digikam::AbstractDetector](#).

## 9.42 Digikam::Akonadilface Class Reference

Inheritance diagram for Digikam::Akonadilface:



### Signals

- void **signalContactTriggered** (const `QString` &)

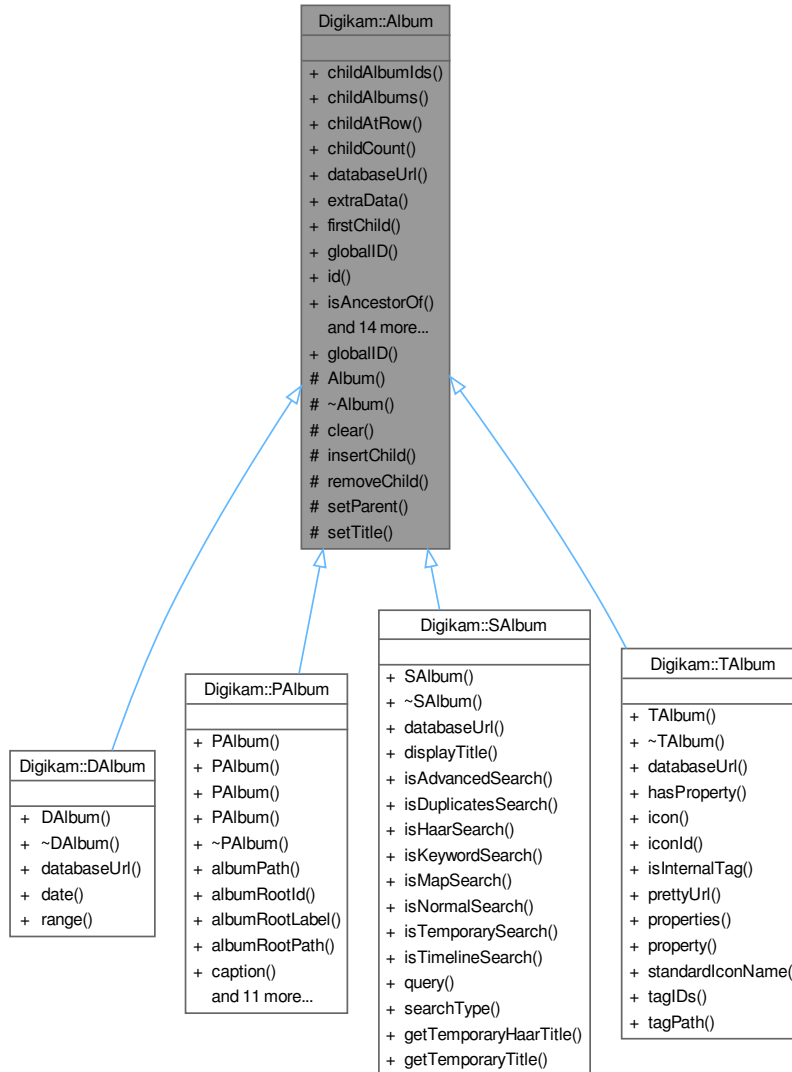
### Public Member Functions

- **Akonadilface** (`QMenu *const parent`)

## 9.43 Digikam::Album Class Reference

Abstract base class for all album types.

Inheritance diagram for Digikam::Album:



### Public Types

- enum `Type` {  
`PHYSICAL = 0`, `TAG`, `DATE`, `SEARCH`,  
`FACE` }

### Public Member Functions

- `QList< int > childAlbumIds` (bool recursive=false)



- AlbumList [childAlbums](#) (bool recursive=false)
- [Album](#) \* [childAtRow](#) (int row) const
- int [childCount](#) () const
- virtual [CoreDbUrl](#) [databaseUrl](#) () const =0
- void \* [extraData](#) (const void \*const key) const  
*Retrieve the associated extra data associated with key.*
- [Album](#) \* [firstChild](#) () const
- int [globalID](#) () const  
*An album ID is only unique among the set of all Albums of its Type.*
- int [id](#) () const  
*Each album has a ID uniquely identifying it in the set of Albums of a Type.*
- bool [isAncestorOf](#) ([Album](#) \*const album) const
- bool [isRoot](#) () const
- bool [isTrashAlbum](#) () const
- bool [isUsedByLabelsTree](#) () const
- [Album](#) \* [lastChild](#) () const
- [Album](#) \* [next](#) () const
- [Album](#) \* [parent](#) () const
- void [prepareForDeletion](#) ()  
*For secure deletion in an album model, call this function beforehand.*
- [Album](#) \* [prev](#) () const
- void [removeExtraData](#) (const void \*const key)  
*Remove the associated extra data associated with key.*
- int [rowFromAlbum](#) () const
- void [setExtraData](#) (const void \*const key, void \*const value)  
*This allows to associate some "extra" data to a [Album](#).*
- void [setUsedByLabelsTree](#) (bool isUsed)  
*Sets the property `m_usedByLabelsTree` to true if the search album was created using the Colors and labels tree view.*
- QString [title](#) () const
- [Type](#) [type](#) () const

### Static Public Member Functions

- static int [globalID](#) ([Type](#) type, int id)  
*Produces the global id.*

### Protected Member Functions

- [Album](#) ([Album::Type](#) type, int id, bool root)  
*Constructor.*
- virtual [~Album](#) ()  
*Destructor.*
- void [clear](#) ()  
*Delete all child albums and also remove any associated extra data.*
- void [insertChild](#) ([Album](#) \*const child)
- void [removeChild](#) ([Album](#) \*const child)
- void [setParent](#) ([Album](#) \*const [parent](#))
- void [setTitle](#) (const QString &[title](#))

## Friends

- class **AlbumManager**

### 9.43.1 Detailed Description

A class which provides an abstraction for a type [Album](#). This class is meant to be derived and every time a new [Album](#) Type is defined add a enum corresponding to that to [Album::Type](#)

This class provides a means of building a tree representation for Albums

#### See also

[Album::setParent\(\)](#).

### 9.43.2 Member Enumeration Documentation

#### 9.43.2.1 Type

enum [Digikam::Album::Type](#)

#### Enumerator

PHYSICAL	A physical album type. See also <a href="#">PAlbum</a>
TAG	A tag album type. See also <a href="#">TAlbum</a>
DATE	A date album type. See also <a href="#">DAlbum</a>
SEARCH	A search album type. See also <a href="#">SAlbum</a>
FACE	A faces album type. See also FAAlbum

### 9.43.3 Constructor & Destructor Documentation

#### 9.43.3.1 ~Album()

```
Digikam::Album::~Album ( ) [protected], [virtual]
```

this will also recursively delete all child Albums

### 9.43.4 Member Function Documentation

#### 9.43.4.1 childAlbumIds()

```
QList< int > Digikam::Album::childAlbumIds (
    bool recursive = false )
```

##### Returns

a list of all child Albums

#### 9.43.4.2 childAlbums()

```
AlbumList Digikam::Album::childAlbums (
    bool recursive = false )
```

##### Returns

a list of all child Albums

#### 9.43.4.3 childAtRow()

```
Album * Digikam::Album::childAtRow (
    int row ) const
```

##### Returns

the child of this album at row

#### 9.43.4.4 childCount()

```
int Digikam::Album::childCount ( ) const
```

##### Returns

the childCount of the album

#### 9.43.4.5 databaseUrl()

```
virtual CoreDbUrl Digikam::Album::databaseUrl ( ) const [pure virtual]
```

##### Returns

the kde url of the album

Implemented in [Digikam::PAlbum](#), [Digikam::TAlbum](#), [Digikam::DAlbum](#), and [Digikam::SAlbum](#).

#### 9.43.4.6 extraData()

```
void * Digikam::Album::extraData (
    const void *const key ) const
```

## Parameters

<i>key</i>	the key of the extra data
------------	---------------------------

## See also

[setExtraData](#)

[extraData](#)

**9.43.4.7 firstChild()**

```
Album * Digikam::Album::firstChild ( ) const
```

## Returns

the first child of this album or 0 if no children

**9.43.4.8 globalID() [1/2]**

```
int Digikam::Album::globalID ( ) const
```

This is a global Identifier which will uniquely identifying the [Album](#) among all Albums

## Note

If you are adding a new [Album](#) Type make sure to update this implementation.

You can always get the ID of the album using something like

```
int albumID = rootAlbum->globalID() - album->globalID();
```

## Returns

the globalID of the album

## See also

[id\(\)](#)

**9.43.4.9 globalID() [2/2]**

```
int Digikam::Album::globalID (
    Type type,
    int id ) [static]
```

## Parameters

<i>type</i>	The type of the album
<i>id</i>	the (type-specific) id of the album

**Returns**

the global id

**9.43.4.10 id()**

```
int Digikam::Album::id ( ) const
```

**Note**

The ID for a root [Album](#) is always 0

**Returns**

the ID of the album

**See also**

[globalID\(\)](#)

**9.43.4.11 isAncestorOf()**

```
bool Digikam::Album::isAncestorOf (
    Album *const album ) const
```

**Returns**

true if the album is in the parent hierarchy

**Parameters**

<i>album</i>	the album to check whether it belongs in the child hierarchy
--------------	--

**9.43.4.12 isRoot()**

```
bool Digikam::Album::isRoot ( ) const
```

**Returns**

true is the album is a Root [Album](#)

**9.43.4.13 isTrashAlbum()**

```
bool Digikam::Album::isTrashAlbum ( ) const
```

**Returns**

true if the album was created to be a trash virtual album

#### 9.43.4.14 isUsedByLabelsTree()

```
bool Digikam::Album::isUsedByLabelsTree ( ) const
```

##### Returns

true if the [Album](#) was created by Labels Tree

#### 9.43.4.15 lastChild()

```
Album * Digikam::Album::lastChild ( ) const
```

##### Returns

the last child of this album or 0 if no children

#### 9.43.4.16 next()

```
Album * Digikam::Album::next ( ) const
```

##### Returns

the next sibling of this album of this album or 0 if no next sibling

##### See also

[AlbumIterator](#)

#### 9.43.4.17 parent()

```
Album * Digikam::Album::parent ( ) const
```

##### Returns

the parent album for this album

#### 9.43.4.18 prev()

```
Album * Digikam::Album::prev ( ) const
```

##### Returns

the previous sibling of this album of this album or 0 if no previous sibling

##### See also

[AlbumIterator](#)

#### 9.43.4.19 removeExtraData()

```
void Digikam::Album::removeExtraData (
    const void *const key )
```

## Parameters

<i>key</i>	the key of the extra data
------------	---------------------------

## See also

[setExtraData](#)

[extraData](#)

**9.43.4.20 rowFromAlbum()**

```
int Digikam::Album::rowFromAlbum ( ) const
```

## Returns

the `rowFromAlbum` of the album

**9.43.4.21 setExtraData()**

```
void Digikam::Album::setExtraData (
    const void *const key,
    void *const value )
```

As one [Album](#) can be used by several objects (often views) which all need to add some data, you have to use a key to reference your extra data within the [Album](#).

That way a [Album](#) can hold and provide access to all those views separately.

for eg,

```
album->setExtraData( this, searchFolderItem );
```

and can later access the `searchFolderItem` by doing

```
SearchFolderItem *item = static_cast<SearchFolderItem*>(album->extraData(this));
```

Note: you have to remove and destroy the data you associated yourself when you don't need it anymore!

## Parameters

<i>key</i>	the key of the extra data
<i>value</i>	the value of the extra data

## See also

[extraData](#)

[removeExtraData](#)

**9.43.4.22 setUsedByLabelsTree()**

```
void Digikam::Album::setUsedByLabelsTree (
    bool isUsed )
```

## Parameters

<code>isUsed</code>	=> the status of the usage
---------------------	----------------------------

**9.43.4.23 title()**

```
QString Digikam::Album::title ( ) const
```

## Returns

the `title` aka name of the album

**9.43.4.24 type()**

```
Album::Type Digikam::Album::type ( ) const
```

## Returns

the type of album

## See also

[Type](#)

## 9.44 Digikam::AlbumChangeset Class Reference

**Public Types**

- enum **Operation** {  
    **Unknown** , **Added** , **Deleted** , **Renamed** ,  
    **PropertiesChanged** }

**Public Member Functions**

- **AlbumChangeset** (int albumId, Operation operation)
- int **albumId** () const
- Operation **operation** () const
- [AlbumChangeset](#) & **operator**<< (const QDBusArgument &argument)
- const [AlbumChangeset](#) & **operator**>> (QDBusArgument &argument) const



## 9.45 Digikam::AlbumCopyMoveHint Class Reference

### Public Member Functions

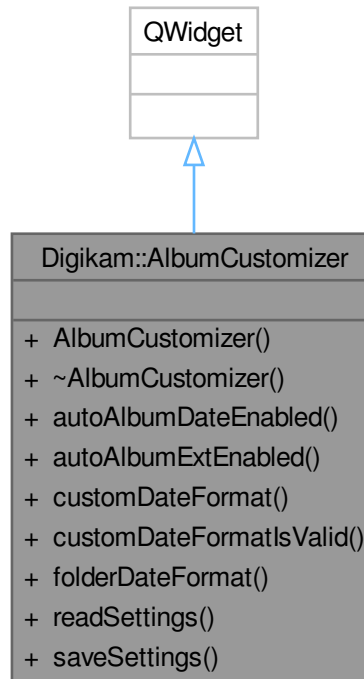
- **AlbumCopyMoveHint** ()=default  
*An [AlbumCopyMoveHint](#) describes an existing album and a destination to which this album is expected to be copied, moved or renamed.*
- **AlbumCopyMoveHint** (int srcAlbumRootId, int srcAlbum, int dstAlbumRootId, const QString &dstRelativePath)
- int **albumIdSrc** () const
- int **albumRootIdDst** () const
- int **albumRootIdSrc** () const
- CollectionScannerHints::DstPath **dst** () const
- bool **isDstAlbum** (int albumRootId, const QString &relativePath) const
- bool **isSrcAlbum** (int albumRootId, int albumId) const
- **operator const CollectionScannerHints::Album &** () const
- **operator const CollectionScannerHints::DstPath &** () const
- [AlbumCopyMoveHint](#) & **operator<<** (const QDBusArgument &argument)
- bool **operator==** (const CollectionScannerHints::Album &src) const
- bool **operator==** (const CollectionScannerHints::DstPath &dst) const
- const [AlbumCopyMoveHint](#) & **operator>>** (QDBusArgument &argument) const
- QT\_HASH\_TYPE **qHash** () const
- QString **relativePathDst** () const
- CollectionScannerHints::Album **src** () const

### Protected Attributes

- CollectionScannerHints::DstPath **m\_dst**
- CollectionScannerHints::Album **m\_src**

## 9.46 Digikam::AlbumCustomizer Class Reference

Inheritance diagram for Digikam::AlbumCustomizer:



### Public Types

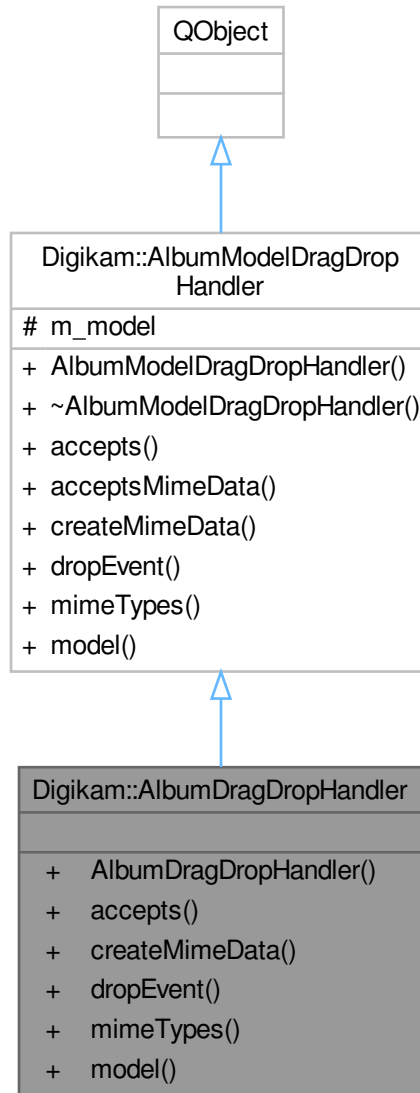
- enum **DateFormatOptions** { **IsoDateFormat** = 0 , **TextDateFormat** , **LocalDateFormat** , **CustomDateFormat** }

### Public Member Functions

- **AlbumCustomizer** (QWidget \*const parent=nullptr)
- bool **autoAlbumDateEnabled** () const
- bool **autoAlbumExtEnabled** () const
- QString **customDateFormat** () const
- bool **customDateFormatIsValid** () const
- int **folderDateFormat** () const
- void **readSettings** (const KConfigGroup &group)
- void **saveSettings** (KConfigGroup &group)

## 9.47 Digikam::AlbumDragDropHandler Class Reference

Inheritance diagram for Digikam::AlbumDragDropHandler:



### Public Member Functions

- **AlbumDragDropHandler** (`AlbumModel *const model`)
- Qt::DropAction **accepts** (`const QDropEvent *e, const QModelIndex &dropIndex`) override  
*Returns if the given mime data is accepted for drop on dropIndex.*
- QMimeData \* **createMimeData** (`const QList< Album * > &`) override  
*Create a mime data object for starting a drag from the given Albums.*
- bool **dropEvent** (`QAbstractItemView *view, const QDropEvent *e, const QModelIndex &droppedOn`) override  
*Gives the view and the occurring drop event.*

- QStringList [mimeTypes](#) () const override  
*Returns the supported mime types.*
- AlbumModel \* [model](#) () const

## Public Member Functions inherited from [Digikam::AlbumModelDragDropHandler](#)

- AlbumModelDragDropHandler ([AbstractAlbumModel](#) \*model)
- virtual bool [acceptsMimeData](#) (const QMimeData \*data)  
*Returns if the given mime data can be handled.*
- [AbstractAlbumModel](#) \* [model](#) () const

## Additional Inherited Members

## Protected Attributes inherited from [Digikam::AlbumModelDragDropHandler](#)

- [AbstractAlbumModel](#) \* [m\\_model](#) = nullptr

## 9.47.1 Member Function Documentation

### 9.47.1.1 [accepts\(\)](#)

```
Qt::DropAction Digikam::AlbumDragDropHandler::accepts (
    const QDropEvent * e,
    const QModelIndex & dropIndex ) [override], [virtual]
```

Returns the proposed action, or Qt::IgnoreAction if not accepted.

Reimplemented from [Digikam::AlbumModelDragDropHandler](#).

### 9.47.1.2 [createMimeData\(\)](#)

```
QMimeData * Digikam::AlbumDragDropHandler::createMimeData (
    const QList< Album * > & ) [override], [virtual]
```

Reimplemented from [Digikam::AlbumModelDragDropHandler](#).

### 9.47.1.3 [dropEvent\(\)](#)

```
bool Digikam::AlbumDragDropHandler::dropEvent (
    QAbstractItemView * view,
    const QDropEvent * e,
    const QModelIndex & droppedOn ) [override], [virtual]
```

The index is the index where the drop was dropped on. It may be invalid (dropped on decoration, viewport) Returns true if the event is to be accepted.

Reimplemented from [Digikam::AlbumModelDragDropHandler](#).

#### 9.47.1.4 mimeTypees()

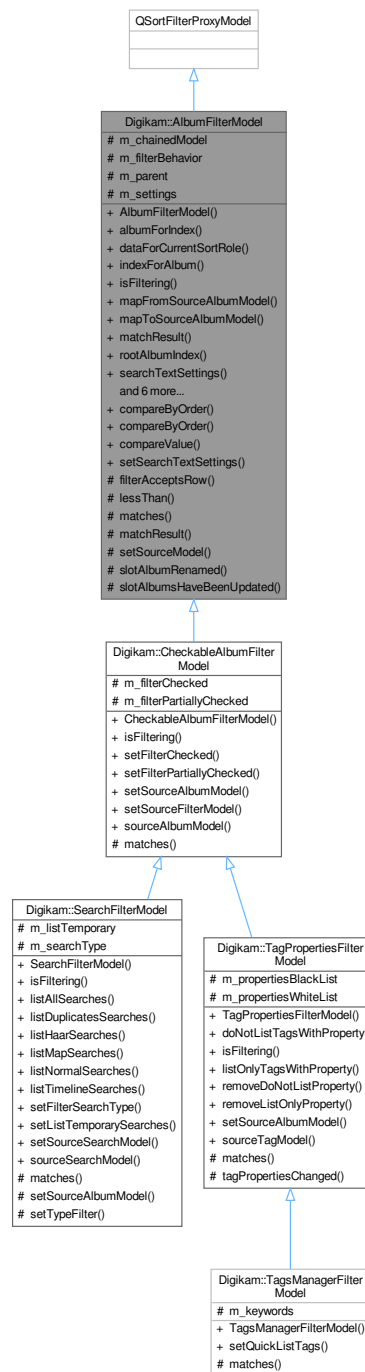
```
QStringList Digikam::AlbumDragDropHandler::mimeTypees ( ) const [override], [virtual]
```

Called by the default implementation of model's [mimeTypees\(\)](#).

Reimplemented from [Digikam::AlbumModelDragDropHandler](#).

## 9.48 Digikam::AlbumFilterModel Class Reference

Inheritance diagram for Digikam::AlbumFilterModel:



### Public Types

- enum [FilterBehavior](#) { [SimpleFiltering](#) , [FullFiltering](#) , [StrictFiltering](#) }
- enum [MatchResult](#) { [NoMatch](#) = 0 , [DirectMatch](#) , [ParentMatch](#) , [ChildMatch](#) , [SpecialMatch](#) }

## Public Slots

- void [setSearchTextSettings](#) (const [SearchTextSettings](#) &settings)  
*Accepts new settings used for filtering and applies them to the model.*

## Signals

- void [hasSearchResult](#) (bool hasResult)  
*Indicates whether the newly applied filter results in a search result or not.*
- void [searchTextSettingsAboutToChange](#) (bool searched, bool willSearch)  
*This signal indicates that a new [SearchTextSettings](#) arrived and is about to be applied to the model.*
- void [searchTextSettingsChanged](#) (bool wasSearching, bool searched)  
*Indicates that new search text settings were applied.*
- void **signalFilterChanged** ()  
*Indicates that a new filter was applied to the model.*

## Public Member Functions

- **AlbumFilterModel** (QObject \*const parent=nullptr)
- **Album \* albumForIndex** (const QModelIndex &index) const  
*Convenience methods.*
- QVariant **dataForCurrentSortRole** ([Album](#) \*album) const
- QModelIndex **indexForAlbum** ([Album](#) \*album) const
- virtual bool **isFiltering** () const  
*Returns if the currently applied filters will result in any filtering.*
- QModelIndex **mapFromSourceAlbumModel** (const QModelIndex &index) const
- QModelIndex **mapToSourceAlbumModel** (const QModelIndex &index) const
- **MatchResult matchResult** (const QModelIndex &index) const  
*Returns the MatchResult of an index of this model.*
- QModelIndex **rootAlbumIndex** () const
- [SearchTextSettings](#) **searchTextSettings** () const  
*Returns the settings currently used for filtering.*
- void **setFilterBehavior** ([FilterBehavior](#) behavior)  
*Sets the filter behavior.*
- void **setSourceAlbumModel** ([AbstractAlbumModel](#) \*const source)  
*Sets the source model.*
- void **setSourceFilterModel** ([AlbumFilterModel](#) \*const source)  
*Sets a chained filter model.*
- [AbstractAlbumModel](#) \* **sourceAlbumModel** () const
- [AlbumFilterModel](#) \* **sourceFilterModel** () const
- void **updateFilter** ()  
*Force invalidateFilter() externally.*

## Static Public Member Functions

- template<typename T >  
static int **compareByOrder** (const T &a, const T &b, Qt::SortOrder sortOrder)
- static int **compareByOrder** (int compareResult, Qt::SortOrder sortOrder)  
*Takes a typical result from a compare method (0 is equal, -1 is less than, 1 is greater than) and applies the given sort order to it.*
- template<typename T >  
static int **compareValue** (const T &a, const T &b)  
*Returns the usual compare result of -1, 0, or 1 for lessThan, equals and greaterThan.*

## Protected Slots

- void **slotAlbumRenamed** ([Album](#) \*album)
- void **slotAlbumsHaveBeenUpdated** (int type)

## Protected Member Functions

- bool **filterAcceptsRow** (int source\_row, const QModelIndex &source\_parent) const override
- bool **lessThan** (const QModelIndex &left, const QModelIndex &right) const override
- virtual bool **matches** ([Album](#) \*album) const  
*This method provides the basic match checking algorithm.*
- [MatchResult](#) **matchResult** ([Album](#) \*album) const  
*Returns if the filter matches this album (same logic as filterAcceptsRow).*
- void **setSourceModel** ([QAbstractItemModel](#) \*const model) override  
*Use setSourceAlbumModel.*

## Protected Attributes

- [QPointer](#)< [AlbumFilterModel](#) > **m\_chainedModel** = nullptr
- [FilterBehavior](#) **m\_filterBehavior** = [FullFiltering](#)
- [QObject](#) \* **m\_parent** = nullptr
- [SearchTextSettings](#) **m\_settings**

## 9.48.1 Member Enumeration Documentation

### 9.48.1.1 FilterBehavior

```
enum Digikam::AlbumFilterModel::FilterBehavior
```

#### Enumerator

SimpleFiltering	If an index does not match, the index and all its children are filtered out. This is the Qt default behavior, but undesirable for album trees.
FullFiltering	Default behavior. If an index matches, it is shown, which directly means all its parents are shown as well. In addition, all its children are shown as well.
StrictFiltering	If an index matches, it is shown, which directly means all its parents are shown as well. Its children are not shown unless they also match.

### 9.48.1.2 MatchResult

```
enum Digikam::AlbumFilterModel::MatchResult
```

#### Enumerator

NoMatch	This enum can be used as a boolean value if match/no match only is needed.
DirectMatch	The index itself is matched.
ParentMatch	A parent if the index is matched.
ChildMatch	A child of the index is matched.
SpecialMatch	The index is matched not because of search settings, but because it has a special type.



## 9.48.2 Member Function Documentation

### 9.48.2.1 hasSearchResult

```
void Digikam::AlbumFilterModel::hasSearchResult (
    bool hasResult ) [signal]
```

#### Parameters

<i>hasResult</i>	true if the new filter matches any album, else false
------------------	--

### 9.48.2.2 isFiltering()

```
bool Digikam::AlbumFilterModel::isFiltering ( ) const [virtual]
```

#### Returns

true if the current selected filter could result in any filtering without checking if this really happens.

Reimplemented in [Digikam::CheckableAlbumFilterModel](#), [Digikam::SearchFilterModel](#), and [Digikam::TagPropertiesFilterModel](#).

### 9.48.2.3 lessThan()

```
bool Digikam::AlbumFilterModel::lessThan (
    const QModelIndex & left,
    const QModelIndex & right ) const [override], [protected]
```

Implementation to sort Tags that contain Unconfirmed Faces, according to the Unconfirmed Face Count.

### 9.48.2.4 matches()

```
bool Digikam::AlbumFilterModel::matches (
    Album * album ) const [protected], [virtual]
```

Return true if this single album matches the current criteria. This method can be overridden to provide custom filtering.

#### Parameters

<i>album</i>	the album to tell if it matches the filter criteria or not.
--------------	---

Reimplemented in [Digikam::CheckableAlbumFilterModel](#), [Digikam::SearchFilterModel](#), [Digikam::TagPropertiesFilterModel](#), and [Digikam::TagsManagerFilterModel](#).

### 9.48.2.5 matchResult() [1/2]

```
AlbumFilterModel::MatchResult Digikam::AlbumFilterModel::matchResult (
    Album * album ) const [protected]
```

An album matches if the search text settings are found in a parent album's title, in the album's title or in a child album's title, or if it is a special album (root) that is never filtered out.

#### 9.48.2.6 `matchResult()` [2/2]

```
AlbumFilterModel::MatchResult Digikam::AlbumFilterModel::matchResult (
    const QModelIndex & index ) const
```

Never returns NoMatch for a valid index, because in this case, the index would rather be filtered out.

#### 9.48.2.7 `searchTextSettings()`

```
SearchTextSettings Digikam::AlbumFilterModel::searchTextSettings ( ) const
```

##### Returns

current settings for filtering.

#### 9.48.2.8 `searchTextSettingsAboutToChange`

```
void Digikam::AlbumFilterModel::searchTextSettingsAboutToChange (
    bool searched,
    bool willSearch ) [signal]
```

##### Parameters

<i>searched</i>	true if filtering by text was enabled before applying the new settings
<i>willSearch</i>	true if the new settings can result in any filtering by text, else false.

#### 9.48.2.9 `searchTextSettingsChanged`

```
void Digikam::AlbumFilterModel::searchTextSettingsChanged (
    bool wasSearching,
    bool searched ) [signal]
```

##### Parameters

<i>wasSearching</i>	true if this is not a new search that
<i>searched</i>	true if the new settings result in any filtering

#### 9.48.2.10 `setFilterBehavior()`

```
void Digikam::AlbumFilterModel::setFilterBehavior (
    FilterBehavior behavior )
```

Default is FullFiltering.

### 9.48.2.11 setSearchTextSettings

```
void Digikam::AlbumFilterModel::setSearchTextSettings (
    const SearchTextSettings & settings ) [slot]
```

#### Parameters

<i>settings</i>	new settings to apply. An empty text will be interpreted as no filtering
-----------------	--

### 9.48.2.12 setSourceAlbumModel()

```
void Digikam::AlbumFilterModel::setSourceAlbumModel (
    AbstractAlbumModel *const source )
```

Note: If a chained filter model is set, it will not be reset, but the source album model will be made source of the chained filter model.

### 9.48.2.13 setSourceFilterModel()

```
void Digikam::AlbumFilterModel::setSourceFilterModel (
    AlbumFilterModel *const source )
```

Note: If a direct source album model is set as current source, it will be set as sourceAlbumModel of the new source filter model.

### 9.48.2.14 setSourceModel()

```
void Digikam::AlbumFilterModel::setSourceModel (
    QAbstractItemModel *const model ) [override], [protected]
```

#### See also

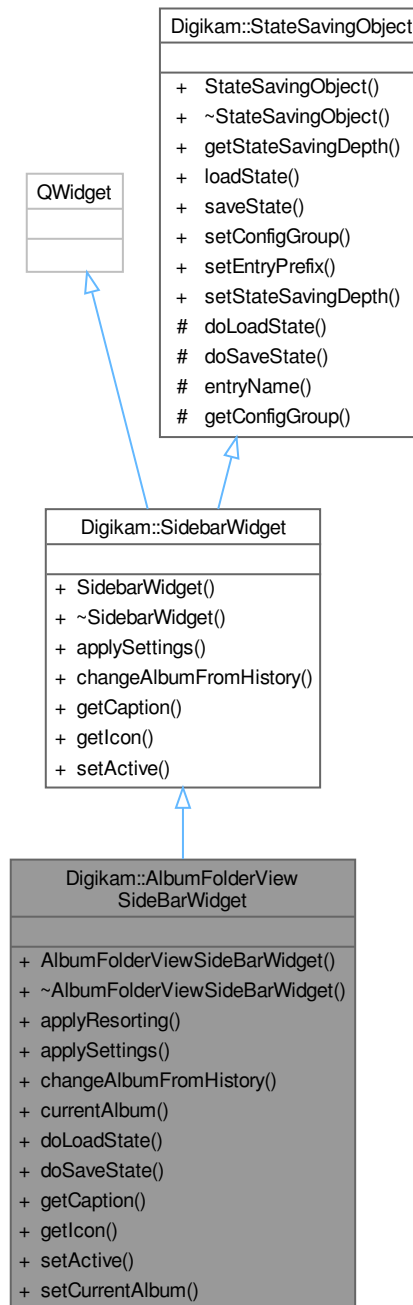
[setSourceAlbumModel](#)

#### Parameters

<i>model</i>	source model
--------------	--------------

## 9.49 Digikam::AlbumFolderViewSideBarWidget Class Reference

Inheritance diagram for Digikam::AlbumFolderViewSideBarWidget:



### Public Slots

- void **setCurrentAlbum** (`PAAlbum *album`)

## Signals

- void **signalFindDuplicates** (const QList< PAlbum \* > &albums)

## Signals inherited from Digikam::SidebarWidget

- void **requestActiveTab** (SidebarWidget \*)  
*This signal can be emitted if this sidebar widget wants to be the one that is active.*
- void **signalNotificationError** (const QString &message, int type)  
*To dispatch error message to temporized pop-up notification widget hosted with icon-view.*

## Public Member Functions

- **AlbumFolderViewSideBarWidget** (QWidget \*const parent, AlbumModel \*const model, AlbumModificationHelper \*const albumModificationHelper)
- void **applyResorting** ()
- void **applySettings** () override  
*This method is invoked when the application settings should be (re-) applied to this widget.*
- void **changeAlbumFromHistory** (const QList< Album \* > &album) override  
*This is called on this widget when the history requires to move back to the specified album.*
- AlbumPointer< PAlbum > **currentAlbum** () const
- void **doLoadState** () override  
*Implement this hook method for state loading.*
- void **doSaveState** () override  
*Implement this hook method for state saving.*
- const QString **getCaption** () override  
*Must be implemented to return the title of this sidebar's tab.*
- const QIcon **getIcon** () override  
*Must be implemented and return the icon that shall be visible for this sidebar widget.*
- void **setActive** (bool active) override  
*This method is called if the visible sidebar widget is changed.*

## Public Member Functions inherited from Digikam::SidebarWidget

- SidebarWidget (QWidget \*const parent)  
*Constructor.*
- ~SidebarWidget () override=default  
*Destructor.*

## Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (QObject \*const host)  
*Constructor.*
- virtual `~StateSavingObject ()`  
*Destructor.*
- [StateSavingDepth](#) `getStateSavingDepth ()` const  
*Returns the depth used for state saving or loading.*
- void `loadState ()`  
*Invokes loading the class' state.*
- void `saveState ()`  
*Invokes saving the class' state.*
- virtual void `setConfigGroup` (const KConfigGroup &group)  
*Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void `setEntryPrefix` (const QString &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void `setStateSavingDepth` (const [StateSavingDepth](#) depth)  
*Sets the depth used for state saving or loading.*

## Additional Inherited Members

## Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { `INSTANCE` , `DIRECT_CHILDREN` , `RECURSIVE` }  
*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

## Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- QString `entryName` (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup `getConfigGroup ()` const  
*Returns the config group that must be used for state saving and loading.*

### 9.49.1 Member Function Documentation

#### 9.49.1.1 `applySettings()`

```
void Digikam::AlbumFolderViewSideBarWidget::applySettings ( ) [override], [virtual]
```

Implements [Digikam::SidebarWidget](#).

#### 9.49.1.2 `changeAlbumFromHistory()`

```
void Digikam::AlbumFolderViewSideBarWidget::changeAlbumFromHistory (
    const QList< Album * > & album ) [override], [virtual]
```

Implements [Digikam::SidebarWidget](#).

### 9.49.1.3 doLoadState()

```
void Digikam::AlbumFolderViewSideBarWidget::doLoadState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

### 9.49.1.4 doSaveState()

```
void Digikam::AlbumFolderViewSideBarWidget::doSaveState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

### 9.49.1.5 getCaption()

```
const QString Digikam::AlbumFolderViewSideBarWidget::getCaption ( ) [override], [virtual]
```

#### Returns

localized title string

Implements [Digikam::SidebarWidget](#).

### 9.49.1.6 getIcon()

```
const QIcon Digikam::AlbumFolderViewSideBarWidget::getIcon ( ) [override], [virtual]
```

#### Returns

pixmap icon

Implements [Digikam::SidebarWidget](#).

### 9.49.1.7 setActive()

```
void Digikam::AlbumFolderViewSideBarWidget::setActive (
    bool active ) [override], [virtual]
```

#### Parameters

<i>active</i>	if true, this widget is the new active widget, if false another widget is active
---------------	--

Implements [Digikam::SidebarWidget](#).

## 9.50 Digikam::AlbumHistory Class Reference

Manages the history of the last visited albums.

Inheritance diagram for Digikam::AlbumHistory:



### Public Slots

- void **slotAlbumCurrentChanged** ()
- void **slotAlbumDeleted** ([Album](#) \*album)



- void **slotAlbumsCleared** ()
- void **slotAlbumSelected** ()
- void **slotClearSelectPAAlbum** (const [ItemInfo](#) &imageInfo)
- void **slotClearSelectTAlbum** (int id)
- void **slotCurrentChange** (const [ItemInfo](#) &info)
- void **slotImageSelected** (const [ItemInfoList](#) &selectedImage)

### Signals

- void **signalSetCurrent** (qulonglong imageId)
- void **signalSetSelectedInfos** (const QList< [ItemInfo](#) > &)

### Public Member Functions

- **AlbumHistory** (QObject \*const parent=nullptr)
- void **addAlbums** (const QList< [Album](#) \* > &albums, QWidget \*const widget, const QHash< [LabelsTreeView::Labels](#), QList< int > > &selectedLabels)
  - AlbumHistory::addAlbums A special overloaded function for handling [AlbumHistory](#) for the [Labels tree-view](#).*
- void **addAlbums** (const QList< [Album](#) \* > &albums, QWidget \*const widget=nullptr)
- void **back** (QList< [Album](#) \* > &album, QWidget \*\*const widget, unsigned int steps=1)
- void **clearHistory** ()
- void **deleteAlbum** ([Album](#) \*const album)
- void **forward** (QList< [Album](#) \* > &album, QWidget \*\*const widget, unsigned int steps=1)
- void **getBackwardHistory** (QStringList &list) const
- void **getCurrentAlbum** ([Album](#) \*\*const album, QWidget \*\*const widget) const
- void **getForwardHistory** (QStringList &list) const
- bool **isBackwardEmpty** () const
- bool **isForwardEmpty** () const
- QHash< [LabelsTreeView::Labels](#), QList< int > > **neededLabels** ()

## 9.50.1 Detailed Description

The user is able to navigate through the albums, he has opened during a session.

## 9.50.2 Member Function Documentation

### 9.50.2.1 addAlbums()

```
void Digikam::AlbumHistory::addAlbums (
    const QList< Album * > & albums,
    QWidget *const widget,
    const QHash< LabelsTreeView::Labels, QList< int > > & selectedLabels )
```

### Author

Mohamed\_Anwer

## 9.51 Digikam::AlbumInfo Class Reference

A container class for transporting album information from the database to [AlbumManager](#).

### Public Types

- typedef QList< [AlbumInfo](#) > **List**

### Public Member Functions

- bool **isNull** () const
- bool **operator**< (const [AlbumInfo](#) &info) const  
*needed for sorting*

### Public Attributes

- int **albumRootId** = 0
- QString **caption**
- QString **category**
- QDate **date**
- qlonglong **iconId** = 0
- int **id** = 0
- QString **relativePath**

## 9.52 Digikam::AlbumIterator Class Reference

Iterate over all children of this [Album](#).

### Public Member Functions

- **AlbumIterator** ([Album](#) \*const album)
- [Album](#) \* **current** () const
- [Album](#) \* **operator\*** ()
- [AlbumIterator](#) & **operator++** ()

### 9.52.1 Detailed Description

#### Note

It will not include the specified album

#### Example usage:

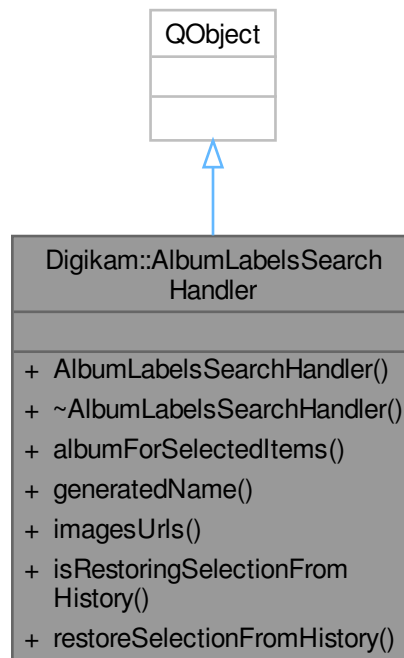
```
AlbumIterator it(album);
while ( it.current() )
{
    qDebug(DIGIKAM_GENERAL_LOG) << "Album: " << it.current()->title();
    ++it;
}
```

#### Warning

Do not delete albums using this iterator.

## 9.53 Digikam::AlbumLabelsSearchHandler Class Reference

Inheritance diagram for Digikam::AlbumLabelsSearchHandler:



### Signals

- void **checkStateChanged** ([Album](#) \*album, Qt::CheckState checkState)

### Public Member Functions

- **AlbumLabelsSearchHandler** ([LabelsTreeView](#) \*const treeWidget)
- [Album](#) \* **albumForSelectedItems** () const
- QString **generatedName** () const
- QList< QUrl > **imagesUrls** () const  
*Gets the list of images generated, for exporting.*
- bool **isRestoringSelectionFromHistory** () const
- void **restoreSelectionFromHistory** (const QHash< [LabelsTreeView::Labels](#), QList< int > > &neededLabels)  
*Restores the selection of the tree-view from history.*

### 9.53.1 Member Function Documentation

#### 9.53.1.1 albumForSelectedItems()

[Album](#) \* Digikam::AlbumLabelsSearchHandler::albumForSelectedItems ( ) const

#### Returns

[Album](#) pointer of the currently selected labels

### 9.53.1.2 generatedName()

```
QString Digikam::AlbumLabelsSearchHandler::generatedName ( ) const
```

#### Returns

A string for a name generated by

#### See also

[generateAlbumNameForExporting\(\)](#)

### 9.53.1.3 imagesUrls()

```
QList< QUrl > Digikam::AlbumLabelsSearchHandler::imagesUrls ( ) const
```

#### Returns

QUrl List of images Urls

### 9.53.1.4 isRestoringSelectionFromHistory()

```
bool Digikam::AlbumLabelsSearchHandler::isRestoringSelectionFromHistory ( ) const
```

#### Returns

true if the tree-view is restoring the selection state from history to block searching until the restoring is done

### 9.53.1.5 restoreSelectionFromHistory()

```
void Digikam::AlbumLabelsSearchHandler::restoreSelectionFromHistory (
    const QHash< LabelsTreeView::Labels, QList< int > > & neededLabels )
```

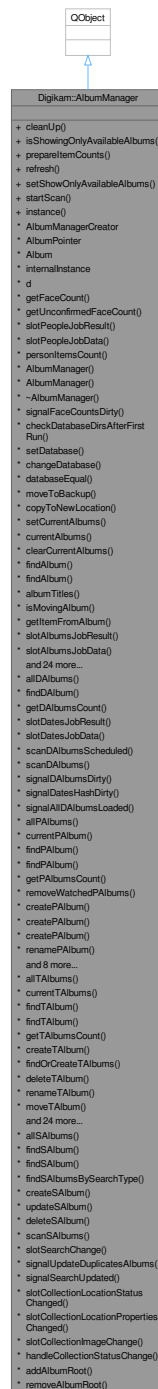
#### Parameters

<i>neededLabels</i>	a hash to restore selection from it
---------------------	-------------------------------------

## 9.54 Digikam::AlbumManager Class Reference

[AlbumManager](#) manages albums: does listing of albums and controls the lifetime of it.

Inheritance diagram for Digikam::AlbumManager:



## Public Member Functions

- void **cleanup** ()

*Stop ongoing operations, prepare for application shutdown.*

- bool **isShowingOnlyAvailableAlbums** () const

- void **prepareItemCounts** ()

*Ensures that valid item counts for physical and tag albums are available.*

- void [refresh](#) ()  
*This is similar to `startScan`, except that it assumes you have run `startScan` at least once.*
- void **setShowOnlyAvailableAlbums** (bool onlyAvailable)
- void [startScan](#) ()  
*starts scanning the `libraryPath` and listing the albums.*

### Static Public Member Functions

- static [AlbumManager](#) \* **instance** ()  
*A convenience function to get the instance of the [AlbumManager](#).*

### Operations on Face Album

- class **AlbumManagerCreator**
- template<class T >  
class **AlbumPointer**
- class **Album**
- QHash< int, int > [getFaceCount](#) () const  
*Returns the latest count for faces as also emitted via `signalFaceCountsDirty`.*
- QHash< int, int > [getUnconfirmedFaceCount](#) () const  
*Returns the latest count for unconfirmed faces only as also emitted via `signalFaceCountsDirty`.*
- void **signalFaceCountsDirty** (const QHash< int, int > &faceCount, const QHash< int, int > &uFaceCount, const QList< int > &toUpdatedFaces)

### Operations with database

- static void **checkDatabaseDirsAfterFirstRun** (const QString &dbPath, const QString &albumPath)  
*Some checks for settings done in first run wizard in case of QSQLite Database.*
- bool [setDatabase](#) (const [DbEngineParameters](#) &params, bool priority, const QString &suggestedAlbumRoot=QString(), bool ignoreDisappearedLocations=false)  
*Initialize.*
- void [changeDatabase](#) (const [DbEngineParameters](#) &params)  
*Sets new database when chosen by the user in setup.*
- bool **databaseEqual** (const [DbEngineParameters](#) &parameters) const  
*Checks if the given database path is equal to the current one.*

### Operations on generic Album

- void [setCurrentAlbums](#) (const QList< [Album](#) \* > &albums)  
*set current album to `albums`.*
- AlbumList [currentAlbums](#) () const
- void **clearCurrentAlbums** ()  
*clear current albums.*
- [Album](#) \* [findAlbum](#) (int gid) const
- [Album](#) \* [findAlbum](#) ([Album::Type](#) type, int id) const
- QHash< int, QString > [albumTitles](#) () const
- bool [isMovingAlbum](#) ([Album](#) \*album) const  
*Returns if the given album is currently being moved, that is, if this album is in between `signalAlbumAboutToBeMoved` and `signalAlbumMoved`.*
- qlonglong [getItemFromAlbum](#) ([Album](#) \*const album, const QString &fileName)

- Returns the id of the item with the given filename in the given *Album*.
- void **signalAlbumAboutToBeAdded** (*Album* \*album, *Album* \*parent, *Album* \*prev)
 

Emitted when an album is about to be added to the given parent (0 if album is root) after the item given by prev (prev is 0 if parent has no children yet).
- void **signalAlbumAdded** (*Album* \*album)
 

Emitted when the album has been added.
- void **signalAlbumAboutToBeDeleted** (*Album* \*album)
 

Emitted when the album is about to be deleted, but is still fully valid.
- void **signalAlbumDeleted** (*Album* \*album)
 

Emitted when the album is deleted, but the object can still be accessed.
- void **signalAlbumHasBeenDeleted** (*Album* \*album)
 

Emitted when the album is deleted, the object can no longer be accessed.
- void **signalAlbumsCleared** ()
- void **signalAlbumCurrentChanged** (const QList< *Album* \* > &albums)
- void **signalAllAlbumsLoaded** ()
- void **signalAlbumIconChanged** (*Album* \*album)
- void **signalAlbumRenamed** (*Album* \*album)
- void **signalAlbumNewPath** (*Album* \*album)
- void **signalAlbumAboutToBeMoved** (*Album* \*album)
 

Emitted when an album is about to be moved.
- void **signalAlbumMoved** (*Album* \*album)
 

Emitted when the album is moved to its new parent.
- void **signalAlbumsUpdated** (int type)
- void **signalShowOnlyAvailableAlbumsChanged** (bool showOnlyAvailableAlbums)
 

Emitted when a change is done on available Albums.

### Operations on Date Album

- AlbumList **allDAAlbums** () const
- *DAAlbum* \* **findDAAlbum** (int id) const
- QMap< YearMonth, int > **getDAAlbumsCount** () const
 

Returns the latest count for DAAlbums as also emitted via *signalDAAlbumsDirty*.
- void **signalDAAlbumsDirty** (const QMap< YearMonth, int > &)
- void **signalDatesHashDirty** (const QHash< QDateTime, int > &)
- void **signalAllDAAlbumsLoaded** ()

### Operations on Physical Album

- AlbumList **allPAAlbums** () const
- *PAAlbum* \* **currentPAAlbum** () const
- *PAAlbum* \* **findPAAlbum** (const QUrl &url) const
 

Given a complete file url (kde url with file protocol), it will try to find a *PAAlbum* corresponding to it.
- *PAAlbum* \* **findPAAlbum** (int id) const
- QHash< int, int > **getPAAlbumsCount** () const
 

Returns the latest count for PAAlbums as also emitted via *signalPAAlbumsDirty*.
- void **removeWatchedPAAlbums** (const *PAAlbum* \*const album)
- *PAAlbum* \* **createPAAlbum** (*PAAlbum* \*parent, const QString &name, const QString &caption, const QDate &date, const QString &category, QString &errMsg)
 

Create a new *PAAlbum* with supplied properties as a child of the parent This is equivalent to creating a new folder on the disk with supplied name in the parent's folder path.

- **PAIbum** \* **createPAIbum** (const QString &albumRootPath, const QString &name, const QString &caption, const QDate &date, const QString &category, QString &errMsg)
 

*Overloaded method.*
- **PAIbum** \* **createPAIbum** (const **CollectionLocation** &location, const QString &name, const QString &caption, const QDate &date, const QString &category, QString &errMsg)
 

*Overloaded method.*
- bool **renamePAIbum** (**PAIbum** \*album, const QString &newName, QString &errMsg)
 

*Renames a PAIbum.*
- bool **updatePAIbumIcon** (**PAIbum** \*album, qlonglong iconID, QString &errMsg)
 

*Update the icon for an album.*
- void **signalPAIbumsDirty** (const QHash< int, int > &)
- void **signalEmptyTrash** ()

### Operations on Tag Album

- AlbumList **allTAIbums** () const
- QList< **TAIbum** \* > **currentTAIbums** () const
 

*This method is not yet used.*
- **TAIbum** \* **findTAIbum** (int id) const
- **TAIbum** \* **findTAIbum** (const QString &tagPath) const
- QHash< int, int > **getTAIbumsCount** () const
 

*Returns the latest count for TAIbums as also emitted via signalTAIbumsDirty.*
- **TAIbum** \* **createTAIbum** (**TAIbum** \*parent, const QString &name, const QString &iconkde, QString &errMsg)
 

*Create a new TAIbum with supplied properties as a child of the parent The tag is added to the database.*
- AlbumList **findOrCreateTAIbums** (const QStringList &tagPaths)
 

*A list of tag paths is supplied.*
- bool **deleteTAIbum** (**TAIbum** \*album, QString &errMsg, QList< qlonglong > \*imageIds=nullptr)
 

*Delete a TAIbum.*
- bool **renameTAIbum** (**TAIbum** \*album, const QString &name, QString &errMsg)
 

*Renames a TAIbum.*
- bool **moveTAIbum** (**TAIbum** \*album, **TAIbum** \*newParent, QString &errMsg)
 

*Move a TAIbum to a new parent.*
- bool **mergeTAIbum** (**TAIbum** \*album, **TAIbum** \*destAlbum, bool dialog, QString &errMsg)
 

*Merge a TAIbum to a TAIbum.*
- bool **updateTAIbumIcon** (**TAIbum** \*album, const QString &iconKDE, qlonglong iconID, QString &errMsg)
 

*Update the icon for a TAIbum.*
- AlbumList **getRecentlyAssignedTags** (bool includeInternal=false) const
 

*Get a list of recently assigned tags (only last 6 tags are listed)*
- QStringList **tagPaths** (const QList< int > &tagIDs, bool leadingSlash=true, bool includeInternal=false) const
 

*Return A list with the tag paths for a list of tag IDs.*
- QStringList **tagNames** (const QList< int > &tagIDs, bool includeInternal=false) const
- QHash< int, QString > **tagPaths** (bool leadingSlash=true, bool includeInternal=false) const
- QHash< int, QString > **tagNames** (bool includeInternal=false) const
- AlbumList **findTagsWithProperty** (const QString &property)
 

*Returns a list of TAIbums which have the given property, or the given property/value combination.*
- AlbumList **findTagsWithProperty** (const QString &property, const QString &value)
- QList< int > **subTags** (int tagId, bool recursive=false) const
 

*TODO.*
- int **findTopId** (int tagId) const
- void **askUserForWriteChangedTAIbumToFiles** (**TAIbum** \*const album)
- void **askUserForWriteChangedTAIbumToFiles** (const QList< qlonglong > &imageIds)
- void **signalTAIbumsDirty** (const QHash< int, int > &)
- void **signalTagPropertiesChanged** (**TAIbum** \*album)



## Operations on Search Album

- AlbumList `allSAlbums` () const
- `SAlbum * findSAlbum` (int id) const
- `SAlbum * findSAlbum` (const QString &name) const
- `QList< SAlbum * > findSAlbumsBySearchType` (int searchType) const
- `SAlbum * createSAlbum` (const QString &name, DatabaseSearch::Type type, const QString &query)
  - Create a new SAlbum with supplied url.*
- bool `updateSAlbum` (`SAlbum *album`, const QString &changedQuery, const QString &changedName=QString(), DatabaseSearch::Type type=DatabaseSearch::UndefinedType)
  - Update the url for a SAlbum.*
- bool `deleteSAlbum` (`SAlbum *album`)
  - Delete a SAlbum from the database.*
- void `signalUpdateDuplicatesAlbums` (const QList< `SAlbum *` > &modifiedAlbums, const QList< qlonglong > &deletedImages)
- void `signalSearchUpdated` (`SAlbum *album`)

### 9.54.1 Detailed Description

For PAlbums and TAlbums, the listing is done by reading the db directly and building the hierarchy of the albums. For DAlbums, since the listing takes time, the work is delegated to a dbjob. Interested frontend entities can connect to the albummanager to receive notifications of new Albums, when Albums are deleted and when the current album is changed.

Additional operations are provided for: creating/deleting/rename Albums, updating icons and moving Albums.

### 9.54.2 Member Function Documentation

#### 9.54.2.1 albumTitles()

```
QHash< int, QString > Digikam::AlbumManager::albumTitles ( ) const
```

##### Returns

A hash with the titles for all album IDs.

#### 9.54.2.2 allDAlbums()

```
AlbumList Digikam::AlbumManager::allDAlbums ( ) const
```

##### Returns

a list of all DAlbums

#### 9.54.2.3 allPAlbums()

```
AlbumList Digikam::AlbumManager::allPAlbums ( ) const
```

##### Returns

a list of all PAlbums

#### 9.54.2.4 allSAlbums()

```
AlbumList Digikam::AlbumManager::allSAlbums ( ) const
```

##### Returns

a list of all SAlbums

#### 9.54.2.5 allTAlbums()

```
AlbumList Digikam::AlbumManager::allTAlbums ( ) const
```

##### Returns

a list of all TAlbums

#### 9.54.2.6 changeDatabase()

```
void Digikam::AlbumManager::changeDatabase (
    const DbEngineParameters & params )
```

Handles user notification about problems. Call this instead of setDatabase when digiKam is up and running.

#### 9.54.2.7 createPAlbum() [1/3]

```
PAlbum * Digikam::AlbumManager::createPAlbum (
    const CollectionLocation & location,
    const QString & name,
    const QString & caption,
    const QDate & date,
    const QString & category,
    QString & errMsg )
```

Here you can supply a collection location (which must be available).

##### Parameters

<i>location</i>	the collection for the new album
<i>name</i>	the name of the new album
<i>caption</i>	the caption for the new album
<i>date</i>	the date for the new album
<i>category</i>	the category for the new album
<i>errMsg</i>	this will contain the error message describing why the operation failed

#### 9.54.2.8 createPAlbum() [2/3]

```
PAlbum * Digikam::AlbumManager::createPAlbum (
```

```

    const QString & albumRootPath,
    const QString & name,
    const QString & caption,
    const QDate & date,
    const QString & category,
    QString & errMsg )

```

Here you can supply an albumRootPath which must correspond to an available collection location.

### 9.54.2.9 createPAlbum() [3/3]

```

PAlbum * Digikam::AlbumManager::createPAlbum (
    PAlbum * parent,
    const QString & name,
    const QString & caption,
    const QDate & date,
    const QString & category,
    QString & errMsg )

```

Also the supplied attributes are written out to the database

#### Note

the signalAlbumAdded will be fired before this function returns. Its recommended to connect to that signal to get notification of new album added

#### Returns

the newly created PAlbum or 0 if it fails

#### Parameters

<i>parent</i>	the parent album under which to create the new Album. Parent must not be root. Otherwise, use the other variants of this method. If parent is root, the albumRootPath must be supplied.
<i>name</i>	the name of the new album
<i>caption</i>	the caption for the new album
<i>date</i>	the date for the new album
<i>category</i>	the category for the new album
<i>errMsg</i>	this will contain the error message describing why the operation failed

### 9.54.2.10 createSAlbum()

```

SAlbum * Digikam::AlbumManager::createSAlbum (
    const QString & name,
    DatabaseSearch::Type type,
    const QString & query )

```

If an existing SAlbum with same name exists this function will return a pointer to that album, instead of creating a new one. A newly created search album is added to the database. For an existing SAlbum, the url is updated and written out to the database

**Note**

the signalAlbumAdded will be fired before this function returns. Its recommended to connect to that signal to get notification of new album added

**Returns**

the newly created [SAlbum](#) or an existing [SAlbum](#) with same name

**Parameters**

<i>name</i>	name for the new search
<i>type</i>	the type of the search
<i>query</i>	search query to use

**9.54.2.11 createTAlbum()**

```
TAlbum * Digikam::AlbumManager::createTAlbum (
    TAlbum * parent,
    const QString & name,
    const QString & iconkde,
    QString & errMsg )
```

**Note**

the signalAlbumAdded will be fired before this function returns. Its recommended to connect to that signal to get notification of new album added

**Returns**

the newly created [TAlbum](#) or 0 if it fails

**Parameters**

<i>parent</i>	the parent album under which to create the new <a href="#">Album</a>
<i>name</i>	the name of the new album
<i>iconkde</i>	the iconkde for the new album (this is a filename which kde iconloader can load up
<i>errMsg</i>	this will contain the error message describing why the operation failed

**9.54.2.12 currentAlbums()**

```
AlbumList Digikam::AlbumManager::currentAlbums ( ) const
```

**Returns**

current albums, previously set up by setCurrentAlbums

### 9.54.2.13 currentPAlbum()

```
PAlbum * Digikam::AlbumManager::currentPAlbum ( ) const
```

#### Returns

the current [PAlbum](#) or null if no one is selected

Temporary fix, to return multiple items, iterate and cast each element

### 9.54.2.14 currentTAlbums()

```
QList< TAlbum * > Digikam::AlbumManager::currentTAlbums ( ) const
```

#### Returns

the current [TAlbum](#) or null if no one is selected

### 9.54.2.15 deleteSAlbum()

```
bool Digikam::AlbumManager::deleteSAlbum (
    SAlbum * album )
```

#### Note

the signalAlbumDeleted will be fired before this function returns. Its recommended to connect to that signal to get notification of album deletes

#### Returns

true if the operation succeeds, false otherwise

#### Parameters

<i>album</i>	the album to delete
--------------	---------------------

### 9.54.2.16 deleteTAlbum()

```
bool Digikam::AlbumManager::deleteTAlbum (
    TAlbum * album,
    QString & errMsg,
    QList< qlonglong > * imageIds = nullptr )
```

The tag is removed from the database

**Note**

the signal `AlbumDeleted` will be fired before this function returns. Its recommended to connect to that signal to get notification of album deletes

**Returns**

true if the operation succeeds or false otherwise

**Parameters**

<i>album</i>	the <code>TAlbum</code> to delete
<i>errMsg</i>	this will contain the error message describing why the
<i>imageIds</i>	list of image ID from the database where tag is removed

**9.54.2.17 findAlbum() [1/2]**

```
Album * Digikam::AlbumManager::findAlbum (
    Album::Type type,
    int id ) const
```

**Returns**

a `Album` with the given type and id

**Parameters**

<i>type</i>	the type of album
<i>id</i>	the id for the album (not the global id)

**9.54.2.18 findAlbum() [2/2]**

```
Album * Digikam::AlbumManager::findAlbum (
    int gid ) const
```

**Returns**

a `Album` with the given globalID

**Parameters**

<i>gid</i>	the global id for the album
------------	-----------------------------

**9.54.2.19 findDAlbum()**

```
DAlbum * Digikam::AlbumManager::findDAlbum (
```

```
int id ) const
```

**Returns**

a [DAAlbum](#) with given ID

**Parameters**

<i>id</i>	the id for the <a href="#">DAAlbum</a>
-----------	--

**9.54.2.20 findOrCreateTAAlbums()**

```
AlbumList Digikam::AlbumManager::findOrCreateTAAlbums (
    const QStringList & tagPaths )
```

If no corresponding [TAAlbum](#) exists, a new one will be created.

**Parameters**

<i>tagPaths</i>	A list of tag paths
-----------------	---------------------

**Returns**

A list of all [TAAlbums](#) for the list (already existing or newly created)

**9.54.2.21 findPAAlbum() [1/2]**

```
PAAlbum * Digikam::AlbumManager::findPAAlbum (
    const QUrl & url ) const
```

**Warning**

This should not be used, unless really necessary

**Returns**

[PAAlbum](#) corresponding to supplied url

**Parameters**

<i>url</i>	the url we need to check
------------	--------------------------

**9.54.2.22 findPAAlbum() [2/2]**

```
PAAlbum * Digikam::AlbumManager::findPAAlbum (
    int id ) const
```

**Returns**

a [PAlbum](#) with given ID

**Parameters**

<i>id</i>	the id for the <a href="#">PAlbum</a>
-----------	---------------------------------------

**9.54.2.23 findSAlbum() [1/2]**

```
SAlbum * Digikam::AlbumManager::findSAlbum (
    const QString & name ) const
```

**Returns**

a [SAlbum](#) with given name, or 0 if not found

**Parameters**

<i>name</i>	the name of the search
-------------	------------------------

**9.54.2.24 findSAlbum() [2/2]**

```
SAlbum * Digikam::AlbumManager::findSAlbum (
    int id ) const
```

**Returns**

a [SAlbum](#) with given ID

**Parameters**

<i>id</i>	the id for the <a href="#">SAlbum</a>
-----------	---------------------------------------

**9.54.2.25 findSAlbumsBySearchType()**

```
QList< SAlbum * > Digikam::AlbumManager::findSAlbumsBySearchType (
    int searchType ) const
```

**Returns**

SAlbums with given type, empty list if not found

**Parameters**

<i>searchType</i>	the type of the search
-------------------	------------------------



### 9.54.2.26 findTAlbum() [1/2]

```
TAlbum * Digikam::AlbumManager::findTAlbum (
    const QString & tagPath ) const
```

#### Returns

a [TAlbum](#) with given tag path, or 0 if not found

#### Parameters

<i>tagPath</i>	the tag path ("People/Friend/John")
----------------	-------------------------------------

### 9.54.2.27 findTAlbum() [2/2]

```
TAlbum * Digikam::AlbumManager::findTAlbum (
    int id ) const
```

#### Returns

a [TAlbum](#) with given ID

#### Parameters

<i>id</i>	the id for the <a href="#">TAlbum</a>
-----------	---------------------------------------

### 9.54.2.28 getDAlbumsCount()

```
 QMap< YearMonth, int > Digikam::AlbumManager::getDAlbumsCount ( ) const
```

#### Returns

count map for DAlbums

### 9.54.2.29 getFaceCount()

```
 QHash< int, int > Digikam::AlbumManager::getFaceCount ( ) const
```

#### Returns

count map for faces (confirmed and unconfirmed combined)

### 9.54.2.30 getItemFromAlbum()

```
 qlonglong Digikam::AlbumManager::getItemFromAlbum (
    Album *const album,
    const QString & fileName )
```

**Parameters**

<i>album</i>	The album in which we search the item.
<i>fileName</i>	The name of the item file.

**Returns**

The item id or -1 if not existent.

**9.54.2.31 getPAlbumsCount()**

```
QHash< int, int > Digikam::AlbumManager::getPAlbumsCount ( ) const
```

**Returns**

count map for PAlbums

**9.54.2.32 getRecentlyAssignedTags()**

```
AlbumList Digikam::AlbumManager::getRecentlyAssignedTags (
    bool includeInternal = false ) const
```

**Returns**

the list of recently assigned TAlbums

**Parameters**

<i>includeInternal</i>	include internal tags in the returned list, or skip them
------------------------	--

**9.54.2.33 getTAlbumsCount()**

```
QHash< int, int > Digikam::AlbumManager::getTAlbumsCount ( ) const
```

**Returns**

count map for TAlbums

**9.54.2.34 getUnconfirmedFaceCount()**

```
QHash< int, int > Digikam::AlbumManager::getUnconfirmedFaceCount ( ) const
```

**Returns**

count map for unconfirmed faces only

**9.54.2.35 isMovingAlbum()**

```
bool Digikam::AlbumManager::isMovingAlbum (
    Album * album ) const
```

In this case, you can preserve state of such an album because the object is guaranteed not to be deleted, even if signalAlbumAboutToBeDeleted is emitted.

**9.54.2.36 mergeTAlbum()**

```
bool Digikam::AlbumManager::mergeTAlbum (
    TAlbum * album,
    TAlbum * destAlbum,
    bool dialog,
    QString & errMsg )
```

This updates the image tags in the database

**Returns**

true if the operation succeeds, false otherwise

**Parameters**

<i>album</i>	the <a href="#">Album</a> which should be merged
<i>destAlbum</i>	the <a href="#">Album</a> to which album should be merged
<i>dialog</i>	show dialog to ask the user if he wants to merge
<i>errMsg</i>	this will contain the error message describing why the operation failed

**9.54.2.37 moveTAlbum()**

```
bool Digikam::AlbumManager::moveTAlbum (
    TAlbum * album,
    TAlbum * newParent,
    QString & errMsg )
```

This updates the tag parent ID in the database

**Returns**

true if the operation succeeds, false otherwise

**Parameters**

<i>album</i>	the <a href="#">Album</a> which should be moved
<i>newParent</i>	the Parent <a href="#">Album</a> to which album should be moved
<i>errMsg</i>	this will contain the error message describing why the operation failed

### 9.54.2.38 refresh()

```
void Digikam::AlbumManager::refresh ( )
```

It checks the database to see if any new albums have been added and updates them accordingly. Use this when a change in the filesystem is detected (but the album library path hasn't changed)

See also

[startScan](#)

### 9.54.2.39 renamePAlbum()

```
bool Digikam::AlbumManager::renamePAlbum (
    PAlbum * album,
    const QString & newName,
    QString & errMsg )
```

This is equivalent to actually renaming the corresponding folder on the disk.

Returns

true if the operation succeeds, false otherwise

Parameters

<i>album</i>	the <a href="#">Album</a> which should be renamed
<i>newName</i>	the new name for the album
<i>errMsg</i>	this will contain the error message describing why the operation failed

### 9.54.2.40 renameTAlbum()

```
bool Digikam::AlbumManager::renameTAlbum (
    TAlbum * album,
    const QString & name,
    QString & errMsg )
```

This updates the tag name in the database

Returns

true if the operation succeeds, false otherwise

Parameters

<i>album</i>	the <a href="#">Album</a> which should be renamed
<i>name</i>	the new name for the album
<i>errMsg</i>	this will contain the error message describing why the operation failed

#### 9.54.2.41 setCurrentAlbums()

```
void Digikam::AlbumManager::setCurrentAlbums (
    const QList< Album * > & albums )
```

Filter out the null pointers

Sort is needed to identify selection correctly, ex [AlbumHistory](#)

#### 9.54.2.42 setDatabase()

```
bool Digikam::AlbumManager::setDatabase (
    const DbEngineParameters & params,
    bool priority,
    const QString & suggestedAlbumRoot = QString(),
    bool ignoreDisappearedLocations = false )
```

Informs the user about failures. Returns true on success, false on failure. A return value of false during startup indicates termination of the program (user is informed)

ignoreDisappearedLocations is intended to be used in tests, because the path of the collection is hardcoded but when executing the test on different computers the collection might not be available at that path

#### 9.54.2.43 signalAlbumAboutToBeMoved

```
void Digikam::AlbumManager::signalAlbumAboutToBeMoved (
    Album * album ) [signal]
```

Signals for deleting and adding will be sent afterwards, but the album object is guaranteed not to be deleted until after signalAlbumMoved.

#### 9.54.2.44 signalAlbumHasBeenDeleted

```
void Digikam::AlbumManager::signalAlbumHasBeenDeleted (
    Album * album ) [signal]
```

For identification purposes, the former album pointer is passed.

#### 9.54.2.45 signalAlbumMoved

```
void Digikam::AlbumManager::signalAlbumMoved (
    Album * album ) [signal]
```

After signalAlbumAboutToBeMoved, all four signals for first deleting and then adding will have been sent.

#### 9.54.2.46 signalShowOnlyAvailableAlbumsChanged

```
void Digikam::AlbumManager::signalShowOnlyAvailableAlbumsChanged (
    bool showsOnlyAvailableAlbums ) [signal]
```

Please note that affected albums may appear or disappear after this signal has been emitted.

**9.54.2.47 startScan()**

```
void Digikam::AlbumManager::startScan ( )
```

If the libraryPath has not changed since the last scan, then nothing happens

See also

[setLibraryPath](#)

[refresh](#)

**9.54.2.48 tagNames() [1/2]**

```
QHash< int, QString > Digikam::AlbumManager::tagNames (
    bool includeInternal = false ) const
```

Returns

A hash with the tag names for all tag IDs.

Parameters

<i>includeInternal</i>	include internal tags in the returned list, or skip them
------------------------	--

**9.54.2.49 tagNames() [2/2]**

```
QStringList Digikam::AlbumManager::tagNames (
    const QList< int > & tagIDs,
    bool includeInternal = false ) const
```

Returns

A list with the tag names for a list of tag IDs.

Parameters

<i>tagIDs</i>	list of tag album IDs
<i>includeInternal</i>	include internal tags in the returned list, or skip them

**9.54.2.50 tagPaths() [1/2]**

```
QHash< int, QString > Digikam::AlbumManager::tagPaths (
    bool leadingSlash = true,
    bool includeInternal = false ) const
```

**Returns**

A hash with the tag paths for all tag IDs.

**Parameters**

<i>leadingSlash</i>	if <code>true</code> return tags with a leading slash
<i>includeInternal</i>	include internal tags in the returned list, or skip them

**9.54.2.51 tagPaths() [2/2]**

```
QStringList Digikam::AlbumManager::tagPaths (
    const QList< int > & tagIDs,
    bool leadingSlash = true,
    bool includeInternal = false ) const
```

**Parameters**

<i>tagIDs</i>	list of tag album IDs
<i>leadingSlash</i>	if <code>true</code> return tags with a leading slash
<i>includeInternal</i>	include internal tags in the returned list, or skip them

**9.54.2.52 updatePAlbumIcon()**

```
bool Digikam::AlbumManager::updatePAlbumIcon (
    PAlbum * album,
    qlonglong iconID,
    QString & errMsg )
```

The `icon` is the name (and not full path) of the file in the album

**Returns**

`true` if the operation succeeds, `false` otherwise

**Parameters**

<i>album</i>	the album for which icon should be changed
<i>iconID</i>	the filename of the new icon
<i>errMsg</i>	if the operation fails, this will contain the error message describing why the operation failed

**9.54.2.53 updateSAlbum()**

```
bool Digikam::AlbumManager::updateSAlbum (
    SAlbum * album,
    const QString & changedQuery,
    const QString & changedName = QString(),
    DatabaseSearch::Type type = DatabaseSearch::UndefinedType )
```

**Returns**

true if the operation succeeds, false otherwise

**Parameters**

<i>album</i>	the album to update
<i>changedQuery</i>	the new query data of the album
<i>changedName</i>	a new name, or null to keep the current name
<i>type</i>	a new type, or UndefinedType to keep the current type

**9.54.2.54 updateTAlbumIcon()**

```
bool Digikam::AlbumManager::updateTAlbumIcon (
    TAlbum * album,
    const QString & iconKDE,
    qlonglong iconID,
    QString & errMsg )
```

**Returns**

true if the operation succeeds, false otherwise

**Parameters**

<i>album</i>	the album for which icon should be changed
<i>iconKDE</i>	a simple filename which can be loaded by KIconLoader
<i>iconID</i>	id of the icon image file
<i>errMsg</i>	this will contain the error message describing why the operation failed

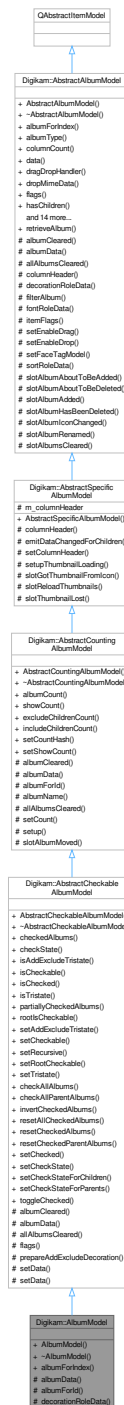
**Note**

if iconKDE is not empty then iconID is used. So if you want to set the icon to a file which can be loaded by QIcon, pass it in as iconKDE. otherwise pass a null QString to iconKDE and set iconID



## 9.55 Digikam::AlbumModel Class Reference

Inheritance diagram for Digikam::AlbumModel:



### Public Member Functions

- **AlbumModel** ([RootAlbumBehavior](#) rootBehavior=[IncludeRootAlbum](#), [QObject](#) \*const parent=nullptr)  
Create a model containing all physical albums.
- **PAAlbum** \* **albumForIndex** (const [QModelIndex](#) &index) const

## Public Member Functions inherited from [Digikam::AbstractCheckableAlbumModel](#)

- [AbstractCheckableAlbumModel](#) ([Album::Type](#) albumType, [Album](#) \*const rootAlbum, [RootAlbumBehavior](#) rootBehavior=[IncludeRootAlbum](#), [QObject](#) \*const parent=nullptr)
 

*Abstract base class that manages the check state of Albums.*
- [QList](#)< [Album](#) \* > **checkedAlbums** () const
 

*Returns a list of album with check state Checked.*
- [Qt::CheckState](#) **checkState** ([Album](#) \*album) const
 

*Returns the check state of the album.*
- bool **isAddExcludeTristate** () const
- bool **isCheckable** () const
- bool **isChecked** ([Album](#) \*album) const
 

*Returns if the given album has the check state Checked.*
- bool **isTristate** () const
- [QList](#)< [Album](#) \* > **partiallyCheckedAlbums** () const
 

*Returns a list of album with partially check state Checked.*
- bool **rootIsCheckable** () const
- void **setAddExcludeTristate** (bool b)
 

*Sets a special tristate mode, which offers the three modes "unchecked", "added" and "excluded", where "excluded" corresponds to partially checked internally, but is reflected in the treeview through the decoration only.*
- void **setCheckable** (bool isCheckable)
 

*Triggers if the albums in this model are checkable.*
- void **setRecursive** (bool recursive)
 

*If an item gets checked, all childs get checked as well, If an item gets unchecked, all childs get unchecked as well.*
- void **setRootCheckable** (bool rootIsCheckable)
 

*Triggers if the root album is checkable.*
- void **setTristate** (bool isTristate)
 

*Triggers if the albums in this model are tristate.*

## Public Member Functions inherited from [Digikam::AbstractCountingAlbumModel](#)

- [AbstractCountingAlbumModel](#) ([Album::Type](#) albumType, [Album](#) \*const rootAlbum, [RootAlbumBehavior](#) rootBehavior=[IncludeRootAlbum](#), [QObject](#) \*const parent=nullptr)
 

*Supports displaying a count alongside the album name in DisplayRole.*
- virtual int **albumCount** ([Album](#) \*album) const
 

*Returns the number of included items for this album.*
- bool **showCount** () const

## Public Member Functions inherited from [Digikam::AbstractSpecificAlbumModel](#)

- [AbstractSpecificAlbumModel](#) ([Album::Type](#) albumType, [Album](#) \*const rootAlbum, [RootAlbumBehavior](#) rootBehavior=[IncludeRootAlbum](#), [QObject](#) \*const parent=nullptr)
 

*Abstract base class, do not instantiate.*

## Public Member Functions inherited from Digikam::AbstractAlbumModel

- [AbstractAlbumModel](#) ([Album::Type](#) albumType, [Album](#) \*const rootAlbum, [RootAlbumBehavior](#) rootBehavior=[IncludeRootAlbum](#), [QObject](#) \*const parent=nullptr)
  - Create an [AbstractAlbumModel](#) object for albums with the given type.*
- [Album](#) \* **albumForIndex** (const [QModelIndex](#) &index) const
  - Returns the album object associated with the given model index.*
- [Album::Type](#) **albumType** () const
  - Returns the [Album::Type](#) of the contained albums.*
- int **columnCount** (const [QModelIndex](#) &parent=[QModelIndex](#)()) const override
- [QVariant](#) **data** (const [QModelIndex](#) &index, int role=[Qt::DisplayRole](#)) const override
- [AlbumModelDragDropHandler](#) \* **dragDropHandler** () const
  - Returns the drag drop handler, or 0 if none is installed.*
- bool **dropMimeData** (const [QMimeData](#) \*data, [Qt::DropAction](#) action, int row, int column, const [QModelIndex](#) &parent) override
- [Qt::ItemFlags](#) **flags** (const [QModelIndex](#) &index) const override
- bool **hasChildren** (const [QModelIndex](#) &parent=[QModelIndex](#)()) const override
- [QVariant](#) **headerData** (int section, [Qt::Orientation](#) orientation, int role=[Qt::DisplayRole](#)) const override
- [QModelIndex](#) **index** (int row, int column, const [QModelIndex](#) &parent=[QModelIndex](#)()) const override
- [QModelIndex](#) **indexForAlbum** ([Album](#) \*album) const
  - Return the [QModelIndex](#) for the given album, or an invalid index if the album is not contained in this model.*
- bool **isFaceTagModel** () const
  - Returns true if the album model a face tag model.*
- [QMimeData](#) \* **mimeData** (const [QModelIndexList](#) &indexes) const override
- [QStringList](#) **mimeTypes** () const override
- [QModelIndex](#) **parent** (const [QModelIndex](#) &index) const override
- [Album](#) \* **rootAlbum** () const
- [RootAlbumBehavior](#) **rootAlbumBehavior** () const
  - Returns the root album behavior set for this model.*
- [QModelIndex](#) **rootAlbumIndex** () const
  - Return the index corresponding to the root album.*
- int **rowCount** (const [QModelIndex](#) &parent=[QModelIndex](#)()) const override
- void **setDragDropHandler** ([AlbumModelDragDropHandler](#) \*handler)
  - Set a drag drop handler.*
- void **setDropIndex** (const [QModelIndex](#) &index)
  - Set current index from [QDragMoveEvent](#).*
- [Qt::DropActions](#) **supportedDropActions** () const override

## Protected Member Functions

- [QVariant](#) **albumData** ([Album](#) \*a, int role) const override
  - For subclassing convenience: A part of the implementation of data()*
- [Album](#) \* **albumForId** (int id) const override
  - need to implement in subclass*
- [QVariant](#) **decorationRoleData** ([Album](#) \*a) const override
  - For subclassing convenience: A part of the implementation of data()*

## Protected Member Functions inherited from [Digikam::AbstractCheckableAlbumModel](#)

- void [albumCleared](#) ([Album](#) \*album) override  
*Notification when an entry is removed.*
- void [allAlbumsCleared](#) () override  
*Notification when all entries are removed.*
- Qt::ItemFlags **flags** (const [QModelIndex](#) &index) const override
- void **prepareAddExcludeDecoration** ([Album](#) \*a, [QPixmap](#) &icon) const  
*If in AddExcludeTristate mode, changes the icon as to indicate the state.*
- bool **setData** (const [QModelIndex](#) &index, const [QVariant](#) &value, int role, bool recursive)
- bool [setData](#) (const [QModelIndex](#) &index, const [QVariant](#) &value, int role=[Qt::EditRole](#)) override

## Protected Member Functions inherited from [Digikam::AbstractCountingAlbumModel](#)

- void [albumCleared](#) ([Album](#) \*album) override  
*Notification when an entry is removed.*
- virtual [QString](#) [albumName](#) ([Album](#) \*a) const  
*Can reimplement in subclass.*
- void [allAlbumsCleared](#) () override  
*Notification when all entries are removed.*
- void **setCount** ([Album](#) \*album, int count)  
*If you do not use setCountHash, excludeChildrenCount and includeChildrenCount, you can set a count here.*
- void **setup** ()  
*Call this method in children class constructors to init signal/slots connections.*

## Protected Member Functions inherited from [Digikam::AbstractSpecificAlbumModel](#)

- [QString](#) [columnHeader](#) () const override  
*For subclassing convenience: A part of the implementation of headerData()*
- void **emitDataChangedForChildren** ([Album](#) \*album)
- void **setColumnHeader** (const [QString](#) &header)
- void **setupThumbnailLoading** ()  
*You need to call this from your constructor if you intend to load the thumbnail facilities of this class.*

## Protected Member Functions inherited from [Digikam::AbstractAlbumModel](#)

- virtual bool [filterAlbum](#) ([Album](#) \*album) const  
*Returns true for those and only those albums that shall be contained in this model.*
- virtual [QVariant](#) [fontRoleData](#) ([Album](#) \*a) const  
*For subclassing convenience: A part of the implementation of data()*
- virtual Qt::ItemFlags **itemFlags** ([Album](#) \*album) const  
*For subclassing convenience: A part of the implementation of itemFlags()*
- void [setEnabledDrag](#) (bool enable)  
*Switch on drag and drop globally for all items.*
- void **setEnabledDrop** (bool enable)
- void **setFaceTagModel** (bool enable)
- virtual [QVariant](#) [sortRoleData](#) ([Album](#) \*a) const  
*For subclassing convenience: A part of the implementation of data()*

## Additional Inherited Members

### Public Types inherited from Digikam::AbstractAlbumModel

- enum `AlbumDataRole` {  
`AlbumTitleRole` = Qt::UserRole , `AlbumTypeRole` = Qt::UserRole + 1 , `AlbumPointerRole` = Qt::UserRole + 2  
, `AlbumIdRole` = Qt::UserRole + 3 ,  
`AlbumGlobalIdRole` = Qt::UserRole + 4 , `AlbumSortRole` = Qt::UserRole + 5 }
  - enum `RootAlbumBehavior` { `IncludeRootAlbum` , `IgnoreRootAlbum` }
- AbstractAlbumModel* is the abstract base class for all models that present *Album* objects as managed by *AlbumManager*.

### Public Slots inherited from Digikam::AbstractCheckableAlbumModel

- void **checkAllAlbums** (const QModelIndex &parent=QModelIndex())  
*Checks all albums beneath the given parent.*
- void **checkAllParentAlbums** (const QModelIndex &child)  
*Checks all parent albums starting at the child, including it.*
- void **invertCheckedAlbums** (const QModelIndex &parent=QModelIndex())  
*Inverts the checked state of all albums under the given parent.*
- void **resetAllCheckedAlbums** ()  
*Resets the checked state of all albums to Qt::Unchecked.*
- void **resetCheckedAlbums** (const QModelIndex &parent=QModelIndex())  
*Resets the checked state of all albums under the given parent.*
- void **resetCheckedParentAlbums** (const QModelIndex &child)  
*Resets the checked state of all parents of the child including it.*
- void **setChecked** (*Album* \*album, bool *isChecked*)  
*Sets the check state of album to Checked or Unchecked.*
- void **setCheckState** (*Album* \*album, Qt::CheckState *state*)  
*Sets the check state of the album.*
- void **setCheckStateForChildren** (*Album* \*album, Qt::CheckState *state*)  
*Sets the checked state recursively for all children of but not for the given album.*
- void **setCheckStateForParents** (*Album* \*album, Qt::CheckState *state*)  
*Sets the checked state recursively for all parents of but not for the given album.*
- void **toggleChecked** (*Album* \*album)  
*Toggles the check state of album between Checked or Unchecked.*

### Public Slots inherited from Digikam::AbstractCountingAlbumModel

- void **excludeChildrenCount** (const QModelIndex &index)  
*Displays only the count of the album, without adding child albums' counts.*
- void **includeChildrenCount** (const QModelIndex &index)  
*Displays sum of the count of the album and child albums' counts.*
- void **setCountHash** (const QHash< int, int > &idCountHash)  
*Enable displaying the count.*
- void **setShowCount** (bool *show*)  
*Call to enable or disable showing the count. Default is false.*

### Signals inherited from [Digikam::AbstractCheckableAlbumModel](#)

- void [checkStateChanged](#) ([Album](#) \*album, Qt::CheckState [checkState](#))  
*Emitted when the check state of an album changes.*

### Signals inherited from [Digikam::AbstractCountingAlbumModel](#)

- void [signalUpdateAlbumCount](#) ([Album](#) \*album)

### Signals inherited from [Digikam::AbstractAlbumModel](#)

- void [rootAlbumAvailable](#) ()  
*This is initialized once after creation, if the root album becomes available, if it was not already available at time of construction.*

### Static Public Member Functions inherited from [Digikam::AbstractAlbumModel](#)

- static [Album](#) \* [retrieveAlbum](#) (const [QModelIndex](#) &index)  
*Returns the album represented by the index.*

### Protected Slots inherited from [Digikam::AbstractCountingAlbumModel](#)

- void [slotAlbumMoved](#) ([Album](#) \*album)

### Protected Slots inherited from [Digikam::AbstractSpecificAlbumModel](#)

- void [slotGotThumbnailFromIcon](#) ([Album](#) \*album, const [QPixmap](#) &thumbnail)
- void [slotReloadThumbnails](#) ()
- void [slotThumbnailLost](#) ([Album](#) \*album)

### Protected Slots inherited from [Digikam::AbstractAlbumModel](#)

- void [slotAlbumAboutToBeAdded](#) ([Album](#) \*album, [Album](#) \*parent, [Album](#) \*prev)
- void [slotAlbumAboutToBeDeleted](#) ([Album](#) \*album)
- void [slotAlbumAdded](#) ([Album](#) \*)
- void [slotAlbumHasBeenDeleted](#) ([Album](#) \*album)
- void [slotAlbumIconChanged](#) ([Album](#) \*album)
- void [slotAlbumRenamed](#) ([Album](#) \*album)
- void [slotAlbumsCleared](#) ()

### Protected Attributes inherited from [Digikam::AbstractSpecificAlbumModel](#)

- [QString](#) [m\\_columnHeader](#)

## 9.55.1 Member Function Documentation

### 9.55.1.1 albumData()

```
QVariant Digikam::AlbumModel::albumData (  
    Album * a,  
    int role ) const [override], [protected], [virtual]
```

#### Note

these can be reimplemented in a subclass

Reimplemented from [Digikam::AbstractCheckableAlbumModel](#).

### 9.55.1.2 albumForId()

```
Album * Digikam::AlbumModel::albumForId (  
    int id ) const [override], [protected], [virtual]
```

Implements [Digikam::AbstractCountingAlbumModel](#).

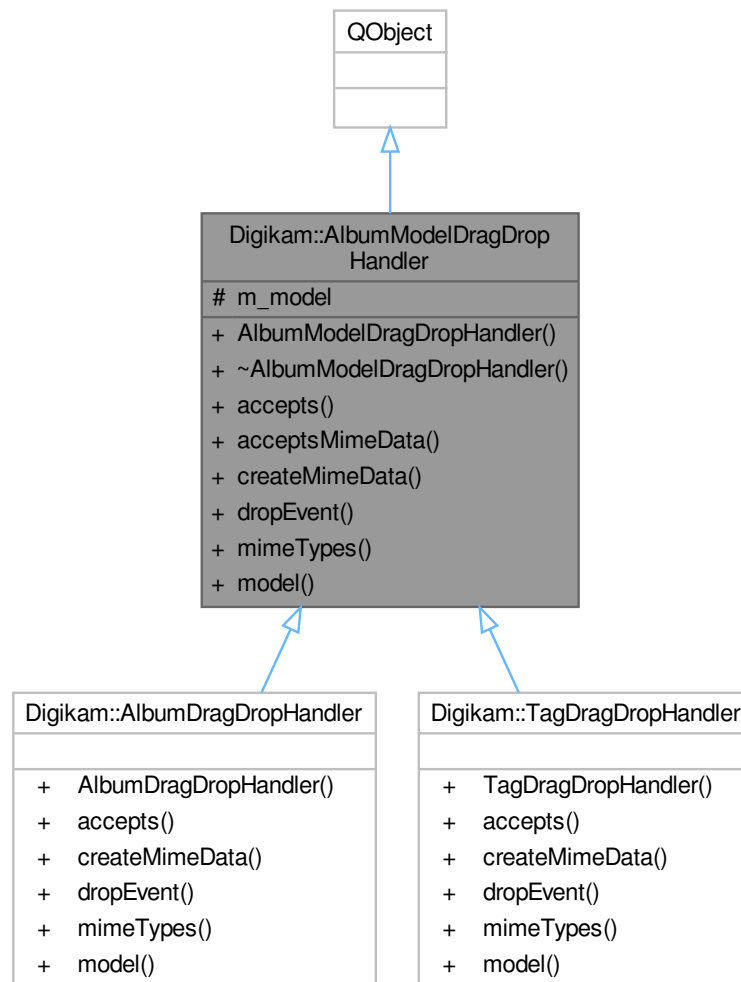
### 9.55.1.3 decorationRoleData()

```
QVariant Digikam::AlbumModel::decorationRoleData (  
    Album * a ) const [override], [protected], [virtual]
```

Reimplemented from [Digikam::AbstractAlbumModel](#).

## 9.56 Digikam::AlbumModelDragDropHandler Class Reference

Inheritance diagram for Digikam::AlbumModelDragDropHandler:



### Public Member Functions

- **AlbumModelDragDropHandler** ([AbstractAlbumModel](#) \*model)
- virtual Qt::DropAction **accepts** (const QDropEvent \*e, const QModelIndex &dropIndex)  
*Returns if the given mime data is accepted for drop on dropIndex.*
- virtual bool **acceptsMimeData** (const QMimeData \*data)  
*Returns if the given mime data can be handled.*
- virtual QMimeData \* **createMimeData** (const QList< [Album](#) \* > &)  
*Create a mime data object for starting a drag from the given Albums.*
- virtual bool **dropEvent** (QAbstractItemView \*view, const QDropEvent \*e, const QModelIndex &droppedOn)  
*Gives the view and the occurring drop event.*
- virtual QStringList **mimeTypees** () const  
*Returns the supported mime types.*
- [AbstractAlbumModel](#) \* **model** () const



## Protected Attributes

- [AbstractAlbumModel](#) \* `m_model` = nullptr

## 9.56.1 Member Function Documentation

### 9.56.1.1 accepts()

```
Qt::DropAction Digikam::AlbumModelDragDropHandler::accepts (
    const QDropEvent * e,
    const QModelIndex & dropIndex ) [virtual]
```

Returns the proposed action, or Qt::IgnoreAction if not accepted.

Reimplemented in [Digikam::AlbumDragDropHandler](#), and [Digikam::TagDragDropHandler](#).

### 9.56.1.2 acceptsMimeData()

```
bool Digikam::AlbumModelDragDropHandler::acceptsMimeData (
    const QMimeData * data ) [virtual]
```

`acceptsMimeData` shall return true if a drop of the given mime data will be accepted on any index or place at all. If this returns false, the more specific method [accepts\(\)](#) will not be called for this drag. The default implementation uses [mimeTypes\(\)](#) to check for supported mime types. There is usually no need to reimplement this.

### 9.56.1.3 createMimeData()

```
QMimeData * Digikam::AlbumModelDragDropHandler::createMimeData (
    const QList< Album * > & ) [virtual]
```

Reimplemented in [Digikam::AlbumDragDropHandler](#), and [Digikam::TagDragDropHandler](#).

### 9.56.1.4 dropEvent()

```
bool Digikam::AlbumModelDragDropHandler::dropEvent (
    QAbstractItemView * view,
    const QDropEvent * e,
    const QModelIndex & droppedOn ) [virtual]
```

The index is the index where the drop was dropped on. It may be invalid (dropped on decoration, viewport) Returns true if the event is to be accepted.

Reimplemented in [Digikam::AlbumDragDropHandler](#), and [Digikam::TagDragDropHandler](#).

### 9.56.1.5 mimeTypes()

```
QStringList Digikam::AlbumModelDragDropHandler::mimeTypes ( ) const [virtual]
```

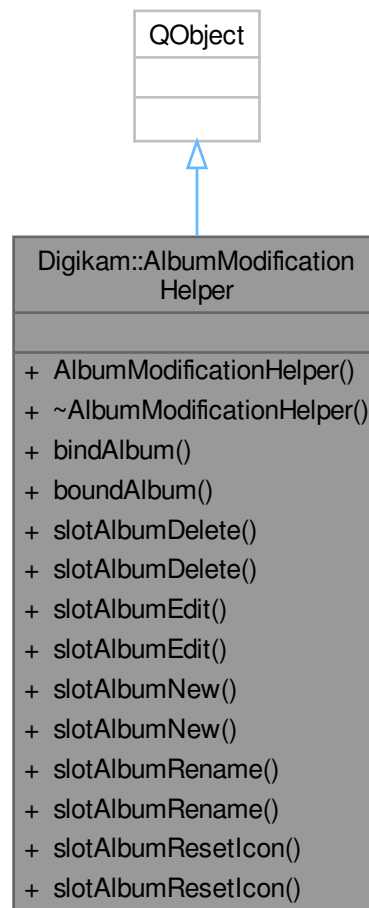
Called by the default implementation of model's [mimeTypes\(\)](#).

Reimplemented in [Digikam::AlbumDragDropHandler](#), and [Digikam::TagDragDropHandler](#).

## 9.57 Digikam::AlbumModificationHelper Class Reference

Utility class providing methods to modify physical albums ([PAlbum](#)) in a way useful to implement views.

Inheritance diagram for Digikam::AlbumModificationHelper:



### Public Slots

- void **slotAlbumDelete** ()
- void [slotAlbumDelete](#) ([PAlbum](#) \*album)
  - Deletes the given album after waiting for a graphical confirmation of the user.*
- void **slotAlbumEdit** ()
- void [slotAlbumEdit](#) ([PAlbum](#) \*album)
  - Graphically edits the properties of the given album.*
- [PAlbum](#) \* **slotAlbumNew** ()
- [PAlbum](#) \* [slotAlbumNew](#) ([PAlbum](#) \*parentAlbum)
  - Creates a new album under the given parent.*
- void **slotAlbumRename** ()

- void `slotAlbumRename` (`PAlbum *album`)  
*Renames the given album.*
- void `slotAlbumResetIcon` ()
- void `slotAlbumResetIcon` (`PAlbum *album`)

## Public Member Functions

- `AlbumModificationHelper` (`QObject *const parent`, `QWidget *const dialogParent`)  
*Constructor.*
- `~AlbumModificationHelper` () override  
*Destructor.*
- void `bindAlbum` (`QAction *const action`, `PAlbum *const parent`) const  
*Sets the album that the given action operates on.*
- `PAlbum * boundAlbum` (`QObject *const action`) const  
*Returns the album bound with bindAlbum.*

## 9.57.1 Detailed Description

### Author

jwienke

## 9.57.2 Constructor & Destructor Documentation

### 9.57.2.1 AlbumModificationHelper()

```
Digikam::AlbumModificationHelper::AlbumModificationHelper (
    QObject *const parent,
    QWidget *const dialogParent ) [explicit]
```

#### Parameters

<i>parent</i>	the parent for qt parent child mechanism
<i>dialogParent</i>	parent widget for dialogs displayed by this object

## 9.57.3 Member Function Documentation

### 9.57.3.1 bindAlbum()

```
void Digikam::AlbumModificationHelper::bindAlbum (
    QAction *const action,
    PAlbum *const parent ) const
```

You must call `bindTag` and then connect the action's triggered to the desired slot, `slotTagNew()`, `slotTagEdit()` or `slotTagDelete()`. Note: Changes the Action's user data.

### 9.57.3.2 boundAlbum()

```
PAlbum * Digikam::AlbumModificationHelper::boundAlbum (
    QObject *const action ) const
```

The given QObject shall be a QAction, but for convenience the given object will be checked with qobject\_cast first, so you can pass QObject::sender().

### 9.57.3.3 slotAlbumDelete

```
void Digikam::AlbumModificationHelper::slotAlbumDelete (
    PAlbum * album ) [slot]
```

#### Parameters

<i>album</i>	the album to delete
--------------	---------------------

### 9.57.3.4 slotAlbumEdit

```
void Digikam::AlbumModificationHelper::slotAlbumEdit (
    PAlbum * album ) [slot]
```

#### Parameters

<i>album</i>	the album to edit
--------------	-------------------

### 9.57.3.5 slotAlbumNew

```
PAlbum * Digikam::AlbumModificationHelper::slotAlbumNew (
    PAlbum * parentAlbum ) [slot]
```

The user will be prompted for the settings of the new album.

#### Parameters

<i>parentAlbum</i>	parent album for the new one
--------------------	------------------------------

#### Returns

the new album or 0 if no album was created

### 9.57.3.6 slotAlbumRename

```
void Digikam::AlbumModificationHelper::slotAlbumRename (
    PAlbum * album ) [slot]
```

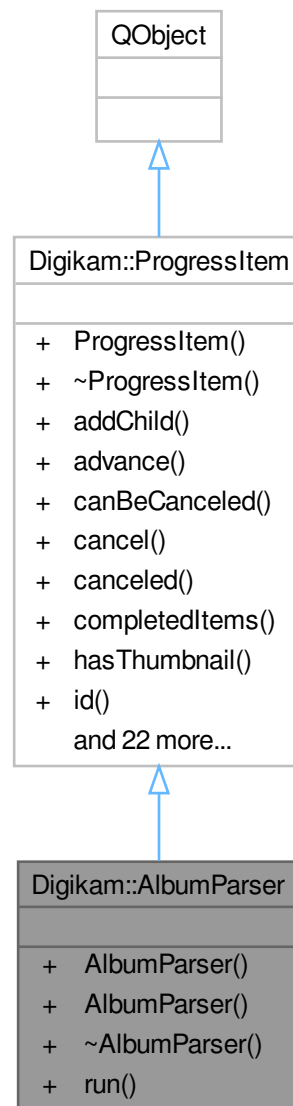
The user will be prompted for a new name.

## Parameters

<i>album</i>	the album to rename
--------------	---------------------

## 9.58 Digikam::AlbumParser Class Reference

Inheritance diagram for Digikam::AlbumParser:



### Signals

- void **signalComplete** (const QList< QUrl > &)

## Signals inherited from [Digikam::ProgressItem](#)

- void [progressItemAdded](#) ([ProgressItem](#) \*item)  
*Emitted when a new [ProgressItem](#) is added.*
- void [progressItemCanceled](#) ([ProgressItem](#) \*item)  
*Emitted when an item was canceled.*
- void [progressItemCanceledById](#) (const QString &id)
- void [progressItemCompleted](#) ([ProgressItem](#) \*item)  
*Emitted when a progress item was completed.*
- void [progressItemLabel](#) ([ProgressItem](#) \*item, const QString &label)  
*Emitted when the label of an item changed.*
- void [progressItemProgress](#) ([ProgressItem](#) \*item, unsigned int v)  
*Emitted when the progress value of an item changes.*
- void [progressItemStatus](#) ([ProgressItem](#) \*item, const QString &mess)  
*Emitted when the status message of an item changed.*
- void [progressItemThumbnail](#) ([ProgressItem](#) \*item, const QPixmap &thumb)  
*Emitted when the thumbnail data must be set in item.*
- void [progressItemUsesBusyIndicator](#) ([ProgressItem](#) \*item, bool value)  
*Emitted when the busy indicator state of an item changes.*

## Public Member Functions

- [AlbumParser](#) ([Album](#) \*const album)  
*Constructor to work on recursive mode from album.*
- [AlbumParser](#) (const [ItemInfoList](#) &infoList)  
*Constructor to work on image list.*
- void [run](#) ()

## Public Member Functions inherited from [Digikam::ProgressItem](#)

- [ProgressItem](#) ([ProgressItem](#) \*const parent, const QString &id, const QString &label, const QString &status, bool canBeCanceled, bool hasThumb)
- void [addChild](#) ([ProgressItem](#) \*const kiddo)
- bool [advance](#) (unsigned int v)  
*Advance total items processed by n values and update percentage in progressbar.*
- bool [canBeCanceled](#) () const
- void [cancel](#) ()
- bool [canceled](#) () const
- unsigned int [completedItems](#) () const
- bool [hasThumbnail](#) () const
- const QString & [id](#) () const
- bool [incCompletedItems](#) (unsigned int v=1)
- void [incTotalItems](#) (unsigned int v=1)
- const QString & [label](#) () const
- [ProgressItem](#) \* [parent](#) () const
- unsigned int [progress](#) () const
- void [removeChild](#) ([ProgressItem](#) \*const kiddo)
- void [reset](#) ()  
*Reset the progress value of this item to 0 and the status string to the empty string.*
- void [setComplete](#) ()  
*Tell the item it has finished.*

- bool **setCompletedItems** (unsigned int v)
- void **setLabel** (const QString &v)
- void **setProgress** (unsigned int v)
  - Set the progress (percentage of completion) value of this item.*
- void **setShowAtStart** (bool showAtStart)
  - Set the property to pop-up item when it's added in progress manager.*
- void **setStatus** (const QString &v)
  - Set the string to be used for showing this item's current status.*
- void **setThumbnail** (const QIcon &icon)
  - Sets whether this item has a thumbnail.*
- void **setTotalItems** (unsigned int v)
- void **setUsesBusyIndicator** (bool useBusyIndicator)
  - Sets whether this item uses a busy indicator instead of real progress for its progress bar.*
- bool **showAtStart** () const
- const QString & **status** () const
- bool **totalCompleted** () const
- unsigned int **totalItems** () const
- void **updateProgress** ()
  - Recalculate progress according to total/completed items and update.*
- bool **usesBusyIndicator** () const

## 9.59 Digikam::AlbumPointer< T > Class Template Reference

You can use [AlbumPointer](#) to store a guarded pointer to [Album](#) or one of the subclasses (use template parameter).

### Public Member Functions

- **AlbumPointer** (const [AlbumPointer](#)< T > &p)
- **AlbumPointer** (T \*const a)
- **operator T\*** () const
- bool **operator!** () const
- T & **operator\*** () const
- T \* **operator->** () const
- [AlbumPointer](#)< T > & **operator=** (const [AlbumPointer](#)< T > &p)
- [AlbumPointer](#)< T > & **operator=** (T \*const a)

### Friends

- class **AlbumManager**

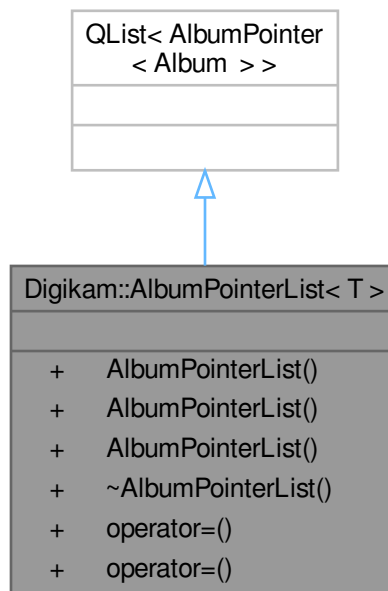
### 9.59.1 Detailed Description

```
template<class T = Album>
class Digikam::AlbumPointer< T >
```

The pointer will be set to 0 when the album object is deleted.

## 9.60 Digikam::AlbumPointerList< T > Class Template Reference

Inheritance diagram for Digikam::AlbumPointerList< T >:



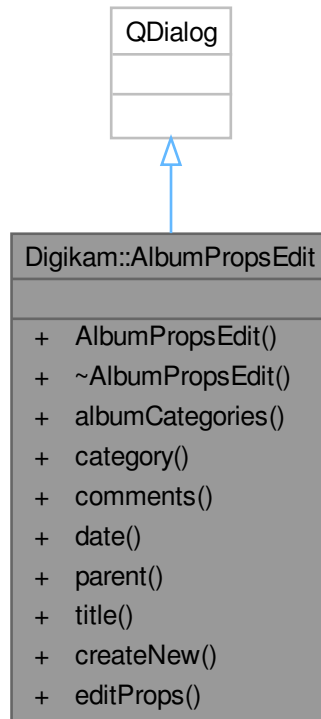
### Public Member Functions

- `AlbumPointerList` (const [AlbumPointerList< T >](#) &list)
- `AlbumPointerList` (const [QList< T \\* >](#) &list)
- [AlbumPointerList< T >](#) & **operator=** (const [AlbumPointerList< T >](#) &list)
- [AlbumPointerList< T >](#) & **operator=** (const [QList< T \\* >](#) &list)



## 9.61 Digikam::AlbumPropsEdit Class Reference

Inheritance diagram for Digikam::AlbumPropsEdit:



### Public Member Functions

- **AlbumPropsEdit** ([PAlbum](#) \*const album, bool create=false)
- QStringList **albumCategories** () const
- QString **category** () const
- QString **comments** () const
- QDate **date** () const
- int **parent** () const
- QString **title** () const

### Static Public Member Functions

- static bool **createNew** ([PAlbum](#) \*const parent, QString &title, QString &comments, QDate &date, QString &category, QStringList &albumCategories, int &parentSelector)
- static bool **editProps** ([PAlbum](#) \*const album, QString &title, QString &comments, QDate &date, QString &category, QStringList &albumCategories)

## 9.62 Digikam::AlbumRootChangeset Class Reference

### Public Types

- enum **Operation** { **Unknown** , **Added** , **Deleted** , **PropertiesChanged** }

### Public Member Functions

- **AlbumRootChangeset** (int albumRootId, Operation operation)
- int **albumRootId** () const
- Operation **operation** () const
- **AlbumRootChangeset** & **operator**<< (const QDBusArgument &argument)
- const **AlbumRootChangeset** & **operator**>> (QDBusArgument &argument) const

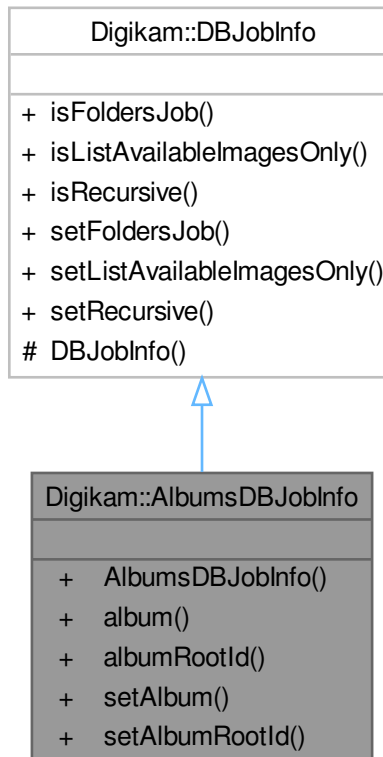
## 9.63 Digikam::AlbumRootInfo Class Reference

### Public Attributes

- int **caseSensitivity** = 0
- int **id** = 0
- QString **identifier**
- QString **label**
- QString **specificPath**
- int **status** = 0
- int **type** = 0

## 9.64 Digikam::AlbumsDBJobInfo Class Reference

Inheritance diagram for Digikam::AlbumsDBJobInfo:



### Public Member Functions

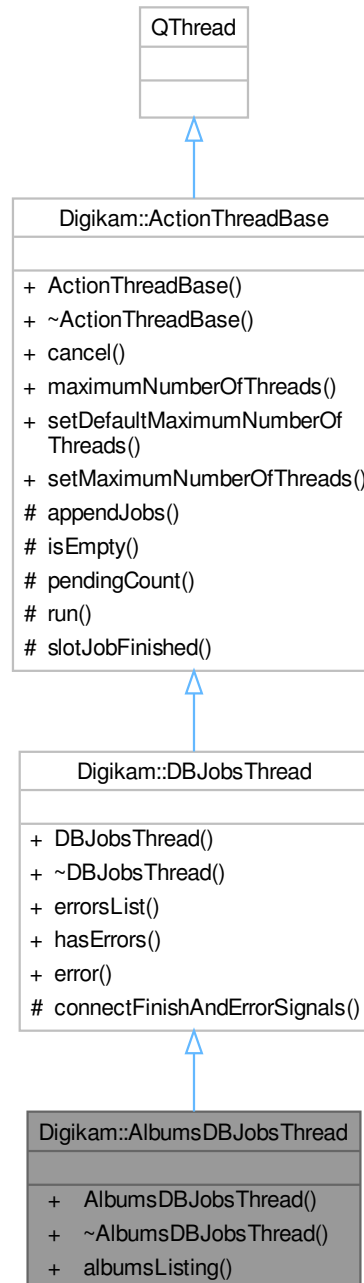
- `QString album ()`
- `int albumRootId ()`
- `void setAlbum (const QString &album)`
- `void setAlbumRootId (int id)`

### Public Member Functions inherited from [Digikam::DBJobInfo](#)

- `bool isFoldersJob () const`
- `bool isListAvailableImagesOnly () const`
- `bool isRecursive () const`
- `void setFoldersJob ()`
- `void setListAvailableImagesOnly ()`
- `void setRecursive ()`

## 9.65 Digikam::AlbumsDBJobsThread Class Reference

Inheritance diagram for Digikam::AlbumsDBJobsThread:



### Signals

- void **faceFoldersData** (const QMap< QString, QHash< int, int > > &)
- void **foldersData** (const QHash< int, int > &)

## Signals inherited from [Digikam::DBJobsThread](#)

- void **data** (const QList< [ItemLISTERRecord](#) > &records)
- void **finished** ()

## Public Member Functions

- **AlbumsDBJobsThread** (QObject \*const parent)
- void **albumsListing** (const [AlbumsDBJobInfo](#) &info)  
*Starts PAlbums listing and scanning job(s)*

## Public Member Functions inherited from [Digikam::DBJobsThread](#)

- **DBJobsThread** (QObject \*const parent)
- QList< QString > & **errorsList** ()  
*A method to get all errors reported from jobs.*
- bool **hasErrors** ()  
*hasErrors: a method to check for jobs errors*

## Public Member Functions inherited from [Digikam::ActionThreadBase](#)

- **ActionThreadBase** (QObject \*const parent=nullptr)
- void **cancel** (bool isCancel=true)  
*Cancel processing of current jobs under progress.*
- int **maximumNumberOfThreads** () const  
*Return the maximum number of threads used to parallelize collection of job processing.*
- void **setDefaultMaximumNumberOfThreads** ()  
*Reset maximum number of threads used to parallelize collection of job processing to max core detected on computer.*
- void **setMaximumNumberOfThreads** (int n)  
*Adjust maximum number of threads used to parallelize collection of job processing.*

## Additional Inherited Members

## Public Slots inherited from [Digikam::DBJobsThread](#)

- void **error** (const QString &errString)  
*Appends the error string to m\_errorsList.*

## Protected Slots inherited from [Digikam::ActionThreadBase](#)

- void **slotJobFinished** ()

## Protected Member Functions inherited from [Digikam::DBJobsThread](#)

- void **connectFinishAndErrorSignals** (DBJob \*const j)  
*Connects the signals of job to the signals of the thread.*

## Protected Member Functions inherited from [Digikam::ActionThreadBase](#)

- void [appendJobs](#) (const [ActionJobCollection](#) &jobs)  
*Append a collection of jobs to process into QThreadPool.*
- bool [isEmpty](#) () const  
*Return true if list of pending jobs to process is empty.*
- int [pendingCount](#) () const  
*Return the number of pending jobs to process.*
- void [run](#) () override  
*Main thread loop used to process jobs in todo list.*

### 9.65.1 Member Function Documentation

#### 9.65.1.1 albumsListing()

```
void Digikam::AlbumsDBJobsThread::albumsListing (  
    const AlbumsDBJobInfo & info )
```

##### Parameters

<i>info</i>	represents the albums job info
-------------	--------------------------------

## 9.66 Digikam::AlbumSelectComboBox Class Reference

Inheritance diagram for Digikam::AlbumSelectComboBox:



### Public Slots

- void **hidePopup** () override
- virtual void **updateText** ()

*Updates the text describing the selection ("3 Albums selected").*

## Public Member Functions

- **AlbumSelectComboBox** (QWidget \*const parent=nullptr)
- QSortFilterProxyModel \* **filterModel** () const  
*Return the filter model in use.*
- bool **isCheckable** () const
- **AbstractCheckableAlbumModel** \* **model** () const  
*Returns the source model.*
- void **setAlbumModels** (**AbstractCheckableAlbumModel** \*model, **AlbumFilterModel** \*filterModel=nullptr)
- void **setAllSelectedText** (bool all)  
*Enable or disable the text used to describe the status when all album is selected.*
- void **setCheckable** (bool checkable)  
*Enable checkboxes next to the items.*
- void **setCloseOnActivate** (bool close)  
*Enable closing when an item was activated (clicked).*
- void **setDefaultAlbumModel** ()  
*Once after creation, call one of these three methods.*
- void **setDefaultTagModel** ()
- void **setNoSelectionText** (const QString &text)  
*Sets the text that is used to describe the state when no album is selected.*
- void **setRecursive** (bool recursive)  
*If all subalbums shall be selected when parent will be selected.*
- void **setShowCheckStateSummary** (bool show)  
*If the box is checkable, enable showing a resume a la "3 Albums checked" in the combo box text.*

## Public Member Functions inherited from [Digikam::TreeViewLineEditComboBox](#)

- [TreeViewLineEditComboBox](#) (QWidget \*const parent=nullptr)  
*This class provides a [TreeViewComboBox](#) with a read-only line edit.*
- void **installView** (QAbstractItemView \*view=nullptr) override  
*Replace the standard combo box list view with a QTreeView.*
- void **setLineEdit** (QLineEdit \*edit)
- void **setLineEditText** (const QString &text)  
*Set the text of the line edit (the text that is visible if the popup is not opened).*

## Public Member Functions inherited from [Digikam::TreeViewComboBox](#)

- [TreeViewComboBox](#) (QWidget \*parent=nullptr)  
*This class provides a QComboBox with a QTreeView instead of the usual QListView.*
- QTreeView \* **view** () const  
*Returns the QTreeView of this class.*

## Public Member Functions inherited from [Digikam::StayPoppedUpComboBox](#)

- [StayPoppedUpComboBox](#) (QWidget \*const parent=nullptr)  
*This class provides an abstract QComboBox with a custom view (which is created by implementing subclasses) instead of the usual QListView.*



**Public Member Functions inherited from [Digikam::ModelIndexBasedComboBox](#)**

- [ModelIndexBasedComboBox](#) (QWidget \*const parent=nullptr)  
*QComboBox has a current index based on a single integer.*
- QModelIndex **currentIndex** () const
- void **hidePopup** () override
- void **setCurrentIndex** (const QModelIndex &index)
- void **showPopup** () override

**Protected Member Functions**

- void [installView](#) (QAbstractItemView \*view=nullptr) override  
*Replace the standard combo box list view with a QTreeView.*

**Protected Member Functions inherited from [Digikam::TreeViewLineEditComboBox](#)**

- virtual void [installLineEdit](#) ()  
*Sets a line edit.*

**Protected Member Functions inherited from [Digikam::TreeViewComboBox](#)**

- void [sendViewportEventToView](#) (QEvent \*e) override  
*Implement in subclass: Send the given event to the viewportEvent() method of m\_view.*

**Protected Member Functions inherited from [Digikam::StayPoppedUpComboBox](#)**

- bool **eventFilter** (QObject \*watched, QEvent \*event) override
- void [installView](#) (QAbstractItemView \*view)  
*Replace the standard combo box list view with the given view.*

**Additional Inherited Members****Protected Attributes inherited from [Digikam::TreeViewLineEditComboBox](#)**

- QLineEdit \* **m\_comboLineEdit** = nullptr

**Protected Attributes inherited from [Digikam::StayPoppedUpComboBox](#)**

- QAbstractItemView \* **m\_view** = nullptr

**Protected Attributes inherited from [Digikam::ModelIndexBasedComboBox](#)**

- QPersistentModelIndex **m\_currentIndex**

## 9.66.1 Member Function Documentation

### 9.66.1.1 installView()

```
void Digikam::AlbumSelectComboBox::installView (
    QAbstractItemView * view = nullptr ) [override], [protected], [virtual]
```

Call this after installing an appropriate model.

Reimplemented from [Digikam::TreeViewComboBox](#).

### 9.66.1.2 model()

```
AbstractCheckableAlbumModel * Digikam::AlbumSelectComboBox::model ( ) const
```

Retrieve selection information from here.

### 9.66.1.3 setCheckable()

```
void Digikam::AlbumSelectComboBox::setCheckable (
    bool checkable )
```

Default: true

### 9.66.1.4 setCloseOnActivate()

```
void Digikam::AlbumSelectComboBox::setCloseOnActivate (
    bool close )
```

Default: false.

### 9.66.1.5 setDefaultAlbumModel()

```
void Digikam::AlbumSelectComboBox::setDefaultAlbumModel ( )
```

Use the first one if you want a standard combo box for PAbums and the second one for tags, while the third allows you to provide custom source and filter models. The first two also set a default noSelectionText. Customize afterwards if required.

### 9.66.1.6 setNoSelectionText()

```
void Digikam::AlbumSelectComboBox::setNoSelectionText (
    const QString & text )
```

This may be something like "Any album" or "No tag selected". Depends on the default line edit implementation of [TreeViewLineEditComboBox](#).

### 9.66.1.7 setShowCheckStateSummary()

```
void Digikam::AlbumSelectComboBox::setShowCheckStateSummary (
    bool show )
```

Default: True

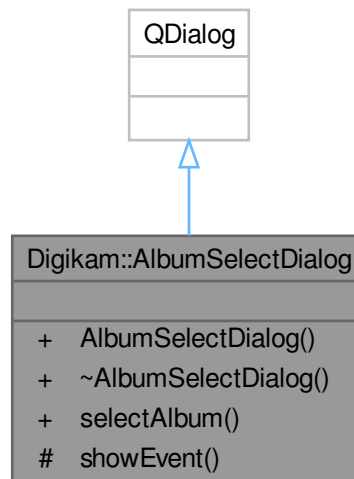
### 9.66.1.8 updateText

```
void Digikam::AlbumSelectComboBox::updateText ( ) [virtual], [slot]
```

Can be overridden to customize the default text.

## 9.67 Digikam::AlbumSelectDialog Class Reference

Inheritance diagram for Digikam::AlbumSelectDialog:



### Public Member Functions

- **AlbumSelectDialog** (QWidget \*const parent, [PAlbum](#) \*const albumToSelect, const QString &header=QString())

### Static Public Member Functions

- static [PAlbum](#) \* **selectAlbum** (QWidget \*const parent, [PAlbum](#) \*const albumToSelect, const QString &header=QString())

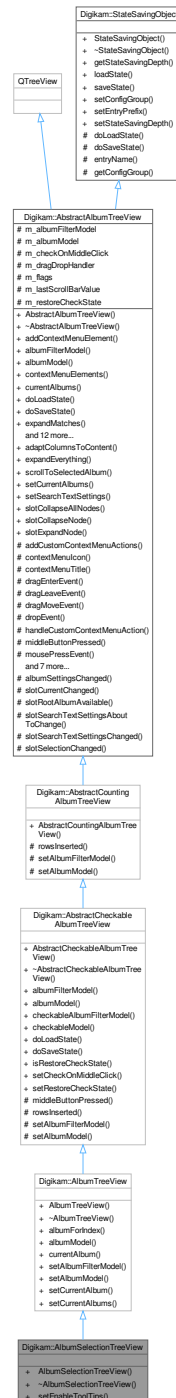
## Protected Member Functions

- void **showEvent** (QShowEvent \*) override

## 9.68 Digikam::AlbumSelectionTreeView Class Reference

[Album](#) tree view used in the left sidebar to select PAAlbums and perform operations on them via a context menu.

Inheritance diagram for Digikam::AlbumSelectionTreeView:



## Signals

- void [signalFindDuplicates](#) (const QList< [PAlbum](#) \* > &albums)  
*Emitted if a find duplicates search shall be invoked on the given album.*

## Signals inherited from [Digikam::AbstractAlbumTreeView](#)

- void **currentAlbumChanged** ([Album](#) \*currentAlbum)  
*Emitted when the currently selected album changes.*
- void [selectedAlbumsChanged](#) (const QList< [Album](#) \* > &selectedAlbums)  
*Emitted when the current selection changes.*

## Public Member Functions

- **AlbumSelectionTreeView** (QWidget \*const parent, [AlbumModel](#) \*const model, [AlbumModificationHelper](#) \*const albumModificationHelper)
- void **setEnabledToolTips** (bool enable)  
*Sets whether this widget shall display tool tips or not.*

## Public Member Functions inherited from [Digikam::AlbumTreeView](#)

- **AlbumTreeView** (QWidget \*const parent=nullptr, [Flags](#) flags=DefaultFlags)
- [PAlbum](#) \* **albumForIndex** (const [QModelIndex](#) &index) const
- [AlbumModel](#) \* **albumModel** () const
- [PAlbum](#) \* **currentAlbum** () const
- void **setAlbumFilterModel** ([CheckableAlbumFilterModel](#) \*const filterModel)
- void **setAlbumModel** ([AlbumModel](#) \*const model)

## Public Member Functions inherited from [Digikam::AbstractCheckableAlbumTreeView](#)

- [AbstractCheckableAlbumTreeView](#) (QWidget \*const parent, [Flags](#) flags)  
*Models of these view can be checkable, they need not.*
- [CheckableAlbumFilterModel](#) \* **albumFilterModel** () const
- [AbstractCheckableAlbumModel](#) \* **albumModel** () const  
*Manage check state through the model directly.*
- [CheckableAlbumFilterModel](#) \* **checkableAlbumFilterModel** () const
- [AbstractCheckableAlbumModel](#) \* **checkableModel** () const
- void **doLoadState** () override  
*Implements state loading for the album tree view in a somewhat clumsy procedure because the model may not be fully loaded when this method is called.*
- void **doSaveState** () override  
*Implement this hook method for state saving.*
- bool **isRestoreCheckState** () const  
*Tells if the check state is restored while loading / saving state.*
- void **setCheckOnMiddleClick** (bool doThat)  
*Enable checking on middle mouse button click (default: on).*
- void **setRestoreCheckState** (bool restore)  
*Set whether to restore check state or not.*

## Public Member Functions inherited from [Digikam::AbstractCountingAlbumTreeView](#)

- [AbstractCountingAlbumTreeView](#) (QWidget \*const parent, Flags flags)

## Public Member Functions inherited from [Digikam::AbstractAlbumTreeView](#)

- [AbstractAlbumTreeView](#) (QWidget \*const parent, Flags flags)  
*Constructs an album tree view.*
- void **addContextMenuElement** ([ContextMenuElement](#) \*const element)
- [AlbumFilterModel](#) \* **albumFilterModel** () const
- [AbstractSpecificAlbumModel](#) \* **albumModel** () const
- QList< [ContextMenuElement](#) \* > **contextMenuElements** () const
- template<class A >  
QList< A \* > **currentAlbums** ()
- bool **expandMatches** (const QModelIndex &index)  
*Ensures that every current match is visible by expanding all parent entries.*
- QModelIndex **indexVisuallyAt** (const QPoint &p)  
*This is a combination of `indexAt()` checked with `visualRect()`.*
- void **removeContextMenuElement** ([ContextMenuElement](#) \*const element)
- QList< [Album](#) \* > **selectedItems** ()  
*selectedItems()* -
- void **setAlbumManagerCurrentAlbum** (const bool setCurrentAlbum)  
*Some treeviews shall control the global current album kept by `AlbumManager`.*
- void **setContextMenuIcon** (const QPixmap &pixmap)  
*Set the context menu title and icon.*
- void **setContextMenuTitle** (const QString &title)
- void **setEnabledContextMenu** (const bool enable)  
*Determines the global decision to show a popup menu or not.*
- void **setExpandNewCurrentItem** (const bool doThat)  
*Expand an item when making it the new current item.*
- void **setExpandOnSingleClick** (const bool doThat)  
*Enable expanding of tree items on single click on the item (default: off)*
- void **setSelectAlbumOnClick** (const bool selectOnClick)  
*Sets whether to select an album on click via the album manager or not.*
- void **setSelectOnContextMenu** (const bool select)  
*Sets whether to select the album under the mouse cursor on a context menu request (so that the album is shown using the album manager) or not.*
- bool **viewportEvent** (QEvent \*event) override  
*For internal use only.*

## Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (QObject \*const host)  
*Constructor.*
- virtual ~[StateSavingObject](#) ()  
*Destructor.*
- [StateSavingDepth](#) **getStateSavingDepth** () const  
*Returns the depth used for state saving or loading.*
- void **loadState** ()  
*Invokes loading the class' state.*

- void **saveState** ()  
*Invokes saving the class' state.*
- virtual void **setConfigGroup** (const KConfigGroup &group)  
*Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void **setEntryPrefix** (const QString &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void **setStateSavingDepth** (const [StateSavingDepth](#) depth)  
*Sets the depth used for state saving or loading.*

### Additional Inherited Members

### Public Types inherited from [Digikam::AbstractAlbumTreeView](#)

- enum [Flag](#) {  
[CreateDefaultModel](#) , [CreateDefaultFilterModel](#) , [CreateDefaultDelegate](#) , [ShowCountAccordingToSettings](#) ,  
[AlwaysShowInclusiveCounts](#) , **DefaultFlags** = [CreateDefaultFilterModel](#) | [CreateDefaultDelegate](#) | Show↔  
CountAccordingToSettings }
- typedef QFlags< [Flag](#) > **Flags**

### Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }  
*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

### Public Slots inherited from [Digikam::AlbumTreeView](#)

- void **setCurrentAlbum** (int albumId, bool selectInAlbumManager=true)
- void **setCurrentAlbums** (const QList< [Album](#) \* > &albums, bool selectInAlbumManager=true)

### Public Slots inherited from [Digikam::AbstractAlbumTreeView](#)

- void **adaptColumnsToContent** ()  
*Adapt the column sizes to the contents of the tree view.*
- void **expandEverything** (const QModelIndex &index)  
*Expands the complete tree under the given index.*
- void **scrollToSelectedAlbum** ()  
*Scrolls to the first selected album if there is one.*
- void **setCurrentAlbums** (const QList< [Album](#) \* > &albums, bool selectInAlbumManager=true)  
*Selects the given album.*
- void **setSearchTextSettings** (const [SearchTextSettings](#) &settings)
- void **slotCollapseAllNodes** ()  
*slotCollapseAllNodes - collapse all nodes without root node*
- void **slotCollapseNode** ()  
*slotCollapseNode - collapse recursively selected nodes*
- void **slotExpandNode** ()  
*slotExpandNode - expands recursively selected nodes*

### Protected Slots inherited from [Digikam::AbstractAlbumTreeView](#)

- void **albumSettingsChanged** ()
- void **slotCurrentChanged** ()
- virtual void **slotRootAlbumAvailable** ()  
*override if implemented behavior is not as intended*
- void **slotSearchTextSettingsAboutToChange** (bool searched, bool willSearch)
- void **slotSearchTextSettingsChanged** (bool wasSearching, bool searching)
- void **slotSelectionChanged** ()

### Protected Member Functions inherited from [Digikam::AbstractCheckableAlbumTreeView](#)

- void **middleButtonPressed** (Album \*a) override
- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- void **setAlbumFilterModel** (CheckableAlbumFilterModel \*const filterModel)
- void **setAlbumModel** (AbstractCheckableAlbumModel \*const model)

### Protected Member Functions inherited from [Digikam::AbstractCountingAlbumTreeView](#)

- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- void **setAlbumFilterModel** (AlbumFilterModel \*const filterModel)
- void **setAlbumModel** (AbstractCountingAlbumModel \*const model)

### Protected Member Functions inherited from [Digikam::AbstractAlbumTreeView](#)

- virtual void **addCustomContextMenuActions** (ContextMenuHelper &cmh, Album \*album)  
*Hook method to add custom actions to the generated context menu.*
- virtual QPixmap **contextMenuIcon** () const  
*Hook method that can be implemented to return a special icon used for the context menu.*
- virtual QString **contextMenuTitle** () const  
*Hook method to implement that returns the title for the context menu.*
- void **dragEnterEvent** (QDragEnterEvent \*e) override
- void **dragLeaveEvent** (QDragLeaveEvent \*e) override
- void **dragMoveEvent** (QDragMoveEvent \*e) override
- void **dropEvent** (QDropEvent \*e) override
- virtual void **handleCustomContextMenuAction** (QAction \*action, const AlbumPointer< Album > &album)  
*Hook method to handle the custom context menu actions that were added with addCustomContextMenuActions.*
- void **mousePressEvent** (QMouseEvent \*e) override  
*Other helper methods.*
- virtual QPixmap  **pixmapForDrag** (const QStyleOptionViewItem &option, QList< QModelIndex > indexes)  
*TODO: Move to delegate, when we have one.*
- void **rowsAboutToBeRemoved** (const QModelIndex &parent, int start, int end) override
- void **rowsInserted** (const QModelIndex &index, int start, int end) override
- void **setAlbumFilterModel** (AlbumFilterModel \*const filterModel)
- void **setAlbumModel** (AbstractSpecificAlbumModel \*const model)
- virtual bool **showContextMenuAt** (QContextMenuEvent \*event, Album \*albumForEvent)  
*Hook method to implement that determines if a context menu shall be displayed for the given event at the position coded in the event.*
- void **startDrag** (Qt::DropActions supportedActions) override



## Protected Member Functions inherited from Digikam::StateSavingObject

- QString [entryName](#) (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup [getConfigGroup](#) () const  
*Returns the config group that must be used for state saving and loading.*

## Protected Attributes inherited from Digikam::AbstractAlbumTreeView

- AlbumFilterModel \* [m\\_albumFilterModel](#) = nullptr
- AbstractSpecificAlbumModel \* [m\\_albumModel](#) = nullptr
- bool [m\\_checkOnMiddleClick](#) = false
- AlbumModelDragDropHandler \* [m\\_dragDropHandler](#) = nullptr
- Flags [m\\_flags](#) = DefaultFlags
- int [m\\_lastScrollBarValue](#) = 0
- bool [m\\_restoreCheckState](#) = false

### 9.68.1 Detailed Description

Author

jwienke

### 9.68.2 Member Function Documentation

#### 9.68.2.1 signalFindDuplicates

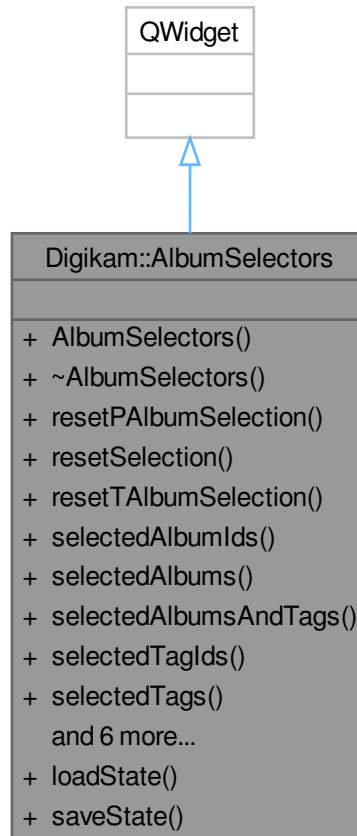
```
void Digikam::AlbumSelectionTreeView::signalFindDuplicates (
    const QList< PAlbum * > & albums ) [signal]
```

Parameters

<i>albums</i>	the album to find duplicates in
---------------	---------------------------------

## 9.69 Digikam::AlbumSelectors Class Reference

Inheritance diagram for Digikam::AlbumSelectors:



### Public Types

- enum **AlbumType** { **PhysAlbum** = 0 , **TagsAlbum** , **All** }
- enum **SelectionType** { **SingleSelection** = 0 , **MultipleSelection** }

### Public Slots

- void [loadState](#) ()  
*Called in constructor.*
- void [saveState](#) ()  
*Save settings in configuration file.*

### Signals

- void **signalSelectionChanged** ()

## Public Member Functions

- [AlbumSelectors](#) (const QString &label, const QString &configName, QWidget \*const parent=nullptr, AlbumType albumType=All, bool allowRecursive=false)  
*Default Constructor.*
- void **resetPAlbumSelection** ()  
*Reset all Physical Albums selection.*
- void **resetSelection** ()  
*Reset all Physical and Tag Albums selection.*
- void **resetTAlbumSelection** ()  
*Reset all Tag Albums selection.*
- QList< int > **selectedAlbumIds** () const  
*Return list of selected physical album ids.*
- AlbumList **selectedAlbums** () const  
*Return list of selected physical albums.*
- AlbumList **selectedAlbumsAndTags** () const  
*Return list of selected physical and tag albums.*
- QList< int > **selectedTagIds** () const  
*Return list of selected tag album ids.*
- AlbumList **selectedTags** () const  
*Return list of selected tag albums.*
- void **setAlbumSelected** (Album \*const album, SelectionType type)  
*Select Physical Album from list.*
- void **setTagSelected** (Album \*const album, SelectionType type)  
*Select Tag Album from list.*
- void **setTypeSelection** (int albumType)  
*Sets the search type selection with the AlbumType.*
- int **typeSelection** () const  
*Returns the selected album type.*
- bool **wholeAlbumsChecked** () const  
*Return true if whole Albums collection option is checked.*
- bool **wholeTagsChecked** () const  
*Return true if whole Tags collection option is checked.*

## 9.69.1 Constructor & Destructor Documentation

### 9.69.1.1 AlbumSelectors()

```
Digikam::AlbumSelectors::AlbumSelectors (
    const QString & label,
    const QString & configName,
    QWidget *const parent = nullptr,
    AlbumType albumType = All,
    bool allowRecursive = false ) [explicit]
```

'label' is front text of label which title widget. 'configName' is name used to store Albums configuration in settings file. 'parent' is parent widget.

## 9.69.2 Member Function Documentation

### 9.69.2.1 loadState

```
void Digikam::AlbumSelectors::loadState ( ) [slot]
```

Restore previous settings saved in configuration file.

### 9.69.2.2 saveState

```
void Digikam::AlbumSelectors::saveState ( ) [slot]
```

Must be called explicitly by host implementation.

### 9.69.2.3 setAlbumSelected()

```
void Digikam::AlbumSelectors::setAlbumSelected (
    Album *const album,
    SelectionType type )
```

If singleSelection is true, only this one is selected from tree-view and all others are deselected.

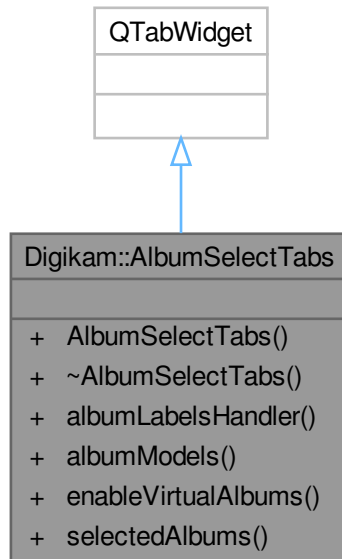
### 9.69.2.4 setTagSelected()

```
void Digikam::AlbumSelectors::setTagSelected (
    Album *const album,
    SelectionType type )
```

If singleSelection is true, only this one is selected from tree-view and all others are deselected.

## 9.70 Digikam::AlbumSelectTabs Class Reference

Inheritance diagram for Digikam::AlbumSelectTabs:



### Signals

- void **signalAlbumSelectionChanged** ()

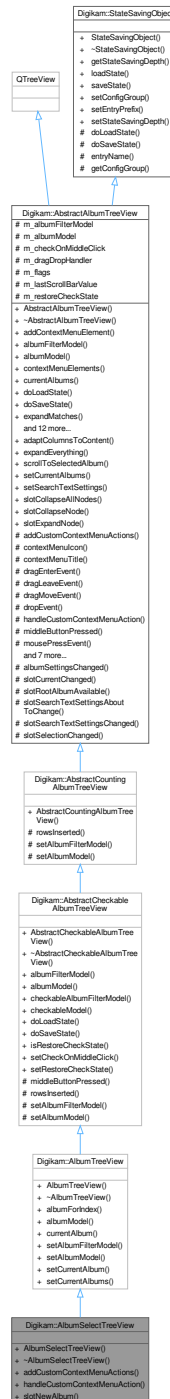
### Public Member Functions

- **AlbumSelectTabs** (const QString &name, QWidget \*const parent=nullptr)
- [AlbumLabelsSearchHandler](#) \* **albumLabelsHandler** () const
- QList< [AbstractCheckableAlbumModel](#) \* > **albumModels** () const
- void **enableVirtualAlbums** (bool flag=true)
- AlbumList **selectedAlbums** () const

## 9.71 Digikam::AlbumSelectTreeView Class Reference

Enables a simple context menu only for creating a new album.

Inheritance diagram for Digikam::AlbumSelectTreeView:



## Public Slots

- void **slotNewAlbum** ()

*Shows a dialog to create a new album under the selected album in this view.*

## Public Slots inherited from [Digikam::AlbumTreeView](#)

- void **setCurrentAlbum** (int albumId, bool selectInAlbumManager=true)

- void **setCurrentAlbums** (const QList< Album \* > &albums, bool selectInAlbumManager=true)

### Public Slots inherited from Digikam::AbstractAlbumTreeView

- void **adaptColumnsToContent** ()  
*Adapt the column sizes to the contents of the tree view.*
- void **expandEverything** (const QModelIndex &index)  
*Expands the complete tree under the given index.*
- void **scrollToSelectedAlbum** ()  
*Scrolls to the first selected album if there is one.*
- void **setCurrentAlbums** (const QList< Album \* > &albums, bool selectInAlbumManager=true)  
*Selects the given album.*
- void **setSearchTextSettings** (const SearchTextSettings &settings)
- void **slotCollapseAllNodes** ()  
*slotCollapseAllNodes - collapse all nodes without root node*
- void **slotCollapseNode** ()  
*slotCollapseNode - collapse recursively selected nodes*
- void **slotExpandNode** ()  
*slotExpandNode - expands recursively selected nodes*

### Public Member Functions

- **AlbumSelectTreeView** (AlbumModel \*const model, AlbumModificationHelper \*const albumModificationHelper, QWidget \*const parent=nullptr)  
*Constructor.*
- **~AlbumSelectTreeView** () override  
*Destructor.*
- void **addCustomContextMenuActions** (ContextMenuHelper &cmh, Album \*album) override  
*Hook method to add custom actions to the generated context menu.*
- void **handleCustomContextMenuAction** (QAction \*action, const AlbumPointer< Album > &album) override  
*Hook method to handle the custom context menu actions that were added with addCustomContextMenuActions.*

### Public Member Functions inherited from Digikam::AlbumTreeView

- **AlbumTreeView** (QWidget \*const parent=nullptr, Flags flags=DefaultFlags)
- **PAbum \* albumForIndex** (const QModelIndex &index) const
- **AlbumModel \* albumModel** () const
- **PAbum \* currentAlbum** () const
- void **setAlbumFilterModel** (CheckableAlbumFilterModel \*const filterModel)
- void **setAlbumModel** (AlbumModel \*const model)

## Public Member Functions inherited from [Digikam::AbstractCheckableAlbumTreeView](#)

- [AbstractCheckableAlbumTreeView](#) (QWidget \*const parent, Flags flags)
  - Models of these view can be checkable, they need not.*
- [CheckableAlbumFilterModel](#) \* [albumFilterModel](#) () const
- [AbstractCheckableAlbumModel](#) \* [albumModel](#) () const
  - Manage check state through the model directly.*
- [CheckableAlbumFilterModel](#) \* [checkableAlbumFilterModel](#) () const
- [AbstractCheckableAlbumModel](#) \* [checkableModel](#) () const
- void [doLoadState](#) () override
  - Implements state loading for the album tree view in a somewhat clumsy procedure because the model may not be fully loaded when this method is called.*
- void [doSaveState](#) () override
  - Implement this hook method for state saving.*
- bool [isRestoreCheckState](#) () const
  - Tells if the check state is restored while loading / saving state.*
- void [setCheckOnMiddleClick](#) (bool doThat)
  - Enable checking on middle mouse button click (default: on).*
- void [setRestoreCheckState](#) (bool restore)
  - Set whether to restore check state or not.*

## Public Member Functions inherited from [Digikam::AbstractCountingAlbumTreeView](#)

- [AbstractCountingAlbumTreeView](#) (QWidget \*const parent, Flags flags)

## Public Member Functions inherited from [Digikam::AbstractAlbumTreeView](#)

- [AbstractAlbumTreeView](#) (QWidget \*const parent, Flags flags)
  - Constructs an album tree view.*
- void [addContextMenuElement](#) ([ContextMenuElement](#) \*const element)
- [AlbumFilterModel](#) \* [albumFilterModel](#) () const
- [AbstractSpecificAlbumModel](#) \* [albumModel](#) () const
- QList< [ContextMenuElement](#) \* > [contextMenuElements](#) () const
- template<class A >
  - QList< A \* > [currentAlbums](#) ()
- bool [expandMatches](#) (const QModelIndex &index)
  - Ensures that every current match is visible by expanding all parent entries.*
- QModelIndex [indexVisuallyAt](#) (const QPoint &p)
  - This is a combination of [indexAt\(\)](#) checked with [visualRect\(\)](#).*
- void [removeContextMenuElement](#) ([ContextMenuElement](#) \*const element)
- QList< [Album](#) \* > [selectedItems](#) ()
  - [selectedItems\(\)](#) -*
- void [setAlbumManagerCurrentAlbum](#) (const bool setCurrentAlbum)
  - Some treeviews shall control the global current album kept by [AlbumManager](#).*
- void [setContextMenuIcon](#) (const QPixmap &pixmap)
  - Set the context menu title and icon.*
- void [setContextMenuTitle](#) (const QString &title)
- void [setEnabledContextMenu](#) (const bool enable)
  - Determines the global decision to show a popup menu or not.*
- void [setExpandNewCurrentItem](#) (const bool doThat)
  - Expand an item when making it the new current item.*



- void **setExpandOnSingleClick** (const bool doThat)  
*Enable expanding of tree items on single click on the item (default: off)*
- void **setSelectAlbumOnClick** (const bool selectOnClick)  
*Sets whether to select an album on click via the album manager or not.*
- void **setSelectOnContextMenu** (const bool select)  
*Sets whether to select the album under the mouse cursor on a context menu request (so that the album is shown using the album manager) or not.*
- bool **viewportEvent** (QEvent \*event) override  
*For internal use only.*

### Public Member Functions inherited from Digikam::StateSavingObject

- [StateSavingObject](#) (QObject \*const host)  
*Constructor.*
- virtual **~StateSavingObject** ()  
*Destructor.*
- [StateSavingDepth](#) **getStateSavingDepth** () const  
*Returns the depth used for state saving or loading.*
- void **loadState** ()  
*Invokes loading the class' state.*
- void **saveState** ()  
*Invokes saving the class' state.*
- virtual void **setConfigGroup** (const KConfigGroup &group)  
*Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void **setEntryPrefix** (const QString &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void **setStateSavingDepth** (const [StateSavingDepth](#) depth)  
*Sets the depth used for state saving or loading.*

### Additional Inherited Members

### Public Types inherited from Digikam::AbstractAlbumTreeView

- enum [Flag](#) {  
[CreateDefaultModel](#) , [CreateDefaultFilterModel](#) , [CreateDefaultDelegate](#) , [ShowCountAccordingToSettings](#) ,  
[AlwaysShowInclusiveCounts](#) , **DefaultFlags** = [CreateDefaultFilterModel](#) | [CreateDefaultDelegate](#) | [ShowCountAccordingToSettings](#) }
- typedef QFlags< [Flag](#) > **Flags**

### Public Types inherited from Digikam::StateSavingObject

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }  
*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

### Signals inherited from Digikam::AbstractAlbumTreeView

- void **currentAlbumChanged** ([Album](#) \*currentAlbum)  
*Emitted when the currently selected album changes.*
- void **selectedAlbumsChanged** (const QList< [Album](#) \* > &selectedAlbums)  
*Emitted when the current selection changes.*

### Protected Slots inherited from [Digikam::AbstractAlbumTreeView](#)

- void **albumSettingsChanged** ()
- void **slotCurrentChanged** ()
- virtual void **slotRootAlbumAvailable** ()  
*override if implemented behavior is not as intended*
- void **slotSearchTextSettingsAboutToChange** (bool searched, bool willSearch)
- void **slotSearchTextSettingsChanged** (bool wasSearching, bool searching)
- void **slotSelectionChanged** ()

### Protected Member Functions inherited from [Digikam::AbstractCheckableAlbumTreeView](#)

- void **middleButtonPressed** (Album \*a) override
- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- void **setAlbumFilterModel** ([CheckableAlbumFilterModel](#) \*const filterModel)
- void **setAlbumModel** ([AbstractCheckableAlbumModel](#) \*const model)

### Protected Member Functions inherited from [Digikam::AbstractCountingAlbumTreeView](#)

- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- void **setAlbumFilterModel** ([AlbumFilterModel](#) \*const filterModel)
- void **setAlbumModel** ([AbstractCountingAlbumModel](#) \*const model)

### Protected Member Functions inherited from [Digikam::AbstractAlbumTreeView](#)

- virtual QPixmap **contextMenuIcon** () const  
*Hook method that can be implemented to return a special icon used for the context menu.*
- virtual QString **contextMenuTitle** () const  
*Hook method to implement that returns the title for the context menu.*
- void **dragEnterEvent** (QDragEnterEvent \*e) override
- void **dragLeaveEvent** (QDragLeaveEvent \*e) override
- void **dragMoveEvent** (QDragMoveEvent \*e) override
- void **dropEvent** (QDropEvent \*e) override
- void **mousePressEvent** (QMouseEvent \*e) override  
*Other helper methods.*
- virtual QPixmap  **pixmapForDrag** (const QStyleOptionViewItem &option, QList< QModelIndex > indexes)  
*TODO: Move to delegate, when we have one.*
- void **rowsAboutToBeRemoved** (const QModelIndex &parent, int start, int end) override
- void **rowsInserted** (const QModelIndex &index, int start, int end) override
- void **setAlbumFilterModel** ([AlbumFilterModel](#) \*const filterModel)
- void **setAlbumModel** ([AbstractSpecificAlbumModel](#) \*const model)
- virtual bool **showContextMenuAt** (QContextMenuEvent \*event, Album \*albumForEvent)  
*Hook method to implement that determines if a context menu shall be displayed for the given event at the position coded in the event.*
- void **startDrag** (Qt::DropActions supportedActions) override

### Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- QString **entryName** (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup **getConfigGroup** () const  
*Returns the config group that must be used for state saving and loading.*

## Protected Attributes inherited from [Digikam::AbstractAlbumTreeView](#)

- [AlbumFilterModel](#) \* `m_albumFilterModel` = nullptr
- [AbstractSpecificAlbumModel](#) \* `m_albumModel` = nullptr
- bool `m_checkOnMiddleClick` = false
- [AlbumModelDragDropHandler](#) \* `m_dragDropHandler` = nullptr
- Flags `m_flags` = DefaultFlags
- int `m_lastScrollBarValue` = 0
- bool `m_restoreCheckState` = false

### 9.71.1 Detailed Description

#### Author

jwienke

### 9.71.2 Constructor & Destructor Documentation

#### 9.71.2.1 AlbumSelectTreeView()

```
Digikam::AlbumSelectTreeView::AlbumSelectTreeView (
    AlbumModel *const model,
    AlbumModificationHelper *const albumModificationHelper,
    QWidget *const parent = nullptr )
```

#### Parameters

<i>model</i>	album model to work with
<i>albumModificationHelper</i>	helper object for modifying albums
<i>parent</i>	the parent for Qt's parent child mechanism

### 9.71.3 Member Function Documentation

#### 9.71.3.1 addCustomContextMenuActions()

```
void Digikam::AlbumSelectTreeView::addCustomContextMenuActions (
    ContextMenuHelper & cmh,
    Album * album ) [override], [virtual]
```

#### Parameters

<i>cmh</i>	helper object to create the context menu
<i>album</i>	tag on which the context menu will be created. May be null if it is requested on no tag entry

Reimplemented from [Digikam::AbstractAlbumTreeView](#).

### 9.71.3.2 handleCustomContextMenuAction()

```
void Digikam::AlbumSelectTreeView::handleCustomContextMenuAction (
    QAction * action,
    const AlbumPointer< Album > & album ) [override], [virtual]
```

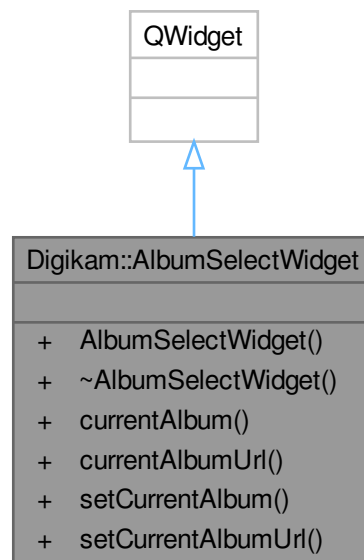
#### Parameters

<i>action</i>	the action that was chosen by the user, may be null if none of the custom actions were selected
<i>album</i>	the tag on which the context menu was requested. May be null if there was no

Reimplemented from [Digikam::AbstractAlbumTreeView](#).

## 9.72 Digikam::AlbumSelectWidget Class Reference

Inheritance diagram for Digikam::AlbumSelectWidget:



#### Signals

- void **completerActivated** ()
- void **itemSelectionChanged** ()

#### Public Member Functions

- **AlbumSelectWidget** (QWidget \*const parent=nullptr, PAlbum \*const albumToSelect=nullptr, bool completerSelect=false)
- PAlbum \* **currentAlbum** () const
- QUrl **currentAlbumUrl** () const
- void **setCurrentAlbum** (PAlbum \*const albumToSelect)
- void **setCurrentAlbumUrl** (const QUrl &albumUrl)

## 9.73 Digikam::AlbumShortInfo Class Reference

### Public Member Functions

- bool **isNull** () const

### Public Attributes

- int **albumRootId** = 0
- int **id** = 0
- QString **relativePath**

## 9.74 Digikam::AlbumSimplified Class Reference

This class is used when parsing response of listAlbums().

### Public Member Functions

- **AlbumSimplified** (const QString &title)
- **AlbumSimplified** (const QString &title, bool uploadable)

### Public Attributes

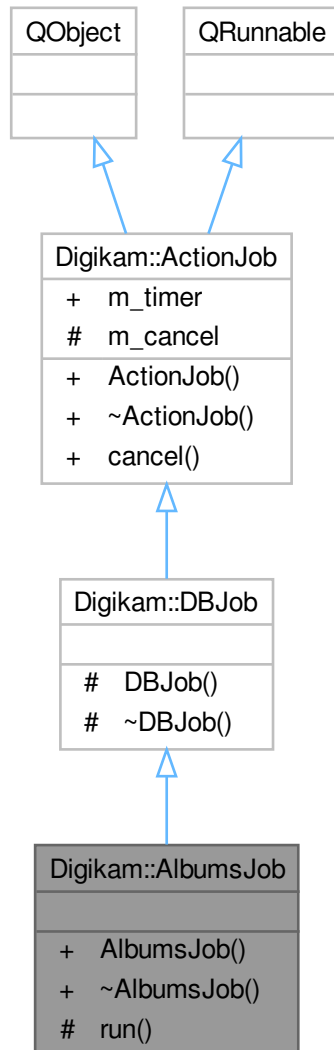
- QStringList **childrenIDs**
- QString **title**
- bool **uploadable** = true

### 9.74.1 Detailed Description

It contains only the most important attributes of an album, which is needed for further usage (e.g upload photos, create new album).

## 9.75 Digikam::AlbumsJob Class Reference

Inheritance diagram for Digikam::AlbumsJob:



### Signals

- void **foldersData** (const QHash< int, int > &)

### Signals inherited from [Digikam::DBJob](#)

- void **data** (const QList< [ItemListerRecord](#) > &records)
- void **error** (const QString &err)

## Signals inherited from [Digikam::ActionJob](#)

- void **signalDone** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job is done.*
- void **signalProgress** (int)  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job progress.*
- void **signalStarted** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager that job is started.*

## Public Member Functions

- **AlbumsJob** (const [AlbumsDBJobInfo](#) &jobInfo)

## Public Member Functions inherited from [Digikam::ActionJob](#)

- **ActionJob** (QObject \*const parent=nullptr)  
*Constructor which delegate deletion of QRunnable instance to [ActionThreadBase](#), not QThreadPool.*
- **~ActionJob** () override  
*Re-implement destructor in you implementation.*

## Protected Member Functions

- void **run** () override

## Additional Inherited Members

## Public Slots inherited from [Digikam::ActionJob](#)

- void **cancel** ()  
*Call this method to cancel job.*

## Public Attributes inherited from [Digikam::ActionJob](#)

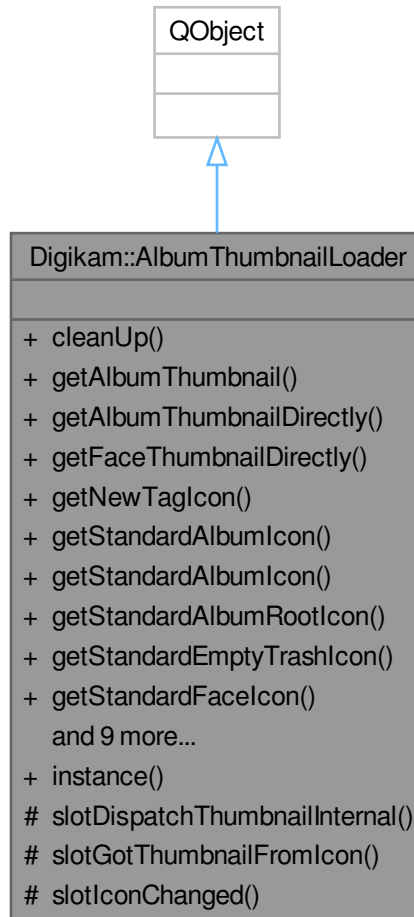
- QElapsedTimer **m\_timer**  
*Timer to determine the running time of the job.*

## Protected Attributes inherited from [Digikam::ActionJob](#)

- bool **m\_cancel** = false  
*You can use this boolean in your implementation to know if job must be canceled.*

## 9.76 Digikam::AlbumThumbnailLoader Class Reference

Inheritance diagram for Digikam::AlbumThumbnailLoader:



### Public Types

- enum `RelativeSize` { `NormalSize` , `SmallerSize` }  
*Album thumbnail size is configurable via the settings menu.*

### Signals

- void `signalDispatchThumbnailInternal` (int albumID, const QPixmap &thumbnail)  
*Internal signal to dispatch Album thumbnail change.*
- void `signalFailed` (Album \*album)  
*This signal is emitted if thumbnail generation for given album failed.*
- void `signalReloadThumbnails` ()  
*Indicates that all album and tag thumbnails need to be reloaded.*
- void `signalThumbnail` (Album \*album, const QPixmap &)  
*This signal is emitted as soon as a thumbnail has become available for given album.*



## Public Member Functions

- void **cleanUp** ()
- bool **getAlbumThumbnail** (PAlbum \*const album)
 

*Request thumbnail for given album.*
- QPixmap **getAlbumThumbnailDirectly** (PAAlbum \*const album)
 

*Request thumbnail for given album, with slightly different behavior than the above method: If the thumbnail is already available in the cache, it is returned.*
- QPixmap **getFaceThumbnailDirectly** (TAlbum \*const album)
 

*Loads face tag thumbnail, like [getTagThumbnailDirectly\(\)](#) but loads thumbnails in the size for faces.*
- QPixmap **getNewTagIcon** (RelativeSize size=NormalSize)
- QPixmap **getStandardAlbumIcon** (PAAlbum \*const album, RelativeSize size=NormalSize)
- QPixmap **getStandardAlbumIcon** (RelativeSize size=NormalSize)
- QPixmap **getStandardAlbumRootIcon** (RelativeSize size=NormalSize)
- QPixmap **getStandardEmptyTrashIcon** (RelativeSize size=NormalSize)
- QPixmap **getStandardFacelIcon** (TAlbum \*const album, RelativeSize size=NormalSize)
- QPixmap **getStandardFullTrashIcon** (RelativeSize size=NormalSize)
- QPixmap **getStandardOfflinelIcon** (RelativeSize size=NormalSize)
- QPixmap **getStandardTagIcon** (RelativeSize size=NormalSize)
 

*Return standard tag and album icons.*
- QPixmap **getStandardTagIcon** (TAlbum \*const album, RelativeSize size=NormalSize)
- QPixmap **getStandardTagRootIcon** (RelativeSize size=NormalSize)
- bool **getTagThumbnail** (TAlbum \*const album, QPixmap &icon)
 

*Behaves similar to the above method.*
- QPixmap **getTagThumbnailDirectly** (TAlbum \*const album)
 

*Loads tag thumbnail, with slightly different behavior than the above method: If the thumbnail is already available in the cache, it is returned, already blended with the standard icon, if requested.*
- void **setThumbnailSize** (int size, int face)
 

*Change the size of the thumbnails.*
- int **thumbnailSize** () const
 

*Get the current default icon size.*

## Static Public Member Functions

- static AlbumThumbnailLoader \* **instance** ()
 

*Return a preview of physical album directly without to use cache.*

## Protected Slots

- void **slotDispatchThumbnailInternal** (int albumID, const QPixmap &thumbnail)
- void **slotGotThumbnailFromIcon** (const LoadingDescription &loadingDescription, const QPixmap &pixmap)
- void **slotIconChanged** (Album \*album)

## Friends

- class AlbumThumbnailLoaderCreator

## 9.76.1 Member Enumeration Documentation

### 9.76.1.1 RelativeSize

```
enum Digikam::AlbumThumbnailLoader::RelativeSize
```

Some widgets use smaller icons than other widgets. These widgets do not need to know the currently set icon size from the setup and calculate a smaller size, but can simply request a relatively smaller icon. Depending on the user-chosen icon size, this size may in fact not be smaller than the normal size.

## 9.76.2 Member Function Documentation

### 9.76.2.1 getAlbumThumbnail()

```
bool Digikam::AlbumThumbnailLoader::getAlbumThumbnail (
    PAlbum *const album )
```

The thumbnail will be loaded and returned asynchronously by the signals. If no thumbnail is associated with given album, no action will be taken, and false is returned.

### 9.76.2.2 getAlbumThumbnailDirectly()

```
QPixmap Digikam::AlbumThumbnailLoader::getAlbumThumbnailDirectly (
    PAlbum *const album )
```

If the icon is not yet loaded, it will be returned asynchronously by the signals, and a default icon is returned here. If no icon is associated, the default icon is returned.

### 9.76.2.3 getStandardTagIcon()

```
QPixmap Digikam::AlbumThumbnailLoader::getStandardTagIcon (
    RelativeSize size = NormalSize )
```

The third methods check if album is the root, and returns the standard icon or the root standard icon.

### 9.76.2.4 getTagThumbnail()

```
bool Digikam::AlbumThumbnailLoader::getTagThumbnail (
    TAlbum *const album,
    QPixmap & icon )
```

Tag thumbnails will be processed as appropriate. Tags may have associated an icon that is loaded synchronously by the system icon loader. In this case, icon is set to this icon, and false is returned. If no icon is associated with the tag, icon is set to null, and false is returned. If a custom icon is associated with the tag, it is loaded asynchronously, icon is set to null, and true is returned. Tag thumbnails are always smaller than album thumbnails - as small as an album thumbnail with SmallerSize. They are supposed to be blended into the standard tag icon obtained below, or used as is when SmallerSize is requested anyway.

#### Returns

Returns true if icon is loaded asynchronously.

### 9.76.2.5 getTagThumbnailDirectly()

```
QPixmap Digikam::AlbumThumbnailLoader::getTagThumbnailDirectly (
    TAlbum *const album )
```

If the icon is not yet loaded, it will be returned asynchronously by the signals (unblended), and a default icon is returned here. If no icon is associated, the default icon is returned.

### 9.76.2.6 instance()

```
AlbumThumbnailLoader * Digikam::AlbumThumbnailLoader::instance ( ) [static]
```

Size of image can be passed as argument.

### 9.76.2.7 setThumbnailSize()

```
void Digikam::AlbumThumbnailLoader::setThumbnailSize (
    int size,
    int face )
```

If the size differs from the current size, signalReloadThumbnails will be emitted.

### 9.76.2.8 signalFailed

```
void Digikam::AlbumThumbnailLoader::signalFailed (
    Album * album ) [signal]
```

Same considerations as above.

### 9.76.2.9 signalReloadThumbnails

```
void Digikam::AlbumThumbnailLoader::signalReloadThumbnails ( ) [signal]
```

This is usually because the icon size has changed in the setup.

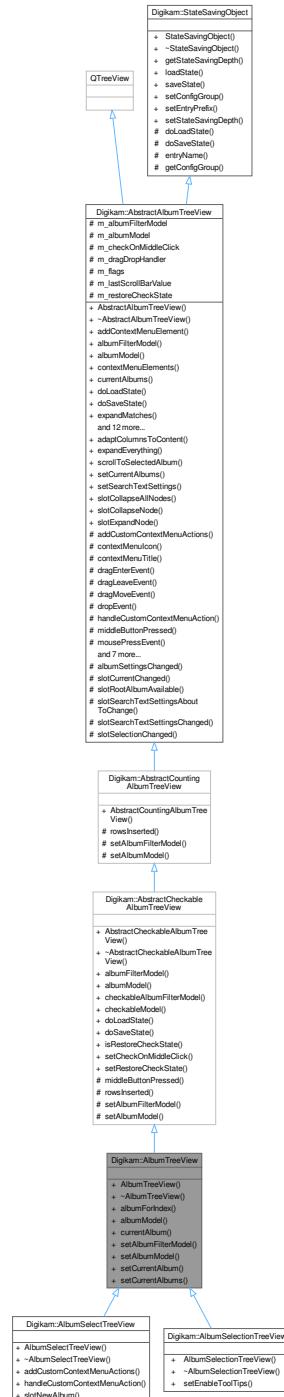
### 9.76.2.10 signalThumbnail

```
void Digikam::AlbumThumbnailLoader::signalThumbnail (
    Album * album,
    const QPixmap & ) [signal]
```

This class is a singleton, so any object connected to this signal might not actually have requested a thumbnail for given url

## 9.77 Digikam::AlbumTreeView Class Reference

Inheritance diagram for Digikam::AlbumTreeView:



### Public Slots

- void **setCurrentAlbum** (int albumId, bool selectInAlbumManager=true)
- void **setCurrentAlbums** (const QList< Album \* > &albums, bool selectInAlbumManager=true)

## Public Slots inherited from [Digikam::AbstractAlbumTreeView](#)

- void **adaptColumnsToContent** ()  
*Adapt the column sizes to the contents of the tree view.*
- void **expandEverything** (const QModelIndex &index)  
*Expands the complete tree under the given index.*
- void **scrollToSelectedAlbum** ()  
*Scrolls to the first selected album if there is one.*
- void **setCurrentAlbums** (const QList< Album \* > &albums, bool selectInAlbumManager=true)  
*Selects the given album.*
- void **setSearchTextSettings** (const SearchTextSettings &settings)
- void **slotCollapseAllNodes** ()  
*slotCollapseAllNodes - collapse all nodes without root node*
- void **slotCollapseNode** ()  
*slotCollapseNode - collapse recursively selected nodes*
- void **slotExpandNode** ()  
*slotExpandNode - expands recursively selected nodes*

## Public Member Functions

- **AlbumTreeView** (QWidget \*const parent=nullptr, Flags flags=DefaultFlags)
- **PAbum** \* **albumForIndex** (const QModelIndex &index) const
- **AlbumModel** \* **albumModel** () const
- **PAbum** \* **currentAlbum** () const
- void **setAlbumFilterModel** (**CheckableAlbumFilterModel** \*const filterModel)
- void **setAlbumModel** (**AlbumModel** \*const model)

## Public Member Functions inherited from [Digikam::AbstractCheckableAlbumTreeView](#)

- **AbstractCheckableAlbumTreeView** (QWidget \*const parent, Flags flags)  
*Models of these view can be checkable, they need not.*
- **CheckableAlbumFilterModel** \* **albumFilterModel** () const
- **AbstractCheckableAlbumModel** \* **albumModel** () const  
*Manage check state through the model directly.*
- **CheckableAlbumFilterModel** \* **checkableAlbumFilterModel** () const
- **AbstractCheckableAlbumModel** \* **checkableModel** () const
- void **doLoadState** () override  
*Implements state loading for the album tree view in a somewhat clumsy procedure because the model may not be fully loaded when this method is called.*
- void **doSaveState** () override  
*Implement this hook method for state saving.*
- bool **isRestoreCheckState** () const  
*Tells if the check state is restored while loading / saving state.*
- void **setCheckOnMiddleClick** (bool doThat)  
*Enable checking on middle mouse button click (default: on).*
- void **setRestoreCheckState** (bool restore)  
*Set whether to restore check state or not.*

## Public Member Functions inherited from [Digikam::AbstractCountingAlbumTreeView](#)

- **AbstractCountingAlbumTreeView** (QWidget \*const parent, Flags flags)

## Public Member Functions inherited from [Digikam::AbstractAlbumTreeView](#)

- [AbstractAlbumTreeView](#) (QWidget \*const parent, Flags flags)  
*Constructs an album tree view.*
- void **addContextMenuElement** ([ContextMenuElement](#) \*const element)
- [AlbumFilterModel](#) \* **albumFilterModel** () const
- [AbstractSpecificAlbumModel](#) \* **albumModel** () const
- QList< [ContextMenuElement](#) \* > **contextMenuElements** () const
- template<class A >  
QList< A \* > **currentAlbums** ()
- bool **expandMatches** (const QModelIndex &index)  
*Ensures that every current match is visible by expanding all parent entries.*
- QModelIndex **indexVisuallyAt** (const QPoint &p)  
*This is a combination of indexAt() checked with visualRect().*
- void **removeContextMenuElement** ([ContextMenuElement](#) \*const element)
- QList< [Album](#) \* > **selectedItems** ()  
*selectedItems()* -
- void **setAlbumManagerCurrentAlbum** (const bool setCurrentAlbum)  
*Some treeviews shall control the global current album kept by [AlbumManager](#).*
- void **setContextMenuIcon** (const QPixmap &pixmap)  
*Set the context menu title and icon.*
- void **setContextMenuTitle** (const QString &title)
- void **setEnabledContextMenu** (const bool enable)  
*Determines the global decision to show a popup menu or not.*
- void **setExpandNewCurrentItem** (const bool doThat)  
*Expand an item when making it the new current item.*
- void **setExpandOnSingleClick** (const bool doThat)  
*Enable expanding of tree items on single click on the item (default: off)*
- void **setSelectAlbumOnClick** (const bool selectOnClick)  
*Sets whether to select an album on click via the album manager or not.*
- void **setSelectOnContextMenu** (const bool select)  
*Sets whether to select the album under the mouse cursor on a context menu request (so that the album is shown using the album manager) or not.*
- bool **viewportEvent** (QEvent \*event) override  
*For internal use only.*

## Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (QObject \*const host)  
*Constructor.*
- virtual ~**StateSavingObject** ()  
*Destructor.*
- [StateSavingDepth](#) **getStateSavingDepth** () const  
*Returns the depth used for state saving or loading.*
- void **loadState** ()  
*Invokes loading the class' state.*
- void **saveState** ()  
*Invokes saving the class' state.*
- virtual void **setConfigGroup** (const KConfigGroup &group)  
*Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void **setEntryPrefix** (const QString &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void **setStateSavingDepth** (const [StateSavingDepth](#) depth)  
*Sets the depth used for state saving or loading.*

## Additional Inherited Members

### Public Types inherited from [Digikam::AbstractAlbumTreeView](#)

- enum [Flag](#) { [CreateDefaultModel](#) , [CreateDefaultFilterModel](#) , [CreateDefaultDelegate](#) , [ShowCountAccordingToSettings](#) , [AlwaysShowInclusiveCounts](#) , **DefaultFlags** = [CreateDefaultFilterModel](#) | [CreateDefaultDelegate](#) | [ShowCountAccordingToSettings](#) }
- typedef QFlags< [Flag](#) > **Flags**

### Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }  
*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

### Signals inherited from [Digikam::AbstractAlbumTreeView](#)

- void **currentAlbumChanged** ([Album](#) \*currentAlbum)  
*Emitted when the currently selected album changes.*
- void **selectedAlbumsChanged** (const QList< [Album](#) \* > &selectedAlbums)  
*Emitted when the current selection changes.*

### Protected Slots inherited from [Digikam::AbstractAlbumTreeView](#)

- void **albumSettingsChanged** ()
- void **slotCurrentChanged** ()
- virtual void **slotRootAlbumAvailable** ()  
*override if implemented behavior is not as intended*
- void **slotSearchTextSettingsAboutToChange** (bool searched, bool willSearch)
- void **slotSearchTextSettingsChanged** (bool wasSearching, bool searching)
- void **slotSelectionChanged** ()

### Protected Member Functions inherited from [Digikam::AbstractCheckableAlbumTreeView](#)

- void **middleButtonPressed** ([Album](#) \*a) override
- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- void **setAlbumFilterModel** ([CheckableAlbumFilterModel](#) \*const filterModel)
- void **setAlbumModel** ([AbstractCheckableAlbumModel](#) \*const model)

### Protected Member Functions inherited from [Digikam::AbstractCountingAlbumTreeView](#)

- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- void **setAlbumFilterModel** ([AlbumFilterModel](#) \*const filterModel)
- void **setAlbumModel** ([AbstractCountingAlbumModel](#) \*const model)

## Protected Member Functions inherited from [Digikam::AbstractAlbumTreeView](#)

- virtual void [addCustomContextMenuActions](#) ([ContextMenuHelper](#) &cmh, [Album](#) \*album)
  - Hook method to add custom actions to the generated context menu.*
- virtual QPixmap [contextMenuIcon](#) () const
  - Hook method that can be implemented to return a special icon used for the context menu.*
- virtual QString [contextMenuTitle](#) () const
  - Hook method to implement that returns the title for the context menu.*
- void **dragEnterEvent** ([QDragEnterEvent](#) \*e) override
- void **dragLeaveEvent** ([QDragLeaveEvent](#) \*e) override
- void **dragMoveEvent** ([QDragMoveEvent](#) \*e) override
- void **dropEvent** ([QDropEvent](#) \*e) override
- virtual void [handleCustomContextMenuAction](#) ([QAction](#) \*action, const [AlbumPointer](#)< [Album](#) > &album)
  - Hook method to handle the custom context menu actions that were added with [addCustomContextMenuActions](#).*
- void **mousePressEvent** ([QMouseEvent](#) \*e) override
  - Other helper methods.*
- virtual QPixmap [pixmapForDrag](#) (const [QStyleOptionViewItem](#) &option, [QList](#)< [QModelIndex](#) > indexes)
  - TODO: Move to delegate, when we have one.*
- void **rowsAboutToBeRemoved** (const [QModelIndex](#) &parent, int start, int end) override
- void **rowsInserted** (const [QModelIndex](#) &index, int start, int end) override
- void **setAlbumFilterModel** ([AlbumFilterModel](#) \*const filterModel)
- void **setAlbumModel** ([AbstractSpecificAlbumModel](#) \*const model)
- virtual bool [showContextMenuAt](#) ([QContextMenuEvent](#) \*event, [Album](#) \*albumForEvent)
  - Hook method to implement that determines if a context menu shall be displayed for the given event at the position coded in the event.*
- void **startDrag** ([Qt::DropActions](#) supportedActions) override

## Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- QString [entryName](#) (const QString &base) const
  - Always use this method to create config group entry names.*
- [KConfigGroup](#) [getConfigGroup](#) () const
  - Returns the config group that must be used for state saving and loading.*

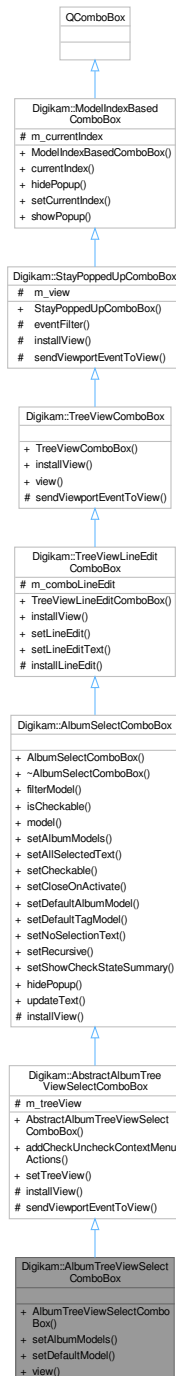
## Protected Attributes inherited from [Digikam::AbstractAlbumTreeView](#)

- [AlbumFilterModel](#) \* [m\\_albumFilterModel](#) = nullptr
- [AbstractSpecificAlbumModel](#) \* [m\\_albumModel](#) = nullptr
- bool [m\\_checkOnMiddleClick](#) = false
- [AlbumModelDragDropHandler](#) \* [m\\_dragDropHandler](#) = nullptr
- Flags [m\\_flags](#) = [DefaultFlags](#)
- int [m\\_lastScrollBarValue](#) = 0
- bool [m\\_restoreCheckState](#) = false



## 9.78 Digikam::AlbumTreeViewSelectComboBox Class Reference

Inheritance diagram for Digikam::AlbumTreeViewSelectComboBox:



### Public Member Functions

- **AlbumTreeViewSelectComboBox** (QWidget \*const parent=nullptr)
- void **setAlbumModels** (AlbumModel \*model, CheckableAlbumFilterModel \*filterModel=nullptr)
- void **setDefaultModel** ()
- AlbumTreeView \* **view** () const

## Public Member Functions inherited from [Digikam::AbstractAlbumTreeViewSelectComboBox](#)

- [AbstractAlbumTreeViewSelectComboBox](#) (QWidget \*const parent=nullptr)  
*Abstract class.*
- void [addCheckUncheckContextMenuActions](#) ()  
*Enables a context menu which contains options to check or uncheck groups of albums, given you have a checkable model.*
- void [setTreeView](#) ([AbstractAlbumTreeView](#) \*const treeView)  
*Set a tree view created by you instead of creating a default view (in the subclasses).*

## Public Member Functions inherited from [Digikam::AlbumSelectComboBox](#)

- [AlbumSelectComboBox](#) (QWidget \*const parent=nullptr)
- QSortFilterProxyModel \* [filterModel](#) () const  
*Return the filter model in use.*
- bool [isCheckable](#) () const
- [AbstractCheckableAlbumModel](#) \* [model](#) () const  
*Returns the source model.*
- void [setAlbumModels](#) ([AbstractCheckableAlbumModel](#) \*model, [AlbumFilterModel](#) \*filterModel=nullptr)
- void [setAllSelectedText](#) (bool all)  
*Enable or disable the text used to describe the status when all album is selected.*
- void [setCheckable](#) (bool checkable)  
*Enable checkboxes next to the items.*
- void [setCloseOnActivate](#) (bool close)  
*Enable closing when an item was activated (clicked).*
- void [setDefaultAlbumModel](#) ()  
*Once after creation, call one of these three methods.*
- void [setDefaultTagModel](#) ()
- void [setNoSelectionText](#) (const QString &text)  
*Sets the text that is used to describe the state when no album is selected.*
- void [setRecursive](#) (bool recursive)  
*If all subalbums shall be selected when parent will be selected.*
- void [setShowCheckStateSummary](#) (bool show)  
*If the box is checkable, enable showing a resume a la "3 Albums checked" in the combo box text.*

## Public Member Functions inherited from [Digikam::TreeViewLineEditComboBox](#)

- [TreeViewLineEditComboBox](#) (QWidget \*const parent=nullptr)  
*This class provides a [TreeViewComboBox](#) with a read-only line edit.*
- void [installView](#) (QAbstractItemView \*view=nullptr) override  
*Replace the standard combo box list view with a QTreeView.*
- void [setLineEdit](#) (QLineEdit \*edit)
- void [setLineEditText](#) (const QString &text)  
*Set the text of the line edit (the text that is visible if the popup is not opened).*

## Public Member Functions inherited from [Digikam::TreeViewComboBox](#)

- [TreeViewComboBox](#) (QWidget \*parent=nullptr)  
*This class provides a QComboBox with a QTreeView instead of the usual QListView.*
- QTreeView \* [view](#) () const  
*Returns the QTreeView of this class.*

## Public Member Functions inherited from [Digikam::StayPoppedUpComboBox](#)

- [StayPoppedUpComboBox](#) (QWidget \*const parent=nullptr)

*This class provides an abstract QComboBox with a custom view (which is created by implementing subclasses) instead of the usual QListView.*

## Public Member Functions inherited from [Digikam::ModelIndexBasedComboBox](#)

- [ModelIndexBasedComboBox](#) (QWidget \*const parent=nullptr)  
*QComboBox has a current index based on a single integer.*
- QModelIndex **currentIndex** () const
- void **hidePopup** () override
- void **setCurrentIndex** (const QModelIndex &index)
- void **showPopup** () override

## Additional Inherited Members

## Public Slots inherited from [Digikam::AlbumSelectComboBox](#)

- void **hidePopup** () override
- virtual void **updateText** ()

*Updates the text describing the selection ("3 Albums selected").*

## Protected Member Functions inherited from [Digikam::AbstractAlbumTreeViewSelectComboBox](#)

- void **installView** (QAbstractItemView \*view=nullptr) override  
*Replace the standard combo box list view with a QTreeView.*
- void **sendViewportEventToView** (QEvent \*e) override

*Implement in subclass: Send the given event to the viewportEvent() method of m\_view.*

## Protected Member Functions inherited from [Digikam::AlbumSelectComboBox](#)

- void **installView** (QAbstractItemView \*view=nullptr) override

*Replace the standard combo box list view with a QTreeView.*

## Protected Member Functions inherited from [Digikam::TreeViewLineEditComboBox](#)

- virtual void **installLineEdit** ()

*Sets a line edit.*

## Protected Member Functions inherited from [Digikam::TreeViewComboBox](#)

- void **sendViewportEventToView** (QEvent \*e) override

*Implement in subclass: Send the given event to the viewportEvent() method of m\_view.*

### Protected Member Functions inherited from [Digikam::StayPoppedUpComboBox](#)

- bool `eventFilter` (QObject \*watched, QEvent \*event) override
- void `installView` (QAbstractItemView \*view)

*Replace the standard combo box list view with the given view.*

### Protected Attributes inherited from [Digikam::AbstractAlbumTreeViewSelectComboBox](#)

- [AbstractAlbumTreeView](#) \* `m_treeView` = nullptr

### Protected Attributes inherited from [Digikam::TreeViewLineEditComboBox](#)

- QLineEdit \* `m_comboLineEdit` = nullptr

### Protected Attributes inherited from [Digikam::StayPoppedUpComboBox](#)

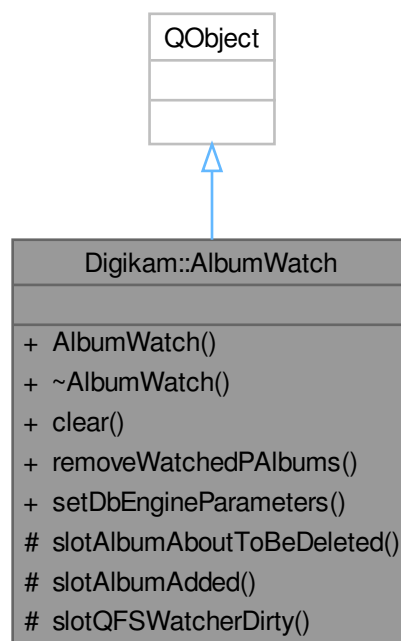
- QAbstractItemView \* `m_view` = nullptr

### Protected Attributes inherited from [Digikam::ModelIndexBasedComboBox](#)

- QPersistentModelIndex `m_currentIndex`

## 9.79 Digikam::AlbumWatch Class Reference

Inheritance diagram for Digikam::AlbumWatch:



### Public Member Functions

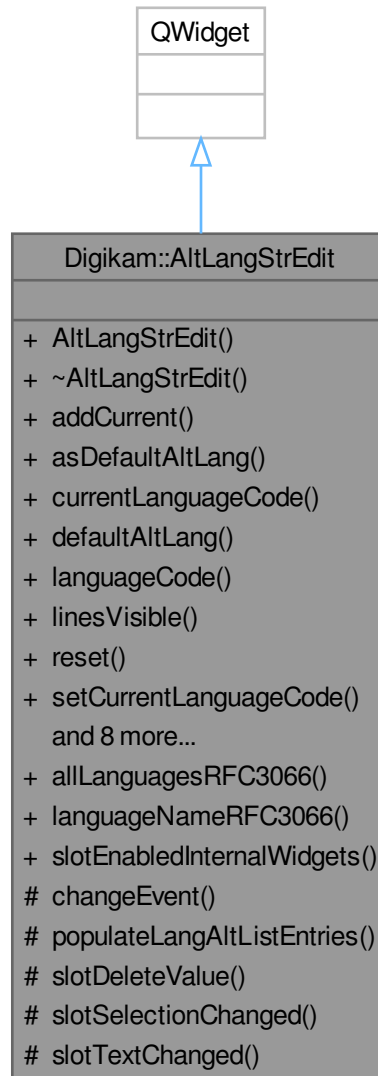
- **AlbumWatch** ([AlbumManager](#) \*const parent=nullptr)
- void **clear** ()
- void **removeWatchedPAlbums** (const [PAlbum](#) \*const album)
- void **setDbEngineParameters** (const [DbEngineParameters](#) &params)

### Protected Slots

- void **slotAlbumAboutToBeDeleted** ([Album](#) \*album)
- void **slotAlbumAdded** ([Album](#) \*album)
- void **slotQFSWatcherDirty** (const QString &path)

## 9.80 Digikam::AltLangStrEdit Class Reference

Inheritance diagram for Digikam::AltLangStrEdit:



### Public Slots

- void [slotEnabledInternalWidgets](#) (bool)  
*Can be used to turn on/off visibility of internal widgets.*

### Signals

- void **signalModified** (const QString &lang, const QString &text)

- *Emitted when the user changes the text for the current language.*
- void **signalSelectionChanged** (const QString &lang)
- *Emitted when the current language changed.*
- void **signalValueAdded** (const QString &lang, const QString &text)
- *Emitted when an entry for a new language is added.*
- void **signalValueDeleted** (const QString &lang)
- *Emitted when the entry for a language is removed.*

### Public Member Functions

- [AltLangStrEdit](#) (QWidget \*const parent, unsigned int lines=3)
- *Default constructor.*
- void [addCurrent](#) ()
- *Ensure that the current language is added to the list of entries, even if the text is empty.*
- bool **asDefaultAltLang** () const
- QString **currentLanguageCode** () const
- QString **defaultAltLang** () const
- QString **languageCode** (int index) const
- uint **linesVisible** () const
- void **reset** ()
- *Reset widget, clear all entries.*
- void **setCurrentLanguageCode** (const QString &lang)
- void [setLinesVisible](#) (uint lines)
- *Fix lines visible in text editor to lines.*
- void **setPlaceholderText** (const QString &msg)
- void [setTitle](#) (const QString &title)
- *Create a title widget with a QLabel and relevant text.*
- void [setTitleWidget](#) (QWidget \*const twdg)
- *Create a title with a specific widget instance (aka a QCheckBox for ex).*
- virtual void **setValues** (const [MetaEngine::AltLangMap](#) &values)
- [DTextEdit](#) \* **textEdit** () const
- QWidget \* [titleWidget](#) () const
- *Return the current title widget instance.*
- [MetaEngine::AltLangMap](#) & **values** () const

### Static Public Member Functions

- static QStringList **allLanguagesRFC3066** ()
- *Return all language codes available following the RFC 3066.*
- static QString **languageNameRFC3066** (const QString &code)
- *Return the literal name of RFC 3066 language code (format FR-fr for ex).*

### Protected Slots

- void **slotDeleteValue** ()
- void **slotSelectionChanged** ()
- void **slotTextChanged** ()

## Protected Member Functions

- void **changeEvent** (QEvent \*e) override
- void **populateLangAltListEntries** ()

## Friends

- class **Private**

## 9.80.1 Constructor & Destructor Documentation

### 9.80.1.1 AltLangStrEdit()

```
Digikam::AltLangStrEdit::AltLangStrEdit (
    QWidget *const parent,
    unsigned int lines = 3 ) [explicit]
```

Use lines to use a specific number of lines with text editor.

## 9.80.2 Member Function Documentation

### 9.80.2.1 addCurrent()

```
void Digikam::AltLangStrEdit::addCurrent ( )
```

[signalValueAdded\(\)](#) will be emitted.

### 9.80.2.2 setLinesVisible()

```
void Digikam::AltLangStrEdit::setLinesVisible (
    uint lines )
```

If zero, do not fix layout to number of lines visible.

### 9.80.2.3 setTitle()

```
void Digikam::AltLangStrEdit::setTitle (
    const QString & title )
```

If a title widget already exists, it's replaced.

### 9.80.2.4 setTitleWidget()

```
void Digikam::AltLangStrEdit::setTitleWidget (
    QWidget *const twdg )
```

If a title widget already exists, it's replaced.



### 9.80.2.5 slotEnabledInternalWidgets

```
void Digikam::AltLangStrEdit::slotEnabledInternalWidgets (
    bool b ) [slot]
```

This do not includes the title widget.

### 9.80.2.6 titleWidget()

```
QWidget * Digikam::AltLangStrEdit::titleWidget ( ) const
```

If no previous call of [setTitle\(\)](#) or [setWidgetTitle\(\)](#), this function will return nullptr.

## 9.81 Digikam::AnimatedClearButton Class Reference

Inheritance diagram for Digikam::AnimatedClearButton:



### Public Slots

- void **animateVisible** (bool visible)  
*Set visible, possibly with animation.*
- void **setDirectlyVisible** (bool visible)  
*Set visible without animation.*
- void **slotPixmapEnabled** (bool b)  
*Set enabled state for drawing the pixmap.*

## Signals

- void **clicked** ()
- void **visibleChanged** (bool v)

## Public Member Functions

- **AnimatedClearButton** (QWidget \*const parent=nullptr)
- QPixmap  **pixmap** () const
- void **setPixmap** (const QPixmap &p)
- void **setShallBeShown** (bool show)  
*Sets a primary condition for the button to be shown.*
- QSize **sizeHint** () const override
- void **stayVisibleWhenAnimatedOut** (bool stayVisible)  
*This parameter determines the behavior when the animation to hide the widget has finished: If stayVisible is true, the widget remains visible, but paints nothing.*

## Protected Slots

- void **updateAnimationSettings** ()
- void **visibleChanged** ()

## Protected Member Functions

- void **mousePressEvent** (QMouseEvent \*event) override
- void **paintEvent** (QPaintEvent \*event) override

## 9.81.1 Member Function Documentation

### 9.81.1.1 setShallBeShown()

```
void Digikam::AnimatedClearButton::setShallBeShown (
    bool show )
```

If false, [animateVisible\(\)](#) will have no effect.

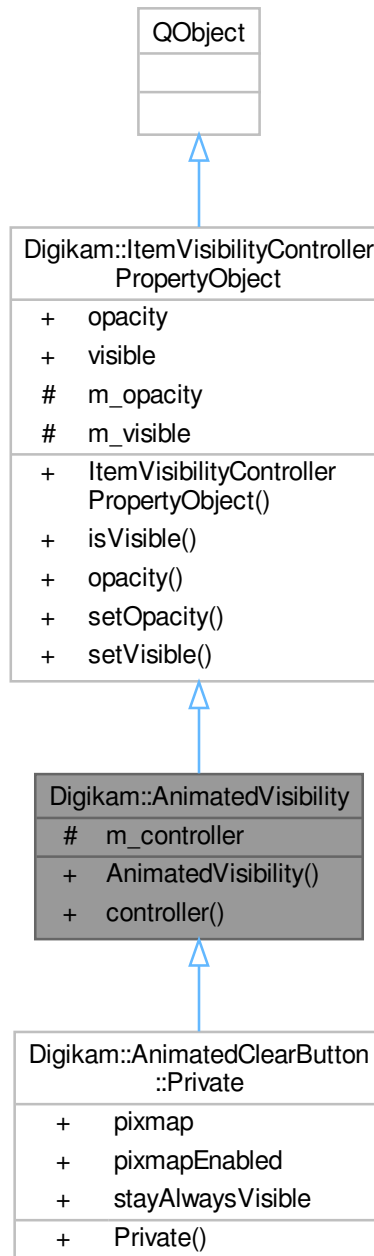
### 9.81.1.2 stayVisibleWhenAnimatedOut()

```
void Digikam::AnimatedClearButton::stayVisibleWhenAnimatedOut (
    bool stayVisible )
```

If stayVisible is false, setVisible(false) is called, which removes the widget for layouting etc. Default: false

## 9.82 Digikam::AnimatedVisibility Class Reference

Inheritance diagram for Digikam::AnimatedVisibility:



### Public Member Functions

- [AnimatedVisibility](#) (`QObject *const parent=nullptr`)  
*A convenience class: The property object brings its own controller.*
- [ItemVisibilityController](#) \* **controller** () const

## Public Member Functions inherited from Digikam::ItemVisibilityControllerPropertyObject

- [ItemVisibilityControllerPropertyObject](#) (QObject \*const parent=nullptr)  
*You can use this object as a container providing the properties set by [ItemVisibilityController](#).*
- bool **isVisible** () const
- qreal **opacity** () const
- void **setOpacity** (qreal opacity)
- void **setVisible** (bool visible)

### Protected Attributes

- [ItemVisibilityController](#) \* **m\_controller** = nullptr

## Protected Attributes inherited from Digikam::ItemVisibilityControllerPropertyObject

- qreal **m\_opacity** = 0.0
- bool **m\_visible** = false

### Additional Inherited Members

## Signals inherited from Digikam::ItemVisibilityControllerPropertyObject

- void **opacityChanged** ()
- void **visibleChanged** ()

## Properties inherited from Digikam::ItemVisibilityControllerPropertyObject

- qreal **opacity**
- bool **visible**

## 9.82.1 Constructor & Destructor Documentation

### 9.82.1.1 AnimatedVisibility()

```
Digikam::AnimatedVisibility::AnimatedVisibility (
    QObject *const parent = nullptr ) [explicit]
```

Ready to use: Just construct an object and connect to the signals. Please note the difference between controller()->setVisible() and setVisible(): You want to call the controller's method!

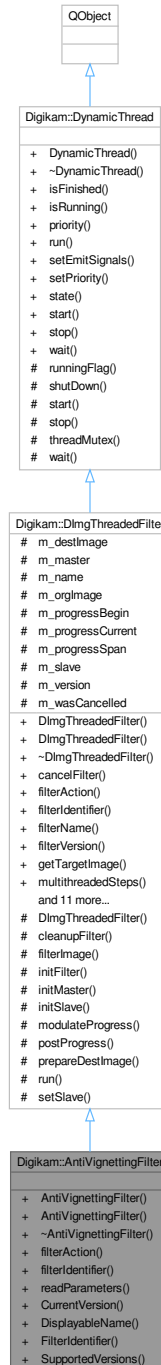
## 9.83 Digikam::AntiVignettingContainer Class Reference

### Public Attributes

- bool **addvignetting** = true
- double **density** = 2.0
- double **innerradius** = 1.0
- double **outerradius** = 1.0
- double **power** = 1.0
- double **xshift** = 0.0
- double **yshift** = 0.0

## 9.84 Digikam::AntiVignettingFilter Class Reference

Inheritance diagram for Digikam::AntiVignettingFilter:



### Public Member Functions

- **AntiVignettingFilter** (`Dlmg *const orgImage`, `QObject *const parent=nullptr`, `const AntiVignettingContainer &settings=AntiVignettingContainer()`)

- **AntiVignettingFilter** (QObject \*const parent=nullptr)
- **FilterAction filterAction** () override  
*Returns the action description corresponding to currently set options.*
- **QString filterIdentifier** () const override  
*Return the identifier for this filter in the image history.*
- void **readParameters** (const **FilterAction** &action) override

## Public Member Functions inherited from Digikam::DImgThreadedFilter

- **DImgThreadedFilter** (DImg \*const orgImage, QObject \*const parent, const QString &name=QString())  
*Constructs a filter with all arguments (ready to use).*
- **DImgThreadedFilter** (QObject \*const parent=nullptr, const QString &name=QString())  
*Constructs a filter without argument.*
- virtual void **cancelFilter** ()  
*Cancel the threaded computation.*
- const QString & **filterName** ()
- int **filterVersion** () const
- **DImg getTargetImage** ()
- QList< int > **multithreadedSteps** (int stop, int start=0) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool **parametersSuccessfullyRead** () const  
*Optional: error handling for readParameters.*
- virtual QString **readParametersError** (const **FilterAction** &actionThatFailed) const
- void **setFilterName** (const QString &name)
- void **setFilterVersion** (int version)  
*Replaying a filter action: Set the filter version.*
- void **setOriginalImage** (const **DImg** &orgImage)
- void **setupAndStartDirectly** (const **DImg** &orgImage, **DImgThreadedFilter** \*const master, int progress←Begin=0, int progressEnd=100)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void **setupFilter** (const **DImg** &orgImage)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void **startFilter** ()  
*Start the threaded computation.*
- virtual void **startFilterDirectly** ()  
*Start computation of this filter, directly in this thread.*
- virtual QList< int > **supportedVersions** () const

## Public Member Functions inherited from Digikam::DynamicThread

- **DynamicThread** (QObject \*const parent=nullptr)  
*This class extends QRunnable, so you have to reimplement virtual void run().*
- **~DynamicThread** () override  
*The destructor calls stop() and wait(), but if you, in your destructor, delete any data that is accessed by your run() method, you must call stop() and wait() before yourself.*
- bool **isFinished** () const
- bool **isRunning** () const
- QThread::Priority **priority** () const
- void **setEmitSignals** (bool emitThem)
- void **setPriority** (QThread::Priority priority)  
*Sets the priority for this dynamic thread.*
- State **state** () const

### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*



## Protected Member Functions inherited from Digikam::DImgThreadedFilter

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from Digikam::DynamicThread

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from Digikam::DImgThreadedFilter

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.84.1 Member Function Documentation

### 9.84.1.1 filterAction()

`FilterAction` Digikam::AntiVignettingFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.84.1.2 filterIdentifier()

`QString` Digikam::AntiVignettingFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

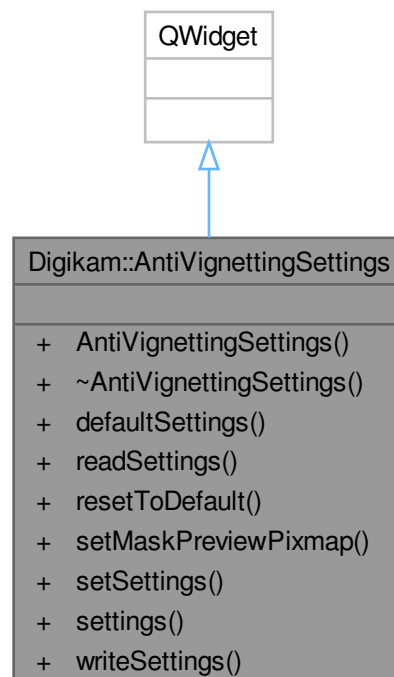
### 9.84.1.3 readParameters()

```
void Digikam::AntiVignettingFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

## 9.85 Digikam::AntiVignettingSettings Class Reference

Inheritance diagram for Digikam::AntiVignettingSettings:



## Signals

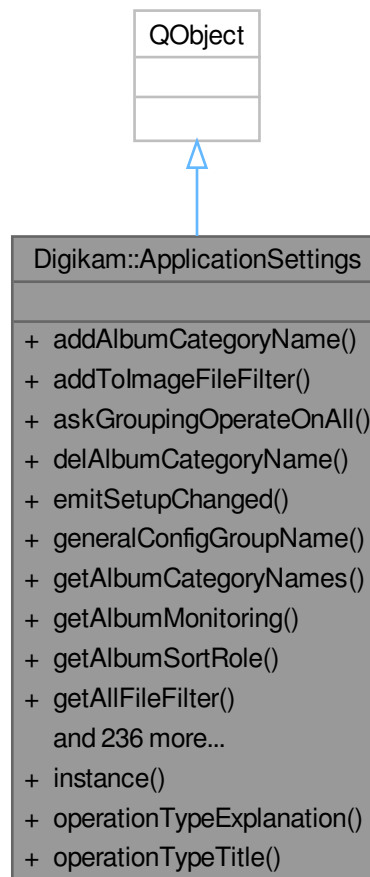
- void **signalSettingsChanged** ()

## Public Member Functions

- **AntiVignettingSettings** (QWidget \*parent)
- [AntiVignettingContainer](#) **defaultSettings** () const
- void **readSettings** (const KConfigGroup &group)
- void **resetToDefault** ()
- void **setMaskPreviewPixmap** (const QPixmap &pix)
- void **setSettings** (const [AntiVignettingContainer](#) &settings)
- [AntiVignettingContainer](#) **settings** () const
- void **writeSettings** (KConfigGroup &group)

## 9.86 Digikam::ApplicationSettings Class Reference

Inheritance diagram for Digikam::ApplicationSettings:



## Public Types

- enum **AlbumSortRole** { **ByFolder** = 0 , **ByCategory** , **ByDate** }
- enum **ApplyToEntireGroup** { **No** = 0 , **Yes** , **Ask** }
- enum **ItemLeftClickAction** { **ShowPreview** = 0 , **StartEditor** , **ShowOnTable** , **OpenDefault** }
- typedef QHash< [OperationType](#), ApplicationSettings::ApplyToEntireGroup > **OperationModes**
- typedef QHash< [OperationType](#), QString > **OperationStrings**
- enum [StringComparisonType](#) { **Natural** = 0 , **Normal** }

*Possible ways of comparing strings.*

## Signals

- void **balooSettingsChanged** ()
- void **recurseSettingsChanged** ()
- void **setupChanged** ()

## Public Member Functions

- bool **addAlbumCategoryName** (const QString &name) const
- void **addToImageFileFilter** (const QString &extensions)
- bool **askGroupingOperateOnAll** ([OperationType](#) type)  
*Asks the user whether the operation should be performed on all grouped images or just the first.*
- bool **delAlbumCategoryName** (const QString &name) const
- void **emitSetupChanged** ()
- QString **generalConfigGroupName** () const
- QStringList **getAlbumCategoryNames** () const
- bool **getAlbumMonitoring** () const
- AlbumSortRole **getAlbumSortRole** () const
- QString **getAllFileFilter** () const
- bool **getAllGroupsOpen** () const
- QFont **getApplicationFont** () const
- QString **getApplicationStyle** () const
- bool **getApplySidebarChangesDirectly** () const
- QString **getAudioFileFilter** () const
- bool **getCleanAtStart** () const
- QString **getCurrentTheme** () const
- bool **getDatabaseDirSetAtCmd** () const
- [DbEngineParameters](#) **getDbEngineParameters** () const
- int **getDefaultIconSize** () const
- bool **getDetectFacesInNewImages** () const
- bool **getDrawFramesToGrouped** () const
- int **getDuplicatesAlbumTagRelation** () const
- [Haarface::RefImageSelMethod](#) **getDuplicatesRefImageSelMethod** () const
- int **getDuplicatesSearchLastMaxSimilarity** () const
- int **getDuplicatesSearchLastMinSimilarity** () const
- int **getDuplicatesSearchRestrictions** () const
- bool **getExpandNewCurrentItem** () const
- int **getFaceDetectionAccuracy** () const
- [FaceScanSettings::FaceDetectionModel](#) **getFaceDetectionModel** () const
- [FaceScanSettings::FaceDetectionSize](#) **getFaceDetectionSize** () const
- int **getFaceRecognitionAccuracy** () const
- [FaceScanSettings::FaceRecognitionModel](#) **getFaceRecognitionModel** () const
- ApplyToEntireGroup **getGroupingOperateOnAll** ([OperationType](#) type) const

*Tells whether an operation should be performed on all grouped items or just the head item.*

- bool **getHelpBoxNotificationSeen** ()
- bool **getIconShowAspectRatio** () const
- bool **getIconShowColorLabel** () const
- bool **getIconShowComments** () const
- bool **getIconShowCoordinates** () const
- bool **getIconShowDate** () const
- bool **getIconShowFullscreen** () const
- bool **getIconShowImageFormat** () const
- bool **getIconShowModDate** () const
- bool **getIconShowName** () const
- bool **getIconShowOverlays** () const

*Determines whether the overlay buttons should be displayed on the icons.*

- bool **getIconShowPickLabel** () const
- bool **getIconShowRating** () const
- bool **getIconShowResolution** () const
- bool **getIconShowSize** () const
- bool **getIconShowTags** () const
- bool **getIconShowTitle** () const
- QString **getIconTheme** () const
- QFont **getIconViewFont** () const
- QString **getImageFileFilter** () const
- int **getImageSeparationMode** () const
- int **getImageSeparationSortOrder** () const
- int **getImageSorting** () const
- int **getImageSortOrder** () const
- int **getItemLeftClickAction** () const
- int **getMinimumSimilarityBound** () const
- QString **getMovieFileFilter** () const
- [PreviewSettings](#) **getPreviewSettings** () const
- bool **getPreviewShowIcons** () const
- bool **getPreviewSmoothScaled** () const
- int **getRatingFilterCond** () const
- QString **getRawFileFilter** () const
- bool **getRecurseAlbums** () const
- bool **getRecurseTags** () const
- bool **getScaleFitToWindow** () const
- bool **getScanAtStart** () const
- bool **getScrollItemToCenter** () const
- bool **getSelectFirstAlbumItem** () const
- bool **getShowAlbumToolTips** () const
- bool **getShowFolderTreeViewItemsCount** () const
- bool **getShowPermanentDeleteDialog** () const
- bool **getShowSplashScreen** () const
- bool **getShowThumbbar** () const
- bool **getShowToolTips** () const
- bool **getShowTrashDeleteDialog** () const
- [DMultiTabBar::TextStyle](#) **getSidebarTitleStyle** () const
- [StringComparisonType](#) **getStringComparisonType** () const

*Tells in which way strings are compared at the moment.*

- bool **getSyncBalooToDigikam** () const
- bool **getSyncDigikamToBaloo** () const
- QFont **getToolTipsFont** () const
- bool **getToolTipsShowAlbumCaption** () const

- bool **getToolTipsShowAlbumCategory** () const
- bool **getToolTipsShowAlbumCollection** () const
- bool **getToolTipsShowAlbumDate** () const
- bool **getToolTipsShowAlbumName** () const
- bool **getToolTipsShowAlbumPreview** () const
- bool **getToolTipsShowAlbumTitle** () const
- bool **getToolTipsShowComments** () const
- bool **getToolTipsShowFileDate** () const
- bool **getToolTipsShowFileName** () const
- bool **getToolTipsShowFileSize** () const
- bool **getToolTipsShowImageAR** () const
- bool **getToolTipsShowImageDim** () const
- bool **getToolTipsShowImageType** () const
- bool **getToolTipsShowLabelRating** () const
- bool **getToolTipsShowPhotoDate** () const
- bool **getToolTipsShowPhotoExpo** () const
- bool **getToolTipsShowPhotoFlash** () const
- bool **getToolTipsShowPhotoFocal** () const
- bool **getToolTipsShowPhotoLens** () const
- bool **getToolTipsShowPhotoMake** () const
- bool **getToolTipsShowPhotoMode** () const
- bool **getToolTipsShowPhotoWB** () const
- bool **getToolTipsShowTags** () const
- bool **getToolTipsShowTitles** () const
- bool **getToolTipsShowVideoAspectRatio** () const
- bool **getToolTipsShowVideoAudioBitRate** () const
- bool **getToolTipsShowVideoAudioChannelType** () const
- bool **getToolTipsShowVideoAudioCodec** () const
- bool **getToolTipsShowVideoDuration** () const
- bool **getToolTipsShowVideoFrameRate** () const
- bool **getToolTipsShowVideoVideoCodec** () const
- int **getTreeViewFaceSize** () const
- QFont **getTreeViewFont** () const
- int **getTreeViewIconSize** () const
- int **getUpdateType** () const
- bool **getUpdateWithDebug** () const
- bool **getUseNativeFileDialog** () const
- bool **getUseTrash** () const
- [VersionManagerSettings](#) **getVersionManagerSettings** () const
- bool **isStringTypeNatural** () const
- bool [readMsgBoxShouldBeShown](#) (const QString &dontShowAgainName)
- void **readSettings** ()
- void [saveMsgBoxShouldBeShown](#) (const QString &dontShowAgainName)
  - *Save the fact that the message box should not be shown again.*
- void **saveSettings** ()
- void **setAlbumCategoryNames** (const QStringList &list)
- void **setAlbumMonitoring** (bool val)
- void **setAlbumSortRole** (const AlbumSortRole role)
- void **setAllGroupsOpen** (bool val)
- void **setApplicationFont** (const QFont &fnt)
- void **setApplicationStyle** (const QString &style)
- void **setApplySidebarChangesDirectly** (bool val)
- void **setCleanAtStart** (bool val)
- void **setCurrentTheme** (const QString &theme)

- void **setDatabaseDirSetAtCmd** (bool val)
- void **setDbEngineParameters** (const [DbEngineParameters](#) &params)
- void **setDefaultIconSize** (int val)
- void **setDetectFacesInNewImages** (bool val)
- void **setDrawFramesToGrouped** (bool val)
- void **setDuplicatesAlbumTagRelation** (int val)
- void **setDuplicatesReferenceImageSelectionMethod** ([Haarface::RefImageSelMethod](#) val)
- void **setDuplicatesSearchLastMaxSimilarity** (int val)
- void **setDuplicatesSearchLastMinSimilarity** (int val)
- void **setDuplicatesSearchRestrictions** (int val)
- void **setExpandNewCurrentItem** (bool val)
- void **setFaceDetectionAccuracy** (int value)
- void **setFaceDetectionModel** ([FaceScanSettings::FaceDetectionModel](#) model)
- void **setFaceDetectionSize** ([FaceScanSettings::FaceDetectionSize](#) size)
- void **setFaceRecognitionAccuracy** (int value)
- void **setFaceRecognitionModel** ([FaceScanSettings::FaceRecognitionModel](#) model)
- void **setGroupingOperateOnAll** ([OperationType](#) type, ApplyToEntireGroup applyAll)

*Defines whether an operation should be performed on all grouped items or just the head item.*

- void **setHelpBoxNotificationSeen** (bool val)
- void **setIconShowAspectRatio** (bool val)
- void **setIconShowColorLabel** (bool val)
- void **setIconShowComments** (bool val)
- void **setIconShowCoordinates** (bool val)
- void **setIconShowDate** (bool val)
- void **setIconShowFullscreen** (bool val)
- void **setIconShowImageFormat** (bool val)
- void **setIconShowModDate** (bool val)
- void **setIconShowName** (bool val)
- void **setIconShowOverlays** (bool val)

*Sets the visibility of the overlay buttons on the image icons.*

- void **setIconShowPickLabel** (bool val)
- void **setIconShowRating** (bool val)
- void **setIconShowResolution** (bool val)
- void **setIconShowSize** (bool val)
- void **setIconShowTags** (bool val)
- void **setIconShowTitle** (bool val)
- void **setIconTheme** (const QString &theme)
- void **setIconViewFont** (const QFont &font)
- void **setImageSeparationMode** (int mode)
- void **setImageSeparationSortOrder** (int order)
- void **setImageSorting** (int sorting)

*means ascending or descending*

- void **setImageSortOrder** (int order)
- void **setItemLeftClickAction** (int action)
- void **setMinimumSimilarityBound** (int val)
- void **setPreviewSettings** (const [PreviewSettings](#) &settings)
- void **setPreviewShowIcons** (bool val)
- void **setPreviewSmoothScaled** (bool val)
- void **setRatingFilterCond** (int val)
- void **setRecurseAlbums** (bool val)
- void **setRecurseTags** (bool val)
- void **setScaleFitToWindow** (bool val)
- void **setScanAtStart** (bool val)
- void **setScrollItemToCenter** (bool val)

- void **setSelectFirstAlbumItem** (bool val)
- void **setShowAlbumToolTips** (bool val)
- void **setShowFolderTreeViewItemsCount** (bool val)
- void **setShowOnlyPersonTagsInPeopleSidebar** (bool val)
- void **setShowPermanentDeleteDialog** (bool val)
- void **setShowSplashScreen** (bool val)
- void **setShowThumbbar** (bool val)
- void **setShowToolTips** (bool val)
- void **setShowTrashDeleteDialog** (bool val)
- void **setSidebarTitleStyle** ([DMultiTabBar::TextStyle](#) style)
- void **setStringComparisonType** ([ApplicationSettings::StringComparisonType](#) val)
  - Defines the way in which string comparisons are performed.*
- void **setSyncBalooToDigikam** (bool val)
- void **setSyncDigikamToBaloo** (bool val)
- void **setToolTipsFont** (const QFont &font)
- void **setToolTipsShowAlbumCaption** (bool val)
- void **setToolTipsShowAlbumCategory** (bool val)
- void **setToolTipsShowAlbumCollection** (bool val)
- void **setToolTipsShowAlbumDate** (bool val)
- void **setToolTipsShowAlbumName** (bool val)
- void **setToolTipsShowAlbumPreview** (bool val)
- void **setToolTipsShowAlbumTitle** (bool val)
- void **setToolTipsShowComments** (bool val)
- void **setToolTipsShowFileDate** (bool val)
- void **setToolTipsShowFileName** (bool val)
- void **setToolTipsShowFileSize** (bool val)
- void **setToolTipsShowImageAR** (bool val)
- void **setToolTipsShowImageDim** (bool val)
- void **setToolTipsShowImageType** (bool val)
- void **setToolTipsShowLabelRating** (bool val)
- void **setToolTipsShowPhotoDate** (bool val)
- void **setToolTipsShowPhotoExpo** (bool val)
- void **setToolTipsShowPhotoFlash** (bool val)
- void **setToolTipsShowPhotoFocal** (bool val)
- void **setToolTipsShowPhotoLens** (bool val)
- void **setToolTipsShowPhotoMake** (bool val)
- void **setToolTipsShowPhotoMode** (bool val)
- void **setToolTipsShowPhotoWB** (bool val)
- void **setToolTipsShowTags** (bool val)
- void **setToolTipsShowTitles** (bool val)
- void **setToolTipsShowVideoAspectRatio** (bool val)
- void **setToolTipsShowVideoAudioBitRate** (bool val)
- void **setToolTipsShowVideoAudioChannelType** (bool val)
- void **setToolTipsShowVideoAudioCodec** (bool val)
- void **setToolTipsShowVideoDuration** (bool val)
- void **setToolTipsShowVideoFrameRate** (bool val)
- void **setToolTipsShowVideoVideoCodec** (bool val)
- void **setTreeViewFaceSize** (int val)
- void **setTreeViewFont** (const QFont &font)
- void **setTreeViewIconSize** (int val)
- void **setUpdateType** (int type)
- void **setUpdateWithDebug** (bool dbg)
- void **setUseNativeFileDialog** (bool val)
- void **setUseTrash** (bool val)
- void **setVersionManagerSettings** (const [VersionManagerSettings](#) &settings)
- bool **showAlbumToolTipsIsValid** () const
- bool **showOnlyPersonTagsInPeopleSidebar** () const
- bool **showToolTipsIsValid** () const



## Static Public Member Functions

- static [ApplicationSettings](#) \* **instance** ()
- static QString [operationTypeExplanation](#) ([OperationType](#) type)  
*Gives a translated explanation of the operation and an empty string, if there is none (e.g.*
- static QString [operationTypeTitle](#) ([OperationType](#) type)  
*Gives the translated title/short explanation of the operation.*

## Friends

- class [ApplicationSettingsCreator](#)

## 9.86.1 Member Enumeration Documentation

### 9.86.1.1 StringComparisonType

```
enum Digikam::ApplicationSettings::StringComparisonType
```

#### Enumerator

Natural	Natural compare using KStringHandler::naturalCompare.
Normal	Normal comparison using Qt's compare function.

## 9.86.2 Member Function Documentation

### 9.86.2.1 askGroupingOperateOnAll()

```
bool Digikam::ApplicationSettings::askGroupingOperateOnAll (
    OperationType type )
```

Also supplies an option to remember the answer.

#### Parameters

<i>type</i>	Operation to be performed
-------------	---------------------------

#### Returns

Whether to apply to all images or just one

### 9.86.2.2 getGroupingOperateOnAll()

```
ApplicationSettings::ApplyToEntireGroup Digikam::ApplicationSettings::getGroupingOperateOnAll
(
    OperationType type ) const
```

**Parameters**

<i>type</i>	Operation to be performed
-------------	---------------------------

**Returns**

Whether to apply to all images or just one, or ask

**9.86.2.3 getStringComparisonType()**

```
ApplicationSettings::StringComparisonType Digikam::ApplicationSettings::getStringComparisonType ( ) const
```

**Returns**

string comparison type to use.

**9.86.2.4 operationTypeExplanation()**

```
QString Digikam::ApplicationSettings::operationTypeExplanation (
    OperationType type ) [static]
```

for tooltips)

**Parameters**

<i>type</i>	Operation to be performed
-------------	---------------------------

**Returns**

Translated operation explanation

**9.86.2.5 operationTypeTitle()**

```
QString Digikam::ApplicationSettings::operationTypeTitle (
    OperationType type ) [static]
```

**Parameters**

<i>type</i>	Operation to be performed
-------------	---------------------------

**Returns**

Translated operation title/short explanation

**9.86.2.6 readMsgBoxShouldBeShown()**

```
bool Digikam::ApplicationSettings::readMsgBoxShouldBeShown (
    const QString & dontShowAgainName )
```

**Returns**

true if the corresponding message box should be shown.

**Parameters**

<i>dontShowAgainName</i>	the name that identify the message box.
--------------------------	---

**9.86.2.7 saveMsgBoxShouldBeShown()**

```
void Digikam::ApplicationSettings::saveMsgBoxShouldBeShown (
    const QString & dontShowAgainName )
```

**Parameters**

<i>dontShowAgainName</i>	the name that identify the message box. If empty, this method does nothing.
--------------------------	---

**9.86.2.8 setGroupingOperateOnAll()**

```
void Digikam::ApplicationSettings::setGroupingOperateOnAll (
    OperationType type,
    ApplicationSettings::ApplyToEntireGroup applyAll )
```

**Parameters**

<i>type</i>	Operation to be performed
<i>applyAll</i>	Whether to apply to all images or just one, or ask

**9.86.2.9 setStringComparisonType()**

```
void Digikam::ApplicationSettings::setStringComparisonType (
    ApplicationSettings::StringComparisonType val )
```

**Parameters**

<i>val</i>	new way to compare strings
------------	----------------------------

## 9.87 Digikam::AssignedBatchTools Class Reference

Container to assign Batch tools and settings to an item by Url.

### Public Member Functions

- QString **targetSuffix** (bool \*const extSet=nullptr) const

### Public Attributes

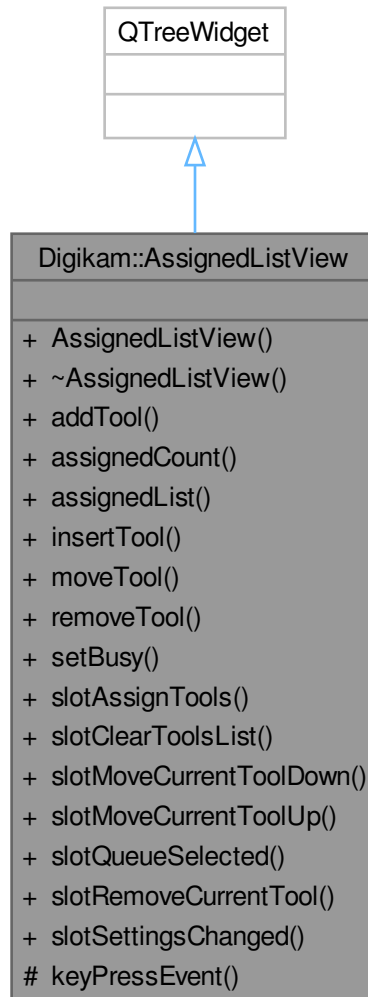
- QString **m\_destFileName**
- QUrl **m\_itemUrl**
- [BatchSetList](#) **m\_toolsList**

### 9.87.1 Detailed Description

Url is used only with [ActionThread](#) class.

## 9.88 Digikam::AssignedListView Class Reference

Inheritance diagram for Digikam::AssignedListView:



### Public Slots

- void **slotAssignTools** (const QMap< int, QString > &)
- void **slotClearToolsList** ()
- void **slotMoveCurrentToolDown** ()
- void **slotMoveCurrentToolUp** ()
- void **slotQueueSelected** (int, const [QueueSettings](#) &, const [AssignedBatchTools](#) &)
- void **slotRemoveCurrentTool** ()
- void **slotSettingsChanged** (const [BatchToolSet](#) &)

## Signals

- void **signalAssignedToolsChanged** (const [AssignedBatchTools](#) &)
- void **signalToolSelected** (const [BatchToolSet](#) &)

## Public Member Functions

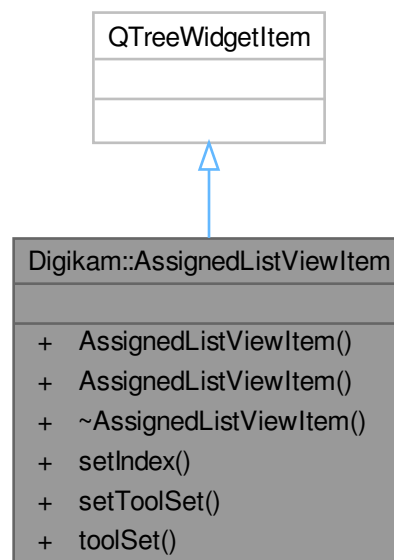
- **AssignedListView** (QWidget \*const parent)
- [AssignedListViewItem](#) \* **addTool** (const [BatchToolSet](#) &set)
- int **assignedCount** ()
- [AssignedBatchTools](#) **assignedList** ()
- [AssignedListViewItem](#) \* **insertTool** ([AssignedListViewItem](#) \*const preceding, const [BatchToolSet](#) &set)
- [AssignedListViewItem](#) \* **moveTool** ([AssignedListViewItem](#) \*const preceding, const [BatchToolSet](#) &set)
- bool **removeTool** (const [BatchToolSet](#) &set)
- void **setBusy** (bool b)

## Protected Member Functions

- void **keyPressEvent** (QKeyEvent \*) override

## 9.89 Digikam::AssignedListViewItem Class Reference

Inheritance diagram for Digikam::AssignedListViewItem:

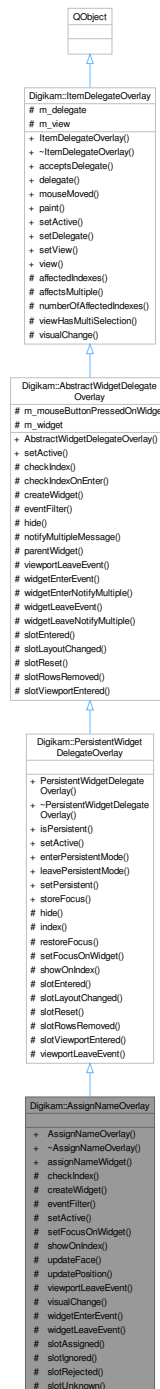


### Public Member Functions

- **AssignedListItem** (QTreeWidgetItem \*const parent)
- **AssignedListItem** (QTreeWidgetItem \*const parent, QTreeWidgetItem \*const preceding)
- void **setIndex** (int index)
- void **setToolSet** (const [BatchToolSet](#) &set)
- [BatchToolSet](#) **toolSet** ()

## 9.90 Digikam::AssignNameOverlay Class Reference

Inheritance diagram for Digikam::AssignNameOverlay:



### Signals

- void **confirmFaces** (const QList< QModelIndex > &indexes, int tagId)
- void **ignoreFaces** (const QList< QModelIndex > &indexes)
- void **removeFaces** (const QList< QModelIndex > &indexes)
- void **unknownFaces** (const QList< QModelIndex > &indexes)



## Signals inherited from [Digikam::ItemDelegateOverlay](#)

- void **hideNotification** ()
- void **requestNotification** (const QModelIndex &index, const QString &message)
- void **update** (const QModelIndex &index)

## Public Member Functions

- **AssignNameOverlay** (QObject \*const parent)
- [AssignNameWidget](#) \* **assignNameWidget** () const

## Public Member Functions inherited from [Digikam::PersistentWidgetDelegateOverlay](#)

- [PersistentWidgetDelegateOverlay](#) (QObject \*const parent)
 

*This class offers additional / modified behavior: When a "persistent" mode is entered, it will not move by mouse hover, but stay and only move on mouse click.*
- bool **isPersistent** () const
- void [setActive](#) (bool active) override
 

*If active is true, this will call [createWidget\(\)](#), initialize the widget for use, and setup connections for the virtual slots.*

## Public Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- [AbstractWidgetDelegateOverlay](#) (QObject \*const parent)
 

*This class provides functionality for using a widget in an overlay.*

## Public Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- **ItemDelegateOverlay** (QObject \*const parent=nullptr)
- virtual bool **acceptsDelegate** (QAbstractItemDelegate \*) const
- QAbstractItemDelegate \* **delegate** () const
- virtual void [mouseMoved](#) (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)
 

*Only these two methods are implemented as virtual methods.*
- virtual void **paint** (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index)
- void **setDelegate** (QAbstractItemDelegate \*delegate)
- void **setView** (QAbstractItemView \*view)
- QAbstractItemView \* **view** () const

## Protected Slots

- void **slotAssigned** (const [TaggingAction](#) &action, const [ItemInfo](#) &, const QVariant &faceIdentifier)
- void **slotIgnored** (const [ItemInfo](#) &, const QVariant &faceIdentifier)
- void **slotRejected** (const [ItemInfo](#) &, const QVariant &faceIdentifier)
- void **slotUnknown** (const [ItemInfo](#) &, const QVariant &faceIdentifier)

## Protected Slots inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

## Protected Slots inherited from [Digikam::ItemDelegateOverlay](#)

### Protected Member Functions

- bool [checkIndex](#) (const QModelIndex &index) const override  
*Return true here if you want to show the overlay for the given index.*
- QWidget \* [createWidget](#) () override  
*Create your widget here.*
- bool [eventFilter](#) (QObject \*o, QEvent \*e) override
- void [setActive](#) (bool) override  
*If active is true, this will call [createWidget\(\)](#), initialize the widget for use, and setup connections for the virtual slots.*
- void [setFocusOnWidget](#) () override  
*Reimplement to set the focus on the correct subwidget.*
- void [showOnIndex](#) (const QModelIndex &index) override  
*see [slotEntered\(\)](#)*
- void [updateFace](#) ()
- void [updatePosition](#) ()
- void [viewportLeaveEvent](#) (QObject \*obj, QEvent \*event) override  
*Called when a QEvent::Leave of the viewport is received.*
- void [visualChange](#) () override  
*Called when any change from the delegate occurs - when the overlay is installed, when size hints, styles or fonts change.*
- void [widgetEnterEvent](#) () override  
*Called when a QEvent::Enter resp.*
- void [widgetLeaveEvent](#) () override

## Protected Member Functions inherited from [Digikam::PersistentWidgetDelegateOverlay](#)

- void [hide](#) () override  
*Called when the widget shall be hidden (mouse cursor left index, viewport, uninstalled etc.).*
- QModelIndex [index](#) () const
- void [restoreFocus](#) ()
- void [slotEntered](#) (const QModelIndex &index) override  
*Most overlays reimplement this slot to get the starting point for repositioning a widget etc.*
- void [slotLayoutChanged](#) () override
- void [slotReset](#) () override  
*Default implementations of these three slots call [hide\(\)](#)*
- void [slotRowsRemoved](#) (const QModelIndex &parent, int start, int end) override
- void [slotViewportEntered](#) () override
- void [viewportLeaveEvent](#) (QObject \*obj, QEvent \*event) override  
*Called when a QEvent::Leave of the viewport is received.*

## Protected Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- bool [checkIndexOnEnter](#) (const QModelIndex &index) const  
*Utility method called from [slotEntered](#).*
- bool [eventFilter](#) (QObject \*obj, QEvent \*event) override
- virtual QString [notifyMultipleMessage](#) (const QModelIndex &, int number)
- QWidget \* [parentWidget](#) () const  
*Returns the widget to be used as parent for your widget created in [createWidget\(\)](#)*
- void [widgetEnterNotifyMultiple](#) (const QModelIndex &index)  
*A sample implementation for above methods.*
- void [widgetLeaveNotifyMultiple](#) ()

## Protected Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- `QList< QModelIndex > affectedIndexes (const QModelIndex &index) const`
- `bool affectsMultiple (const QModelIndex &index) const`  
*For the context that an overlay can affect multiple items: Assuming the currently overlaid index is given.*
- `int numberOfAffectedIndexes (const QModelIndex &index) const`
- `bool viewHasMultiSelection () const`  
*Utility method.*

## Additional Inherited Members

## Public Slots inherited from [Digikam::PersistentWidgetDelegateOverlay](#)

- `void enterPersistentMode ()`
- `void leavePersistentMode ()`
- `void setPersistent (bool persistent)`  
*Enters persistent mode.*
- `void storeFocus ()`

## Protected Attributes inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- `bool m_mouseButtonPressedOnWidget = false`
- `QWidget * m_widget = nullptr`

## Protected Attributes inherited from [Digikam::ItemDelegateOverlay](#)

- `QAbstractItemDelegate * m_delegate = nullptr`
- `QAbstractItemView * m_view = nullptr`

## 9.90.1 Member Function Documentation

### 9.90.1.1 `checkIndex()`

```
bool Digikam::AssignNameOverlay::checkIndex (
    const QModelIndex & index ) const [override], [protected], [virtual]
```

The default implementation returns true.

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.90.1.2 `createWidget()`

```
QWidget * Digikam::AssignNameOverlay::createWidget ( ) [override], [protected], [virtual]
```

When creating the object, pass [parentWidget\(\)](#) as parent widget. Ownership of the object is passed. It will be deleted in `setActive(false)`.

Implements [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.90.1.3 setActive()

```
void Digikam::AssignNameOverlay::setActive (
    bool active ) [override], [protected], [virtual]
```

If active is false, this will delete the widget and disconnect all signal from model and view to this object (!)

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.90.1.4 setFocusOnWidget()

```
void Digikam::AssignNameOverlay::setFocusOnWidget ( ) [override], [protected], [virtual]
```

Default implementation sets focus on widget()

Reimplemented from [Digikam::PersistentWidgetDelegateOverlay](#).

### 9.90.1.5 showOnIndex()

```
void Digikam::AssignNameOverlay::showOnIndex (
    const QModelIndex & index ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::PersistentWidgetDelegateOverlay](#).

### 9.90.1.6 updateFace()

```
void Digikam::AssignNameOverlay::updateFace ( ) [protected]
```

The order to plug these functions is important, since `setUserData()` controls how the Overlay appears on a particular face.

### 9.90.1.7 viewportLeaveEvent()

```
void Digikam::AssignNameOverlay::viewportLeaveEvent (
    QObject * obj,
    QEvent * event ) [override], [protected], [virtual]
```

The default implementation [hide\(\)](#)s.

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.90.1.8 visualChange()

```
void Digikam::AssignNameOverlay::visualChange ( ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ItemDelegateOverlay](#).

### 9.90.1.9 widgetEnterEvent()

```
void Digikam::AssignNameOverlay::widgetEnterEvent ( ) [override], [protected], [virtual]
```

QEvent::Leave event for the widget is received. The default implementation does nothing.

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

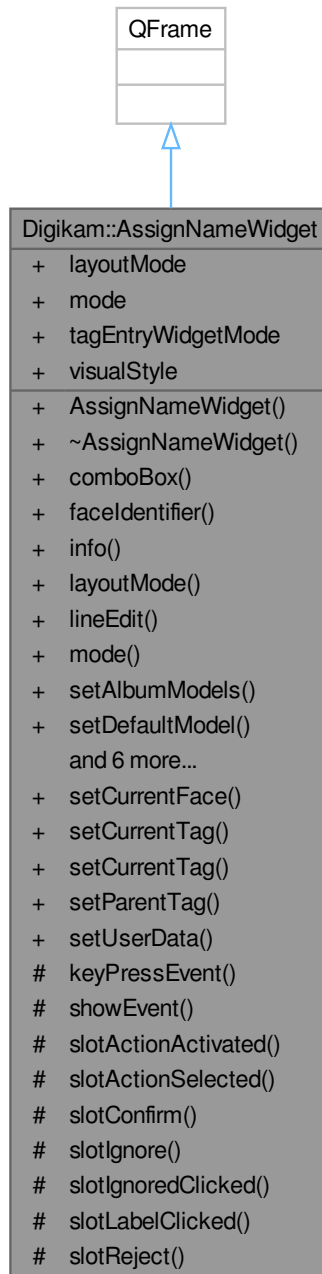
### 9.90.1.10 widgetLeaveEvent()

```
void Digikam::AssignNameOverlay::widgetLeaveEvent ( ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

## 9.91 Digikam::AssignNameWidget Class Reference

Inheritance diagram for Digikam::AssignNameWidget:



### Public Types

- enum **LayoutMode** { **InvalidLayout** , **FullLine** , **TwoLines** , **Compact** }

- enum **Mode** { **InvalidMode** , **UnconfirmedEditMode** , **ConfirmedMode** , **ConfirmedEditMode** , **IgnoredMode** }
- enum **TagEntryWidgetMode** { **InvalidTagEntryWidgetMode** , **AddTagsComboBoxMode** , **AddTagsLineEditMode** }
- enum **VisualStyle** { **InvalidVisualStyle** , **StyledFrame** , **TranslucentDarkRound** , **TranslucentThemedFrameless** }

## Public Slots

- void **setCurrentFace** (const [FaceTagsface](#) &face)
- void **setCurrentTag** (int tagId)  
*Sets the suggested (UnconfirmedEditMode) or assigned (ConfirmedMode) tag to be displayed.*
- void **setCurrentTag** ([TAlbum](#) \*album)
- void **setParentTag** ([TAlbum](#) \*album)  
*Set a parent tag for suggesting a parent tag for a new tag, and a default action.*
- void **setUserData** (const [ItemInfo](#) &info, const [QVariant](#) &faceIdentifier=[QVariant](#)())  
*The identifying information emitted with the signals.*

## Signals

- void **assigned** (const [TaggingAction](#) &action, const [ItemInfo](#) &info, const [QVariant](#) &faceIdentifier)  
*A name has been assigned to the associated face.*
- void **ignored** (const [ItemInfo](#) &info, const [QVariant](#) &faceIdentifier)
- void **ignoredClicked** (const [ItemInfo](#) &info, const [QVariant](#) &faceIdentifier)  
*In IgnoredMode, this signal is emitted when the user clicked on the label.*
- void **labelClicked** (const [ItemInfo](#) &info, const [QVariant](#) &faceIdentifier)  
*In ConfirmedMode, this signal is emitted when the user clicked on the label.*
- void **rejected** (const [ItemInfo](#) &info, const [QVariant](#) &faceIdentifier)  
*The suggestion has been rejected and the face will be moved to Unknown.*
- void **selected** (const [TaggingAction](#) &action, const [ItemInfo](#) &info, const [QVariant](#) &faceIdentifier)  
*An action has been selected.*

## Public Member Functions

- **AssignNameWidget** ([QWidget](#) \*const parent=nullptr)  
*Please take care: you must set all four modes before usage!*
- **AddTagsComboBox** \* **comboBox** () const  
*The combo box or line edit in use, if any.*
- [QVariant](#) **faceIdentifier** () const
- [ItemInfo](#) **info** () const
- [LayoutMode](#) **layoutMode** () const
- **AddTagsLineEdit** \* **lineEdit** () const
- [Mode](#) **mode** () const
- void **setAlbumModels** ([TagModel](#) \*const model, [TagPropertiesFilterModel](#) \*const filteredModel, [CheckableAlbumFilterModel](#) \*const filterModel)  
*Set the tag model to use for completion.*
- void **setDefaultModel** ()
- void **setLayoutMode** ([LayoutMode](#) mode)
- void **setMode** ([Mode](#) mode)
- void **setTagEntryWidgetMode** ([TagEntryWidgetMode](#) mode)
- void **setVisualStyle** ([VisualStyle](#) style)
- [TagEntryWidgetMode](#) **tagEntryWidgetMode** () const
- [VisualStyle](#) **visualStyle** () const

## Protected Slots

- void **slotActionActivated** (const [TaggingAction](#) &action)
- void **slotActionSelected** (const [TaggingAction](#) &action)
- void **slotConfirm** ()
- void **slotIgnore** ()
- void **slotIgnoredClicked** ()
- void **slotLabelClicked** ()
- void **slotReject** ()

## Protected Member Functions

- void **keyPressEvent** (QKeyEvent \*e) override
- void **showEvent** (QShowEvent \*e) override

## Properties

- LayoutMode **layoutMode**
- Mode **mode**
- TagEntryWidgetMode **tagEntryWidgetMode**
- VisualStyle **visualStyle**

## 9.91.1 Member Function Documentation

### 9.91.1.1 assigned

```
void Digikam::AssignNameWidget::assigned (
    const TaggingAction & action,
    const ItemInfo & info,
    const QVariant & faceIdentifier ) [signal]
```

This can be an existing tag, or a new tag, as described by [TaggingAction](#). For convenience, info() and faceIdentifier() are provided.

### 9.91.1.2 rejected

```
void Digikam::AssignNameWidget::rejected (
    const ItemInfo & info,
    const QVariant & faceIdentifier ) [signal]
```

For convenience, info() and faceIdentifier() are provided.

### 9.91.1.3 selected

```
void Digikam::AssignNameWidget::selected (
    const TaggingAction & action,
    const ItemInfo & info,
    const QVariant & faceIdentifier ) [signal]
```

This purely signals user interaction, no fixed decision - mouse hover may be enough to emit this signal. The action may be invalid (user switched back to empty selection).



#### 9.91.1.4 setMode()

```
void Digikam::AssignNameWidget::setMode (
    Mode mode )
```

Reject tooltip and icon should be updated even if the same mode is passed, because Unconfirmed and Unknown. Faces have the same mode but different tooltips and icons.

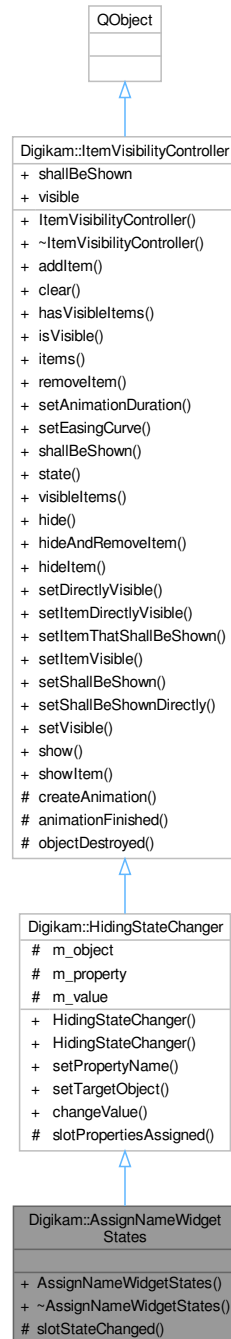
#### 9.91.1.5 setUserData

```
void Digikam::AssignNameWidget::setUserData (
    const ItemInfo & info,
    const QVariant & faceIdentifier = QVariant() ) [slot]
```

Ignored faces are drawn over with a different overlay, as Reject button should be disabled.

## 9.92 Digikam::AssignNameWidgetStates Class Reference

Inheritance diagram for Digikam::AssignNameWidgetStates:



### Public Member Functions

- **AssignNameWidgetStates** ([FacelItem](#) \*const item)

## Public Member Functions inherited from Digikam::HidingStateChanger

- [HidingStateChanger](#) (QObject \*const parent=nullptr)
 

*This class provides a state change while fading in and out: When changeValue is called, first the items are hidden, when this is finished, the property is assigned to the object.*
- **HidingStateChanger** (QObject \*const target, const QByteArray &property, QObject \*const parent=nullptr)
 

*Convenience constructor: Sets target and property name.*
- void **setProperty** (const QByteArray &propertyName)
- void **setTargetObject** (QObject \*const object)

## Public Member Functions inherited from Digikam::ItemVisibilityController

- **ItemVisibilityController** (QObject \*const parent=nullptr)
- void **addItem** (QObject \*const object)
 

*Add and remove objects.*
- void **clear** ()
 

*Remove all animations.*
- bool **hasVisibleItems** ([IncludeFadingOutMode](#) mode=[IncludeFadingOut](#)) const
 

*This returns the "result" of isVisible and shallBeShown: Something is indeed visible on the scene.*
- bool **isVisible** () const
- QList< QObject \* > **items** () const
 

*Returns all items under control.*
- void **removeItem** (QObject \*const object)
- void **setAnimationDuration** (int msec)
- void **setEasingCurve** (const QEasingCurve &easing)
 

*Allows to change the default parameters of all animations.*
- bool **shallBeShown** () const
- [State](#) **state** () const
- QList< QObject \* > **visibleItems** ([IncludeFadingOutMode](#) mode=[IncludeFadingOut](#)) const
 

*Returns all currently visible items.*

## Protected Slots

- void **slotStateChanged** ()

## Protected Slots inherited from Digikam::HidingStateChanger

- void **slotPropertiesAssigned** (bool)

## Protected Slots inherited from Digikam::ItemVisibilityController

- void **animationFinished** ()
- void **objectDestroyed** (QObject \*)

## Additional Inherited Members

## Public Types inherited from Digikam::ItemVisibilityController

- enum [IncludeFadingOutMode](#) { [IncludeFadingOut](#) , [ExcludeFadingOut](#) }
  - enum [State](#) { [Hidden](#) , [FadingIn](#) , [Visible](#) , [FadingOut](#) }
- This class handles complex visibility situations for items.*

## Public Slots inherited from [Digikam::HidingStateChanger](#)

- void **changeValue** (const QVariant &value)

## Public Slots inherited from [Digikam::ItemVisibilityController](#)

- void **hide** ()
- void **hideAndRemoveItem** (QObject \*item)
  - Hide the item, and then remove it.*
- void **hideItem** (QObject \*item)
- void **setDirectlyVisible** (bool visible)
- void **setItemDirectlyVisible** (QObject \*item, bool visible)
- void **setItemThatShallBeShown** (QObject \*item)
  - Sets a single item to be shown.*
- void **setItemVisible** (QObject \*item, bool visible)
- void **setShallBeShown** (bool shallBeShown)
  - Adjusts the first condition - the items are shown if shallBeShown is true and isVisible is true.*
- void **setShallBeShownDirectly** (bool shallBeShown)
- void **setVisible** (bool visible)
- void **show** ()
  - Adjusts the main condition.*
- void **showItem** (QObject \*item)
  - Shows or hides a single item.*

## Signals inherited from [Digikam::HidingStateChanger](#)

- void **finished** ()
  - Emitted when the items were hidden, the target object's property changed, and the items shown again.*
- void **stateChanged** ()
  - Emitted when the items were hidden and the target object's property changed.*

## Signals inherited from [Digikam::ItemVisibilityController](#)

- void **hiddenAndRemoved** (QObject \*item)
  - Emitted when hideAndRemoveItem has finished.*
- void **propertiesAssigned** (bool visible)
  - Emitted when the (main) transition has finished.*
- void **propertiesAssigned** (QObject \*item, bool visible)
  - Emitted when a transition for a single item finished (see setItemVisible())*

## Protected Member Functions inherited from [Digikam::ItemVisibilityController](#)

- virtual QPropertyAnimation \* **createAnimation** (QObject \*item)
  - Creates the animation for showing and hiding the given item.*

## Protected Attributes inherited from [Digikam::HidingStateChanger](#)

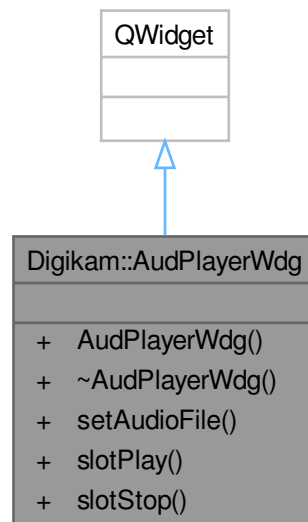
- QObject \* **m\_object** = nullptr
- QByteArray **m\_property**
- QVariant **m\_value**

### Properties inherited from [Digikam::ItemVisibilityController](#)

- bool **shallBeShown**
- bool **visible**

## 9.93 Digikam::AudPlayerWdg Class Reference

Inheritance diagram for Digikam::AudPlayerWdg:



### Public Slots

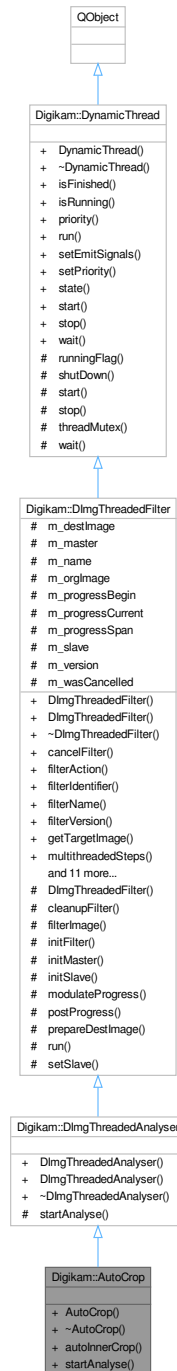
- void **slotPlay** ()
- void **slotStop** ()

### Public Member Functions

- **AudPlayerWdg** (`QWidget *const parent=nullptr`)
- void **setAudioFile** (`const QString &afile`)

## 9.94 Digikam::AutoCrop Class Reference

Inheritance diagram for Digikam::AutoCrop:



### Public Member Functions

- **AutoCrop** (`DImg *const orgImage, QObject *const parent=nullptr`)  
*Standard constructor with image container to parse.*

- QRect **autoInnerCrop** () const  
*Return inner crop area detected by [startAnalyse\(\)](#).*
- void **startAnalyse** () override  
*Perform auto-crop analyze to find best inner crop.*

### Public Member Functions inherited from [Digikam::DImgThreadedAnalyser](#)

- [DImgThreadedAnalyser](#) (DImg \*const orgImage, QObject \*const parent=nullptr, const QString &name=QString())  
*Constructs an image analyser with all arguments (ready to use).*
- [DImgThreadedAnalyser](#) (QObject \*const parent=nullptr, const QString &name=QString())  
*Constructs a filter without argument.*

### Public Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) (DImg \*const orgImage, QObject \*const parent, const QString &name=QString())  
*Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter](#) (QObject \*const parent=nullptr, const QString &name=QString())  
*Constructs a filter without argument.*
- virtual void **cancelFilter** ()  
*Cancel the threaded computation.*
- const QString & **filterName** ()
- int **filterVersion** () const
- [DImg](#) **getTargetImage** ()
- QList< int > **multithreadedSteps** (int stop, int start=0) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool **parametersSuccessfullyRead** () const  
*Optional: error handling for readParameters.*
- virtual QString **readParametersError** (const [FilterAction](#) &actionThatFailed) const
- void **setFilterName** (const QString &name)
- void **setFilterVersion** (int version)  
*Replaying a filter action: Set the filter version.*
- void **setOriginalImage** (const [DImg](#) &orgImage)
- void **setupAndStartDirectly** (const [DImg](#) &orgImage, [DImgThreadedFilter](#) \*const master, int progress←Begin=0, int progressEnd=100)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void **setupFilter** (const [DImg](#) &orgImage)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void **startFilter** ()  
*Start the threaded computation.*
- virtual void **startFilterDirectly** ()  
*Start computation of this filter, directly in this thread.*

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread](#) (QObject \*const parent=nullptr)
  - This class extends QRunnable, so you have to reimplement virtual void [run\(\)](#).*
- [~DynamicThread](#) () override
  - The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool [isFinished](#) () const
- bool [isRunning](#) () const
- QThread::Priority [priority](#) () const
- void [setEmitSignals](#) (bool emitThem)
- void [setPriority](#) (QThread::Priority priority)
  - Sets the priority for this dynamic thread.*
- State [state](#) () const

## Additional Inherited Members

## Public Types inherited from [Digikam::DynamicThread](#)

- enum [State](#) { [Inactive](#) , [Scheduled](#) , [Running](#) , [Deactivating](#) }

## Public Slots inherited from [Digikam::DynamicThread](#)

- void [start](#) ()
- void [stop](#) ()
  - Stop computation, sets the running flag to false.*
- void [wait](#) ()
  - Waits until the thread finishes.*

## Signals inherited from [Digikam::DImgThreadedFilter](#)

- void [finished](#) (bool success)
  - Emitted when the computation has completed.*
- void [progress](#) (int progress)
  - Emitted when progress info from the calculation is available.*
- void [started](#) ()
  - This signal is emitted when image data is available and the computation has started.*

## Signals inherited from [Digikam::DynamicThread](#)

- void [finished](#) ()
- void [starting](#) ()
  - Emitted if emitSignals is enabled.*



## Protected Member Functions inherited from Digikam::DImgThreadedFilter

- **DImgThreadedFilter** (**DImgThreadedFilter** \*const master, const **DImg** &orgImage, const **DImg** &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void **cleanupFilter** ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void **initFilter** ()  
*Start filter operation before threaded method.*
- void **initMaster** ()
- void **initSlave** (**DImgThreadedFilter** \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int **modulateProgress** (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void **postProgress** (int progress)  
*Emit progress info.*
- void **run** () override  
*List of threaded operations by filter.*
- void **setSlave** (**DImgThreadedFilter** \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from Digikam::DynamicThread

- bool **runningFlag** () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void **shutDown** ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void **start** (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void **stop** (const QMutexLocker< QMutex > &locker)
- QMutex \* **threadMutex** () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void **wait** (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from Digikam::DImgThreadedFilter

- **DImg m\_destImage**  
*Output image data.*
- **DImgThreadedFilter** \* **m\_master** = nullptr  
*The master of this slave filter.*
- QString **m\_name**  
*Filter name.*
- **DImg m\_orgImage**  
*Copy of original Image data.*
- int **m\_progressBegin** = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int **m\_progressCurrent** = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int **m\_progressSpan** = 0
- **DImgThreadedFilter** \* **m\_slave** = nullptr  
*The current slave.*
- int **m\_version** = 1
- bool **m\_wasCancelled** = false

## 9.94.1 Member Function Documentation

### 9.94.1.1 startAnalyse()

```
void Digikam::AutoCrop::startAnalyse ( ) [override], [virtual]
```

Use [autoInnerCrop\(\)](#) to get computed area. This would be done in 4 steps

1. Search column wise: (a) From the left to the right, this is to get the left boundary (b) From the right to the left, this is to get the right boundary
2. Search row wise : (a) From the top to the bottom, this is to get the top boundary (b) From the bottom to the top, this is to get the bottom boundary

Implements [Digikam::DImgThreadedAnalyser](#).

## 9.95 Digikam::AutoExpoFilter Class Reference

Inheritance diagram for Digikam::AutoExpoFilter:



### Public Member Functions

- **AutoExpoFilter** (`Dimg *const orgImage`, `const Dimg *const reflImage`, `QObject *const parent=nullptr`)
- **AutoExpoFilter** (`QObject *const parent=nullptr`)

- [FilterAction filterAction](#) () override  
*Returns the action description corresponding to currently set options.*
- [QString filterIdentifier](#) () const override  
*Return the identifier for this filter in the image history.*
- void [readParameters](#) (const [FilterAction](#) &action) override

### Public Member Functions inherited from [Digikam::WBFilter](#)

- **WBFilter** (const [WBContainer](#) &settings, [DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100)
- **WBFilter** ([DImg](#) \*const orgImage, [QObject](#) \*const parent=nullptr, const [WBContainer](#) &settings=[WBContainer](#)())
- **WBFilter** ([QObject](#) \*const parent=nullptr)
- [FilterAction filterAction](#) () override  
*Returns the action description corresponding to currently set options.*
- [QString filterIdentifier](#) () const override  
*Return the identifier for this filter in the image history.*
- void [readParameters](#) (const [FilterAction](#) &action) override

### Public Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImg](#) \*const orgImage, [QObject](#) \*const parent, const [QString](#) &name=[QString](#)())  
*Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter](#) ([QObject](#) \*const parent=nullptr, const [QString](#) &name=[QString](#)())  
*Constructs a filter without argument.*
- virtual void [cancelFilter](#) ()  
*Cancel the threaded computation.*
- const [QString](#) & **filterName** ()
- int **filterVersion** () const
- [DImg](#) **getTargetImage** ()
- [QList](#)< int > [multithreadedSteps](#) (int stop, int start=0) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool [parametersSuccessfullyRead](#) () const  
*Optional: error handling for readParameters.*
- virtual [QString](#) **readParametersError** (const [FilterAction](#) &actionThatFailed) const
- void **setFilterName** (const [QString](#) &name)
- void [setFilterVersion](#) (int version)  
*Replaying a filter action: Set the filter version.*
- void **setOriginalImage** (const [DImg](#) &orgImage)
- void **setupAndStartDirectly** (const [DImg](#) &orgImage, [DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void [setupFilter](#) (const [DImg](#) &orgImage)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void **startFilter** ()  
*Start the threaded computation.*
- virtual void **startFilterDirectly** ()  
*Start computation of this filter, directly in this thread.*
- virtual [QList](#)< int > **supportedVersions** () const

## Public Member Functions inherited from Digikam::DynamicThread

- [DynamicThread](#) (QObject \*const parent=nullptr)  
*This class extends QRunnable, so you have to reimplement virtual void [run\(\)](#).*
- [~DynamicThread](#) () override  
*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool [isFinished](#) () const
- bool [isRunning](#) () const
- QThread::Priority [priority](#) () const
- void [setEmitSignals](#) (bool emitThem)
- void [setPriority](#) (QThread::Priority priority)  
*Sets the priority for this dynamic thread.*
- State [state](#) () const

## Static Public Member Functions

- static int [CurrentVersion](#) ()
- static QString [DisplayableName](#) ()
- static QString [FilterIdentifier](#) ()
- static QList< int > [SupportedVersions](#) ()

## Static Public Member Functions inherited from Digikam::WBFilter

- static void [autoExposureAdjustement](#) (const [DImg](#) \*const img, double &black, double &expo)
- static void [autoWBAdjustementFromColor](#) (const QColor &tc, double &temperature, double &green)
- static int [CurrentVersion](#) ()
- static QString [DisplayableName](#) ()
- static QString [FilterIdentifier](#) ()
- static QList< int > [SupportedVersions](#) ()

## Additional Inherited Members

## Public Types inherited from Digikam::DynamicThread

- enum [State](#) { [Inactive](#) , [Scheduled](#) , [Running](#) , [Deactivating](#) }

## Public Slots inherited from Digikam::DynamicThread

- void [start](#) ()
- void [stop](#) ()  
*Stop computation, sets the running flag to false.*
- void [wait](#) ()  
*Waits until the thread finishes.*

## Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

## Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from [Digikam::WBFilter](#)

- void **filterImage** () override  
*Main image filter method.*

## Protected Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void **cleanupFilter** ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void **initFilter** ()  
*Start filter operation before threaded method.*
- void **initMaster** ()
- void **initSlave** ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int **modulateProgress** (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void **postProgress** (int progress)  
*Emit progress info.*
- virtual void **prepareDestImage** ()
- void **run** () override  
*List of threaded operations by filter.*
- void **setSlave** ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool **runningFlag** () const volatile  
*In you [run\(\)](#) method, you shall regularly check for [runningFlag\(\)](#) and cleanup and return if false.*
- void **shutDown** ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call [stop\(\)](#) and [wait\(\)](#), knowing that nothing will call [start\(\)](#) anymore after this 3) Be sure the thread will never be running at destruction.*
- void **start** (QMutexLocker< QMutex > &locker)  
*Doing the same as [start\(\)](#), [stop\(\)](#) and [wait](#) above, provide it with a locked QMutexLocker on mutex().*
- void **stop** (const QMutexLocker< QMutex > &locker)
- QMutex \* **threadMutex** () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void **wait** (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from [Digikam::WBFilter](#)

- [WBContainer](#) **m\_settings**

## Protected Attributes inherited from [Digikam::DImgThreadedFilter](#)

- [DImg](#) **m\_destImage**  
*Output image data.*
- [DImgThreadedFilter](#) \* **m\_master** = nullptr  
*The master of this slave filter.*
- QString **m\_name**  
*Filter name.*
- [DImg](#) **m\_orgImage**  
*Copy of original Image data.*
- int **m\_progressBegin** = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int **m\_progressCurrent** = 0  
*To prevent signals bombarding with progress indicator value in [postProgress\(\)](#).*
- int **m\_progressSpan** = 0
- [DImgThreadedFilter](#) \* **m\_slave** = nullptr  
*The current slave.*
- int **m\_version** = 1
- bool **m\_wasCancelled** = false

### 9.95.1 Member Function Documentation

#### 9.95.1.1 [filterAction\(\)](#)

[FilterAction](#) Digikam::AutoExpoFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.95.1.2 filterIdentifier()

```
QString Digikam::AutoExpoFilter::filterIdentifier ( ) const [inline], [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

### 9.95.1.3 readParameters()

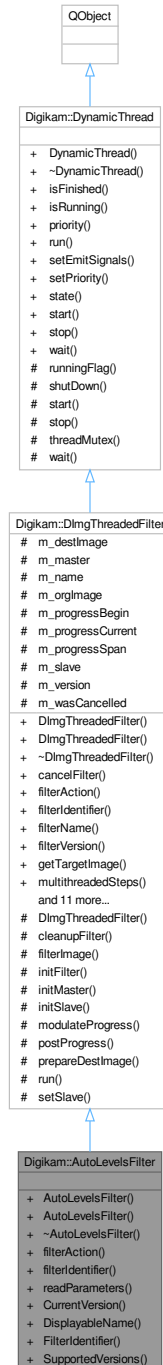
```
void Digikam::AutoExpoFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).



## 9.96 Digikam::AutoLevelsFilter Class Reference

Inheritance diagram for Digikam::AutoLevelsFilter:



### Public Member Functions

- **AutoLevelsFilter** (`Dimg *const orgImage`, `const Dimg *const reflImage`, `QObject *const parent=nullptr`)
- **AutoLevelsFilter** (`QObject *const parent=nullptr`)

- [FilterAction filterAction \(\)](#) override  
*Returns the action description corresponding to currently set options.*
- [QString filterIdentifier \(\)](#) const override  
*Return the identifier for this filter in the image history.*
- void [readParameters \(const FilterAction &action\)](#) override

## Public Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter \(DImg \\*const orgImage, QObject \\*const parent, const QString &name=QString\(\)\)](#)  
*Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter \(QObject \\*const parent=nullptr, const QString &name=QString\(\)\)](#)  
*Constructs a filter without argument.*
- virtual void [cancelFilter \(\)](#)  
*Cancel the threaded computation.*
- const [QString &filterName \(\)](#)
- int [filterVersion \(\)](#) const
- [DImg getTargetImage \(\)](#)
- [QList< int > multithreadedSteps \(int stop, int start=0\)](#) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool [parametersSuccessfullyRead \(\)](#) const  
*Optional: error handling for readParameters.*
- virtual [QString readParametersError \(const FilterAction &actionThatFailed\)](#) const
- void [setFilterName \(const QString &name\)](#)
- void [setFilterVersion \(int version\)](#)  
*Replaying a filter action: Set the filter version.*
- void [setOriginalImage \(const DImg &orgImage\)](#)
- void [setupAndStartDirectly \(const DImg &orgImage, DImgThreadedFilter \\*const master, int progress←Begin=0, int progressEnd=100\)](#)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void [setupFilter \(const DImg &orgImage\)](#)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void [startFilter \(\)](#)  
*Start the threaded computation.*
- virtual void [startFilterDirectly \(\)](#)  
*Start computation of this filter, directly in this thread.*
- virtual [QList< int > supportedVersions \(\)](#) const

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread \(QObject \\*const parent=nullptr\)](#)  
*This class extends QRunnable, so you have to reimplement virtual void [run\(\)](#).*
- [~DynamicThread \(\)](#) override  
*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool [isFinished \(\)](#) const
- bool [isRunning \(\)](#) const
- [QThread::Priority priority \(\)](#) const
- void [setEmitSignals \(bool emitThem\)](#)
- void [setPriority \(QThread::Priority priority\)](#)  
*Sets the priority for this dynamic thread.*
- State [state \(\)](#) const

### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from [Digikam::DImgThreadedFilter](#)

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.96.1 Member Function Documentation

### 9.96.1.1 filterAction()

`FilterAction` Digikam::AutoLevelsFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.96.1.2 filterIdentifier()

`QString` Digikam::AutoLevelsFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.96.1.3 readParameters()

```
void Digikam::AutoLevelsFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

## 9.97 Digikam::AutoTagsAssign Class Reference

### Public Member Functions

- **AutoTagsAssign** ([AutoTagsScanSettings::DetectorModel](#) model=[AutoTagsScanSettings::YOLOV5NANO](#))
- `QList< QString >` **generateTagsList** (const [DImg](#) &inputImage)
- `QList< QString >` **generateTagsList** (const [QImage](#) &inputImage)
- `QList< QList< QString > >` **generateTagsList** (const `QList< DImg >` &inputImages, int batchSize) const  
*Run in batch return the list of tags name corresponding to.*
- `QList< QList< QString > >` **generateTagsList** (const `QList< QString >` &inputImagePaths, int batchSize) const
- `QList< QString >` **generateTagsList** (const `QString` &inputImagePath)
- `QList< QString >` **getPredefinedTagsPath** ( ) const
- `cv::Mat` **prepareForDetection** (const [DImg](#) &inputImage) const
- `cv::Mat` **prepareForDetection** (const [QImage](#) &inputImage) const
- `std::vector< cv::Mat >` **prepareForDetection** (const `QList< DImg >` &inputImages, int batchSize) const
- `std::vector< cv::Mat >` **prepareForDetection** (const `QList< QString >` &inputImagePaths, int batchSize) const
- `cv::Mat` **prepareForDetection** (const `QString` &inputImagePath) const

## 9.97.1 Member Function Documentation

### 9.97.1.1 generateTagsList()

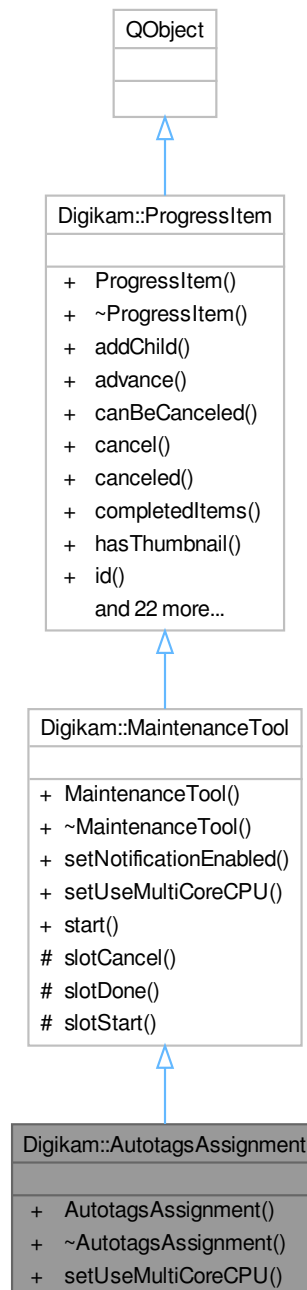
```
QList< QList< QString > > Digikam::AutoTagsAssign::generateTagsList (
    const QList< DImg > & inputImages,
    int batchSize ) const
```

#### Note

the batch size is fixed depending on the deep NN model we choose.

## 9.98 Digikam::AutotagsAssignment Class Reference

Inheritance diagram for Digikam::AutotagsAssignment:



### Public Member Functions

- [AutotagsAssignment](#) ([AutoTagsScanSettings::ScanMode](#) mode, const AlbumList &list, int modelType, const QStringList &langs, [ProgressItem](#) \*const parent=nullptr)

*Constructor using AlbumList as argument.*

- void [setUseMultiCoreCPU](#) (bool b) override

*Re-implement this method if your tool is able to use multi-core CPU to process item in parallel.*

## Public Member Functions inherited from [Digikam::MaintenanceTool](#)

- **MaintenanceTool** (const QString &id, [ProgressItem](#) \*const parent=nullptr)
- void **setNotificationEnabled** (bool b)

*If true, show a notification message on desktop notification manager with time elapsed to run process.*

## Public Member Functions inherited from [Digikam::ProgressItem](#)

- **ProgressItem** ([ProgressItem](#) \*const parent, const QString &id, const QString &label, const QString &status, bool canBeCanceled, bool hasThumb)
- void **addChild** ([ProgressItem](#) \*const kiddo)
- bool **advance** (unsigned int v)

*Advance total items processed by n values and update percentage in progressbar.*

- bool **canBeCanceled** () const
- void **cancel** ()
- bool **canceled** () const
- unsigned int **completedItems** () const
- bool **hasThumbnail** () const
- const QString & **id** () const
- bool **incCompletedItems** (unsigned int v=1)
- void **incTotalItems** (unsigned int v=1)
- const QString & **label** () const
- [ProgressItem](#) \* **parent** () const
- unsigned int **progress** () const
- void **removeChild** ([ProgressItem](#) \*const kiddo)
- void **reset** ()

*Reset the progress value of this item to 0 and the status string to the empty string.*

- void **setComplete** ()
- bool **setCompletedItems** (unsigned int v)
- void **setLabel** (const QString &v)
- void **setProgress** (unsigned int v)

*Set the progress (percentage of completion) value of this item.*

- void **setShowAtStart** (bool showAtStart)

*Set the property to pop-up item when it's added in progress manager.*

- void **setStatus** (const QString &v)

*Set the string to be used for showing this item's current status.*

- void **setThumbnail** (const QIcon &icon)

*Sets whether this item has a thumbnail.*

- void **setTotalItems** (unsigned int v)
- void **setUsesBusyIndicator** (bool useBusyIndicator)

*Sets whether this item uses a busy indicator instead of real progress for its progress bar.*

- bool **showAtStart** () const
- const QString & **status** () const
- bool **totalCompleted** () const
- unsigned int **totalItems** () const
- void **updateProgress** ()

*Recalculate progress according to total/completed items and update.*

- bool **usesBusyIndicator** () const

## Additional Inherited Members

### Public Slots inherited from [Digikam::MaintenanceTool](#)

- void **start** ()

### Signals inherited from [Digikam::MaintenanceTool](#)

- void **signalCanceled** ()  
*Emit when process is canceled.*
- void **signalComplete** ()  
*Emit when process is done (not canceled).*

### Signals inherited from [Digikam::ProgressItem](#)

- void **progressItemAdded** ([ProgressItem](#) \*item)  
*Emitted when a new [ProgressItem](#) is added.*
- void **progressItemCanceled** ([ProgressItem](#) \*item)  
*Emitted when an item was canceled.*
- void **progressItemCanceledById** (const QString &id)
- void **progressItemCompleted** ([ProgressItem](#) \*item)  
*Emitted when a progress item was completed.*
- void **progressItemLabel** ([ProgressItem](#) \*item, const QString &label)  
*Emitted when the label of an item changed.*
- void **progressItemProgress** ([ProgressItem](#) \*item, unsigned int v)  
*Emitted when the progress value of an item changes.*
- void **progressItemStatus** ([ProgressItem](#) \*item, const QString &mess)  
*Emitted when the status message of an item changed.*
- void **progressItemThumbnail** ([ProgressItem](#) \*item, const QPixmap &thumb)  
*Emitted when the thumbnail data must be set in item.*
- void **progressItemUsesBusyIndicator** ([ProgressItem](#) \*item, bool value)  
*Emitted when the busy indicator state of an item changes.*

### Protected Slots inherited from [Digikam::MaintenanceTool](#)

- virtual void **slotCancel** ()
- virtual void **slotDone** ()
- virtual void **slotStart** ()

## 9.98.1 Constructor & Destructor Documentation

### 9.98.1.1 AutotagsAssignment()

```
Digikam::AutotagsAssignment::AutotagsAssignment (
    AutoTagsScanSettings::ScanMode mode,
    const AlbumList & list,
    int modelType,
    const QStringList & langs,
    ProgressItem *const parent = nullptr ) [explicit]
```

If list is empty, whole Albums collection is processed.



## 9.98.2 Member Function Documentation

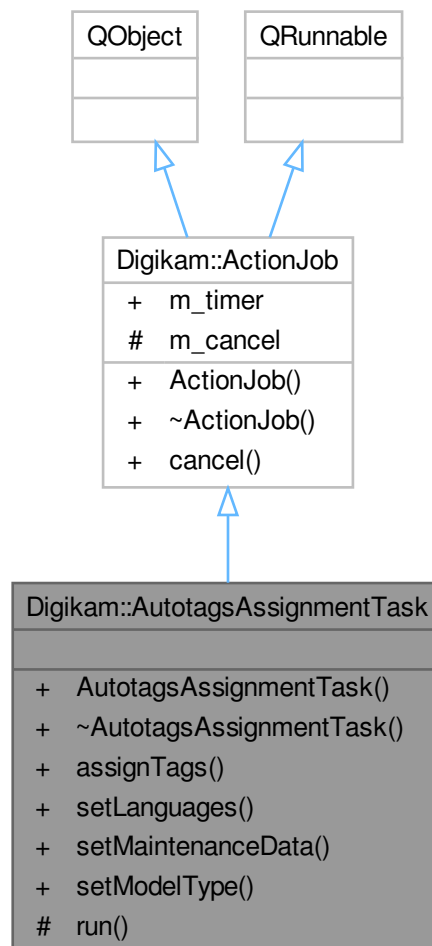
### 9.98.2.1 setUseMultiCoreCPU()

```
void Digikam::AutotagsAssignment::setUseMultiCoreCPU (
    bool ) [override], [virtual]
```

Reimplemented from [Digikam::MaintenanceTool](#).

## 9.99 Digikam::AutotagsAssignmentTask Class Reference

Inheritance diagram for Digikam::AutotagsAssignmentTask:



### Signals

- void **signalFinished** (const [ItemInfo](#) &, const QImage &, const QStringList &)

## Signals inherited from [Digikam::ActionJob](#)

- void **signalDone** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job is done.*
- void **signalProgress** (int)  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job progress.*
- void **signalStarted** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager that job is started.*

## Public Member Functions

- void **assignTags** (const QString &pathImage, const QList< QString > &tagsList)
- void **setLanguages** (const QStringList &langs)
- void **setMaintenanceData** ([MaintenanceData](#) \*const data=nullptr)
- void **setModelType** (int modelType)

## Public Member Functions inherited from [Digikam::ActionJob](#)

- **ActionJob** (QObject \*const parent=nullptr)  
*Constructor which delegate deletion of [QRunnable](#) instance to [ActionThreadBase](#), not [QThreadPool](#).*
- **~ActionJob** () override  
*Re-implement destructor in you implementation.*

## Protected Member Functions

- void **run** () override

## Additional Inherited Members

## Public Slots inherited from [Digikam::ActionJob](#)

- void **cancel** ()  
*Call this method to cancel job.*

## Public Attributes inherited from [Digikam::ActionJob](#)

- QElapsedTimer **m\_timer**  
*Timer to determine the running time of the job.*

## Protected Attributes inherited from [Digikam::ActionJob](#)

- bool **m\_cancel** = false  
*You can use this boolean in your implementation to know if job must be canceled.*

## 9.100 Digikam::AutoTagsScanSettings Class Reference

### Public Types

- enum [DetectorModel](#) { [YOLOV5NANO](#) = 0 , [YOLOV5XLARGE](#) , [RESNET50](#) }  
*Objects detection model.*
- enum [ScanMode](#) { [AllItems](#) = 0 , [NonAssignedItems](#) }  
*Scanning modes.*

### Public Attributes

- AlbumList **albums**  
*Albums to scan.*
- QStringList **langs**
- [ScanMode](#) **mode** = [AllItems](#)
- [DetectorModel](#) **modelType** = [YOLOV5NANO](#)
- bool **wholeAlbums** = false  
*Whole albums checked.*

### 9.100.1 Member Enumeration Documentation

#### 9.100.1.1 DetectorModel

enum [Digikam::AutoTagsScanSettings::DetectorModel](#)

##### Enumerator

YOLOV5NANO	YOLO nano neural network model.
YOLOV5XLARGE	YOLO large neural network model.

#### 9.100.1.2 ScanMode

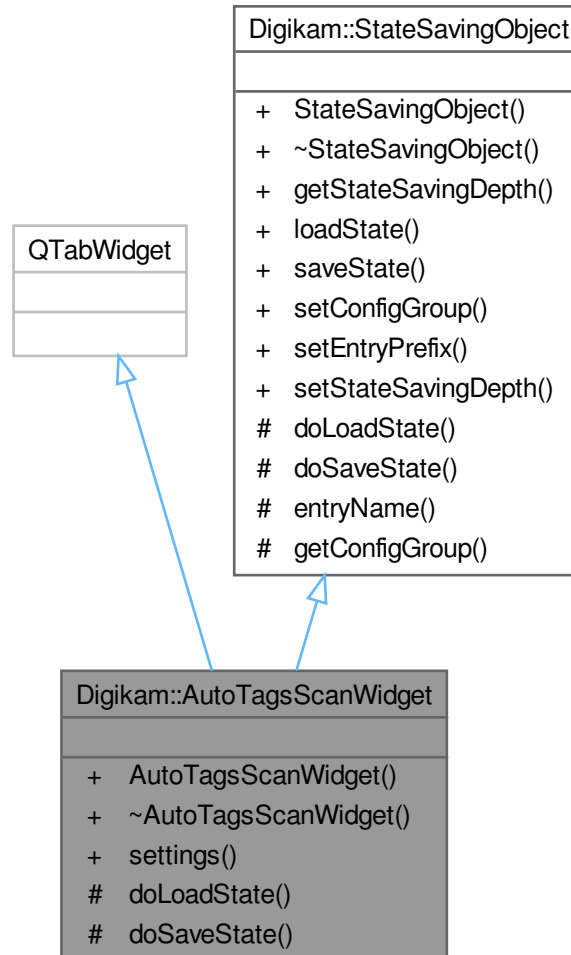
enum [Digikam::AutoTagsScanSettings::ScanMode](#)

##### Enumerator

AllItems	Clean all tags already assigned and re-scan all items.
NonAssignedItems	Scan only items with no tags assigned.

## 9.101 Digikam::AutoTagsScanWidget Class Reference

Inheritance diagram for Digikam::AutoTagsScanWidget:



### Public Member Functions

- **AutoTagsScanWidget** (`QWidget *const parent=nullptr`)
- [AutoTagsScanSettings](#) **settings** () const

### Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (`QObject *const host`)
  - Constructor.*
- virtual `~StateSavingObject ()`
  - Destructor.*
- [StateSavingDepth](#) `getStateSavingDepth` () const

- Returns the depth used for state saving or loading.*

  - void **loadState** ()
    - Invokes loading the class' state.*
  - void **saveState** ()
    - Invokes saving the class' state.*
  - virtual void **setConfigGroup** (const KConfigGroup &group)
    - Sets a dedicated config group that will be used to store and reload the state from.*
  - virtual void **setEntryPrefix** (const QString &prefix)
    - Define a prefix that will be used for every entry in the config group.*
  - void **setStateSavingDepth** (const [StateSavingDepth](#) depth)
    - Sets the depth used for state saving or loading.*

### Protected Member Functions

- void **doLoadState** () override
  - Implement this hook method for state loading.*
- void **doSaveState** () override
  - Implement this hook method for state saving.*

### Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- QString **entryName** (const QString &base) const
  - Always use this method to create config group entry names.*
- KConfigGroup **getConfigGroup** () const
  - Returns the config group that must be used for state saving and loading.*

### Additional Inherited Members

### Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }
  - This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

## 9.101.1 Member Function Documentation

### 9.101.1.1 doLoadState()

```
void Digikam::AutoTagsScanWidget::doLoadState ( ) [override], [protected], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

### 9.101.1.2 doSaveState()

```
void Digikam::AutoTagsScanWidget::doSaveState ( ) [override], [protected], [virtual]
```

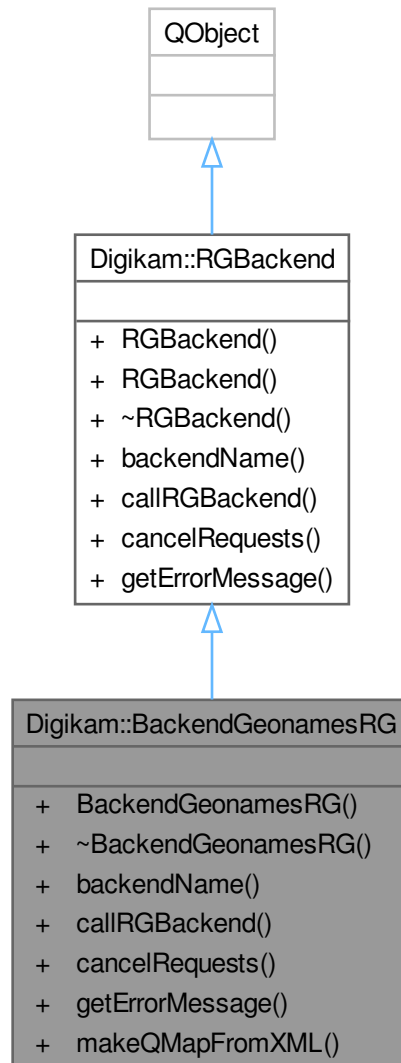
Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

## 9.102 Digikam::BackendGeonamesRG Class Reference

This class calls Geonames' reverse geocoding service.

Inheritance diagram for Digikam::BackendGeonamesRG:



### Public Member Functions

- [BackendGeonamesRG](#) (QObject \*const parent)  
*Constructor.*
- [~BackendGeonamesRG](#) () override  
*Destructor.*
- QString [backendName](#) () override
- void [callRGBackend](#) (const QList< [RGInfo](#) > &rgList, const QString &language) override

*Takes coordinates from each image and then connects to Open Street Map's reverse geocoding service.*

- void [cancelRequests](#) () override
- QString [getErrorMessage](#) () override
- QMap< QString, QString > [makeQMapFromXML](#) (const QString &xmlData)

*The data is returned from Open Street Map in a XML.*

## Public Member Functions inherited from [Digikam::RGBackend](#)

- **RGBackend** (QObject \*const parent)

*Constructor.*

## Additional Inherited Members

## Signals inherited from [Digikam::RGBackend](#)

- void **signalRGReady** (const QList< [RGInfo](#) > &)

*Emitted whenever some items are ready.*

## 9.102.1 Constructor & Destructor Documentation

### 9.102.1.1 BackendGeonamesRG()

```
Digikam::BackendGeonamesRG::BackendGeonamesRG (
    QObject *const parent ) [explicit]
```

#### Parameters

<i>parent</i>	the parent object.
---------------	--------------------

## 9.102.2 Member Function Documentation

### 9.102.2.1 backendName()

```
QString Digikam::BackendGeonamesRG::backendName ( ) [override], [virtual]
```

#### Returns

Backend name.

Reimplemented from [Digikam::RGBackend](#).

### 9.102.2.2 callRGBackend()

```
void Digikam::BackendGeonamesRG::callRGBackend (
    const QList< RGInfo > & rgList,
    const QString & language ) [override], [virtual]
```

## Parameters

<i>rgList</i>	A list containing information needed in reverse geocoding process. At this point, it contains only coordinates.
<i>language</i>	The language in which the data will be returned.

Implements [Digikam::RGBackend](#).

### 9.102.2.3 cancelRequests()

```
void Digikam::BackendGeonamesRG::cancelRequests ( ) [override], [virtual]
```

Implements [Digikam::RGBackend](#).

### 9.102.2.4 getErrorMessage()

```
QString Digikam::BackendGeonamesRG::getErrorMessage ( ) [override], [virtual]
```

## Returns

Error message, if any.

Reimplemented from [Digikam::RGBackend](#).

### 9.102.2.5 makeQMapFromXML()

```
QMap< QString, QString > Digikam::BackendGeonamesRG::makeQMapFromXML (
    const QString & xmlData )
```

This function translates the XML into a QMap.

## Parameters

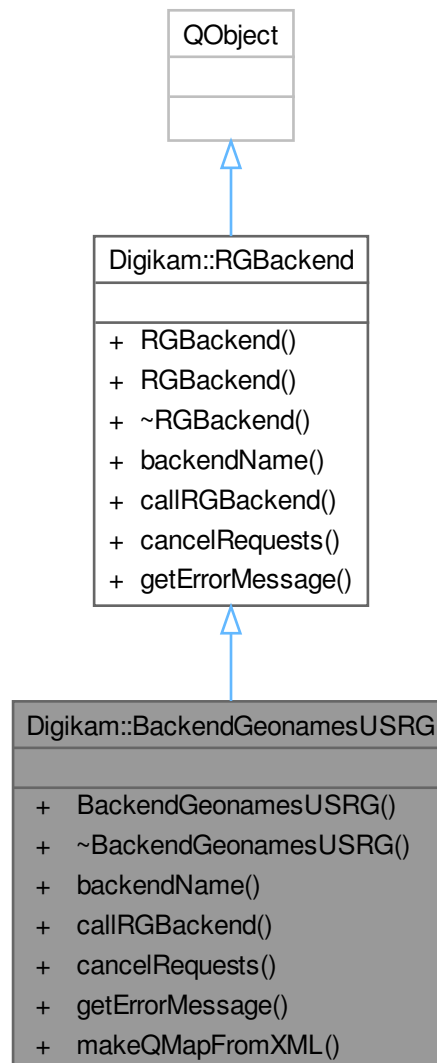
<i>xmlData</i>	The returned XML.
----------------	-------------------

## 9.103 Digikam::BackendGeonamesUSRG Class Reference

This class calls Geonames' get address service available only for USA locations.



Inheritance diagram for Digikam::BackendGeonamesUSRG:



## Public Member Functions

- [BackendGeonamesUSRG](#) (`QObject *const parent`)  
*Constructor.*
- `~BackendGeonamesUSRG ()` override  
*Destructor.*
- `QString backendName ()` override
- `void callIRGBBackend (const QList< RGInfo > &rgList, const QString &language)` override  
*Takes the coordinate of each image and then connects to Open Street Map's reverse geocoding service.*
- `void cancelRequests ()` override
- `QString getErrorMessage ()` override
- `QMap< QString, QString > makeQMapFromXML (const QString &xmlData)`  
*The data is returned from Open Street Map in a XML.*

## Public Member Functions inherited from [Digikam::RGBackend](#)

- **RGBackend** (QObject \*const parent)  
*Constructor.*

### Additional Inherited Members

## Signals inherited from [Digikam::RGBackend](#)

- void **signalRGReady** (const QList< [RGInfo](#) > &)  
*Emitted whenever some items are ready.*

## 9.103.1 Constructor & Destructor Documentation

### 9.103.1.1 BackendGeonamesUSRG()

```
Digikam::BackendGeonamesUSRG::BackendGeonamesUSRG (
    QObject *const parent ) [explicit]
```

#### Parameters

<i>parent</i>	the parent object.
---------------	--------------------

## 9.103.2 Member Function Documentation

### 9.103.2.1 backendName()

```
QString Digikam::BackendGeonamesUSRG::backendName ( ) [override], [virtual]
```

#### Returns

Backend name.

Reimplemented from [Digikam::RGBackend](#).

### 9.103.2.2 callRGBackend()

```
void Digikam::BackendGeonamesUSRG::callRGBackend (
    const QList< RGInfo > & rgList,
    const QString & language ) [override], [virtual]
```

#### Parameters

<i>rgList</i>	A list containing information needed in reverse geocoding process. At this point, it contains only coordinates.
<i>language</i>	The language in which the data will be returned.

Implements [Digikam::RGBBackend](#).

### 9.103.2.3 cancelRequests()

```
void Digikam::BackendGeonamesUSRG::cancelRequests ( ) [override], [virtual]
```

Implements [Digikam::RGBBackend](#).

### 9.103.2.4 getErrorMessage()

```
QString Digikam::BackendGeonamesUSRG::getErrorMessage ( ) [override], [virtual]
```

#### Returns

Error message, if any.

Reimplemented from [Digikam::RGBBackend](#).

### 9.103.2.5 makeQMapFromXML()

```
QMap< QString, QString > Digikam::BackendGeonamesUSRG::makeQMapFromXML (
    const QString & xmlData )
```

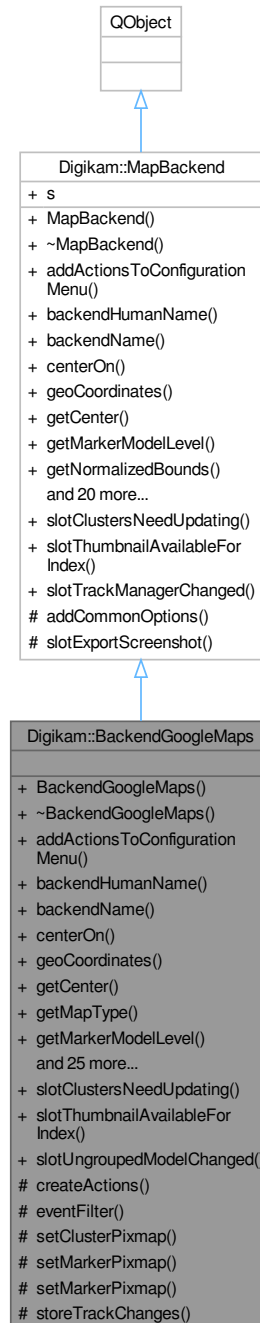
This function translates the XML into a QMap.

#### Parameters

<i>xmlData</i>	The returned XML.
----------------	-------------------

## 9.104 Digikam::BackendGoogleMaps Class Reference

Inheritance diagram for Digikam::BackendGoogleMaps:



### Public Slots

- void **slotClustersNeedUpdating** () override
- void **slotThumbnailAvailableForIndex** (const QVariant &index, const QPixmap &pixmap) override
- void **slotUngroupedModelChanged** (const int mindex)

## Public Slots inherited from [Digikam::MapBackend](#)

- virtual void **slotClustersNeedUpdating** ()=0
- virtual void **slotThumbnailAvailableForIndex** (const QVariant &index, const QPixmap &pixmap)
- virtual void **slotTrackManagerChanged** ()

## Public Member Functions

- **BackendGoogleMaps** (const QExplicitlySharedDataPointer< [GeofaceSharedData](#) > &sharedData, QObject \*const parent=nullptr)
- [~BackendGoogleMaps](#) () override
- void **addActionToConfigurationMenu** (QMenu \*const configurationMenu) override
- QString **backendHumanName** () const override
- QString **backendName** () const override
- void **centerOn** (const Marble::GeoDataLatLonBox &latLonBox, const bool useSaneZoomLevel) override
- bool **geoCoordinates** (const QPoint &point, [GeoCoordinates](#) \*const coordinates) const override
- [GeoCoordinates](#) **getCenter** () const override
- QString **getMapType** () const
- int **getMarkerModelLevel** () override
- [GeoCoordinates::PairList](#) **getNormalizedBounds** () override
- QString **getZoom** () const override
- bool **isReady** () const override
- QSize **mapSize** () const override
- QWidget \* **mapWidget** () override
- void **mapWidgetDocked** (const bool state) override
- void **mouseModeChanged** () override
- void **readSettingsFromGroup** (const KConfigGroup \*const group) override
- void **regionSelectionChanged** () override
- void **releaseWidget** ([GeofaceInternalWidgetInfo](#) \*const info) override
- void **reload** () override
- void **saveSettingsToGroup** (KConfigGroup \*const group) override
- bool **screenCoordinates** (const [GeoCoordinates](#) &coordinates, QPoint \*const point) override
- void **setActive** (const bool state) override
- void **setCenter** (const [GeoCoordinates](#) &coordinate) override
- void **setMapType** (const QString &newMapType)
- void **setShowMapTypeControl** (const bool state)
- void **setShowNavigationControl** (const bool state)
- void **setShowScaleControl** (const bool state)
- void **setZoom** (const QString &newZoom) override
- void **updateActionAvailability** () override
- void **updateClusters** () override
- void **updateMarkers** () override
- void **zoomIn** () override
- void **zoomOut** () override

## Public Member Functions inherited from [Digikam::MapBackend](#)

- **MapBackend** (const QExplicitlySharedDataPointer< [GeofaceSharedData](#) > &sharedData, QObject \*const parent)

### Protected Member Functions

- void **createActions** ()
- bool **eventFilter** (QObject \*object, QEvent \*event) override
- void **setClusterPixmap** (const int clusterId, const QPoint &centerPoint, const QPixmap &clusterPixmap)
- void **setMarkerPixmap** (const int modelId, const int markerId, const QPoint &centerPoint, const QPixmap &markerPixmap)
- void **setMarkerPixmap** (const int modelId, const int markerId, const QPoint &centerPoint, const QSize &iconSize, const QUrl &iconUrl)
- void **storeTrackChanges** (const TrackManager::TrackChanges trackChanges)

### Protected Member Functions inherited from [Digikam::MapBackend](#)

- void **addCommonOptions** (QMenu \*const configurationMenu)

### Additional Inherited Members

### Signals inherited from [Digikam::MapBackend](#)

- void **signalBackendReadyChanged** (const QString &backendName)
- void **signalClustersClicked** (const QList &clusterIndices)
- void **signalClustersMoved** (const QList &clusterIndices, const QPair< int, QModelIndex > &snapTarget)
- void **signalMarkersMoved** (const QList &markerIndices)
- void **signalSelectionHasBeenMade** (const Digikam::GeoCoordinates::Pair &coordinates)
- void **signalZoomChanged** (const QString &newZoom)

### Public Attributes inherited from [Digikam::MapBackend](#)

- const QExplicitlySharedDataPointer< [GeofaceSharedData](#) > **s**

### Protected Slots inherited from [Digikam::MapBackend](#)

- void **slotExportScreenshot** ()

## 9.104.1 Constructor & Destructor Documentation

### 9.104.1.1 ~BackendGoogleMaps()

```
Digikam::BackendGoogleMaps::~BackendGoogleMaps ( ) [override]
```

## 9.104.2 Member Function Documentation

### 9.104.2.1 addActionstoConfigurationMenu()

```
void Digikam::BackendGoogleMaps::addActionstoConfigurationMenu (
    QMenu *const configurationMenu ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

### 9.104.2.2 backendHumanName()

```
QString Digikam::BackendGoogleMaps::backendHumanName ( ) const [override], [virtual]
```

Implements [Digikam::MapBackend](#).

### 9.104.2.3 backendName()

```
QString Digikam::BackendGoogleMaps::backendName ( ) const [override], [virtual]
```

Implements [Digikam::MapBackend](#).

### 9.104.2.4 centerOn()

```
void Digikam::BackendGoogleMaps::centerOn (
    const Marble::GeoDataLatLonBox & latLonBox,
    const bool useSaneZoomLevel ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

### 9.104.2.5 geoCoordinates()

```
bool Digikam::BackendGoogleMaps::geoCoordinates (
    const QPoint & point,
    GeoCoordinates *const coordinates ) const [override], [virtual]
```

Implements [Digikam::MapBackend](#).

### 9.104.2.6 getCenter()

```
GeoCoordinates Digikam::BackendGoogleMaps::getCenter ( ) const [override], [virtual]
```

Implements [Digikam::MapBackend](#).

### 9.104.2.7 getMarkerModelLevel()

```
int Digikam::BackendGoogleMaps::getMarkerModelLevel ( ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

### 9.104.2.8 getNormalizedBounds()

```
GeoCoordinates::PairList Digikam::BackendGoogleMaps::getNormalizedBounds ( ) [override],
[virtual]
```

Implements [Digikam::MapBackend](#).

### 9.104.2.9 getZoom()

```
QString Digikam::BackendGoogleMaps::getZoom ( ) const [override], [virtual]
```

Implements [Digikam::MapBackend](#).

### 9.104.2.10 isReady()

```
bool Digikam::BackendGoogleMaps::isReady ( ) const [override], [virtual]
```

Implements [Digikam::MapBackend](#).

### 9.104.2.11 mapSize()

```
QSize Digikam::BackendGoogleMaps::mapSize ( ) const [override], [virtual]
```

Implements [Digikam::MapBackend](#).

### 9.104.2.12 mapWidget()

```
QWidget * Digikam::BackendGoogleMaps::mapWidget ( ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

### 9.104.2.13 mapWidgetDocked()

```
void Digikam::BackendGoogleMaps::mapWidgetDocked (
    const bool state ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

### 9.104.2.14 mouseModeChanged()

```
void Digikam::BackendGoogleMaps::mouseModeChanged ( ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

### 9.104.2.15 readSettingsFromGroup()

```
void Digikam::BackendGoogleMaps::readSettingsFromGroup (
    const KConfigGroup *const group ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).



**9.104.2.16 regionSelectionChanged()**

```
void Digikam::BackendGoogleMaps::regionSelectionChanged ( ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

**9.104.2.17 releaseWidget()**

```
void Digikam::BackendGoogleMaps::releaseWidget (
    GeoInterfaceInternalWidgetInfo *const info ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

**9.104.2.18 reload()**

```
void Digikam::BackendGoogleMaps::reload ( ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

**9.104.2.19 saveSettingsToGroup()**

```
void Digikam::BackendGoogleMaps::saveSettingsToGroup (
    KConfigGroup *const group ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

**9.104.2.20 screenCoordinates()**

```
bool Digikam::BackendGoogleMaps::screenCoordinates (
    const GeoCoordinates & coordinates,
    QPoint *const point ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

**9.104.2.21 setActive()**

```
void Digikam::BackendGoogleMaps::setActive (
    const bool state ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

**9.104.2.22 setCenter()**

```
void Digikam::BackendGoogleMaps::setCenter (
    const GeoCoordinates & coordinate ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

#### 9.104.2.23 setMarkerPixmap()

```
void Digikam::BackendGoogleMaps::setMarkerPixmap (
    const int modelId,
    const int markerId,
    const QPoint & centerPoint,
    const QSize & iconSize,
    const QUrl & iconUrl ) [protected]
```

#### 9.104.2.24 setZoom()

```
void Digikam::BackendGoogleMaps::setZoom (
    const QString & newZoom ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

#### 9.104.2.25 updateActionAvailability()

```
void Digikam::BackendGoogleMaps::updateActionAvailability ( ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

#### 9.104.2.26 updateClusters()

```
void Digikam::BackendGoogleMaps::updateClusters ( ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

#### 9.104.2.27 updateMarkers()

```
void Digikam::BackendGoogleMaps::updateMarkers ( ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

#### 9.104.2.28 zoomIn()

```
void Digikam::BackendGoogleMaps::zoomIn ( ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

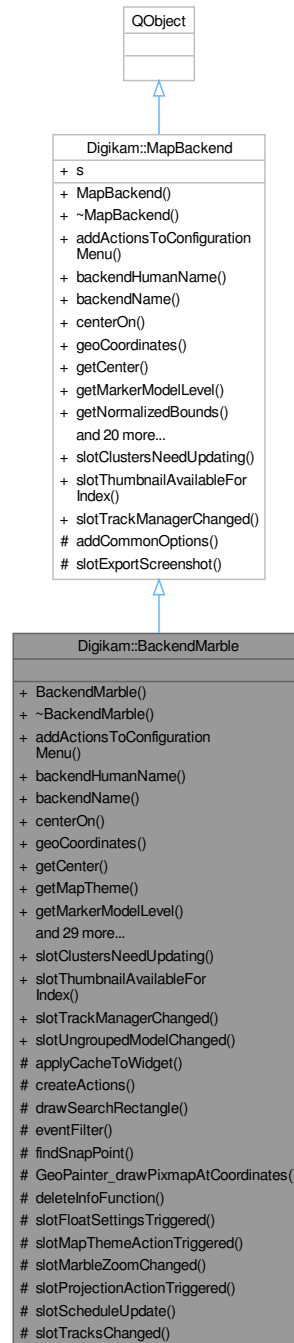
#### 9.104.2.29 zoomOut()

```
void Digikam::BackendGoogleMaps::zoomOut ( ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

## 9.105 Digikam::BackendMarble Class Reference

Inheritance diagram for Digikam::BackendMarble:



### Public Slots

- void **slotClustersNeedUpdating** () override
- void **slotThumbnailAvailableForIndex** (const QVariant &index, const QPixmap &pixmap) override
- void **slotTrackManagerChanged** () override
- void **slotUngroupedModelChanged** (const int index)

## Public Slots inherited from [Digikam::MapBackend](#)

- virtual void **slotClustersNeedUpdating** ()=0
- virtual void **slotThumbnailAvailableForIndex** (const QVariant &index, const QPixmap &pixmap)
- virtual void **slotTrackManagerChanged** ()

## Public Member Functions

- **BackendMarble** (const QExplicitlySharedDataPointer< [GeofaceSharedData](#) > &sharedData, QObject \*const parent=nullptr)
- [~BackendMarble](#) () override
- void [addActionToConfigurationMenu](#) (QMenu \*const configurationMenu) override
- QString [backendHumanName](#) () const override
- QString [backendName](#) () const override
- void [centerOn](#) (const Marble::GeoDataLatLonBox &box, const bool useSaneZoomLevel) override
- bool [geoCoordinates](#) (const QPoint &point, [GeoCoordinates](#) \*const coordinates) const override
- [GeoCoordinates](#) [getCenter](#) () const override
- QString [getMapTheme](#) () const
- int [getMarkerModelLevel](#) () override
- GeoCoordinates::PairList [getNormalizedBounds](#) () override
- QString [getProjection](#) () const
- QString [getZoom](#) () const override
- bool [isReady](#) () const override
- QSize [mapSize](#) () const override
- QWidget \* [mapWidget](#) () override
- void [mapWidgetDocked](#) (const bool state) override
- void [marbleCustomPaint](#) (Marble::GeoPainter \*painter)
- void [mouseModeChanged](#) () override
- void [readSettingsFromGroup](#) (const KConfigGroup \*const group) override
- void [regionSelectionChanged](#) () override
- void [releaseWidget](#) ([GeofaceInternalWidgetInfo](#) \*const info) override
- void [reload](#) () override
- void [saveSettingsToGroup](#) (KConfigGroup \*const group) override
- bool [screenCoordinates](#) (const [GeoCoordinates](#) &coordinates, QPoint \*const point) override
- void [setActive](#) (const bool state) override
- void [setCenter](#) (const [GeoCoordinates](#) &coordinate) override
- void [setMapTheme](#) (const QString &newMapTheme)
- void [setProjection](#) (const QString &newProjection)
- void [setShowCompass](#) (const bool state)
- void [setShowNavigation](#) (const bool state)
- void [setShowOverviewMap](#) (const bool state)
- void [setShowScaleBar](#) (const bool state)
- void [setZoom](#) (const QString &newZoom) override
- void [updateActionAvailability](#) () override
- void [updateClusters](#) () override
- void [updateMarkers](#) () override
- void [zoomIn](#) () override
- void [zoomOut](#) () override

## Public Member Functions inherited from [Digikam::MapBackend](#)

- **MapBackend** (const QExplicitlySharedDataPointer< [GeofaceSharedData](#) > &sharedData, QObject \*const parent)

### Protected Slots

- void **slotFloatSettingsTriggered** (QAction \*action)
- void **slotMapThemeActionTriggered** (QAction \*action)
- void **slotMarbleZoomChanged** ()
- void **slotProjectionActionTriggered** (QAction \*action)
- void **slotScheduleUpdate** ()
- void **slotTracksChanged** (const QList< TrackManager::TrackChanges > &trackChanges)

### Protected Slots inherited from [Digikam::MapBackend](#)

- void **slotExportScreenshot** ()

### Protected Member Functions

- void **applyCacheToWidget** ()
- void **createActions** ()
- void **drawSearchRectangle** (Marble::GeoPainter \*const painter, const GeoCoordinates::Pair &searchRectangle, const bool isOldRectangle)
- bool **eventFilter** (QObject \*object, QEvent \*event) override
- bool **findSnapPoint** (const QPoint &actualPoint, QPoint \*const snapPoint, [GeoCoordinates](#) \*const snapCoordinates, QPair< int, QModelIndex > \*const snapTargetIndex)
- void **GeoPainter\_drawPixmapAtCoordinates** (Marble::GeoPainter \*const painter, const QPixmap &pixmap, const [GeoCoordinates](#) &coordinates, const QPoint &basePoint)

*Replacement for Marble::GeoPainter::drawPixmap which takes a pixel offset.*

### Protected Member Functions inherited from [Digikam::MapBackend](#)

- void **addCommonOptions** (QMenu \*const configurationMenu)

### Static Protected Member Functions

- static void **deleteInfoFunction** ([GeofaceInternalWidgetInfo](#) \*const info)

### Additional Inherited Members

### Signals inherited from [Digikam::MapBackend](#)

- void **signalBackendReadyChanged** (const QString &backendName)
- void **signalClustersClicked** (const QList &clusterIndices)
- void **signalClustersMoved** (const QList &clusterIndices, const QPair< int, QModelIndex > &snapTarget)
- void **signalMarkersMoved** (const QList &markerIndices)
- void **signalSelectionHasBeenMade** (const Digikam::GeoCoordinates::Pair &coordinates)
- void **signalZoomChanged** (const QString &newZoom)

### Public Attributes inherited from [Digikam::MapBackend](#)

- const QExplicitlySharedDataPointer< [GeofaceSharedData](#) > **s**

## 9.105.1 Constructor & Destructor Documentation

### 9.105.1.1 ~BackendMarble()

Digikam::BackendMarble::~~BackendMarble ( ) [override]

## 9.105.2 Member Function Documentation

### 9.105.2.1 addActionstoConfigurationMenu()

```
void Digikam::BackendMarble::addActionstoConfigurationMenu (
    QMenu *const configurationMenu ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

### 9.105.2.2 applyCacheToWidget()

```
void Digikam::BackendMarble::applyCacheToWidget ( ) [protected]
```

### 9.105.2.3 backendHumanName()

```
QString Digikam::BackendMarble::backendHumanName ( ) const [override], [virtual]
```

Implements [Digikam::MapBackend](#).

### 9.105.2.4 backendName()

```
QString Digikam::BackendMarble::backendName ( ) const [override], [virtual]
```

Implements [Digikam::MapBackend](#).

### 9.105.2.5 centerOn()

```
void Digikam::BackendMarble::centerOn (
    const Marble::GeoDataLatLonBox & box,
    const bool useSaneZoomLevel ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

### 9.105.2.6 eventFilter()

```
bool Digikam::BackendMarble::eventFilter (
    QObject * object,
    QEvent * event ) [override], [protected]
```

**9.105.2.7 geoCoordinates()**

```
bool Digikam::BackendMarble::geoCoordinates (
    const QPoint & point,
    GeoCoordinates *const coordinates ) const [override], [virtual]
```

Implements [Digikam::MapBackend](#).

**9.105.2.8 GeoPainter\_drawPixmapAtCoordinates()**

```
void Digikam::BackendMarble::GeoPainter_drawPixmapAtCoordinates (
    Marble::GeoPainter *const painter,
    const QPixmap & pixmap,
    const GeoCoordinates & coordinates,
    const QPoint & offsetPoint ) [protected]
```

**Parameters**

<i>painter</i>	Marble::GeoPainter on which to draw the pixmap
<i>pixmap</i>	Pixmap to be drawn
<i>coordinates</i>	<a href="#">GeoCoordinates</a> where the image is to be drawn
<i>offsetPoint</i>	Point in the <code>pixmap</code> which should be at <code>coordinates</code>

**9.105.2.9 getCenter()**

```
GeoCoordinates Digikam::BackendMarble::getCenter ( ) const [override], [virtual]
```

Implements [Digikam::MapBackend](#).

**9.105.2.10 getMarkerModelLevel()**

```
int Digikam::BackendMarble::getMarkerModelLevel ( ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

**9.105.2.11 getNormalizedBounds()**

```
GeoCoordinates::PairList Digikam::BackendMarble::getNormalizedBounds ( ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

**9.105.2.12 getProjection()**

```
QString Digikam::BackendMarble::getProjection ( ) const
```

**9.105.2.13 getZoom()**

```
QString Digikam::BackendMarble::getZoom ( ) const [override], [virtual]
```

Implements [Digikam::MapBackend](#).

**9.105.2.14 isReady()**

```
bool Digikam::BackendMarble::isReady ( ) const [override], [virtual]
```

Implements [Digikam::MapBackend](#).

**9.105.2.15 mapSize()**

```
QSize Digikam::BackendMarble::mapSize ( ) const [override], [virtual]
```

Implements [Digikam::MapBackend](#).

**9.105.2.16 mapWidget()**

```
QWidget * Digikam::BackendMarble::mapWidget ( ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

**9.105.2.17 mapWidgetDocked()**

```
void Digikam::BackendMarble::mapWidgetDocked (
    const bool state ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

**9.105.2.18 marbleCustomPaint()**

```
void Digikam::BackendMarble::marbleCustomPaint (
    Marble::GeoPainter * painter )
```

**9.105.2.19 mouseModeChanged()**

```
void Digikam::BackendMarble::mouseModeChanged ( ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

**9.105.2.20 readSettingsFromGroup()**

```
void Digikam::BackendMarble::readSettingsFromGroup (
    const KConfigGroup *const group ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).



**9.105.2.21 regionSelectionChanged()**

```
void Digikam::BackendMarble::regionSelectionChanged ( ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

**9.105.2.22 releaseWidget()**

```
void Digikam::BackendMarble::releaseWidget (
    GeoInterfaceInternalWidgetInfo *const info ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

**9.105.2.23 reload()**

```
void Digikam::BackendMarble::reload ( ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

**9.105.2.24 saveSettingsToGroup()**

```
void Digikam::BackendMarble::saveSettingsToGroup (
    KConfigGroup *const group ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

**9.105.2.25 screenCoordinates()**

```
bool Digikam::BackendMarble::screenCoordinates (
    const GeoCoordinates & coordinates,
    QPoint *const point ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

**9.105.2.26 setActive()**

```
void Digikam::BackendMarble::setActive (
    const bool state ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

**9.105.2.27 setCenter()**

```
void Digikam::BackendMarble::setCenter (
    const GeoCoordinates & coordinate ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

**9.105.2.28 setZoom()**

```
void Digikam::BackendMarble::setZoom (
    const QString & newZoom ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

**9.105.2.29 slotScheduleUpdate**

```
void Digikam::BackendMarble::slotScheduleUpdate ( ) [protected], [slot]
```

**9.105.2.30 updateActionAvailability()**

```
void Digikam::BackendMarble::updateActionAvailability ( ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

**9.105.2.31 updateClusters()**

```
void Digikam::BackendMarble::updateClusters ( ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

**9.105.2.32 updateMarkers()**

```
void Digikam::BackendMarble::updateMarkers ( ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

**9.105.2.33 zoomIn()**

```
void Digikam::BackendMarble::zoomIn ( ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

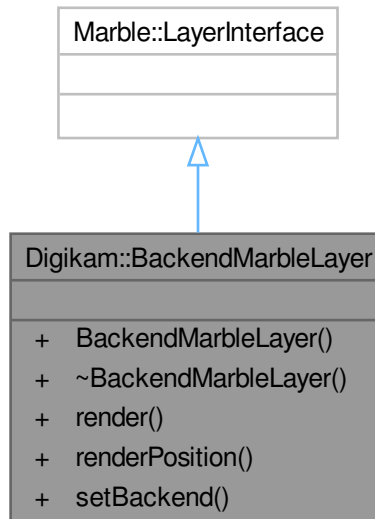
**9.105.2.34 zoomOut()**

```
void Digikam::BackendMarble::zoomOut ( ) [override], [virtual]
```

Implements [Digikam::MapBackend](#).

## 9.106 Digikam::BackendMarbleLayer Class Reference

Inheritance diagram for Digikam::BackendMarbleLayer:



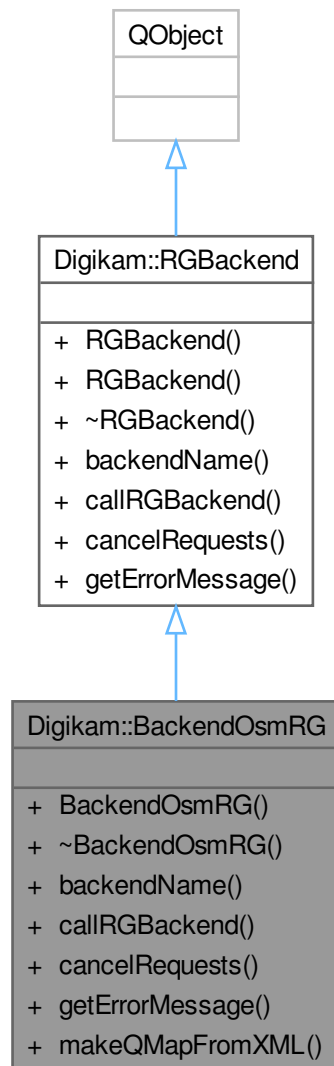
### Public Member Functions

- **BackendMarbleLayer** ([BackendMarble](#) \*const pMarbleBackend)
- bool **render** ([Marble::GeoPainter](#) \*painter, [Marble::ViewportParams](#) \*viewport, const [QString](#) &render←Pos=[QLatin1String](#)("NONE"), [Marble::GeoSceneLayer](#) \*layer=nullptr) override
- [QStringList](#) **renderPosition** () const override
- void **setBackend** ([BackendMarble](#) \*const pMarbleBackend)

## 9.107 Digikam::BackendOsmRG Class Reference

This class calls Open Street Map's reverse geocoding service.

Inheritance diagram for Digikam::BackendOsmRG:



## Public Member Functions

- [BackendOsmRG](#) (QObject \*const parent)  
*Constructor.*
- [~BackendOsmRG](#) () override  
*Destructor.*
- QString [backendName](#) () override
- void [callIRGBBackend](#) (const QList< [RGInfo](#) > &rgList, const QString &language) override  
*Takes the coordinate of each image and then connects to Open Street Map's reverse geocoding service.*
- void [cancelRequests](#) () override
- QString [getErrorMessage](#) () override
- QMap< QString, QString > [makeQMapFromXML](#) (const QString &xmlData)  
*The data is returned from Open Street Map in a XML.*

## Public Member Functions inherited from [Digikam::RGBackend](#)

- **RGBackend** (QObject \*const parent)

*Constructor.*

### Additional Inherited Members

## Signals inherited from [Digikam::RGBackend](#)

- void **signalRGReady** (const QList< [RGInfo](#) > &)

*Emitted whenever some items are ready.*

## 9.107.1 Constructor & Destructor Documentation

### 9.107.1.1 BackendOsmRG()

```
Digikam::BackendOsmRG::BackendOsmRG (
    QObject *const parent ) [explicit]
```

#### Parameters

<i>parent</i>	the parent object.
---------------	--------------------

## 9.107.2 Member Function Documentation

### 9.107.2.1 backendName()

```
QString Digikam::BackendOsmRG::backendName ( ) [override], [virtual]
```

#### Returns

Backend name.

Reimplemented from [Digikam::RGBackend](#).

### 9.107.2.2 callRGBackend()

```
void Digikam::BackendOsmRG::callRGBackend (
    const QList< RGInfo > & rgList,
    const QString & language ) [override], [virtual]
```

#### Parameters

<i>rgList</i>	A list containing information needed in reverse geocoding process. At this point, it contains only coordinates.
<i>language</i>	The language in which the data will be returned.

Implements [Digikam::RGBBackend](#).

### 9.107.2.3 cancelRequests()

```
void Digikam::BackendOsmRG::cancelRequests ( ) [override], [virtual]
```

Implements [Digikam::RGBBackend](#).

### 9.107.2.4 getErrorMessage()

```
QString Digikam::BackendOsmRG::getErrorMessage ( ) [override], [virtual]
```

#### Returns

Error message, if any.

Reimplemented from [Digikam::RGBBackend](#).

### 9.107.2.5 makeQMapFromXML()

```
QMap< QString, QString > Digikam::BackendOsmRG::makeQMapFromXML (
    const QString & xmlData )
```

This function translates the XML into a QMap.

#### Parameters

<i>xmlData</i>	The returned XML.
----------------	-------------------

## 9.108 Digikam::BalooInfo Class Reference

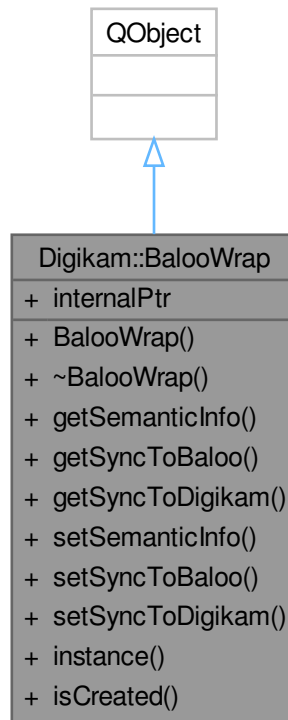
### Public Attributes

- QString **comment**
- int **rating** = -1
- QStringList **tags**

## 9.109 Digikam::BalooWrap Class Reference

The [BalooWrap](#) class is a singleton class which offer functionality for reading and writing image comment, tags and rating from Baloo to digiKam and from digiKam to Baloo.

Inheritance diagram for Digikam::BalooWrap:



### Public Member Functions

- `BalooInfo getSemanticInfo (const QUrl &url) const`  
*getSemanticInfo - used by `ItemScanner` to retrieve all information tags, comment, rating*
- `bool getSyncToBaloo () const`
- `bool getSyncToDigikam () const`
- `void setSemanticInfo (const QUrl &url, const BalooInfo &blInfo)`  
*setSemanticInfo - generic method to set all data from digiKam to Baloo*
- `void setSyncToBaloo (bool value)`
- `void setSyncToDigikam (bool value)`

### Static Public Member Functions

- `static BalooWrap * instance ()`
- `static bool isCreated ()`

### Static Public Attributes

- `static QPointer< BalooWrap > internalPtr = QPointer<BalooWrap>()`  
*internalPtr - singleton implementation*

### 9.109.1 Detailed Description

The singleton functionality is required because it also watches for changes in Baloo and notify digiKam, so it could trigger a scan

### 9.109.2 Member Function Documentation

#### 9.109.2.1 getSemanticInfo()

```
BalooInfo Digikam::BalooWrap::getSemanticInfo (
    const QUrl & url ) const
```

##### Parameters

<i>url</i>	- image url
------------	-------------

##### Returns

- container class for tags, comment, rating

#### 9.109.2.2 setSemanticInfo()

```
void Digikam::BalooWrap::setSemanticInfo (
    const QUrl & url,
    const BalooInfo & bInfo )
```

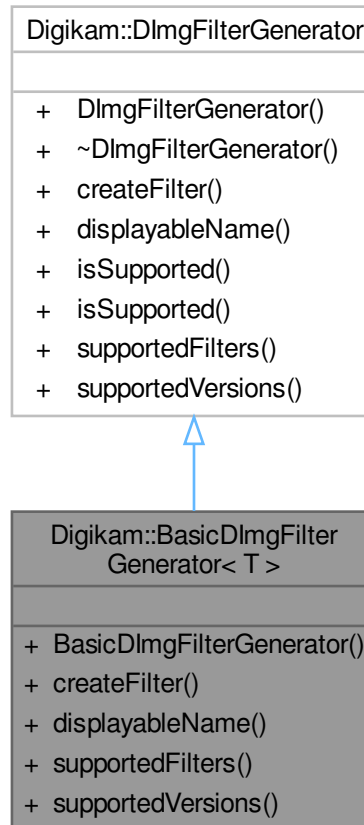
##### Parameters

<i>url</i>	- image url
<i>bInfo</i>	- container class for tags, comment, rating



## 9.110 Digikam::BasicDImgFilterGenerator< T > Class Template Reference

Inheritance diagram for Digikam::BasicDImgFilterGenerator< T >:



### Public Member Functions

- `BasicDImgFilterGenerator ()=default`  
*A sample implementation for one [DImgThreadedFilter](#) class.*
- `DImgThreadedFilter * createFilter (const QString &filterIdentifier, int version) override`  
*Create the filter for the given combination of identifier and version.*
- `QString displayName (const QString &filterIdentifier) override`  
*Returns a QString with filter name for displaying in views.*
- `QStringList supportedFilters () override`  
*Returns a list with identifiers of supported filters.*
- `QList< int > supportedVersions (const QString &filterIdentifier) override`  
*Returns a list with the supported versions for the given identifier.*

## Public Member Functions inherited from [Digikam::DImgFilterGenerator](#)

- virtual bool [isSupported](#) (const QString &filterIdentifier)  
*Convenience methods.*
- virtual bool [isSupported](#) (const QString &filterIdentifier, int version)

### 9.110.1 Constructor & Destructor Documentation

#### 9.110.1.1 BasicDImgFilterGenerator()

```
template<class T >
Digikam::BasicDImgFilterGenerator< T >::BasicDImgFilterGenerator ( ) [default]
```

The class must provide two static methods, [FilterIdentifier\(\)](#) and [SupportedVersions\(\)](#).

### 9.110.2 Member Function Documentation

#### 9.110.2.1 createFilter()

```
template<class T >
DImgThreadedFilter * Digikam::BasicDImgFilterGenerator< T >::createFilter (
    const QString & filterIdentifier,
    int version ) [inline], [override], [virtual]
```

Implements [Digikam::DImgFilterGenerator](#).

#### 9.110.2.2 displayableName()

```
template<class T >
QString Digikam::BasicDImgFilterGenerator< T >::displayableName (
    const QString & filterIdentifier ) [inline], [override], [virtual]
```

Implements [Digikam::DImgFilterGenerator](#).

#### 9.110.2.3 supportedFilters()

```
template<class T >
QStringList Digikam::BasicDImgFilterGenerator< T >::supportedFilters ( ) [inline], [override],
[virtual]
```

Implements [Digikam::DImgFilterGenerator](#).

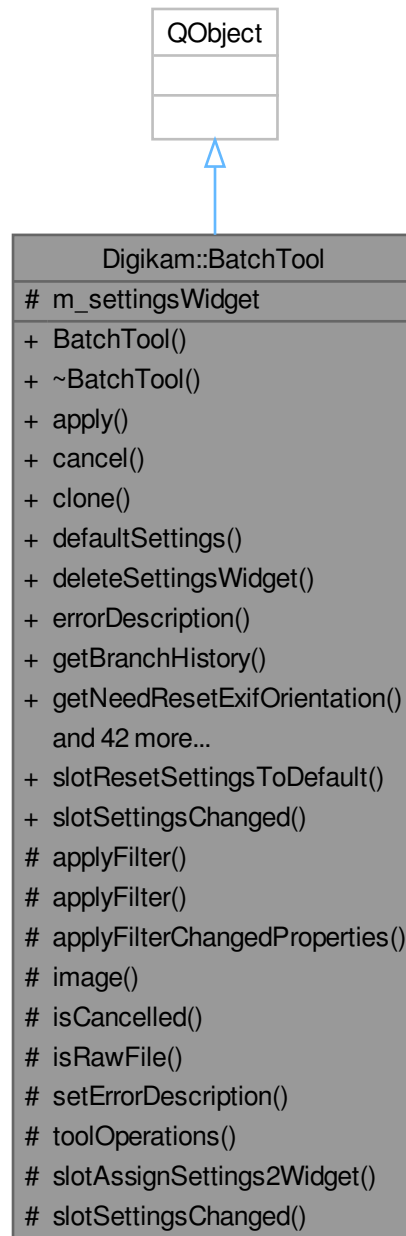
#### 9.110.2.4 supportedVersions()

```
template<class T >
QList< int > Digikam::BasicDImgFilterGenerator< T >::supportedVersions (
    const QString & filterIdentifier ) [inline], [override], [virtual]
```

Implements [Digikam::DImgFilterGenerator](#).

## 9.111 Digikam::BatchTool Class Reference

Inheritance diagram for Digikam::BatchTool:



### Public Types

- enum `BatchToolGroup` {  
`BaseTool = 0`, `CustomTool`, `ColorTool`, `EnhanceTool`,  
`TransformTool`, `DecorateTool`, `FiltersTool`, `ConvertTool`,  
`MetadataTool` }

## Public Slots

- void **slotResetSettingsToDefault** ()
- void **slotSettingsChanged** (const [BatchToolSettings](#) &settings)

## Signals

- void **signalAssignSettings2Widget** ()  
*Only used internally.*
- void **signalSettingsChanged** (const [BatchToolSettings](#) &)
- void **signalVisible** (bool)

## Public Member Functions

- [BatchTool](#) (const QString &name, [BatchToolGroup](#) group, QObject \*const parent=nullptr)  
*Tool data and properties management.*
- bool **apply** ()  
*Apply all change to perform by this tool.*
- virtual void **cancel** ()  
*Re-implement this method is you want customize cancellation of tool, for ex.*
- virtual [BatchTool](#) \* **clone** (QObject \*const parent=nullptr) const =0  
*Clone this tool without to create settings widget.*
- virtual [BatchToolSettings](#) **defaultSettings** ()=0  
*Re-implement this method to initialize Settings Widget value with default settings.*
- void **deleteSettingsWidget** ()  
*Delete dedicated settings widget registered with [registerSettingsWidget\(\)](#).*
- QString **errorDescription** () const  
*Get description of an error which appear during [apply\(\)](#) method.*
- bool **getBranchHistory** () const
- bool **getNeedResetExifOrientation** () const  
*Returns true if the Exif orientation tag should be reset after tool operation.*
- bool **getResetExifOrientationAllowed** () const  
*Returns true if the Exif orientation tag is allowed to be reset after tool operation.*
- [DImg](#) **imageData** () const
- [ItemInfo](#) **imageInfo** () const
- QUrl **inputUrl** () const
- [IOFileSettings](#) **ioFileSettings** () const  
*Return IOFile settings used during tool operations.*
- bool **isLastChainedTool** () const
- bool **loadToDImg** () const  
*Load image data using input Url set by [setInputUrl\(\)](#) to instance of internal [DImg](#) container.*
- virtual QString **outputSuffix** () const  
*Re-implement this method if tool change file extension during batch process (ex: "png").*
- QUrl **outputUrl** () const
- [DPluginBqm](#) \* **plugin** () const
- [DRawDecoderSettings](#) **rawDecodingSettings** () const  
*Return RAW decoding settings used during tool operations.*
- virtual void **registerSettingsWidget** ()  
*Setup dedicated settings widget.*
- bool **savefromDImg** () const  
*Save image data from instance of internal [DImg](#) container using :*

- void **setBranchHistory** (bool branch=true)
 

*Applies only when the file is actually saved on disk, and takes the history since the loading from disk to set the first added step as creating a branch.*
- void **setDRawDecoderSettings** (const [DRawDecoderSettings](#) &settings)
 

*Set-up RAW decoding settings no use during tool operations.*
- void **setImageData** (const [DImg](#) &img)
 

*Manage instance of current image data container loaded by this tool.*
- void **setInputUrl** (const [QUrl](#) &inputUrl)
 

*Manage current input url processed by this tool.*
- void **setIOFileSettings** (const [IOFileSettings](#) &settings)
 

*Set-up IOFile settings no use during tool operations.*
- void **setItemInfo** (const [ItemInfo](#) &info)
 

*Manage instance of current image info loaded by this tool.*
- void **setLastChainedTool** (bool last)
 

*Manage flag properties to indicate if this tool is last one to process on current item.*
- void **setNeedResetExifOrientation** (bool reset)
 

*Set that the Exif orientation flag should be reset to NORMAL after tool operation.*
- void **setOutputUrl** (const [QUrl](#) &outputUrl)
 

*Manage current output url processed by this tool.*
- void **setOutputUrlFromInputUrl** ()
 

*Set output url using input url content + annotation based on time stamp + file extension defined by [outputSuffix\(\)](#).*
- void **setPlugin** ([DPluginBqm](#) \*const plugin)
- void **setRawLoadingRules** ([QueueSettings::RawLoadingRule](#) rule)
 

*Set that RAW files loading rule to use (demosaicing or JPEG embedded).*
- void **setResetExifOrientationAllowed** (bool reset)
 

*Set that the Exif orientation flag is allowed be reset to NORMAL after tool operation.*
- void **setSaveAsNewVersion** (bool fork=true)
 

*Sets if the history added by tools shall be made a branch (new version).*
- void **setSettings** (const [BatchToolSettings](#) &settings)
 

*Manage settings values to tool.*
- [BatchToolSettings](#) **settings** () const
- [QWidget](#) \* **settingsWidget** () const
 

*Settings widget management. NOTE: do not use these methods in multi-threading part ([ActionThread](#)), only in main thread (GUI)*
- void **setToolDescription** (const [QString](#) &toolDescription)
 

*Manage Tool description.*
- void **setToolIcon** (const [QIcon](#) &icon)
- void **setToolIconName** (const [QString](#) &iconName)
 

*Manage Tool icon name.*
- void **setToolTitle** (const [QString](#) &toolTitle)
 

*Manage Tool title.*
- void **setWorkingUrl** (const [QUrl](#) &workingUrl)
 

*Manage current working url used by this tool to process items.*
- [QString](#) **toolDescription** () const
- [BatchToolGroup](#) **toolGroup** () const
 

*Return group of tool.*
- [QString](#) **toolGroupToString** () const
 

*Return group of tool name as string.*
- [QIcon](#) **toolIcon** () const
- [QString](#) **toolTitle** () const
- virtual int **toolVersion** () const
 

*Return version of tool.*
- [QUrl](#) **workingUrl** () const

## Protected Slots

- virtual void `slotAssignSettings2Widget ()=0`  
*Re-implement this method to customize how all settings values must be assigned to settings widget.*
- virtual void `slotSettingsChanged ()=0`

## Protected Member Functions

- void `applyFilter (DImgBuiltinFilter *const filter)`
- void `applyFilter (DImgThreadedFilter *const filter)`  
*Use this if you have a filter ready to run.*
- void `applyFilterChangedProperties (DImgThreadedFilter *const filter)`
- `DImg & image () const`  
*Return a reference of internal `DImg` container used to modify image data.*
- bool `isCancelled () const`  
*Return true if `cancel()` have been called.*
- bool `isRawFile (const QUrl &url) const`  
*Method to check if file pointed by url is a RAW image.*
- void `setErrorDescription (const QString &errmsg)`  
*Set string to describe an error which appear during `apply()` method.*
- virtual bool `toolOperations ()=0`  
*Re-implement this method to customize all batch operations done by this tool.*

## Protected Attributes

- `QWidget * m_settingsWidget = nullptr`  
*Host settings widget instance.*

## 9.111.1 Member Enumeration Documentation

### 9.111.1.1 BatchToolGroup

enum `Digikam::BatchTool::BatchToolGroup`

#### Enumerator

<code>BaseTool</code>	digikam core tools.
<code>CustomTool</code>	List of tools grouped and customized by users.
<code>ColorTool</code>	Tools to manage image colors (Curves, BCG, etc...)
<code>EnhanceTool</code>	Tools to enhance images (NR, sharp, etc...)
<code>TransformTool</code>	Tools to transform images geometry (resize, rotate, flip, etc...)
<code>DecorateTool</code>	Tools to decorate images (Border, watermark, etc...)
<code>FiltersTool</code>	Tools to apply filters and special effects (film grain, BlurFx, etc...)
<code>ConvertTool</code>	Tools to convert images format (PNG, JPEG, TIFF, etc...)
<code>MetadataTool</code>	Tools to play with metadata.

## 9.111.2 Constructor & Destructor Documentation

### 9.111.2.1 BatchTool()

```
Digikam::BatchTool::BatchTool (
    const QString & name,
    BatchToolGroup group,
    QObject *const parent = nullptr ) [explicit]
```

NOTE: these methods can be used safely in multi-threading part ([ActionThread](#)).

## 9.111.3 Member Function Documentation

### 9.111.3.1 apply()

```
bool Digikam::BatchTool::apply ( )
```

This method call customized [toolOperations\(\)](#).

### 9.111.3.2 applyFilter()

```
void Digikam::BatchTool::applyFilter (
    DImgThreadedFilter *const filter ) [protected]
```

Will call [startFilterDirectly](#) and apply the result to [image\(\)](#).

### 9.111.3.3 cancel()

```
void Digikam::BatchTool::cancel ( ) [virtual]
```

to call a dedicated method to kill sub-threads parented to this tool instance. Unforget to call parent [BatchTool::cancel\(\)](#) method in your customized implementation.

### 9.111.3.4 clone()

```
virtual BatchTool * Digikam::BatchTool::clone (
    QObject *const parent = nullptr ) const [pure virtual]
```

It's a safe construction of tools instance used in multithreading ([ActionThread](#)) to process items in parallel.

### 9.111.3.5 isCancelled()

```
bool Digikam::BatchTool::isCancelled ( ) const [protected]
```

Use this method to stop loop in your [toolOperations\(\)](#) implementation.

### 9.111.3.6 outputSuffix()

```
QString Digikam::BatchTool::outputSuffix ( ) const [virtual]
```

Typically, this is used with tool which convert to new file format. This method return and empty string by default.

### 9.111.3.7 registerSettingsWidget()

```
void Digikam::BatchTool::registerSettingsWidget ( ) [virtual]
```

Default implementation assign no settings view (a message label is just displayed). You need to call default implementation in your child class to init default signals and slots connections, after to have instanced your dedicated settings widget.

### 9.111.3.8 savefromDImg()

```
bool Digikam::BatchTool::savefromDImg ( ) const
```

- output Url set by [setOutputUrl\(\)](#) or [setOutputUrlFromInputUrl\(\)](#)
- output file format set by [outputSuffix\(\)](#). If this one is empty, format of original image is used instead.

### 9.111.3.9 setOutputUrlFromInputUrl()

```
void Digikam::BatchTool::setOutputUrlFromInputUrl ( )
```

if [outputSuffix\(\)](#) return null, file extension is the same than original.

### 9.111.3.10 setSettings()

```
void Digikam::BatchTool::setSettings (
    const BatchToolSettings & settings )
```

See BatchToolSettings container for details.

### 9.111.3.11 settingsWidget()

```
QWidget * Digikam::BatchTool::settingsWidget ( ) const
```

Return dedicated settings widget registered with [registerSettingsWidget\(\)](#).

### 9.111.3.12 signalAssignSettings2Widget

```
void Digikam::BatchTool::signalAssignSettings2Widget ( ) [signal]
```

See [registerSettingsWidget\(\)](#) implementation.



### 9.111.3.13 slotAssignSettings2Widget

```
virtual void Digikam::BatchTool::slotAssignSettings2Widget ( ) [protected], [pure virtual],
[slot]
```

This method is called by [setSettings\(\)](#) through [signalAssignSettings2Widget\(\)](#).

### 9.111.3.14 toolGroup()

```
BatchTool::BatchToolGroup Digikam::BatchTool::toolGroup ( ) const
```

See BatchToolGroup enum for details.

### 9.111.3.15 toolOperations()

```
virtual bool Digikam::BatchTool::toolOperations ( ) [protected], [pure virtual]
```

This method is called by [apply\(\)](#).

### 9.111.3.16 toolVersion()

```
virtual int Digikam::BatchTool::toolVersion ( ) const [inline], [virtual]
```

By default, ID is 1. Re-implement this method and increase this ID when tool settings change.

## 9.112 Digikam::BatchToolSet Class Reference

A container of associated batch tool and settings.

### Public Member Functions

- bool [operator==](#) (const [BatchToolSet](#) &set) const  
*Equality operator which check index, version, name, and group data.*

### Public Attributes

- [BatchTool::BatchToolGroup](#) **group** = [BatchTool::BaseTool](#)
- int **index** = -1  
*Tool identifier data. Index is tool ID from assigned list.*
- QString **name**
- [BatchToolSettings](#) **settings**  
*Settings hosted in this container.*
- int **version** = 0

## 9.112.1 Member Function Documentation

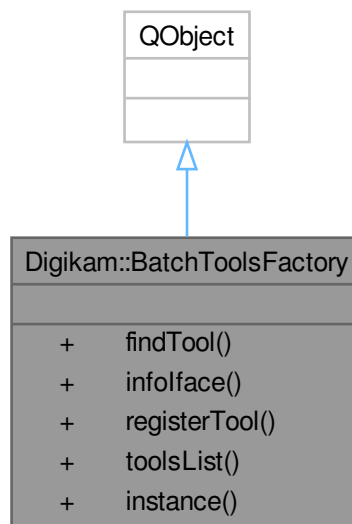
### 9.112.1.1 operator==( )

```
bool Digikam::BatchToolSet::operator==(
    const BatchToolSet & set ) const
```

Settings member is ignored.

## 9.113 Digikam::BatchToolsFactory Class Reference

Inheritance diagram for Digikam::BatchToolsFactory:



### Public Member Functions

- [BatchTool](#) \* **findTool** (const QString &name, [BatchTool::BatchToolGroup](#) group) const
- [BqmInfoface](#) \* **infoface** () const
- void **registerTool** ([BatchTool](#) \*const tool)
- [BatchToolsList](#) **toolsList** () const

### Static Public Member Functions

- static [BatchToolsFactory](#) \* **instance** ()

### Friends

- class **BatchToolsFactoryCreator**

## 9.114 Digikam::BCGContainer Class Reference

### Public Member Functions

- bool **isDefault** () const
- bool **operator==** (const [BCGContainer](#) &other) const
- void **writeToFilterAction** ([FilterAction](#) &action, const QString &prefix=QString()) const

### Static Public Member Functions

- static [BCGContainer](#) **fromFilterAction** (const [FilterAction](#) &action, const QString &prefix=QString())

### Public Attributes

- double **brightness** = 0.0
- int **channel** = LuminosityChannel
- double **contrast** = 0.0
- double **gamma** = 1.0

## 9.115 Digikam::BCGFilter Class Reference

Inheritance diagram for Digikam::BCGFilter:



### Public Member Functions

- **BCGFilter** (const [BCGContainer](#) &settings, [DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100)

- **BCGFilter** (*DImg* \*const orgImage, *QObject* \*const parent=nullptr, const **BCGContainer** &settings=**BCGContainer**())
- **BCGFilter** (*QObject* \*const parent=nullptr)
- **FilterAction** filterAction () override  
*Returns the action description corresponding to currently set options.*
- *QString* filterIdentifier () const override  
*Return the identifier for this filter in the image history.*
- void readParameters (const **FilterAction** &action) override

## Public Member Functions inherited from Digikam::DImgThreadedFilter

- **DImgThreadedFilter** (*DImg* \*const orgImage, *QObject* \*const parent, const *QString* &name=*QString*())  
*Constructs a filter with all arguments (ready to use).*
- **DImgThreadedFilter** (*QObject* \*const parent=nullptr, const *QString* &name=*QString*())  
*Constructs a filter without argument.*
- virtual void cancelFilter ()  
*Cancel the threaded computation.*
- const *QString* & filterName ()
- int filterVersion () const
- **DImg** getTargetImage ()
- *QList*< int > multithreadedSteps (int stop, int start=0) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool parametersSuccessfullyRead () const  
*Optional: error handling for readParameters.*
- virtual *QString* readParametersError (const **FilterAction** &actionThatFailed) const
- void setFilterName (const *QString* &name)
- void setFilterVersion (int version)  
*Replaying a filter action: Set the filter version.*
- void setOriginalImage (const **DImg** &orgImage)
- void setupAndStartDirectly (const **DImg** &orgImage, **DImgThreadedFilter** \*const master, int progress←Begin=0, int progressEnd=100)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void setupFilter (const **DImg** &orgImage)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void startFilter ()  
*Start the threaded computation.*
- virtual void startFilterDirectly ()  
*Start computation of this filter, directly in this thread.*
- virtual *QList*< int > supportedVersions () const

## Public Member Functions inherited from Digikam::DynamicThread

- **DynamicThread** (*QObject* \*const parent=nullptr)  
*This class extends QRunnable, so you have to reimplement virtual void run().*
- ~**DynamicThread** () override  
*The destructor calls stop() and wait(), but if you, in your destructor, delete any data that is accessed by your run() method, you must call stop() and wait() before yourself.*
- bool isFinished () const
- bool isRunning () const
- *QThread*::Priority priority () const
- void setEmitSignals (bool emitThem)
- void setPriority (*QThread*::Priority priority)  
*Sets the priority for this dynamic thread.*
- State state () const

### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from Digikam::DImgThreadedFilter

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from Digikam::DynamicThread

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from Digikam::DImgThreadedFilter

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.115.1 Member Function Documentation

### 9.115.1.1 filterAction()

`FilterAction` Digikam::BCGFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.115.1.2 filterIdentifier()

`QString` Digikam::BCGFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

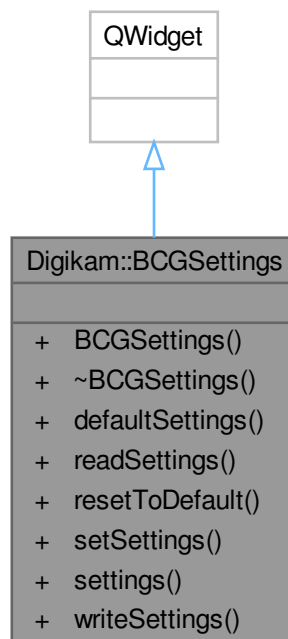
### 9.115.1.3 readParameters()

```
void Digikam::BCGFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

## 9.116 Digikam::BCGSettings Class Reference

Inheritance diagram for Digikam::BCGSettings:





## Signals

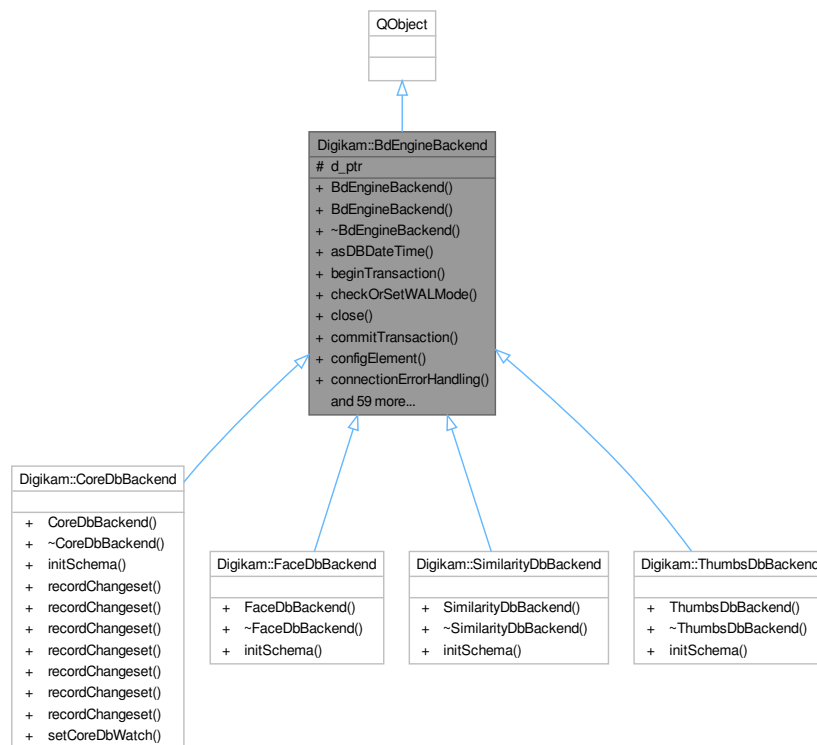
- void **signalSettingsChanged** ()

## Public Member Functions

- **BCGSettings** (QWidget \*const parent)
- **BCGContainer defaultSettings** () const
- void **readSettings** (const KConfigGroup &group)
- void **resetToDefault** ()
- void **setSettings** (const **BCGContainer** &settings)
- **BCGContainer settings** () const
- void **writeSettings** (KConfigGroup &group)

## 9.117 Digikam::BdEngineBackend Class Reference

Inheritance diagram for Digikam::BdEngineBackend:



## Classes

- class **QueryState**

## Public Types

- enum **DbType** { **SQLite** , **MySQL** }
- enum **QueryOperationStatus** { **ExecuteNormal** , **Wait** , **AbortQueries** }
- enum **QueryStateEnum** { **NoErrors** , **SQLException** , **ConnectionError** }
- enum **Status** { **Unavailable** , **Open** , **OpenSchemaChecked** }

## Public Member Functions

- **BdEngineBackend** (const QString &backendName, **DbEngineLocking** \*const locking)  
*Creates a database backend.*
- **BdEngineBackend** (const QString &backendName, **DbEngineLocking** \*const locking, **BdEngineBackend**←  
Private &dd)
- QDateTime **asDBDateTime** (const QDateTime &dateTime) const  
*Depending on the database backend return a local or UTC date format.*
- **BdEngineBackend::QueryState beginTransaction** ()  
*Begin a database transaction.*
- bool **checkOrSetWALMode** ()  
*Check or set WAL mode for SQLite database if enabled in settings.*
- void **close** ()  
*Close the database connection.*
- **BdEngineBackend::QueryState commitTransaction** ()  
*Commit the current database transaction.*
- **DbEngineConfigSettings configElement** () const  
*Return config read from XML, corresponding to this backend's database type.*
- bool **connectionErrorHandling** (int retries)  
*Called when an attempted connection to the database failed.*
- **DbEngineSqlQuery copyQuery** (const **DbEngineSqlQuery** &old)  
*Creates a faithful copy of the passed query, with the current db connection.*
- DbType **databaseType** () const  
*Return the database type.*
- bool **exec** (**DbEngineSqlQuery** &query)  
*Calls exec/execBatch on the query, and handles debug output if something went wrong.*
- bool **execBatch** (**DbEngineSqlQuery** &query)
- **QueryState execDBAction** (const **DbEngineAction** &action, const QMap< QString, QVariant > &bindingMap, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)  
*Performs the database action on the current database.*
- **QueryState execDBAction** (const **DbEngineAction** &action, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)  
*Performs the database action on the current database.*
- **QueryState execDBAction** (const QString &action, const QMap< QString, QVariant > &bindingMap, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- **QueryState execDBAction** (const QString &action, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- QSqlQuery **execDBActionQuery** (const **DbEngineAction** &action, const QMap< QString, QVariant > &bindingMap)  
*Performs the database action on the current database.*
- QSqlQuery **execDBActionQuery** (const QString &action, const QMap< QString, QVariant > &bindingMap)
- **QueryState execDirectSql** (const QString &query)  
*Calls exec on the query, and handles debug output if something went wrong.*
- **QueryState execDirectSqlWithResult** (const QString &query, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)

*Calls exec on the query, and handles debug output if something went wrong.*

- [DbEngineSqlQuery](#) [execQuery](#) (const QString &sql)

*Executes the statement and returns the query object.*

- [DbEngineSqlQuery](#) [execQuery](#) (const QString &sql, const QList< QVariant > &boundValues)
- [DbEngineSqlQuery](#) [execQuery](#) (const QString &sql, const QMap< QString, QVariant > &bindingMap)

*Method which accept a hashmap with key, values which are used for named binding.*

- [DbEngineSqlQuery](#) [execQuery](#) (const QString &sql, const QVariant &boundValue1)
- [DbEngineSqlQuery](#) [execQuery](#) (const QString &sql, const QVariant &boundValue1, const QVariant &boundValue2)
- [DbEngineSqlQuery](#) [execQuery](#) (const QString &sql, const QVariant &boundValue1, const QVariant &boundValue2, const QVariant &boundValue3)
- [DbEngineSqlQuery](#) [execQuery](#) (const QString &sql, const QVariant &boundValue1, const QVariant &boundValue2, const QVariant &boundValue3, const QVariant &boundValue4)
- void [execQuery](#) ([DbEngineSqlQuery](#) &preparedQuery, const QList< QVariant > &boundValues)
- void [execQuery](#) ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &boundValue1)

*Binds the values and executes the prepared query.*

- void [execQuery](#) ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &boundValue1, const QVariant &boundValue2)
- void [execQuery](#) ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &boundValue1, const QVariant &boundValue2, const QVariant &boundValue3)
- void [execQuery](#) ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &boundValue1, const QVariant &boundValue2, const QVariant &boundValue3, const QVariant &boundValue4)
- [QueryState](#) [execSql](#) (const QString &sql, const QList< QVariant > &boundValues, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) [execSql](#) (const QString &sql, const QMap< QString, QVariant > &bindingMap, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)

*Method which accepts a map for named binding.*

- [QueryState](#) [execSql](#) (const QString &sql, const QVariant &boundValue1, const QVariant &boundValue2, const QVariant &boundValue3, const QVariant &boundValue4, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) [execSql](#) (const QString &sql, const QVariant &boundValue1, const QVariant &boundValue2, const QVariant &boundValue3, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) [execSql](#) (const QString &sql, const QVariant &boundValue1, const QVariant &boundValue2, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) [execSql](#) (const QString &sql, const QVariant &boundValue1, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) [execSql](#) (const QString &sql, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)

*Executes the SQL statement, and write the returned data into the values list.*

- [QueryState](#) [execSql](#) ([DbEngineSqlQuery](#) &preparedQuery, const QList< QVariant > &boundValues, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) [execSql](#) ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &boundValue1, const QVariant &boundValue2, const QVariant &boundValue3, const QVariant &boundValue4, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) [execSql](#) ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &boundValue1, const QVariant &boundValue2, const QVariant &boundValue3, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) [execSql](#) ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &boundValue1, const QVariant &boundValue2, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) [execSql](#) ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &boundValue1, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) [execSql](#) ([DbEngineSqlQuery](#) &preparedQuery, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) [execUpsertDBAction](#) (const [DbEngineAction](#) &action, const QVariant &id, const QStringList &fieldNames, const QList< QVariant > &values)

- Performs a special DBAction that is usually needed to "INSERT or UPDATE" entries in a table.*
- [QueryState](#) **execUpsertDBAction** (const QString &action, const QVariant &id, const QStringList &fieldNames, const QList< QVariant > &values)
  - [DbEngineAction](#) **getDBAction** (const QString &actionName) const  
*Returns a database action with name, specified in actionName, for the current database.*
  - [DbEngineSqlQuery](#) **getQuery** ()  
*Creates an empty query object waiting for the statement.*
  - [QueryState](#) **handleQueryResult** ([DbEngineSqlQuery](#) &query, QList< QVariant > \*const values, QVariant \*const lastInsertId)  
*Checks if there was a connection error.*
  - bool **isCompatible** (const [DbEngineParameters](#) &parameters)  
*Checks if the parameters can be used for this database backend.*
  - bool **isInTransaction** () const  
*Returns if the database is in a different thread in a transaction.*
  - bool **isOpen** () const
  - bool **isReady** () const
  - QString **lastError** ()  
*Returns a description of the last error that occurred on this database.*
  - QSqlError **lastSQLError** ()  
*Returns the last error that occurred on this database.*
  - int **maximumBoundValues** () const  
*Returns the maximum number of bound parameters allowed per query.*
  - bool **open** (const [DbEngineParameters](#) &parameters)  
*Open the database connection.*
  - [DbEngineSqlQuery](#) **prepareQuery** (const QString &sql)  
*Creates a query object prepared with the statement, waiting for bound values.*
  - bool **queryErrorHandling** ([DbEngineSqlQuery](#) &query, int retries)  
*Called with a failed query.*
  - QList< QVariant > **readToList** ([DbEngineSqlQuery](#) &query)  
*Reads data of returned result set into a list which is returned.*
  - void **rollbackTransaction** ()  
*Rollback the current database transaction.*
  - void **setDbEngineErrorHandler** ([DbEngineErrorHandler](#) \*const handler)  
*Add a DbEngineErrorHandler.*
  - void **setForeignKeyChecks** (bool check)  
*Enables or disables FOREIGN\_KEY\_CHECKS for the database.*
  - [Status](#) **status** () const  
*Returns the current status of the database backend.*
  - QStringList **tables** ()  
*Returns a list with the names of tables in the database.*
  - bool **transactionErrorHandling** (const QSqlError &lastError, int retries)

### Protected Attributes

- [BdEngineBackendPrivate](#) \*const **d\_ptr** = nullptr

## 9.117.1 Member Enumeration Documentation

### 9.117.1.1 QueryStateEnum

```
enum Digikam::BdEngineBackend::QueryStateEnum
```

## Enumerator

NoErrors	No errors occurred while executing the query.
SQLException	An SQLException has occurred while executing the query.
ConnectionError	An connection error has occurred while executing the query.

## 9.117.1.2 Status

```
enum Digikam::BdEngineBackend::Status
```

## Enumerator

Unavailable	The database is not available, because it has not been opened yet or because of an error condition.
Open	The database is open. It has not been verified that the schema is up to date. This status is sufficient for use in a context where it can be assumed that the necessary schema check has been carried out by a master process.
OpenSchemaChecked	The database is open, and it has been verified that the schema is up to date, or the schema has been updated.

## 9.117.2 Constructor &amp; Destructor Documentation

## 9.117.2.1 BdEngineBackend()

```
Digikam::BdEngineBackend::BdEngineBackend (
    const QString & backendName,
    DbEngineLocking *const locking ) [explicit]
```

The backend name is an arbitrary string that shall be unique for this backend object. It will be used to create unique connection names per backend and thread.

## 9.117.3 Member Function Documentation

## 9.117.3.1 asDBDateTime()

```
QDateTime Digikam::BdEngineBackend::asDBDateTime (
    const QDateTime & dateTime ) const
```

SQLite: local date format MySQL: UTC date format

## 9.117.3.2 checkOrSetWALMode()

```
bool Digikam::BdEngineBackend::checkOrSetWALMode ( )
```

## Returns

true the WAL mode is confirmed enabled.

**9.117.3.3 close()**

```
void Digikam::BdEngineBackend::close ( )
```

Shall only be called from the thread that called [open\(\)](#).

**9.117.3.4 connectionErrorHandling()**

```
bool Digikam::BdEngineBackend::connectionErrorHandling (
    int retries )
```

If it returns true, retry; if it returns false, bail out. Pass the number of connection retries to help with some decisions.

**9.117.3.5 execDBAction() [1/2]**

```
BdEngineBackend::QueryState Digikam::BdEngineBackend::execDBAction (
    const DbEngineAction & action,
    const QMap< QString, QVariant > & bindingMap,
    QList< QVariant > *const values = nullptr,
    QVariant *const lastInsertId = nullptr )
```

Queries by the specified parameters can have named parameters which are substituted with values from the bindingMap parameter. The result values (if any) are stored within the values list.

**9.117.3.6 execDBAction() [2/2]**

```
BdEngineBackend::QueryState Digikam::BdEngineBackend::execDBAction (
    const DbEngineAction & action,
    QList< QVariant > *const values = nullptr,
    QVariant *const lastInsertId = nullptr )
```

Queries by the specified parameters mustn't have named parameters. The result values (if any) are stored within the values list.

**9.117.3.7 execDBActionQuery()**

```
QSqlQuery Digikam::BdEngineBackend::execDBActionQuery (
    const DbEngineAction & action,
    const QMap< QString, QVariant > & bindingMap )
```

Queries by the specified parameters can have named parameters which are substituted with values from the bindingMap parameter. The result values (if any) are stored within the values list. This method returns the last query, which is used to handle special cases.

**9.117.3.8 execDirectSql()**

```
BdEngineBackend::QueryState Digikam::BdEngineBackend::execDirectSql (
    const QString & query )
```

The query is not prepared, which can be fail in certain situations (e.g. trigger statements on QMYSQL).

### 9.117.3.9 execDirectSqlWithResult()

```
BdEngineBackend::QueryState Digikam::BdEngineBackend::execDirectSqlWithResult (
    const QString & query,
    QList< QVariant > *const values = nullptr,
    QVariant *const lastInsertId = nullptr )
```

The query is not prepared, which can be fail in certain situations (e.g. trigger statements on QMYSQL).

### 9.117.3.10 execQuery()

```
DbEngineSqlQuery Digikam::BdEngineBackend::execQuery (
    const QString & sql )
```

Methods are provided for up to four bound values (positional binding), or for a list of bound values.

### 9.117.3.11 execSql() [1/2]

```
BdEngineBackend::QueryState Digikam::BdEngineBackend::execSql (
    const QString & sql,
    const QMap< QString, QVariant > & bindingMap,
    QList< QVariant > *const values = nullptr,
    QVariant *const lastInsertId = nullptr )
```

For special cases it's also possible to add a [DbEngineActionType](#) which wraps another data object (also lists or maps) which can be used as field entry or as value (where it's prepared with positional binding). See more on [DbEngineActionType](#) class. If the wrapped data object is an instance of list, then the elements are separated by comma. If the wrapped data object is an instance of map, then the elements are inserted in the following way: key1=value1, key2=value2,...,keyN=valueN.

### 9.117.3.12 execSql() [2/2]

```
BdEngineBackend::QueryState Digikam::BdEngineBackend::execSql (
    const QString & sql,
    QList< QVariant > *const values = nullptr,
    QVariant *const lastInsertId = nullptr )
```

If you are not interested in the returned data, set values to 0. Methods are provided for up to four bound values (positional binding), or for a list of bound values. If you want the last inserted id (and your query is suitable), set lastInsertId to the address of a QVariant. Additionally, methods are provided for prepared statements.

### 9.117.3.13 execUpsertDBAction()

```
BdEngineBackend::QueryState Digikam::BdEngineBackend::execUpsertDBAction (
    const DbEngineAction & action,
    const QVariant & id,
    const QStringList & fieldNames,
    const QList< QVariant > & values )
```

The corresponding DBAction must contain exactly the named parameters :id, :fieldValueList, :fieldList and :value←→List. You pass the value to be bound to the ":id" field, then two lists of the same size: The first containing the field names, the second one containing the values as QVariants ready for binding.

### 9.117.3.14 `handleQueryResult()`

```
BdEngineBackend::QueryState Digikam::BdEngineBackend::handleQueryResult (
    DbEngineSqlQuery & query,
    QList< QVariant > *const values,
    QVariant *const lastInsertId )
```

If so [BdEngineBackend::ConnectionError](#) is returned. If not, the values are extracted from the query and inserted in the values list, the last insertion id is taken from the query and [BdEngineBackend::NoErrors](#) is returned.

### 9.117.3.15 `isInTransaction()`

```
bool Digikam::BdEngineBackend::isInTransaction ( ) const
```

Note that a transaction does not require holding [CoreDbAccess](#). Note that this does not give information about other processes locking the database.

### 9.117.3.16 `lastError()`

```
QString Digikam::BdEngineBackend::lastError ( )
```

Use [CoreDbAccess::lastError](#) for errors presented to the user. This error will be included in that message. It may be empty.

### 9.117.3.17 `lastSQLError()`

```
QString Digikam::BdEngineBackend::lastSQLError ( )
```

Use [CoreDbAccess::lastError](#) for errors presented to the user. It may be empty.

### 9.117.3.18 `maximumBoundValues()`

```
int Digikam::BdEngineBackend::maximumBoundValues ( ) const
```

This value depends on the database engine.

### 9.117.3.19 `open()`

```
bool Digikam::BdEngineBackend::open (
    const DbEngineParameters & parameters )
```

#### Returns

true on success



### 9.117.3.20 queryErrorHandling()

```
bool Digikam::BdEngineBackend::queryErrorHandling (
    DbEngineSqlQuery & query,
    int retries )
```

Handles certain known errors and debug output. If it returns true, reexecute the query; if it returns false, return it as failed. Pass the number of retries already done for this query to help with some decisions.

### 9.117.3.21 readToList()

```
QList< QVariant > Digikam::BdEngineBackend::readToList (
    DbEngineSqlQuery & query )
```

The read process is column wise, which means all data elements of a row is read, then the resultset is switched to the next row.

### 9.117.3.22 setDbEngineErrorHandler()

```
void Digikam::BdEngineBackend::setDbEngineErrorHandler (
    DbEngineErrorHandler *const handler )
```

This object must be created in the main thread. If a database error occurs, this object can handle problem solving and user interaction.

### 9.117.3.23 setForeignKeyChecks()

```
void Digikam::BdEngineBackend::setForeignKeyChecks (
    bool check )
```

This function depends on the database engine.

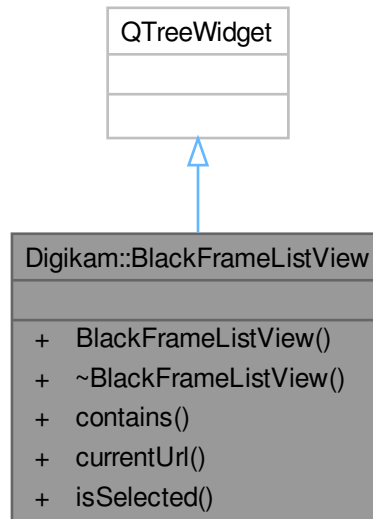
## 9.118 Digikam::BdEngineBackend::QueryState Class Reference

### Public Member Functions

- **QueryState** (const [QueryStateEnum](#) value)
- **operator bool** () const
- **operator QueryStateEnum** () const

## 9.119 Digikam::BlackFrameListView Class Reference

Inheritance diagram for Digikam::BlackFrameListView:



### Signals

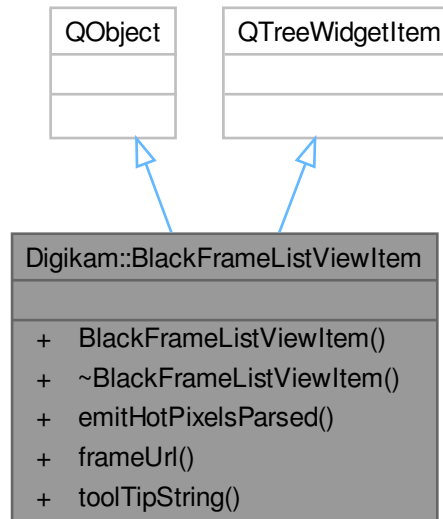
- void **signalBlackFrameRemoved** (const `QUrl` &)
- void **signalBlackFrameSelected** (const `QList`< `HotPixelProps` > &, const `QUrl` &)
- void **signalClearBlackFrameList** ()

### Public Member Functions

- **BlackFrameListView** (`QWidget` \*const parent=nullptr)
- bool **contains** (const `QUrl` &url)
- `QUrl` **currentUrl** ()
- bool **isSelected** (const `QUrl` &url)

## 9.120 Digikam::BlackFrameListViewItem Class Reference

Inheritance diagram for Digikam::BlackFrameListViewItem:



### Public Types

- enum `BlackFrameConst` { `PREVIEW = 0` , `SIZE = 1` , `HOTPIXELS = 2` , `THUMB_WIDTH = 150` }

### Signals

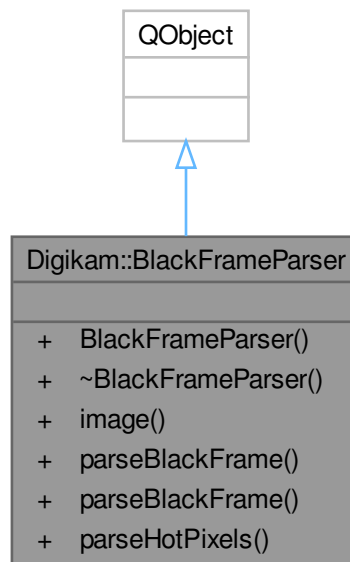
- void `signalHotPixelsParsed` (const QList< [HotPixelProps](#) > &, const QUrl &)

### Public Member Functions

- `BlackFrameListViewItem` (QTreeWidgetItem \*const parent, const QUrl &url)
- void `emitHotPixelsParsed` ()
- QUrl `frameUrl` () const
- QString `toolTipString` () const

## 9.121 Digikam::BlackFrameParser Class Reference

Inheritance diagram for Digikam::BlackFrameParser:



### Signals

- void **signalHotPixelsParsed** (const QList< [HotPixelProps](#) > &)
- void **signalLoadingComplete** ()
- void **signalLoadingProgress** (float)

### Public Member Functions

- **BlackFrameParser** (QObject \*const parent)
- [DImg](#) **image** () const
- void **parseBlackFrame** (const [DImg](#) &img)
- void **parseBlackFrame** (const [QUrl](#) &url)
- void **parseHotPixels** (const [QString](#) &file)

## 9.122 Digikam::BlackFrameToolTip Class Reference

Inheritance diagram for Digikam::BlackFrameToolTip:



### Public Member Functions

- **BlackFrameToolTip** (`QTreeWidgetItem *const view`)
- void **setItem** (`QTreeWidgetItem *const item`)
- void **setToolTipString** (`const QString &tip`)
- void **show** ()

## Public Member Functions inherited from [Digikam::DItemToolTip](#)

- **DItemToolTip** (QWidget \*const parent=nullptr)

## Protected Member Functions

- QRect [repositionRect](#) () override
- QString [tipContents](#) () override

## Protected Member Functions inherited from [Digikam::DItemToolTip](#)

- bool **event** (QEvent \*) override
- void **paintEvent** (QPaintEvent \*) override
- void **renderArrows** ()
- void **reposition** ()
- void **resizeEvent** (QResizeEvent \*) override
- bool **toolTipsEmpty** () const
- void **updateToolTip** ()

## 9.122.1 Member Function Documentation

### 9.122.1.1 [repositionRect\(\)](#)

QRect Digikam::BlackFrameToolTip::repositionRect ( ) [override], [protected], [virtual]

Implements [Digikam::DItemToolTip](#).

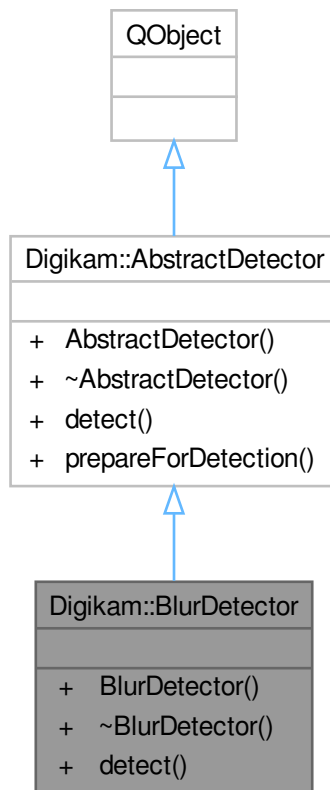
### 9.122.1.2 [tipContents\(\)](#)

QString Digikam::BlackFrameToolTip::tipContents ( ) [override], [protected], [virtual]

Implements [Digikam::DItemToolTip](#).

## 9.123 Digikam::BlurDetector Class Reference

Inheritance diagram for Digikam::BlurDetector:



### Public Member Functions

- `BlurDetector` (const [DImg](#) &image)
- float `detect` (const cv::Mat &image) const override

### Public Member Functions inherited from [Digikam::AbstractDetector](#)

- `AbstractDetector` (`QObject *const parent=nullptr`)

### Additional Inherited Members

### Static Public Member Functions inherited from [Digikam::AbstractDetector](#)

- static cv::Mat `prepareForDetection` (const [DImg](#) &inputImage)

*NOTE: Maybe this function will move to `read_image()` of `imagequalityparser` in case all detectors of IQS use `cv::Mat`.*

## 9.123.1 Member Function Documentation

### 9.123.1.1 detect()

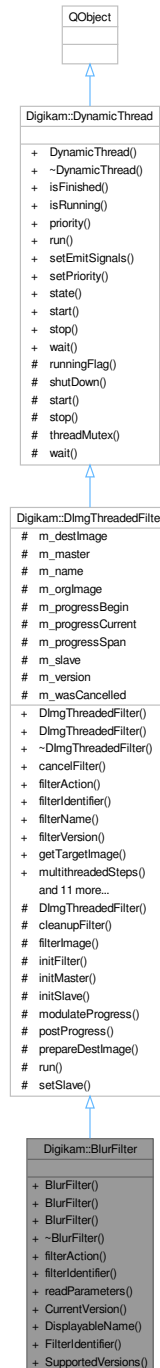
```
float Digikam::BlurDetector::detect (  
    const cv::Mat & image ) const [override], [virtual]
```

Implements [Digikam::AbstractDetector](#).



## 9.124 Digikam::BlurFilter Class Reference

Inheritance diagram for Digikam::BlurFilter:



### Public Member Functions

- **BlurFilter** ([DImg](#) \*const orgImage, [QObject](#) \*const parent=nullptr, int radius=3)
- **BlurFilter** ([DImgThreadedFilter](#) \*const parentFilter, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, int radius=3)

Constructor for slave mode: execute immediately in current thread with specified master filter.

- **BlurFilter** (QObject \*const parent=nullptr)
- **FilterAction filterAction** () override
 

Returns the action description corresponding to currently set options.
- **QString filterIdentifier** () const override
 

Return the identifier for this filter in the image history.
- void **readParameters** (const **FilterAction** &action) override

## Public Member Functions inherited from **Digikam::DImgThreadedFilter**

- **DImgThreadedFilter** (DImg \*const orgImage, QObject \*const parent, const QString &name=QString())
 

Constructs a filter with all arguments (ready to use).
- **DImgThreadedFilter** (QObject \*const parent=nullptr, const QString &name=QString())
 

Constructs a filter without argument.
- virtual void **cancelFilter** ()
 

Cancel the threaded computation.
- const QString & **filterName** ()
- int **filterVersion** () const
- **DImg getTargetImage** ()
- QList< int > **multithreadedSteps** (int stop, int start=0) const
 

This method return a list of steps to process parallelized operation in filter using QtConcurrents API.
- virtual bool **parametersSuccessfullyRead** () const
 

Optional: error handling for readParameters.
- virtual QString **readParametersError** (const **FilterAction** &actionThatFailed) const
- void **setFilterName** (const QString &name)
- void **setFilterVersion** (int version)
 

Replaying a filter action: Set the filter version.
- void **setOriginalImage** (const **DImg** &orgImage)
- void **setupAndStartDirectly** (const **DImg** &orgImage, **DImgThreadedFilter** \*const master, int progress←Begin=0, int progressEnd=100)
 

Initializes the filter for use as a slave and directly starts computation (in-thread)
- void **setupFilter** (const **DImg** &orgImage)
 

You need to call this and then start filter of you used the constructor not setting an original image.
- virtual void **startFilter** ()
 

Start the threaded computation.
- virtual void **startFilterDirectly** ()
 

Start computation of this filter, directly in this thread.
- virtual QList< int > **supportedVersions** () const

## Public Member Functions inherited from **Digikam::DynamicThread**

- **DynamicThread** (QObject \*const parent=nullptr)
 

This class extends **QRunnable**, so you have to reimplement virtual void **run()**.
- **~DynamicThread** () override
 

The destructor calls **stop()** and **wait()**, but if you, in your destructor, delete any data that is accessed by your **run()** method, you must call **stop()** and **wait()** before yourself.
- bool **isFinished** () const
- bool **isRunning** () const
- **QThread::Priority priority** () const
- void **setEmitSignals** (bool emitThem)
- void **setPriority** (QThread::Priority priority)
 

Sets the priority for this dynamic thread.
- State **state** () const

### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from [Digikam::DImgThreadedFilter](#)

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.124.1 Member Function Documentation

### 9.124.1.1 filterAction()

`FilterAction` Digikam::BlurFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.124.1.2 filterIdentifier()

`QString` Digikam::BlurFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

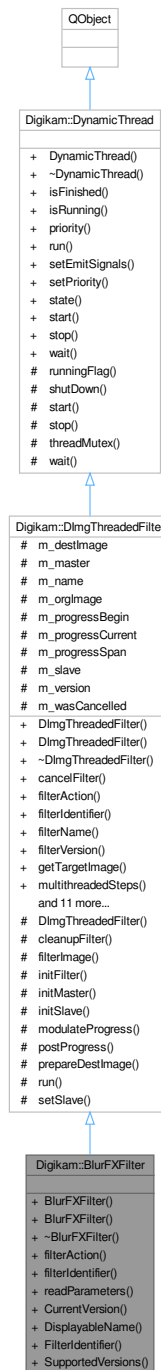
### 9.124.1.3 readParameters()

```
void Digikam::BlurFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

## 9.125 Digikam::BlurFXFilter Class Reference

Inheritance diagram for Digikam::BlurFXFilter:



### Public Types

- enum **BlurFXFilterTypes** {
  - ZoomBlur** = 0 , **RadialBlur** , **FarBlur** , **MotionBlur** , **SoftenerBlur** , **ShakeBlur** , **FocusBlur** , **SmartBlur** , **FrostGlass** , **Mosaic** }

## Public Types inherited from Digikam::DynamicThread

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

## Public Member Functions

- **BlurFXFilter** (**DImg** \*const orgImage, **QObject** \*const parent=nullptr, int blurFXType=ZoomBlur, int distance=100, int level=45)
- **BlurFXFilter** (**QObject** \*const parent=nullptr)
- **FilterAction** filterAction () override  
*Returns the action description corresponding to currently set options.*
- **QString** filterIdentifier () const override  
*Return the identifier for this filter in the image history.*
- void **readParameters** (const **FilterAction** &action) override

## Public Member Functions inherited from Digikam::DImgThreadedFilter

- **DImgThreadedFilter** (**DImg** \*const orgImage, **QObject** \*const parent, const **QString** &name=**QString**())  
*Constructs a filter with all arguments (ready to use).*
- **DImgThreadedFilter** (**QObject** \*const parent=nullptr, const **QString** &name=**QString**())  
*Constructs a filter without argument.*
- virtual void **cancelFilter** ()  
*Cancel the threaded computation.*
- const **QString** & **filterName** ()
- int **filterVersion** () const
- **DImg** **getTargetImage** ()
- **QList**< int > **multithreadedSteps** (int stop, int start=0) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool **parametersSuccessfullyRead** () const  
*Optional: error handling for readParameters.*
- virtual **QString** **readParametersError** (const **FilterAction** &actionThatFailed) const
- void **setFilterName** (const **QString** &name)
- void **setFilterVersion** (int version)  
*Replaying a filter action: Set the filter version.*
- void **setOriginalImage** (const **DImg** &orgImage)
- void **setupAndStartDirectly** (const **DImg** &orgImage, **DImgThreadedFilter** \*const master, int progress←Begin=0, int progressEnd=100)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void **setupFilter** (const **DImg** &orgImage)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void **startFilter** ()  
*Start the threaded computation.*
- virtual void **startFilterDirectly** ()  
*Start computation of this filter, directly in this thread.*
- virtual **QList**< int > **supportedVersions** () const

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread](#) (QObject \*const parent=nullptr)
 

*This class extends QRunnable, so you have to reimplement virtual void [run\(\)](#).*
- [~DynamicThread](#) () override
 

*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool [isFinished](#) () const
- bool [isRunning](#) () const
- QThread::Priority [priority](#) () const
- void [setEmitSignals](#) (bool emitThem)
- void [setPriority](#) (QThread::Priority priority)
 

*Sets the priority for this dynamic thread.*
- State [state](#) () const

## Static Public Member Functions

- static int [CurrentVersion](#) ()
- static QString [DisplayableName](#) ()
- static QString [FilterIdentifier](#) ()
- static QList< int > [SupportedVersions](#) ()

## Additional Inherited Members

## Public Slots inherited from [Digikam::DynamicThread](#)

- void [start](#) ()
- void [stop](#) ()
 

*Stop computation, sets the running flag to false.*
- void [wait](#) ()
 

*Waits until the thread finishes.*

## Signals inherited from [Digikam::DImgThreadedFilter](#)

- void [finished](#) (bool success)
 

*Emitted when the computation has completed.*
- void [progress](#) (int progress)
 

*Emitted when progress info from the calculation is available.*
- void [started](#) ()
 

*This signal is emitted when image data is available and the computation has started.*

## Signals inherited from [Digikam::DynamicThread](#)

- void [finished](#) ()
- void [starting](#) ()
 

*Emitted if emitSignals is enabled.*



## Protected Member Functions inherited from Digikam::DImgThreadedFilter

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from Digikam::DynamicThread

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from Digikam::DImgThreadedFilter

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.125.1 Member Function Documentation

### 9.125.1.1 filterAction()

`FilterAction` Digikam::BlurFXFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.125.1.2 filterIdentifier()

`QString` Digikam::BlurFXFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

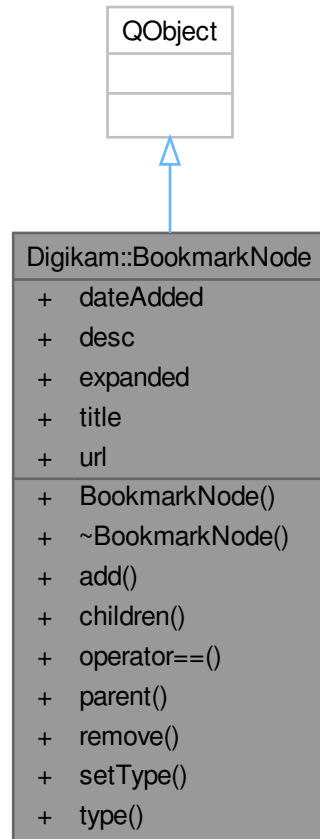
### 9.125.1.3 readParameters()

```
void Digikam::BlurFXFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

## 9.126 Digikam::BookmarkNode Class Reference

Inheritance diagram for Digikam::BookmarkNode:



### Public Types

- enum `Type` {  
`Root`, `Folder`, `Bookmark`, `Separator`,  
`RootFolder` }

### Public Member Functions

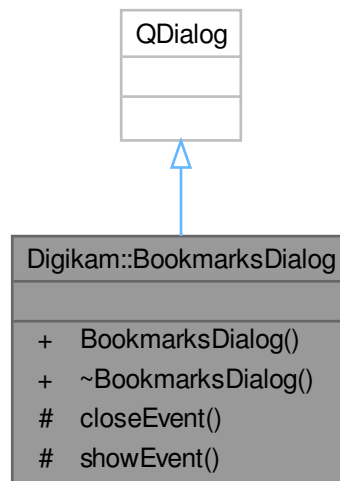
- `BookmarkNode` (Type type=`Root`, `BookmarkNode` \*const parent=nullptr)
- void `add` (`BookmarkNode` \*const child, int offset=-1)
- QList< `BookmarkNode` \* > `children` () const
- bool `operator==` (const `BookmarkNode` &other) const
- `BookmarkNode` \* `parent` () const
- void `remove` (`BookmarkNode` \*const child)
- void `setType` (Type type)
- Type `type` () const

### Public Attributes

- QDateTime **dateAdded**
- QString **desc**
- bool **expanded**
- QString **title**
- QString **url**

## 9.127 Digikam::BookmarksDialog Class Reference

Inheritance diagram for Digikam::BookmarksDialog:



### Public Member Functions

- **BookmarksDialog** (QWidget \*const parent=nullptr, [BookmarksManager](#) \*const mngr=nullptr)

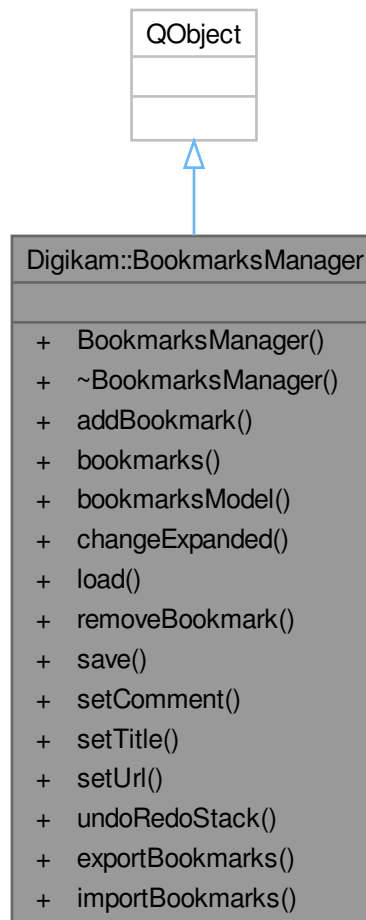
### Protected Member Functions

- void **closeEvent** (QCloseEvent \*) override
- void **showEvent** (QShowEvent \*) override

## 9.128 Digikam::BookmarksManager Class Reference

Bookmark manager, owner of the bookmarks, loads, saves and basic tasks.

Inheritance diagram for Digikam::BookmarksManager:



### Public Slots

- void **exportBookmarks** ()
- void **importBookmarks** ()

### Signals

- void **entryAdded** ([BookmarkNode](#) \*item)
- void **entryChanged** ([BookmarkNode](#) \*item)
- void **entryRemoved** ([BookmarkNode](#) \*parent, int row, [BookmarkNode](#) \*item)

## Public Member Functions

- **BookmarksManager** (const QString &bookmarksFile, QObject \*const parent=nullptr)
- void **addBookmark** ([BookmarkNode](#) \*const parent, [BookmarkNode](#) \*const node, int row=-1)
- [BookmarkNode](#) \* **bookmarks** ()
- [BookmarksModel](#) \* **bookmarksModel** ()
- void **changeExpanded** ()
- void **load** ()
- void **removeBookmark** ([BookmarkNode](#) \*const node)
- void **save** ()
- void **setComment** ([BookmarkNode](#) \*const node, const QString &newDesc)
- void **setTitle** ([BookmarkNode](#) \*const node, const QString &newTitle)
- void **setUrl** ([BookmarkNode](#) \*const node, const QString &newUrl)
- QUndoStack \* **undoRedoStack** () const

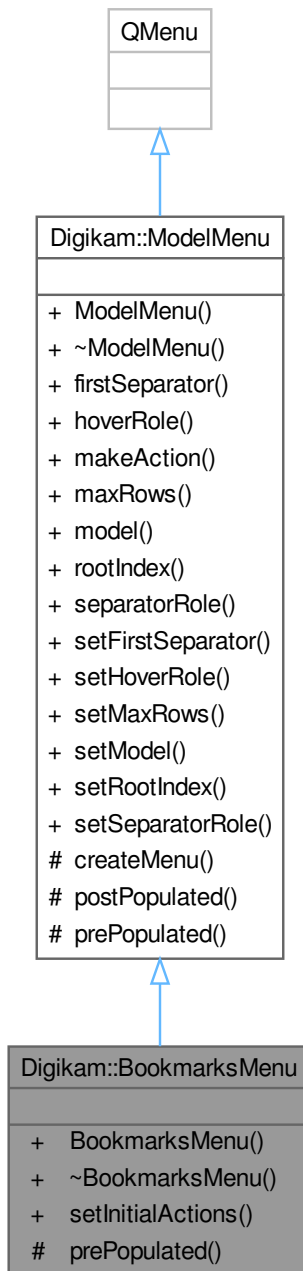
## Friends

- class **ChangeBookmarkCommand**
- class **RemoveBookmarksCommand**

## 9.129 Digikam::BookmarksMenu Class Reference

Menu that is dynamically populated from the bookmarks.

Inheritance diagram for Digikam::BookmarksMenu:



## Signals

- void **openUrl** (const QUrl &url)

## Signals inherited from [Digikam::ModelMenu](#)

- void **activated** (const QModelIndex &index)
- void **hovered** (const QString &text)

## Public Member Functions

- **BookmarksMenu** ([BookmarksManager](#) \*const mngr, QWidget \*const parent=nullptr)
- void **setInitialActions** (const QList< QAction \* > &actions)

## Public Member Functions inherited from [Digikam::ModelMenu](#)

- **ModelMenu** (QWidget \*const parent=nullptr)
- int **firstSeparator** () const
- int **hoverRole** () const
- QAction \* **makeAction** (const QIcon &icon, const QString &text, QObject \*const parent)
- int **maxRows** () const
- QAbstractItemModel \* **model** () const
- QModelIndex **rootIndex** () const
- int **separatorRole** () const
- void **setFirstSeparator** (int offset)
- void **setHoverRole** (int role)
- void **setMaxRows** (int max)
- void **setModel** (QAbstractItemModel \*model)
- void **setRootIndex** (const QModelIndex &index)
- void **setSeparatorRole** (int role)

## Protected Member Functions

- bool [prePopulated](#) () override  
*add any actions before the tree, return true if any actions are added.*

## Protected Member Functions inherited from [Digikam::ModelMenu](#)

- void **createMenu** (const QModelIndex &parent, int max, QMenu \*parentMenu=nullptr, QMenu \*menu=nullptr)  
*put all of the children of parent into menu up to max*
- virtual void **postPopulated** ()  
*add any actions after the tree*

## 9.129.1 Member Function Documentation

### 9.129.1.1 prePopulated()

```
bool Digikam::BookmarksMenu::prePopulated ( ) [override], [protected], [virtual]
```

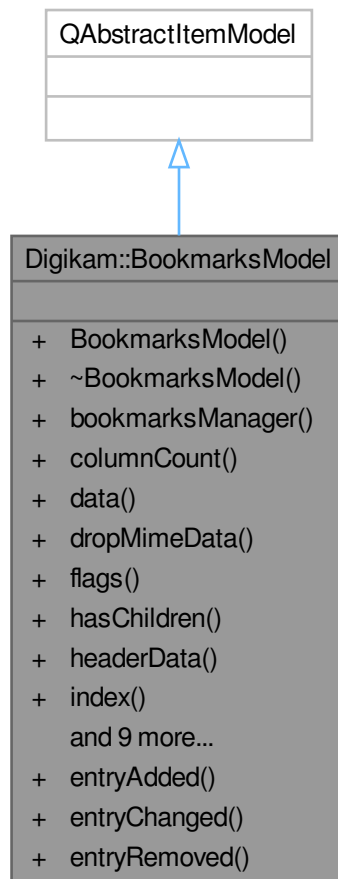
Reimplemented from [Digikam::ModelMenu](#).



## 9.130 Digikam::BookmarksModel Class Reference

[BookmarksModel](#) is a `QAbstractItemModel` wrapper around the `BookmarkManager`.

Inheritance diagram for `Digikam::BookmarksModel`:



### Public Types

- enum **Roles** {  
**TypeRole** = `Qt::UserRole + 1` , **UrlRole** = `Qt::UserRole + 2` , **UrlStringRole** = `Qt::UserRole + 3` , **Separator**↔  
**Role** = `Qt::UserRole + 4` ,  
**DateAddedRole** = `Qt::UserRole + 5` }

### Public Slots

- void **entryAdded** ([BookmarkNode](#) \*item)
- void **entryChanged** ([BookmarkNode](#) \*item)
- void **entryRemoved** ([BookmarkNode](#) \*parent, int row, [BookmarkNode](#) \*item)

## Public Member Functions

- **BookmarksModel** ([BookmarksManager](#) \*const mngr, QObject \*const parent=nullptr)
- [BookmarksManager](#) \* **bookmarksManager** () const
- int **columnCount** (const QModelIndex &parent=QModelIndex()) const override
- QVariant **data** (const QModelIndex &index, int role=Qt::DisplayRole) const override
- bool **dropMimeData** (const QMimeData \*data, Qt::DropAction action, int row, int column, const QModelIndex &parent) override
- Qt::ItemFlags **flags** (const QModelIndex &index) const override
- bool **hasChildren** (const QModelIndex &parent=QModelIndex()) const override
- QVariant **headerData** (int section, Qt::Orientation orientation, int role=Qt::DisplayRole) const override
- QModelIndex **index** ([BookmarkNode](#) \*node) const
- QModelIndex **index** (int, int, const QModelIndex &=QModelIndex()) const override
- QMimeData \* **mimeData** (const QModelIndexList &indexes) const override
- QStringList **mimeTypes** () const override
- [BookmarkNode](#) \* **node** (const QModelIndex &index) const
- QModelIndex **parent** (const QModelIndex &index=QModelIndex()) const override
- bool **removeRows** (int row, int count, const QModelIndex &parent=QModelIndex()) override
- int **rowCount** (const QModelIndex &parent=QModelIndex()) const override
- bool **setData** (const QModelIndex &index, const QVariant &value, int role=Qt::EditRole) override
- Qt::DropActions **supportedDropActions** () const override

## 9.131 Digikam::BorderContainer Class Reference

### Public Types

- enum **BorderTypes** {  
**SolidBorder** = 0 , **NiepceBorder** , **BeveledBorder** , **PineBorder** ,  
**WoodBorder** , **PaperBorder** , **ParqueBorder** , **IceBorder** ,  
**LeafBorder** , **MarbleBorder** , **RainBorder** , **CratersBorder** ,  
**DriedBorder** , **PinkBorder** , **StoneBorder** , **ChalkBorder** ,  
**GraniteBorder** , **RockBorder** , **WallBorder** }

### Static Public Member Functions

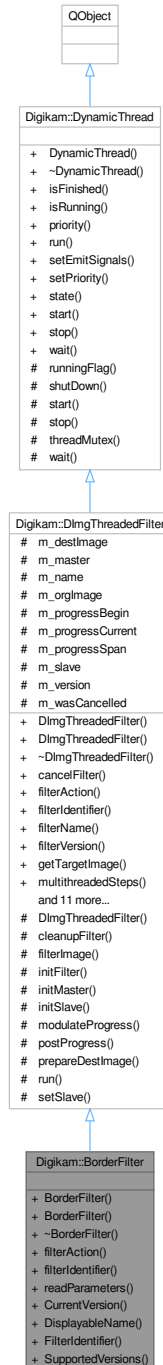
- static QString **getBorderPath** (int border)

### Public Attributes

- QColor **bevelLowerRightColor** = QColor(128, 128, 128)
- QColor **bevelUpperLeftColor** = QColor(192, 192, 192)
- QString **borderPath**
- double **borderPercent** = 0.1
- int **borderType** = 0
- int **borderWidth1** = 0
- int **borderWidth2** = 0
- int **borderWidth3** = 0
- int **borderWidth4** = 0
- QColor **decorativeFirstColor** = QColor(0, 0, 0)
- QColor **decorativeSecondColor** = QColor(0, 0, 0)
- QColor **niepceBorderColor** = QColor(255, 255, 255)
- QColor **niepceLineColor** = QColor(0, 0, 0)
- int **orgHeight** = 0
- int **orgWidth** = 0
- bool **preserveAspectRatio** = true
- QColor **solidColor** = QColor(0, 0, 0)

## 9.132 Digikam::BorderFilter Class Reference

Inheritance diagram for Digikam::BorderFilter:



### Public Member Functions

- **BorderFilter** (`DImg *origImage`, `QObject *const parent=nullptr`, `const BorderContainer &settings=BorderContainer()`)
- **BorderFilter** (`QObject *const parent=nullptr`)

- *Constructor using settings to preserve aspect ratio of image.*
- [FilterAction filterAction](#) () override
  - *Returns the action description corresponding to currently set options.*
- [QString filterIdentifier](#) () const override
  - *Return the identifier for this filter in the image history.*
- void [readParameters](#) (const [FilterAction](#) &action) override

## Public Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImg](#) \*const orgImage, [QObject](#) \*const parent, const [QString](#) &name=[QString](#)())
  - *Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter](#) ([QObject](#) \*const parent=nullptr, const [QString](#) &name=[QString](#)())
  - *Constructs a filter without argument.*
- virtual void [cancelFilter](#) ()
  - *Cancel the threaded computation.*
- const [QString](#) & [filterName](#) ()
- int [filterVersion](#) () const
- [DImg](#) [getTargetImage](#) ()
- [QList](#)< int > [multithreadedSteps](#) (int stop, int start=0) const
  - *This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool [parametersSuccessfullyRead](#) () const
  - *Optional: error handling for readParameters.*
- virtual [QString](#) [readParametersError](#) (const [FilterAction](#) &actionThatFailed) const
- void [setFilterName](#) (const [QString](#) &name)
- void [setFilterVersion](#) (int version)
  - *Replaying a filter action: Set the filter version.*
- void [setOriginalImage](#) (const [DImg](#) &orgImage)
- void [setupAndStartDirectly](#) (const [DImg](#) &orgImage, [DImgThreadedFilter](#) \*const master, int progress←Begin=0, int progressEnd=100)
  - *Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void [setupFilter](#) (const [DImg](#) &orgImage)
  - *You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void [startFilter](#) ()
  - *Start the threaded computation.*
- virtual void [startFilterDirectly](#) ()
  - *Start computation of this filter, directly in this thread.*
- virtual [QList](#)< int > [supportedVersions](#) () const

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread](#) ([QObject](#) \*const parent=nullptr)
  - *This class extends [QRunnable](#), so you have to reimplement virtual void [run\(\)](#).*
- [~DynamicThread](#) () override
  - *The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool [isFinished](#) () const
- bool [isRunning](#) () const
- [QThread::Priority](#) [priority](#) () const
- void [setEmitSignals](#) (bool emitThem)
- void [setPriority](#) ([QThread::Priority](#) priority)
  - *Sets the priority for this dynamic thread.*
- State [state](#) () const

### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

### Protected Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

### Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

### Protected Attributes inherited from [Digikam::DImgThreadedFilter](#)

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.132.1 Member Function Documentation

### 9.132.1.1 filterAction()

`FilterAction` Digikam::BorderFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.132.1.2 filterIdentifier()

`QString` Digikam::BorderFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

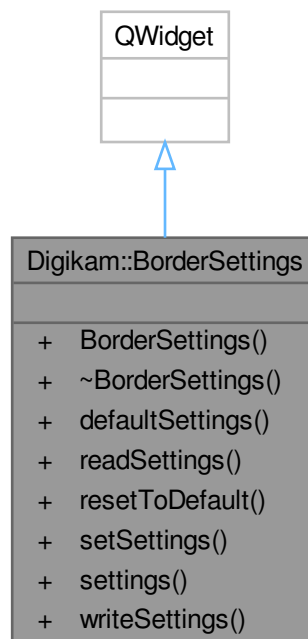
### 9.132.1.3 readParameters()

`void` Digikam::BorderFilter::readParameters (   
     const `FilterAction` & *action* ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

## 9.133 Digikam::BorderSettings Class Reference

Inheritance diagram for Digikam::BorderSettings:



## Signals

- void **signalSettingsChanged** ()

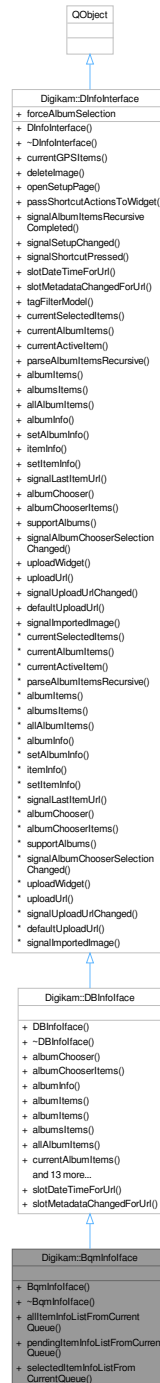
## Public Member Functions

- **BorderSettings** (QWidget \*const parent)
- [BorderContainer](#) **defaultSettings** () const
- void **readSettings** (const KConfigGroup &group)
- void **resetToDefault** ()
- void **setSettings** (const [BorderContainer](#) &settings)
- [BorderContainer](#) **settings** () const
- void **writeSettings** (KConfigGroup &group)



## 9.134 Digikam::BqmInfoface Class Reference

Inheritance diagram for Digikam::BqmInfoface:



### Public Member Functions

- **BqmInfoface** (QObject \*const parent)
- [QueuePoolItemsList allItemInfoListFromCurrentQueue](#) () const

*Return all item info list from the current queue.*

- [QueuePoolItemsList pendingItemInfoListFromCurrentQueue](#) () const

*Return pending item info list from the current queue.*

- [QueuePoolItemsList selectedItemInfoListFromCurrentQueue](#) () const

*Return selected item info list from the current queue.*

## Public Member Functions inherited from [Digikam::DBInfolface](#)

- **DBInfolface** (QObject \*const parent, const QList< QUrl > &lst=QList< QUrl >(), const [OperationType](#) type=[UnspecifiedOps](#))

- QWidget \* [albumChooser](#) (QWidget \*const parent) const override

*Albums chooser view methods (to use items from albums before to process).*

- [DAlbumIDs albumChooserItems](#) () const override
- [DInfoMap albumInfo](#) (int) const override
- QList< QUrl > [albumItems](#) ([Album](#) \*const album) const
- QList< QUrl > [albumItems](#) (int id) const override
- QList< QUrl > [albumsItems](#) (const [DAlbumIDs](#) &) const override
- QList< QUrl > [allAlbumItems](#) () const override
- QList< QUrl > [currentAlbumItems](#) () const override
- QList< [GPSItemContainer](#) \* > [currentGPSItems](#) () const override
- QList< QUrl > [currentSelectedItems](#) () const override

*Low level items and albums methods.*

- QUrl [defaultUploadUrl](#) () const override

*Url to upload new items without to use album selector.*

- void [deleteImage](#) (const QUrl &url) override

*Manipulate with item.*

- [DInfoMap itemInfo](#) (const QUrl &) const override
- void [openSetupPage](#) (SetupPage page) override

*Open configuration dialog page.*

- void [parseAlbumItemsRecursive](#) () override
- QMap< QString, QString > [passShortcutActionsToWidget](#) (QWidget \*const wdg) const override

*Pass extra shortcut actions to widget and return prefixes of shortcuts.*

- void [setItemInfo](#) (const QUrl &, const [DInfoMap](#) &) override
- bool [supportAlbums](#) () const override
- QAbstractItemModel \* [tagFilterModel](#) () override

*Return an instance of tag filter model if host application support this feature, else null pointer.*

- QUrl [uploadUrl](#) () const override
- QWidget \* [uploadWidget](#) (QWidget \*const parent) const override

*Album selector view methods (to upload items from an external place).*

## Public Member Functions inherited from [Digikam::DInfolInterface](#)

- **DInfolInterface** (QObject \*const parent)
- Q\_SIGNAL void [signalAlbumItemsRecursiveCompleted](#) (const QList< QUrl > &imageList)
- Q\_SIGNAL void [signalSetupChanged](#) ()
- Q\_SIGNAL void [signalShortcutPressed](#) (const QString &shortcut, int val)
- virtual Q\_SLOT void [slotDateTimeForUrl](#) (const QUrl &url, const QDateTime &dt, bool updModDate)

*Slot to call when date time stamp from item is changed.*

- virtual Q\_SLOT void [slotMetadataChangedForUrl](#) (const QUrl &url)

*Slot to call when something in metadata from item is changed.*

- virtual `QUrl` **currentActiveItem** () const
- virtual void **setAlbumInfo** (int, const `DInfoMap` &) const
- `Q_SIGNAL` void **signalLastItemUrl** (const `QUrl` &)
  
- `Q_SIGNAL` void **signalAlbumChooserSelectionChanged** ()
  
- `Q_SIGNAL` void **signalUploadUrlChanged** ()
- `Q_SIGNAL` void **signalImportedImage** (const `QUrl` &)

#### Additional Inherited Members

#### Public Types inherited from `Digikam::DInfoInterface`

- typedef `QList< int >` **DAlbumIDs**  
*List of `Album` ids.*
- typedef `QMap< QString, QVariant >` **DInfoMap**  
*Map of properties name and value.*
- enum **SetupPage** { `ExifToolPage` = 0 , `ImageQualityPage` }

#### Public Slots inherited from `Digikam::DBInfolface`

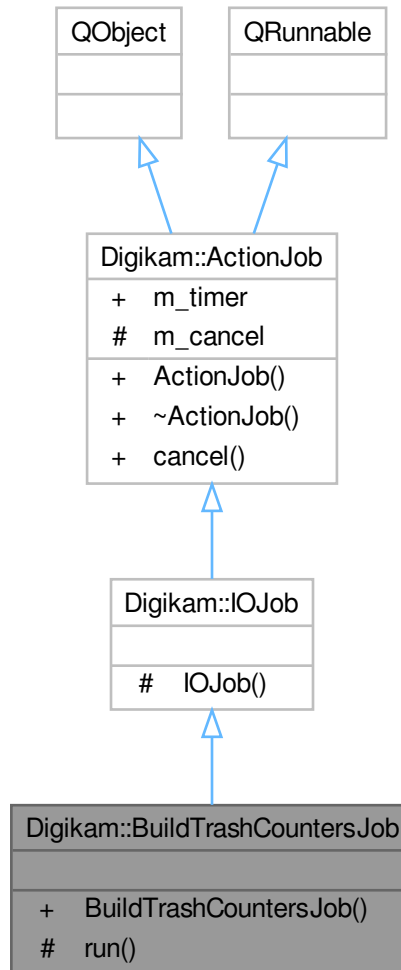
- void **slotDateTimeForUrl** (const `QUrl` &url, const `QDateTime` &dt, bool updModDate) override
- void **slotMetadataChangedForUrl** (const `QUrl` &url) override

#### Public Attributes inherited from `Digikam::DInfoInterface`

- bool **forceAlbumSelection** = false

## 9.135 Digikam::BuildTrashCountersJob Class Reference

Inheritance diagram for Digikam::BuildTrashCountersJob:



### Signals

- void **signalTrashCountersMap** (const QMap< QString, int > &counterMap)

### Signals inherited from [Digikam::IOJob](#)

- void **signalError** (const QString &errMsg)
- void **signalOneProcessed** (const QUrl &url)

## Signals inherited from [Digikam::ActionJob](#)

- void **signalDone** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job is done.*
- void **signalProgress** (int)  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job progress.*
- void **signalStarted** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager that job is started.*

## Protected Member Functions

- void **run** () override

## Additional Inherited Members

## Public Slots inherited from [Digikam::ActionJob](#)

- void **cancel** ()  
*Call this method to cancel job.*

## Public Member Functions inherited from [Digikam::ActionJob](#)

- **ActionJob** (QObject \*const parent=nullptr)  
*Constructor which delegate deletion of QRunnable instance to [ActionThreadBase](#), not QThreadPool.*
- **~ActionJob** () override  
*Re-implement destructor in you implementation.*

## Public Attributes inherited from [Digikam::ActionJob](#)

- QElapsedTimer **m\_timer**  
*Timer to determine the running time of the job.*

## Protected Attributes inherited from [Digikam::ActionJob](#)

- bool **m\_cancel** = false  
*You can use this boolean in your implementation to know if job must be canceled.*

## 9.136 Digikam::BWSepiaContainer Class Reference

### Public Types

- enum [BlackWhiteConversionType](#) {  
[BWNoFilter](#) = 0 , [BWGreenFilter](#) , [BWSepiaFilter](#) , [BWOrangeFilter](#) , [BWRedFilter](#) ,  
[BWYellowFilter](#) , [BWYellowGreenFilter](#) , [BWBlueFilter](#) , [BWGeneric](#) ,  
[BWAgfa200X](#) , [BWAgfapan25](#) , [BWAgfapan100](#) , [BWAgfapan400](#) ,  
[BWIlfordDelta100](#) , [BWIlfordDelta400](#) , [BWIlfordDelta400Pro3200](#) , [BWIlfordFP4](#) ,  
[BWIlfordHP5](#) , [BWIlfordPanF](#) , [BWIlfordXP2Super](#) , [BWKodakTmax100](#) ,  
[BWKodakTmax400](#) , [BWKodakTriX](#) , [BWIlfordSFX200](#) , [BWIlfordSFX400](#) ,  
[BWIlfordSFX800](#) , [BWNoTone](#) , [BWSepiaTone](#) , [BWBrownTone](#) ,  
[BWColdTone](#) , [BWSeleniumTone](#) , [BWPlatinumTone](#) , [BWGreenTone](#) ,  
[BWKodakHIE](#) }

## Public Member Functions

- **BWSepiaContainer** (int ptype)
- **BWSepiaContainer** (int ptype, const [CurvesContainer](#) &container)

## Public Attributes

- [BCGContainer](#) **bcgPrm**
- [CurvesContainer](#) **curvesPrm**
- int **filmType** = [BWGeneric](#)
- int **filterType** = [BWNoFilter](#)
- bool **preview** = false
- int **previewType** = [BWGeneric](#)
- double **strength** = 1.0
- int **toneType** = [BWNoTone](#)

## 9.136.1 Member Enumeration Documentation

### 9.136.1.1 BlackWhiteConversionType

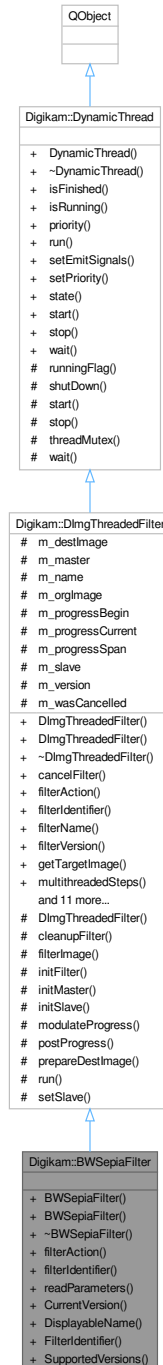
enum [Digikam::BWSepiaContainer::BlackWhiteConversionType](#)

#### Enumerator

BWNoFilter	B&W filter to the front of lens.
BWGeneric	B&W film simulation.
BWIlfordSFX200	Infrared film simulation.
BWNoTone	Chemical color tone filter.
BWKodakHIE	Infrared film simulation.

## 9.137 Digikam::BWSepiaFilter Class Reference

Inheritance diagram for Digikam::BWSepiaFilter:



### Public Member Functions

- **BWSepiaFilter** (`DImg *origImage`, `QObject *const parent=nullptr`, `const BWSepiaContainer &settings=BWSepiaContainer()`)
- **BWSepiaFilter** (`QObject *const parent=nullptr`)

- [FilterAction filterAction \(\)](#) override  
*Returns the action description corresponding to currently set options.*
- [QString filterIdentifier \(\)](#) const override  
*Return the identifier for this filter in the image history.*
- void [readParameters \(const FilterAction &action\)](#) override

## Public Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter \(DImg \\*const orgImage, QObject \\*const parent, const QString &name=QString\(\)\)](#)  
*Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter \(QObject \\*const parent=nullptr, const QString &name=QString\(\)\)](#)  
*Constructs a filter without argument.*
- virtual void [cancelFilter \(\)](#)  
*Cancel the threaded computation.*
- const [QString &filterName \(\)](#)
- int [filterVersion \(\)](#) const
- [DImg getTargetImage \(\)](#)
- [QList< int > multithreadedSteps \(int stop, int start=0\)](#) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool [parametersSuccessfullyRead \(\)](#) const  
*Optional: error handling for readParameters.*
- virtual [QString readParametersError \(const FilterAction &actionThatFailed\)](#) const
- void [setFilterName \(const QString &name\)](#)
- void [setFilterVersion \(int version\)](#)  
*Replaying a filter action: Set the filter version.*
- void [setOriginalImage \(const DImg &orgImage\)](#)
- void [setupAndStartDirectly \(const DImg &orgImage, DImgThreadedFilter \\*const master, int progress←Begin=0, int progressEnd=100\)](#)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void [setupFilter \(const DImg &orgImage\)](#)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void [startFilter \(\)](#)  
*Start the threaded computation.*
- virtual void [startFilterDirectly \(\)](#)  
*Start computation of this filter, directly in this thread.*
- virtual [QList< int > supportedVersions \(\)](#) const

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread \(QObject \\*const parent=nullptr\)](#)  
*This class extends QRunnable, so you have to reimplement virtual void [run\(\)](#).*
- [~DynamicThread \(\)](#) override  
*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool [isFinished \(\)](#) const
- bool [isRunning \(\)](#) const
- [QThread::Priority priority \(\)](#) const
- void [setEmitSignals \(bool emitThem\)](#)
- void [setPriority \(QThread::Priority priority\)](#)  
*Sets the priority for this dynamic thread.*
- State [state \(\)](#) const



### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

### Protected Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

### Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

### Protected Attributes inherited from [Digikam::DImgThreadedFilter](#)

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.137.1 Member Function Documentation

### 9.137.1.1 filterAction()

`FilterAction` Digikam::BWSepiaFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.137.1.2 filterIdentifier()

`QString` Digikam::BWSepiaFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

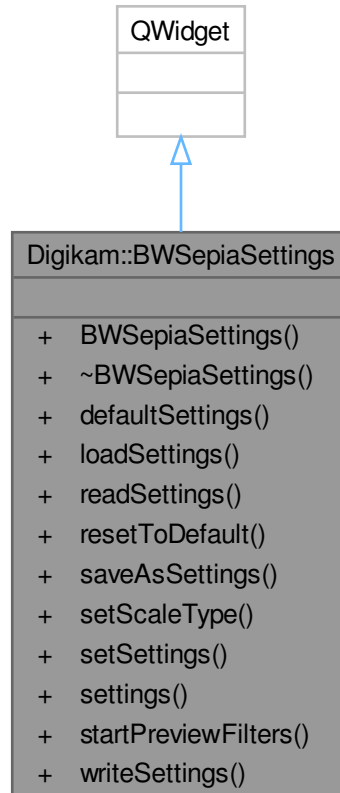
### 9.137.1.3 readParameters()

`void` Digikam::BWSepiaFilter::readParameters (   
     const `FilterAction` & action ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

## 9.138 Digikam::BWSepiaSettings Class Reference

Inheritance diagram for Digikam::BWSepiaSettings:



## Signals

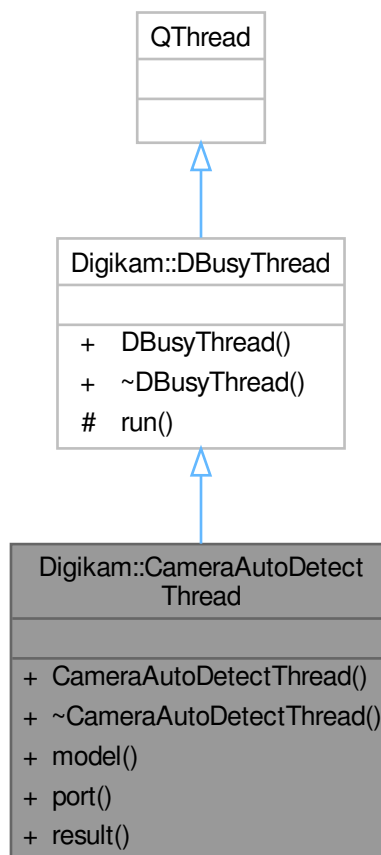
- void **signalSettingsChanged** ()

## Public Member Functions

- **BWSepiaSettings** (QWidget \*const parent, [DImg](#) \*const img)
- [BWSepiaContainer](#) **defaultSettings** () const
- void **loadSettings** ()
- void **readSettings** (KConfigGroup &group)
- void **resetToDefault** ()
- void **saveAsSettings** ()
- void **setScaleType** ([HistogramScale](#) scale)
- void **setSettings** (const [BWSepiaContainer](#) &settings)
- [BWSepiaContainer](#) **settings** () const
- void **startPreviewFilters** ()
- void **writeSettings** (KConfigGroup &group)

## 9.139 Digikam::CameraAutoDetectThread Class Reference

Inheritance diagram for Digikam::CameraAutoDetectThread:



### Public Member Functions

- **CameraAutoDetectThread** (QObject \*const parent)
- QString **model** () const
- QString **port** () const
- int **result** () const

### Public Member Functions inherited from [Digikam::DBusyThread](#)

- **DBusyThread** (QObject \*const parent)

### Additional Inherited Members

### Signals inherited from [Digikam::DBusyThread](#)

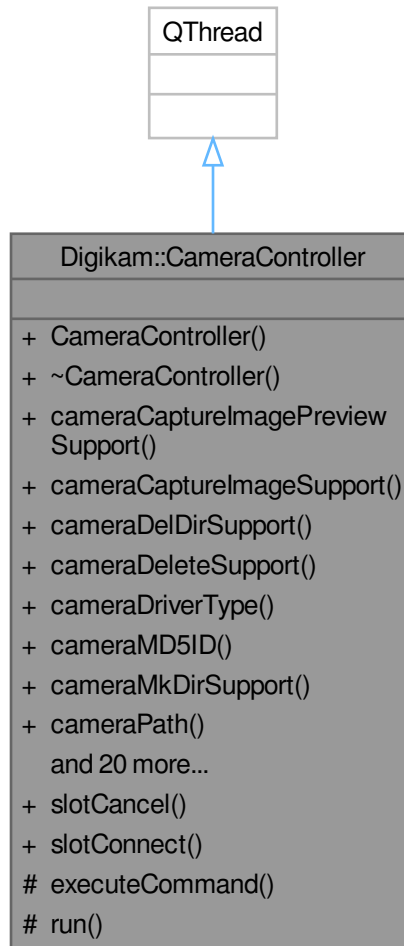
- void **signalComplete** ()

### Protected Member Functions inherited from [Digikam::DBusyThread](#)

- void **run** () override  
*Reimplement this method with your code to run in a separate thread.*

## 9.140 Digikam::CameraController Class Reference

Inheritance diagram for Digikam::CameraController:



### Public Slots

- void **slotCancel** ()
- void **slotConnect** ()

### Signals

- void **signalBusy** (bool val)
- void **signalCameraInformation** (const QString &summary, const QString &>manual, const QString &about)
- void **signalConnected** (bool val)
- void **signalDeleted** (const QString &folder, const QString &file, bool status)
- void **signalDownloaded** (const QString &folder, const QString &file, const QString &temp, int status)

- void **signalFileList** (const CamItemInfoList &infoList)
- void **signalFolderList** (const QStringList &folderList)
- void **signalFreeSpace** (qint64 bytesSize, qint64 bytesAvail)
- void **signalInternalDeleteFailed** (const QString &folder, const QString &file)
- void **signalInternalDownloadFailed** (const QString &folder, const QString &file)
- void **signalInternalLockFailed** (const QString &folder, const QString &file)
- void **signalInternalUploadFailed** (const QString &folder, const QString &file, const QString &src)
- void **signalLocked** (const QString &folder, const QString &file, bool status)
- void **signalLogMsg** (const QString &msg, DHistoryView::EntryType type, const QString &folder, const QString &file)
- void **signalMetadata** (const QString &folder, const QString &file, const [MetaEngineData](#) &exifData)
- void **signalPreview** (const QImage &preview)
- void **signalThumbInfo** (const QString &folder, const QString &file, const [CamItemInfo](#) &itemInfo, const QImage &thumb)
- void **signalThumbInfoFailed** (const QString &folder, const QString &file, const [CamItemInfo](#) &itemInfo)
- void **signalUploaded** (const [CamItemInfo](#) &itemInfo)

### Public Member Functions

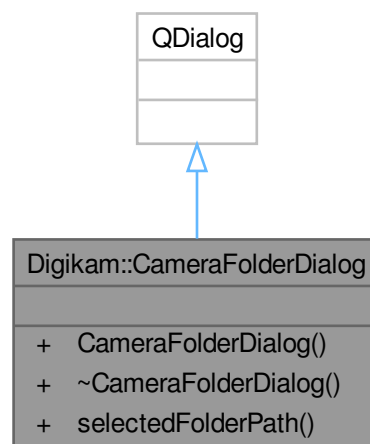
- **CameraController** (QWidget \*const parent, const QString &title, const QString &model, const QString &port, const QString &path)
- bool **cameraCaptureImagePreviewSupport** () const
- bool **cameraCaptureImageSupport** () const
- bool **cameraDelDirSupport** () const
- bool **cameraDeleteSupport** () const
- DKCamera::CameraDriverType **cameraDriverType** () const
- QByteArray **cameraMD5ID** () const
- bool **cameraMkDirSupport** () const
- QString **cameraPath** () const
- bool **cameraThumbnailSupport** () const
- QString **cameraTitle** () const
- bool **cameraUploadSupport** () const
- void **capture** ()
- void **deleteFile** (const QString &folder, const QString &file)
- void **download** (const [DownloadSettings](#) &downloadSettings)
- void **download** (const DownloadSettingsList &list)
- void **getCameraInformation** ()
- void **getFreeSpace** ()
- void **getMetadata** (const QString &folder, const QString &file)
- void **getPreview** ()
- CameraCommand \* **getThumbsInfo** (const CamItemInfoList &infoList, int thumbSize)  
*Get thumbnails for a list of camera items plus advanced information from metadata.*
- void **listFiles** (const QString &folder, bool useMetadata)
- void **listFolders** (const QString &folder=QString())
- void **listRootFolder** (bool useMetadata)
- void **lockFile** (const QString &folder, const QString &file, bool lock)
- QIcon **mimeThumbnail** (const QString &itemName) const
- void **moveThumbsInfo** (CameraCommand \*const cmd)
- void **openFile** (const QString &folder, const QString &file)
- void **upload** (const QFileInfo &srcFileInfo, const QString &destFile, const QString &destFolder)

### Protected Member Functions

- void **executeCommand** (CameraCommand \*const cmd)
- void **run** () override

## 9.141 Digikam::CameraFolderDialog Class Reference

Inheritance diagram for Digikam::CameraFolderDialog:



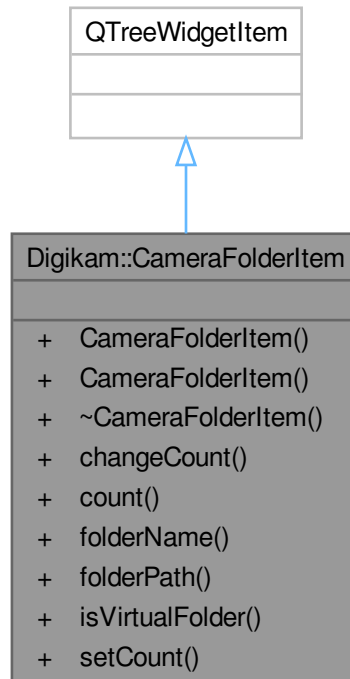
### Public Member Functions

- **CameraFolderDialog** (QWidget \*const parent, const QMap< QString, int > &map, const QString &cameraName, const QString &rootPath)
- QString **selectedFolderPath** () const



## 9.142 Digikam::CameraFolderItem Class Reference

Inheritance diagram for Digikam::CameraFolderItem:

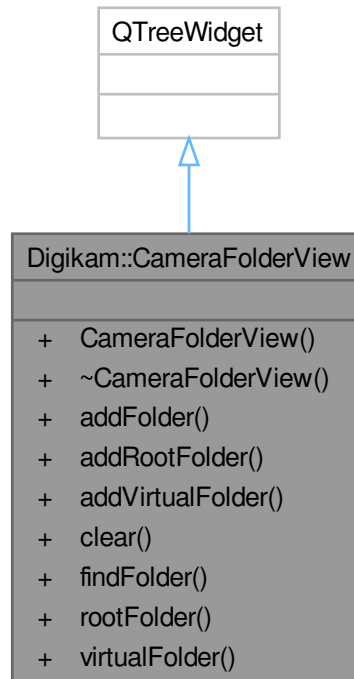


### Public Member Functions

- **CameraFolderItem** (`QTreeWidgetItem *const parent`, `const QString &name`, `const QIcon &icon=QIcon::fromTheme(QLatin1String("folder"))`)
- **CameraFolderItem** (`QTreeWidgetItem *const parent`, `const QString &folderName`, `const QString &folderPath`, `const QIcon &icon=QIcon::fromTheme(QLatin1String("folder"))`)
- void **changeCount** (`int val`)
- int **count** () const
- QString **folderName** () const
- QString **folderPath** () const
- bool **isVirtualFolder** () const
- void **setCount** (`int val`)

## 9.143 Digikam::CameraFolderView Class Reference

Inheritance diagram for Digikam::CameraFolderView:



### Signals

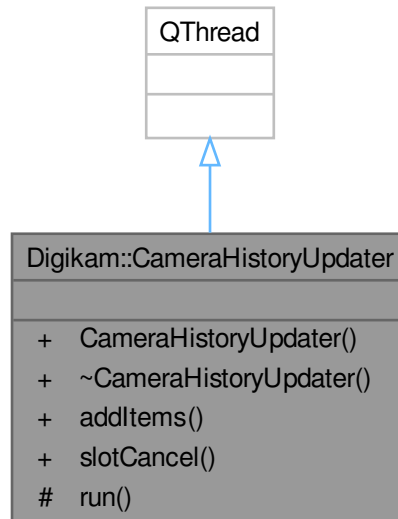
- void **signalCleared** ()
- void **signalFolderChanged** ([CameraFolderItem](#) \*)

### Public Member Functions

- **CameraFolderView** (QWidget \*const parent)
- [CameraFolderItem](#) \* **addFolder** (const QString &folder, const QString &subFolder, int nbltems, const QIcon &icon=QIcon::fromTheme(QLatin1String("folder")))
- void **addRootFolder** (const QString &folder, int nbltems=-1, const QIcon &icon=QIcon::fromTheme(QLatin1String("folder")))
- void **addVirtualFolder** (const QString &name, const QIcon &icon=QIcon::fromTheme(QLatin1String("camera-photo")))
- virtual void **clear** ()
- [CameraFolderItem](#) \* **findFolder** (const QString &folderPath)
- [CameraFolderItem](#) \* **rootFolder** () const
- [CameraFolderItem](#) \* **virtualFolder** () const

## 9.144 Digikam::CameraHistoryUpdater Class Reference

Inheritance diagram for Digikam::CameraHistoryUpdater:



### Public Slots

- void **slotCancel** ()

### Signals

- void **signalBusy** (bool val)
- void **signalHistoryMap** (const CHUpdateItemMap &)

### Public Member Functions

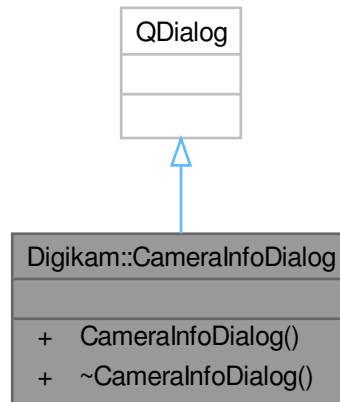
- **CameraHistoryUpdater** (QWidget \*const parent)
- void **addItem** (const QByteArray &id, CHUpdateItemMap &map)

### Protected Member Functions

- void **run** ()

## 9.145 Digikam::CameraInfoDialog Class Reference

Inheritance diagram for Digikam::CameraInfoDialog:

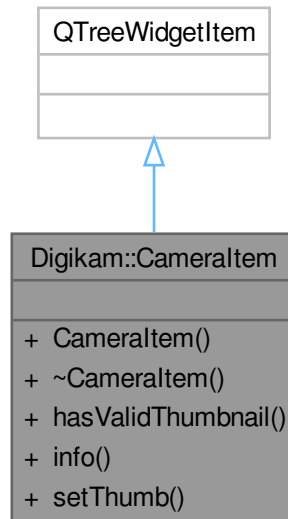


### Public Member Functions

- **CameraInfoDialog** (`QWidget *const parent, const QString &summary, const QString &>manual, const QString &about`)

## 9.146 Digikam::Cameratem Class Reference

Inheritance diagram for Digikam::Cameratem:

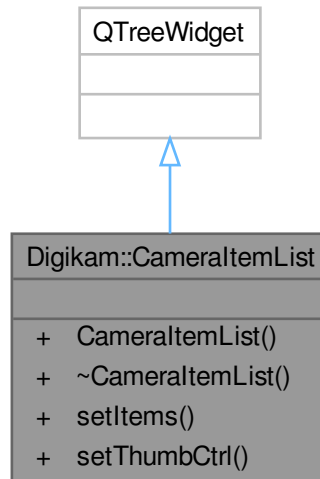


### Public Member Functions

- **Cameratem** (`QTreeWidgetItem *const parent, const CamltemInfo &info`)
- `bool hasValidThumbnail () const`
- `CamltemInfo info () const`
- `void setThumb (const QPixmap &pix, bool hasThumb=true)`

## 9.147 Digikam::CameraItemList Class Reference

Inheritance diagram for Digikam::CameraItemList:

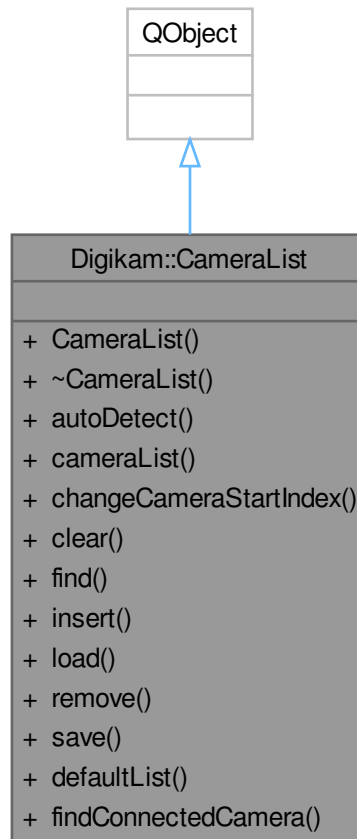


### Public Member Functions

- **CameraItemList** (`QWidget *const parent=nullptr`)
- void **setItems** (`const CamItemInfoList &items`)
- void **setThumbCtrl** (`CameraThumbsCtrl *const ctrl`)

## 9.148 Digikam::CameraList Class Reference

Inheritance diagram for Digikam::CameraList:



### Signals

- void **signalCameraAdded** ([CameraType](#) \*)
- void **signalCameraRemoved** ([QAction](#) \*)

### Public Member Functions

- **CameraList** ([QObject](#) \*const parent, const [QString](#) &file)
- [CameraType](#) \* **autoDetect** (bool &retry)
- [QList](#)< [CameraType](#) \* > \* **cameraList** () const
- bool **changeCameraStartIndex** (const [QString](#) &cameraTitle, int startIndex)
- void **clear** ()
- [CameraType](#) \* **find** (const [QString](#) &title) const
- void **insert** ([CameraType](#) \*const ctype)
- bool **load** ()
- void **remove** ([CameraType](#) \*const ctype)
- bool **save** ()

### Static Public Member Functions

- static [CameraList](#) \* **defaultList** ()
- static bool **findConnectedCamera** (int vendorId, int productId, QString &model, QString &port)

## 9.149 Digikam::CameraMessageBox Class Reference

### Static Public Member Functions

- static void **informationList** ([CameraThumbsCtrl](#) \*const ctrl, QWidget \*const parent, const QString &caption, const QString &text, const CamItemInfoList &items, const QString &dontShowAgainName=QString())  
*Show List of camera items into an informative message box.*
- static int **warningContinueCancelList** ([CameraThumbsCtrl](#) \*const ctrl, QWidget \*const parent, const QString &caption, const QString &text, const CamItemInfoList &items, const QString &dontAskAgainName=QString())  
*Show List of camera items to process into a message box and wait user feedback.*

### 9.149.1 Member Function Documentation

#### 9.149.1.1 warningContinueCancelList()

```
int Digikam::CameraMessageBox::warningContinueCancelList (
    CameraThumbsCtrl *const ctrl,
    QWidget *const parent,
    const QString & caption,
    const QString & text,
    const CamItemInfoList & items,
    const QString & dontAskAgainName = QString() ) [static]
```

Return QMessageBox::Yes or QMessageBox::Cancel

## 9.150 Digikam::CameraNameHelper Class Reference

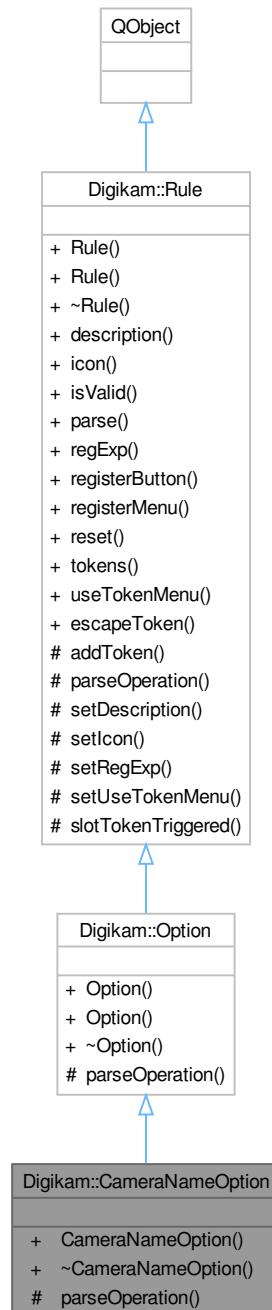
### Static Public Member Functions

- static QString **cameraName** (const QString &name)
- static QString **cameraNameAutoDetected** (const QString &name)
- static QString **createCameraName** (const QString &vendor, const QString &product=QString(), const QString &mode=QString(), bool autoDetected=false)
- static bool **sameDevices** (const QString &deviceA, const QString &deviceB)



## 9.151 Digikam::CameraNameOption Class Reference

Inheritance diagram for Digikam::CameraNameOption:



### Protected Member Functions

- `QString parseOperation (ParseSettings &settings, const QRegularExpressionMatch &match)` override  
*TODO: describe me.*

## Protected Member Functions inherited from [Digikam::Rule](#)

- bool [addToken](#) (const QString &id, const QString &description, const QString &actionName=QString())  
*add a token to the parser, every parser should at least assign one token object*
- void [setDescription](#) (const QString &desc)
- void [setIcon](#) (const QString &pixmap)
- void [setRegExp](#) (const QRegularExpression &regExp)
- void [setUseTokenMenu](#) (bool value)  
*If multiple tokens have been assigned to a rule, a menu will be created.*

## Additional Inherited Members

## Public Types inherited from [Digikam::Rule](#)

- enum [IconType](#) { [Action](#) = 0 , [Dialog](#) }

## Signals inherited from [Digikam::Rule](#)

- void [signalTokenTriggered](#) (const QString &)

## Public Member Functions inherited from [Digikam::Option](#)

- [Option](#) (const QString &name, const QString &description)
- [Option](#) (const QString &name, const QString &description, const QString &icon)

## Public Member Functions inherited from [Digikam::Rule](#)

- [Rule](#) (const QString &name)
- [Rule](#) (const QString &name, const QString &icon)
- QString [description](#) () const
- QPixmap [icon](#) (Rule::IconType type=Rule::Action) const
- bool [isValid](#) () const  
*Checks the validity of the parse object.*
- [ParseResults](#) [parse](#) ([ParseSettings](#) &settings)
- QRegularExpression & [regExp](#) () const  
*TODO: This is probably not needed anymore.*
- QPushButton \* [registerButton](#) (QWidget \*parent)  
*Register a button in the parent object.*
- QAction \* [registerMenu](#) (QMenu \*parent)  
*Register a menu action in the parent object.*
- virtual void [reset](#) ()  
*Resets the parser to its initial state.*
- TokenList & [tokens](#) () const
- bool [useTokenMenu](#) () const  
*Returns true if a token menu is used.*

## Static Public Member Functions inherited from [Digikam::Rule](#)

- static QString [escapeToken](#) (const QString &token)  
*Escape the token characters to make them work in regular expressions.*

## Protected Slots inherited from [Digikam::Rule](#)

- virtual void [slotTokenTriggered](#) (const QString &)

### 9.151.1 Member Function Documentation

#### 9.151.1.1 [parseOperation\(\)](#)

```
QString Digikam::CameraNameOption::parseOperation (
    ParseSettings & settings,
    const QRegularExpressionMatch & match ) [override], [protected], [virtual]
```

##### Parameters

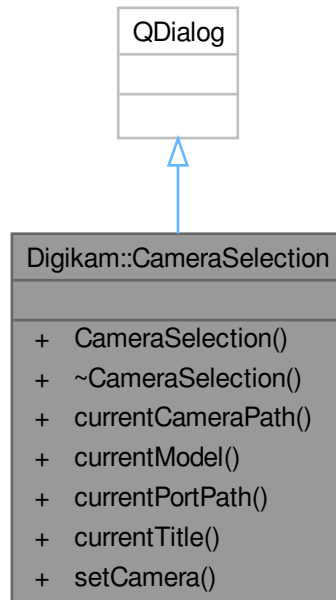
<i>settings</i>	contains settings
<i>match</i>	result of the regular expression match done in <a href="#">Option::parse()</a>

##### Returns

Implements [Digikam::Option](#).

## 9.152 Digikam::CameraSelection Class Reference

Inheritance diagram for Digikam::CameraSelection:



### Signals

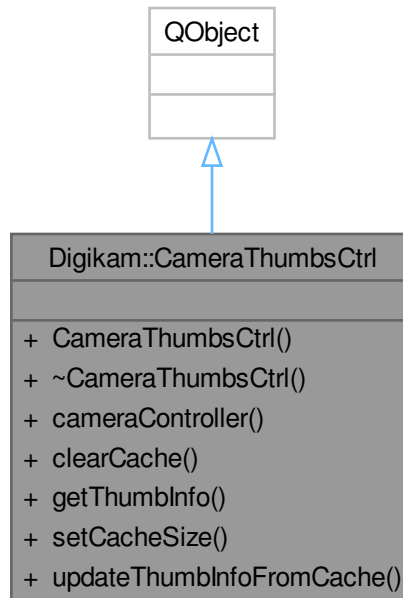
- void **signalOkClicked** (const QString &title, const QString &model, const QString &port, const QString &path)

### Public Member Functions

- **CameraSelection** (QWidget \*const parent=nullptr)
- QString **currentCameraPath** () const
- QString **currentModel** () const
- QString **currentPortPath** () const
- QString **currentTitle** () const
- void **setCamera** (const QString &title, const QString &model, const QString &port, const QString &path)

## 9.153 Digikam::CameraThumbsCtrl Class Reference

Inheritance diagram for Digikam::CameraThumbsCtrl:



### Signals

- void **signalThumbInfoReady** (const [CamItemInfo](#) &)

### Public Member Functions

- **CameraThumbsCtrl** ([CameraController](#) \*const ctrl, QWidget \*const parent)
- [CameraController](#) \* **cameraController** () const  
*Return camera controller instance.*
- void **clearCache** ()
- bool **getThumbInfo** (const [CamItemInfo](#) &info, CachedItem &item) const  
*Fill item with relevant information.*
- void **setCacheSize** (int numberOfItems)
- void **updateThumbInfoFromCache** (const [CamItemInfo](#) &info)  
*Force controller to update info from device in cache.*

### 9.153.1 Member Function Documentation

#### 9.153.1.1 getThumbInfo()

```
bool Digikam::CameraThumbsCtrl::getThumbInfo (
    const CamItemInfo & info,
    CachedItem & item ) const
```

if item is not in cache, return false and information will be dispatched later through signalThumbInfoReady(), else return true and information is available immediately.

## 9.154 Digikam::CameraType Class Reference

### Public Member Functions

- **CameraType** (const [CameraType](#) &ctype)
- **CameraType** (const QString &title, const QString &model, const QString &port, const QString &path, int startingNumber, QAction \*const action=nullptr)
- QAction \* **action** () const
- [ImportUI](#) \* **currentImportUI** () const
- QString **model** () const
- [CameraType](#) & **operator=** (const [CameraType](#) &type)
- QString **path** () const
- QString **port** () const
- void **setAction** (QAction \*const action)
- void **setCurrentImportUI** ([ImportUI](#) \*const importui)
- void **setModel** (const QString &model)
- void **setPath** (const QString &path)
- void **setPort** (const QString &port)
- void **setStartingNumber** (int sn)
- void **setTitle** (const QString &title)
- void **setValid** (bool valid)
- int **startingNumber** () const
- QString **title** () const
- bool **valid** () const

## 9.155 Digikam::CamItemInfo Class Reference

### Public Types

- enum [DownloadStatus](#) {  
[DownloadUnknown](#) = -1 , [DownloadedNo](#) = 0 , [DownloadedYes](#) = 1 , [DownloadFailed](#) = 2 ,  
[DownloadStarted](#) = 3 , [NewPicture](#) = 4 }

### Public Member Functions

- bool **isNull** () const  
*Return true if all member in this container are null.*
- bool **operator!=** (const [CamItemInfo](#) &info) const  
*Compare for camera information un-equality, not including variable values.*
- bool **operator==** (const [CamItemInfo](#) &info) const  
*Compare for camera information equality, not including variable values.*
- QUrl **url** () const  
*Return the local file system (mounted on computer) url to the camera file.*

**Public Attributes**

- int **colorLabel** = NoColorLabel  
*Pre-picklabel value of camera file.*
- QDateTime **ctime**  
*Created time stamp of camera file.*
- int **downloaded** = DownloadUnknown  
*Variable values depending of user actions.*
- QString **downloadName**  
*Preview of the file-name to use during download from camera.*
- QString **folder**  
*Folder path to access to file in camera.*
- int **height** = -1  
*Image height in pixels.*
- qlonglong **id** = -1  
*Unique image id.*
- QString **mime**  
*Type mime of camera file.*
- QString **name**  
*File name in camera file-system.*
- [PhotoInfoContainer](#) **photoInfo**  
*Photo Info from camera file (get from file metadata)*
- int **pickLabel** = NoPickLabel  
*Pre-picklabel value of camera file.*
- bool **previewPossible** = false
- int **rating** = NoRating  
*Pre-rating value of camera file.*
- int **readPermissions** = -1  
*Read permission of camera file.*
- qint64 **size** = -1  
*Static values taken from camera.*
- QList< int > **tagIds**  
*Pre-tags ids of camera file.*
- int **width** = -1  
*Image width in pixels.*
- int **writePermissions** = -1  
*Write permission of camera file.*

**9.155.1 Member Enumeration Documentation****9.155.1.1 DownloadStatus**

```
enum Digikam::CamItemInfo::DownloadStatus
```

**Enumerator**

DownloadUnknown	Download state is unknown.
DownloadedNo	Is not yet downloaded on computer.
DownloadedYes	Is already downloaded on computer.
DownloadFailed	Download is failed or have been aborted by user.
DownloadStarted	Download is under progress.

## 9.155.2 Member Data Documentation

### 9.155.2.1 downloaded

```
int Digikam::CamItemInfo::downloaded = DownloadUnknown
```

Download status of camera file. See DownloadStatus enum for details

### 9.155.2.2 size

```
qint64 Digikam::CamItemInfo::size = -1
```

Camera file size in bytes.

## 9.156 Digikam::CamItemSortSettings Class Reference

### Public Types

- enum **CategorizationMode** { **NoCategories** , **CategoryByFolder** , **CategoryByFormat** , **CategoryByDate** }
- enum **SortOrder** { **AscendingOrder** = Qt::AscendingOrder , **DescendingOrder** = Qt::DescendingOrder , **DefaultOrder** }
- enum **SortRole** { **SortByFileName** , **SortByFilePath** , **SortByCreationDate** , **SortByFileSize** , **SortByDownloadState** , **SortByRating** }

### Public Member Functions

- int **compare** (const [CamItemInfo](#) &left, const [CamItemInfo](#) &right) const  
*Compares the camItemInfos left and right.*
- int **compare** (const [CamItemInfo](#) &left, const [CamItemInfo](#) &right, SortRole sortRole) const
- int **compareCategories** (const [CamItemInfo](#) &left, const [CamItemInfo](#) &right) const  
*Compares the categories of left and right camItemInfos.*
- bool **isCategorized** () const
- bool **lessThan** (const [CamItemInfo](#) &left, const [CamItemInfo](#) &right) const  
*Returns true if left is less than right.*
- bool **lessThan** (const QVariant &left, const QVariant &right) const  
*Returns true if left QVariant is less than right.*
- bool **operator==** (const [CamItemSortSettings](#) &other) const
- void **setCategorizationMode** (CategorizationMode mode)
- void **setCategorizationSortOrder** ([SortOrder](#) order)
- void **setSortOrder** ([SortOrder](#) order)
- void **setSortRole** (SortRole role)
- void **setStringTypeNatural** (bool natural)



## Static Public Member Functions

- `template<typename T >`  
`static int compareByOrder (const T &a, const T &b, Qt::SortOrder sortOrder)`
- `static int compareByOrder (int compareResult, Qt::SortOrder sortOrder)`  
*Takes a typical result from a compare method (0 is equal, -1 is less than, 1 is greater than) and applies the given sort order to it.*
- `template<typename T >`  
`static int compareValue (const T &a, const T &b)`  
*Returns the usual compare result of -1, 0, or 1 for lessThan, equals and greaterThan.*
- `static Qt::SortOrder defaultSortOrderForCategorizationMode (CategorizationMode mode)`
- `static Qt::SortOrder defaultSortOrderForSortRole (SortRole role)`
- `template<typename T >`  
`static bool lessThanByOrder (const T &a, const T &b, Qt::SortOrder sortOrder)`  
*Returns  $a < b$  if `sortOrder` is Ascending, or  $b < a$  if order is descending.*
- `static int naturalCompare (const QString &a, const QString &b, Qt::SortOrder sortOrder, Qt::CaseSensitivity caseSensitive=Qt::CaseSensitive, bool natural=true)`  
*Compares the two string by natural comparison and adheres to given sort order.*

## Public Attributes

- `Qt::CaseSensitivity categorizationCaseSensitivity = Qt::CaseSensitive`
- `CategorizationMode categorizationMode = NoCategories`
- `SortOrder categorizationSortOrder = DefaultOrder`
- `Qt::SortOrder currentCategorizationSortOrder = Qt::AscendingOrder`  
*Only Ascending or Descending, never be DefaultOrder.*
- `Qt::SortOrder currentSortOrder = Qt::AscendingOrder`
- `Qt::CaseSensitivity sortCaseSensitivity = Qt::CaseSensitive`
- `SortOrder sortOrder = DefaultOrder`  
*Camera Items Sorting.*
- `SortRole sortRole = SortByFileName`
- `bool strTypeNatural = true`

## 9.156.1 Member Enumeration Documentation

### 9.156.1.1 SortOrder

```
enum Digikam::CamItemSortSettings::SortOrder
```

#### Enumerator

DefaultOrder	sort order depends on the chosen sort role
--------------	--

## 9.156.2 Member Function Documentation

### 9.156.2.1 compare()

```
int Digikam::CamItemSortSettings::compare (
    const CamItemInfo & left,
    const CamItemInfo & right ) const
```

Return -1 if left is less than right, 1 if left is greater than right, and 0 if left equals right comparing the current sort role's value. Adheres to set sort role and sort order.

### 9.156.2.2 compareCategories()

```
int Digikam::CamItemSortSettings::compareCategories (
    const CamItemInfo & left,
    const CamItemInfo & right ) const
```

It returns -1 if the left camItemInfo is less than right, and 0 if both fall in the same category, and 1 if the left camItemInfo is greater than right. Adheres to set categorization mode and current category sort order.

### 9.156.2.3 lessThan() [1/2]

```
bool Digikam::CamItemSortSettings::lessThan (
    const CamItemInfo & left,
    const CamItemInfo & right ) const
```

Adheres to current sort role and sort order.

### 9.156.2.4 lessThan() [2/2]

```
bool Digikam::CamItemSortSettings::lessThan (
    const QVariant & left,
    const QVariant & right ) const
```

Adheres to current sort role and sort order.

## 9.157 Digikam::Canvas Class Reference

Inheritance diagram for Digikam::Canvas:



### Public Slots

- void **slotCopy** ()
- void **slotCrop** ()

- void **slotFlipHoriz** ()
- void **slotFlipVert** ()
- void **slotRedo** (int steps=1)
- void **slotRestore** ()
- void **slotRotate180** ()
- void **slotRotate270** ()
- void **slotRotate90** ()
- image modifiers*
- void **slotSelectAll** ()
- void **slotSelected** ()
- void **slotSelectionMoved** ()
- void **slotSelectNone** ()
- void **slotUndo** (int steps=1)

## Signals

- void **signalAddedDroppedItems** (QDropEvent \*)
- void **signalChanged** ()
- void **signalLoadingFinished** (const QString &filename, bool success)
- void **signalLoadingProgress** (const QString &filePath, float progress)
- void **signalLoadingStarted** (const QString &filename)
- void **signalPrepareToLoad** ()
- void **signalRedoSteps** (int)
- void **signalRightButtonClicked** ()
- void **signalSavingFinished** (const QString &filename, bool success)
- void **signalSavingProgress** (const QString &filePath, float progress)
- void **signalSavingStarted** (const QString &filename)
- void **signalSelected** (bool)
- void **signalSelectionChanged** (const QRect &)
- void **signalSelectionSetText** (const QRect &)
- void **signalShowNextImage** ()
- void **signalShowPrevImage** ()
- void **signalToggleOffFitToWindow** ()
- void **signalUndoSteps** (int)
- void **signalZoomChanged** (double)

## Signals inherited from [Digikam::GraphicsDImgView](#)

- void **activated** ()
- void **contentsMoved** (bool panningFinished)
- void **contentsMoving** (int, int)
- void **leftButtonClicked** ()
- void **leftButtonDoubleClicked** ()
- void **resized** ()
- void **rightButtonClicked** ()
- void **toNextImage** ()
- void **toPreviousImage** ()
- void **viewportRectChanged** (const QRectF &viewportRect)

## Public Member Functions

- **Canvas** (QWidget \*const parent=nullptr)
- void **abortSaving** ()
- void **applyTransform** (const [IccTransform](#) &transform)  
*Apply Color Management transformation to image (typically working color space).*
- [DImg](#) **currentImage** () const  
*Return a copy of current image loaded in editor.*
- QString **currentImageFileFormat** () const  
*Return the type mime of current image loaded in editor.*
- QString **currentImageFilePath** () const  
*Return the file path of current image loaded in editor.*
- QString **ensureHasCurrentUuid** () const
- bool **exifRotated** () const  
*Return true if image have been rotated following Exif information.*
- void **fitToSelect** ()  
*Change zoom level to fit current selection on canvas size.*
- QRect **getSelectedArea** () const  
*Return the rectangle information of current canvas selection.*
- int **imageHeight** () const  
*Return the height of current image loaded in editor.*
- int **imageWidth** () const  
*Return the width of current image loaded in editor.*
- [EditorCore](#) \* **interface** () const  
*Return the core interface instance of editor.*
- bool **isReadOnly** () const  
*If current image file format is only available in read only, typically all RAW image file formats.*
- void **load** (const QString &filename, [IOFileSettings](#) \*const [IOFileSettings](#))
- void **makeDefaultEditingCanvas** ()
- void **preload** (const QString &filename)
- void **resetImage** ()
- void **setExifOrient** (bool exifOrient)  
*Rotate image following Exif information.*
- void **setExposureSettings** ([ExposureSettingsContainer](#) \*const expoSettings)  
*Apply under.over exposure indicator settings.*
- void **setICCSettings** (const [ICCSettingsContainer](#) &cmSettings)  
*Apply Color management settings (typically screen profile).*
- void **setModified** ()
- void **setSoftProofingEnabled** (bool enable)  
*Turn on/off Color Management Soft proofing mode.*

## Public Member Functions inherited from [Digikam::GraphicsDImgView](#)

- **GraphicsDImgView** (QWidget \*const parent=nullptr)
- int **contentsX** () const
- int **contentsY** () const
- void **drawText** (QPainter \*p, const QRectF &rect, const QString &text)
- void **fitToWindow** ()
- [GraphicsDImgItem](#) \* **item** () const  
*Return the instance of item set by [setItem\(\)](#).*
- [SinglePhotoPreviewLayout](#) \* **layout** () const

- **DImgPreviewItem \* previewItem () const**  
Return a cast of item instance of item set by [setItem\(\)](#) as *DImgPreviewItem* Note: if you store a *GraphicsDImgItem* object using [setItem\(\)](#), this method will return 0.
- void **scrollPointOnPoint** (const QPointF &scenePos, const QPoint &viewportPos)  
Scrolls the view such that scenePos (in scene coordinates is displayed on the viewport at viewportPos (in viewport coordinates).
- void **setContentPos** (int x, int y)
- void **setItem** (*GraphicsDImgItem* \*const item)  
Store internal instance of item as *GraphicsDImgItem*.
- void **toggleFullScreen** (bool set)
- QRect **visibleArea** () const

### Protected Member Functions

- void **addRubber** ()
- void **dragEnterEvent** (QDragEnterEvent \*) override
- void **dragMoveEvent** (QDragMoveEvent \*) override
- void **dropEvent** (QDropEvent \*) override
- void **keyPressEvent** (QKeyEvent \*) override
- void **mousePressEvent** (QMouseEvent \*) override

### Protected Member Functions inherited from [Digikam::GraphicsDImgView](#)

- virtual bool **acceptsMouseClicked** (QMouseEvent \*e)
- void **continuePanning** (const QPoint &pos)
- void **drawForeground** (QPainter \*painter, const QRectF &rect) override
- void **finishPanning** ()
- void **installPanIcon** ()
- void **mouseDoubleClickEvent** (QMouseEvent \*) override
- void **mouseMoveEvent** (QMouseEvent \*) override
- void **mousePressEvent** (QMouseEvent \*) override
- void **mouseReleaseEvent** (QMouseEvent \*) override
- void **resizeEvent** (QResizeEvent \*) override
- void **scrollContentsBy** (int dx, int dy) override
- void **setScaleFitToWindow** (bool value)
- void **setShowText** (bool value)
- void **startPanning** (const QPoint &pos)
- void **wheelEvent** (QWheelEvent \*) override

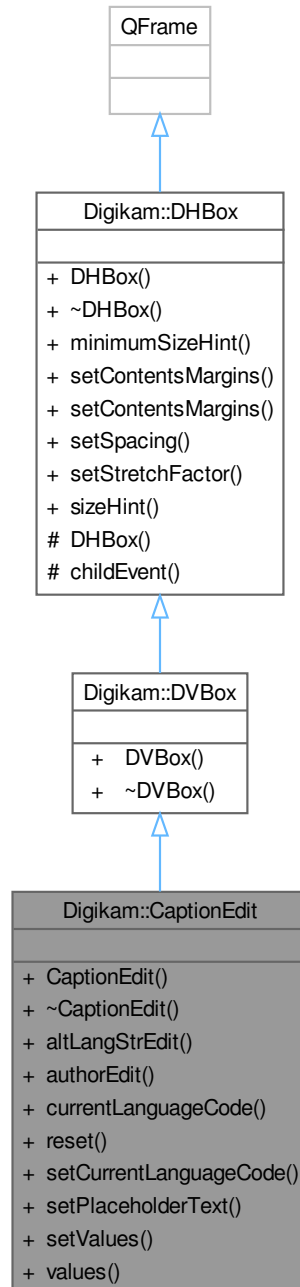
### Additional Inherited Members

### Protected Slots inherited from [Digikam::GraphicsDImgView](#)

- void **slotContentsMoved** ()
- void **slotCornerButtonPressed** ()
- void **slotPanIconHidden** ()
- virtual void **slotPanIconSelectionMoved** (const QRect &, bool)

## 9.158 Digikam::CaptionEdit Class Reference

Inheritance diagram for Digikam::CaptionEdit:



### Signals

- void **signalModified** ()

### Public Member Functions

- **CaptionEdit** (QWidget \*const parent)
- **AltLangStrEdit** \* **altLangStrEdit** () const
- QLineEdit \* **authorEdit** () const
- QString **currentLanguageCode** () const
- void **reset** ()
- void **setCurrentLanguageCode** (const QString &lang)
- void **setPlaceholderText** (const QString &msg)
- void **setValues** (const [CaptionsMap](#) &values)
- [CaptionsMap](#) & **values** () const

### Public Member Functions inherited from [Digikam::DVBox](#)

- **DVBox** (QWidget \*const parent=nullptr)

### Public Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (QWidget \*const parent=nullptr)
- QSize **minimumSizeHint** () const override
- void **setContentsMargins** (const QMargins &margins)
- void **setContentsMargins** (int left, int top, int right, int bottom)
- void **setSpacing** (int space)
- void **setStretchFactor** (QWidget \*const widget, int stretch)
- QSize **sizeHint** () const override

### Additional Inherited Members

### Protected Member Functions inherited from [Digikam::DHBox](#)

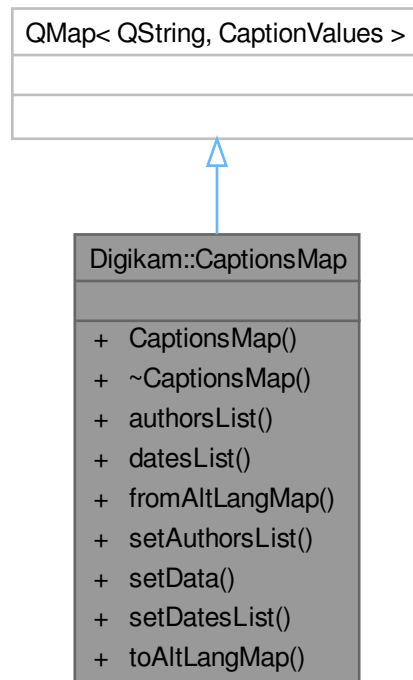
- **DHBox** (bool vertical, QWidget \*const parent)
- void **childEvent** (QChildEvent \*e) override

## 9.159 [Digikam::CaptionsMap](#) Class Reference

A map used to store a list of Alternative Language values + author and date properties The map key is the language code following RFC3066 notation (like "fr-FR" for French), and the [CaptionsMap](#) value all caption properties.



Inheritance diagram for Digikam::CaptionsMap:



## Public Member Functions

- [MetaEngine::AltLangMap](#) `authorsList` () const
- [MetaEngine::AltLangMap](#) `datesList` () const
- void `fromAltLangMap` (const [MetaEngine::AltLangMap](#) &map)
- void `setAuthorsList` (const [MetaEngine::AltLangMap](#) &map, const `QString` &commonAuthor=`QString()`)  
*Sets the author for the comments in the specified languages.*
- void `setData` (const [MetaEngine::AltLangMap](#) &comments, const [MetaEngine::AltLangMap](#) &authors, const `QString` &commonAuthor, const [MetaEngine::AltLangMap](#) &dates)
- void `setDatesList` (const [MetaEngine::AltLangMap](#) &map)
- [MetaEngine::AltLangMap](#) `toAltLangMap` () const

## 9.159.1 Member Function Documentation

### 9.159.1.1 setAuthorsList()

```

void Digikam::CaptionsMap::setAuthorsList (
    const MetaEngine::AltLangMap & map,
    const QString & commonAuthor = QString() )
  
```

If `commonAuthor` is not null, it will be used to set the author of all comments for which the author is not specified in the map.

## 9.160 Digikam::CaptionValues Class Reference

### Public Member Functions

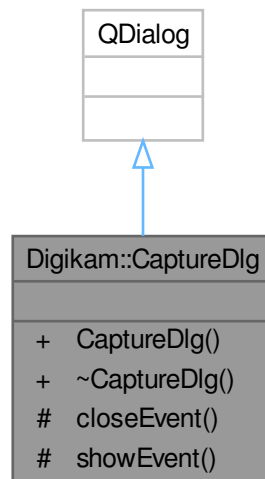
- bool **operator==** (const [CaptionValues](#) &val) const

### Public Attributes

- QString **author**
- QString **caption**
- QDateTime **date**

## 9.161 Digikam::CaptureDlg Class Reference

Inheritance diagram for Digikam::CaptureDlg:



### Public Member Functions

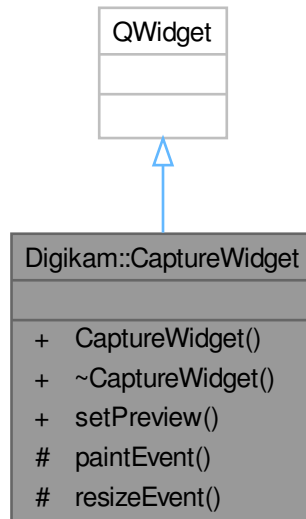
- **CaptureDlg** (QWidget \*const parent, [CameraController](#) \*const controller, const QString &cameraTitle)

### Protected Member Functions

- void **closeEvent** (QCloseEvent \*e) override
- void **showEvent** (QShowEvent \*e) override

## 9.162 Digikam::CaptureWidget Class Reference

Inheritance diagram for Digikam::CaptureWidget:



### Public Member Functions

- **CaptureWidget** (`QWidget *const parent=nullptr`)
- void **setPreview** (`const QImage &preview`)

### Protected Member Functions

- void **paintEvent** (`QPaintEvent *`) override
- void **resizeEvent** (`QResizeEvent *`) override

## 9.163 Digikam::CaseModifier Class Reference

Inheritance diagram for Digikam::CaseModifier:



### Public Member Functions

- QString [parseOperation](#) ([ParseSettings](#) &settings, const QRegularExpressionMatch &match) override  
*TODO: describe me.*

## Public Member Functions inherited from Digikam::Modifier

- **Modifier** (const QString &name, const QString &description)
- **Modifier** (const QString &name, const QString &description, const QString &icon)

## Public Member Functions inherited from Digikam::Rule

- **Rule** (const QString &name)
- **Rule** (const QString &name, const QString &icon)
- QString **description** () const
- QPixmap **icon** (Rule::IconType type=Rule::Action) const
- bool **isValid** () const

*Checks the validity of the parse object.*

- ParseResults **parse** (ParseSettings &settings)
- QRegularExpression & **regExp** () const
- QPushButton \* **registerButton** (QWidget \*parent)

*TODO: This is probably not needed anymore.*

*Register a button in the parent object.*

- QAction \* **registerMenu** (QMenu \*parent)

*Register a menu action in the parent object.*

- virtual void **reset** ()

*Resets the parser to its initial state.*

- TokenList & **tokens** () const
- bool **useTokenMenu** () const

*Returns true if a token menu is used.*

## Additional Inherited Members

## Public Types inherited from Digikam::Rule

- enum **IconType** { **Action** = 0 , **Dialog** }

## Signals inherited from Digikam::Rule

- void **signalTokenTriggered** (const QString &)

## Static Public Member Functions inherited from Digikam::Rule

- static QString **escapeToken** (const QString &token)  
*Escape the token characters to make them work in regular expressions.*

## Protected Slots inherited from Digikam::Rule

- virtual void **slotTokenTriggered** (const QString &)

## Protected Member Functions inherited from [Digikam::Rule](#)

- bool [addToken](#) (const QString &id, const QString &description, const QString &actionName=QString())  
*add a token to the parser, every parser should at least assign one token object*
- void [setDescription](#) (const QString &desc)
- void [setIcon](#) (const QString &pixmap)
- void [setRegExp](#) (const QRegularExpression &regExp)
- void [setUseTokenMenu](#) (bool value)  
*If multiple tokens have been assigned to a rule, a menu will be created.*

### 9.163.1 Member Function Documentation

#### 9.163.1.1 [parseOperation\(\)](#)

```
QString Digikam::CaseModifier::parseOperation (
    ParseSettings & settings,
    const QRegularExpressionMatch & match ) [override], [virtual]
```

##### Parameters

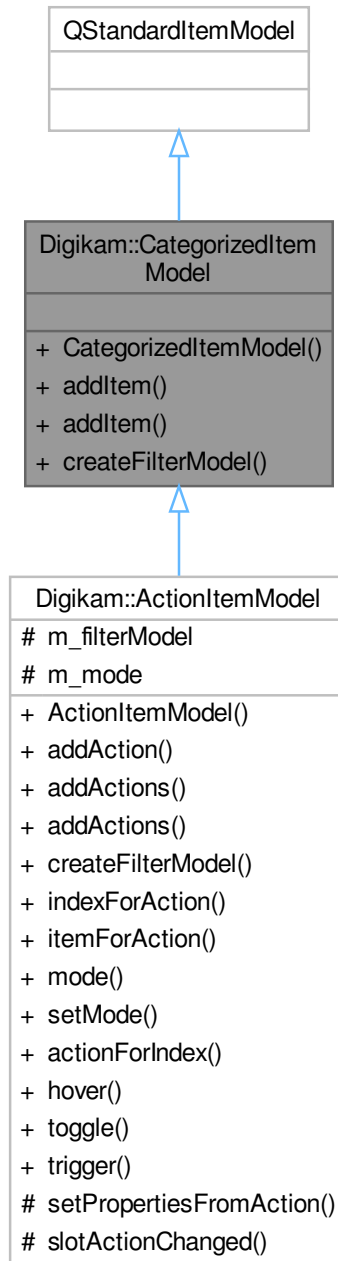
<i>settings</i>	contains settings
<i>match</i>	result of the regular expression match done in <a href="#">Option::parse()</a>

##### Returns

Implements [Digikam::Modifier](#).

## 9.164 Digikam::CategorizedItemModel Class Reference

Inheritance diagram for Digikam::CategorizedItemModel:



### Public Types

- enum `ExtraRoles` { `ItemOrderRole` = `Qt::UserRole + 1` }

## Public Member Functions

- **CategorizedItemModel** (QObject \*const parent=nullptr)
- QStandardItem \* **addItem** (const QString &text, const QIcon &decoration, const QVariant &category, const QVariant &categorySorting=QVariant())
- QStandardItem \* **addItem** (const QString &text, const QVariant &category, const QVariant &categorySorting=QVariant())
- virtual [DCategorizedSortFilterProxyModel](#) \* **createFilterModel** ()

## 9.164.1 Member Enumeration Documentation

### 9.164.1.1 ExtraRoles

enum `Digikam::CategorizedItemModel::ExtraRoles`

#### Enumerator

ItemOrderRole	This role, per default, reflects the order in which items are added.
---------------	--

## 9.165 Digikam::CBCContainer Class Reference

### Public Attributes

- double **alpha** = 1.0
- double **blue** = 1.0
- double **gamma** = 1.0
- double **green** = 1.0
- double **red** = 1.0



## 9.166 Digikam::CBFilter Class Reference

Inheritance diagram for Digikam::CBFilter:



### Public Member Functions

- **CBFilter** (const [CBContainer](#) &settings, [DlmgThreadedFilter](#) \*const master, const [Dlmg](#) &orgImage, [Dlmg](#) &destImage, int progressBegin=0, int progressEnd=100)

- **CBFilter** ([DImg](#) \*const orgImage, [QObject](#) \*const parent=nullptr, const [CBContainer](#) &settings=[CBContainer](#)())
- **CBFilter** ([QObject](#) \*const parent=nullptr)
- **FilterAction** [filterAction](#) () override
 

*Returns the action description corresponding to currently set options.*
- [QString](#) [filterIdentifier](#) () const override
 

*Return the identifier for this filter in the image history.*
- void [readParameters](#) (const [FilterAction](#) &action) override

## Public Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImg](#) \*const orgImage, [QObject](#) \*const parent, const [QString](#) &name=[QString](#)())
 

*Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter](#) ([QObject](#) \*const parent=nullptr, const [QString](#) &name=[QString](#)())
 

*Constructs a filter without argument.*
- virtual void [cancelFilter](#) ()
 

*Cancel the threaded computation.*
- const [QString](#) & **filterName** ()
- int **filterVersion** () const
- [DImg](#) **getTargetImage** ()
- [QList](#)< int > [multithreadedSteps](#) (int stop, int start=0) const
 

*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool [parametersSuccessfullyRead](#) () const
 

*Optional: error handling for readParameters.*
- virtual [QString](#) **readParametersError** (const [FilterAction](#) &actionThatFailed) const
- void **setFilterName** (const [QString](#) &name)
- void [setFilterVersion](#) (int version)
 

*Replaying a filter action: Set the filter version.*
- void **setOriginalImage** (const [DImg](#) &orgImage)
- void **setupAndStartDirectly** (const [DImg](#) &orgImage, [DImgThreadedFilter](#) \*const master, int progress←Begin=0, int progressEnd=100)
 

*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void [setupFilter](#) (const [DImg](#) &orgImage)
 

*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void **startFilter** ()
 

*Start the threaded computation.*
- virtual void **startFilterDirectly** ()
 

*Start computation of this filter, directly in this thread.*
- virtual [QList](#)< int > **supportedVersions** () const

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread](#) ([QObject](#) \*const parent=nullptr)
 

*This class extends [QRunnable](#), so you have to reimplement virtual void [run\(\)](#).*
- ~**DynamicThread** () override
 

*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool **isFinished** () const
- bool **isRunning** () const
- [QThread::Priority](#) **priority** () const
- void **setEmitSignals** (bool emitThem)
- void [setPriority](#) ([QThread::Priority](#) priority)
 

*Sets the priority for this dynamic thread.*
- State **state** () const

### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from [Digikam::DImgThreadedFilter](#)

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.166.1 Member Function Documentation

### 9.166.1.1 filterAction()

`FilterAction` Digikam::CBFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.166.1.2 filterIdentifier()

`QString` Digikam::CBFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

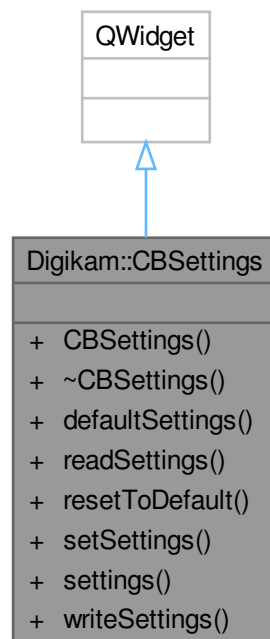
### 9.166.1.3 readParameters()

```
void Digikam::CBFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

## 9.167 Digikam::CBSettings Class Reference

Inheritance diagram for Digikam::CBSettings:



## Signals

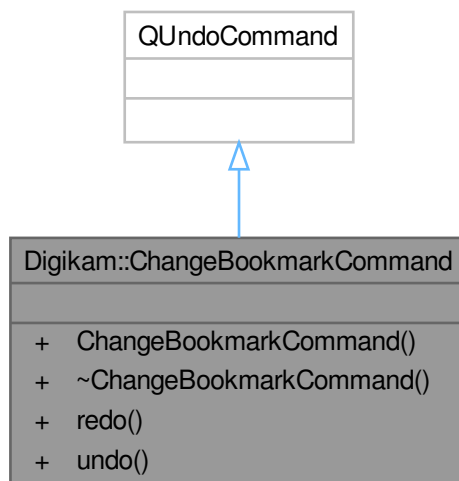
- void **signalSettingsChanged** ()

## Public Member Functions

- **CBSettings** (QWidget \*const parent)
- **CBContainer defaultSettings** () const
- void **readSettings** (const KConfigGroup &group)
- void **resetToDefault** ()
- void **setSettings** (const **CBContainer** &settings)
- **CBContainer settings** () const
- void **writeSettings** (KConfigGroup &group)

## 9.168 Digikam::ChangeBookmarkCommand Class Reference

Inheritance diagram for Digikam::ChangeBookmarkCommand:



## Public Types

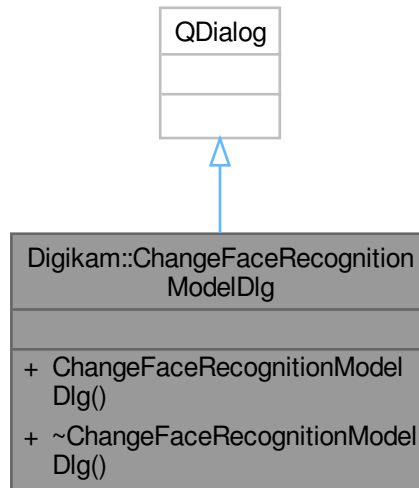
- enum **BookmarkData** { **Url** = 0 , **Title** , **Desc** }

## Public Member Functions

- **ChangeBookmarkCommand** (**BookmarksManager** \*const mngr, **BookmarkNode** \*const node, const QString &newValue, **BookmarkData** type)
- void **redo** () override
- void **undo** () override

## 9.169 Digikam::ChangeFaceRecognitionModelDlg Class Reference

Inheritance diagram for Digikam::ChangeFaceRecognitionModelDlg:

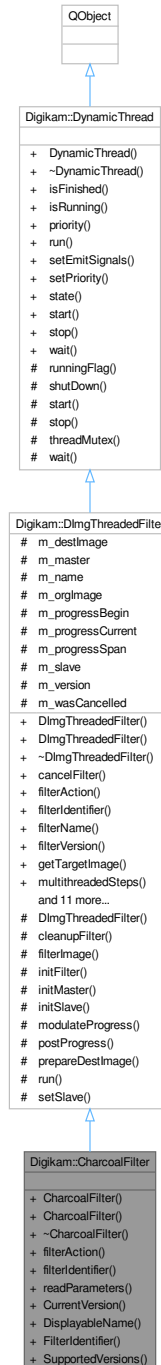


### Public Member Functions

- **ChangeFaceRecognitionModelDlg** (QWidget \*const parent, [FaceScanSettings::FaceRecognitionModel](#) newModel)

## 9.170 Digikam::CharcoalFilter Class Reference

Inheritance diagram for Digikam::CharcoalFilter:



### Public Member Functions

- **CharcoalFilter** (`DImg *const orgImage`, `QObject *const parent=nullptr`, `double pencil=5.0`, `double smooth=10.0`)



- **CharcoalFilter** (QObject \*const parent=nullptr)
- **FilterAction filterAction** () override  
*Returns the action description corresponding to currently set options.*
- **QString filterIdentifier** () const override  
*Return the identifier for this filter in the image history.*
- void **readParameters** (const **FilterAction** &action) override

## Public Member Functions inherited from Digikam::DImgThreadedFilter

- **DImgThreadedFilter** (DImg \*const orgImage, QObject \*const parent, const QString &name=QString())  
*Constructs a filter with all arguments (ready to use).*
- **DImgThreadedFilter** (QObject \*const parent=nullptr, const QString &name=QString())  
*Constructs a filter without argument.*
- virtual void **cancelFilter** ()  
*Cancel the threaded computation.*
- const QString & **filterName** ()
- int **filterVersion** () const
- **DImg getTargetImage** ()
- QList< int > **multithreadedSteps** (int stop, int start=0) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool **parametersSuccessfullyRead** () const  
*Optional: error handling for readParameters.*
- virtual QString **readParametersError** (const **FilterAction** &actionThatFailed) const
- void **setFilterName** (const QString &name)
- void **setFilterVersion** (int version)  
*Replaying a filter action: Set the filter version.*
- void **setOriginalImage** (const **DImg** &orgImage)
- void **setupAndStartDirectly** (const **DImg** &orgImage, **DImgThreadedFilter** \*const master, int progress←Begin=0, int progressEnd=100)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void **setupFilter** (const **DImg** &orgImage)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void **startFilter** ()  
*Start the threaded computation.*
- virtual void **startFilterDirectly** ()  
*Start computation of this filter, directly in this thread.*
- virtual QList< int > **supportedVersions** () const

## Public Member Functions inherited from Digikam::DynamicThread

- **DynamicThread** (QObject \*const parent=nullptr)  
*This class extends QRunnable, so you have to reimplement virtual void run().*
- **~DynamicThread** () override  
*The destructor calls stop() and wait(), but if you, in your destructor, delete any data that is accessed by your run() method, you must call stop() and wait() before yourself.*
- bool **isFinished** () const
- bool **isRunning** () const
- QThread::Priority **priority** () const
- void **setEmitSignals** (bool emitThem)
- void **setPriority** (QThread::Priority priority)  
*Sets the priority for this dynamic thread.*
- State **state** () const

### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from Digikam::DImgThreadedFilter

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from Digikam::DynamicThread

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from Digikam::DImgThreadedFilter

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.170.1 Member Function Documentation

### 9.170.1.1 filterAction()

`FilterAction` Digikam::CharcoalFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.170.1.2 filterIdentifier()

`QString` Digikam::CharcoalFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.170.1.3 readParameters()

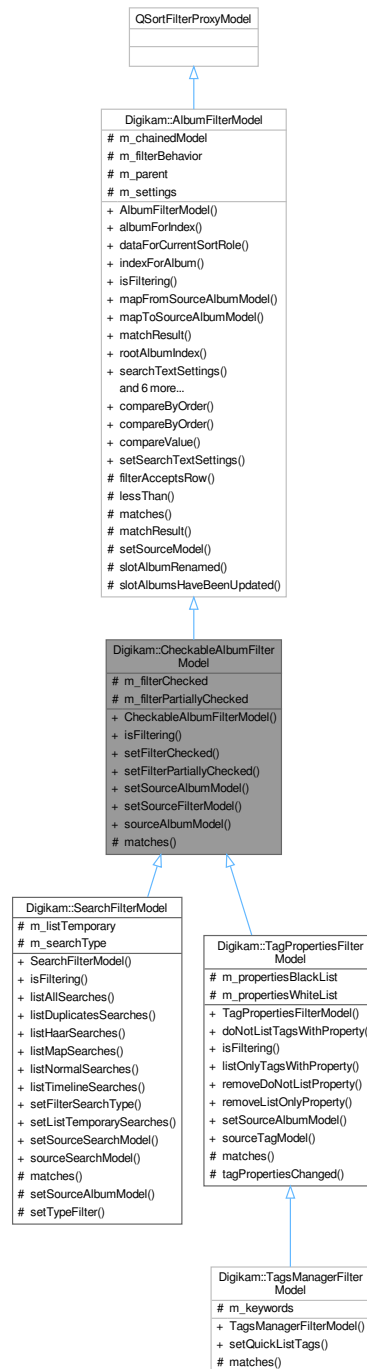
```
void Digikam::CharcoalFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

## 9.171 Digikam::CheckableAlbumFilterModel Class Reference

[Filter](#) model for checkable album models that allows more filtering options based on check state.

Inheritance diagram for Digikam::CheckableAlbumFilterModel:



## Public Member Functions

- **CheckableAlbumFilterModel** (QObject \*const parent=nullptr)
- bool **isFiltering** () const override  
*Returns if the currently applied filters will result in any filtering.*
- void **setFilterChecked** (bool filter)
- void **setFilterPartiallyChecked** (bool filter)

- void **setSourceAlbumModel** ([AbstractCheckableAlbumModel](#) \*const source)
- void **setSourceFilterModel** ([CheckableAlbumFilterModel](#) \*const source)
- [AbstractCheckableAlbumModel](#) \* **sourceAlbumModel** () const

## Public Member Functions inherited from [Digikam::AlbumFilterModel](#)

- **AlbumFilterModel** (QObject \*const parent=nullptr)
- [Album](#) \* **albumForIndex** (const QModelIndex &index) const  
*Convenience methods.*
- QVariant **dataForCurrentSortRole** ([Album](#) \*album) const
- QModelIndex **indexForAlbum** ([Album](#) \*album) const
- QModelIndex **mapFromSourceAlbumModel** (const QModelIndex &index) const
- QModelIndex **mapToSourceAlbumModel** (const QModelIndex &index) const
- [MatchResult](#) **matchResult** (const QModelIndex &index) const  
*Returns the MatchResult of an index of this model.*
- QModelIndex **rootAlbumIndex** () const
- [SearchTextSettings](#) **searchTextSettings** () const  
*Returns the settings currently used for filtering.*
- void **setFilterBehavior** ([FilterBehavior](#) behavior)  
*Sets the filter behavior.*
- void **setSourceAlbumModel** ([AbstractAlbumModel](#) \*const source)  
*Sets the source model.*
- void **setSourceFilterModel** ([AlbumFilterModel](#) \*const source)  
*Sets a chained filter model.*
- [AbstractAlbumModel](#) \* **sourceAlbumModel** () const
- [AlbumFilterModel](#) \* **sourceFilterModel** () const
- void **updateFilter** ()  
*Force invalidateFilter() externally.*

## Protected Member Functions

- bool **matches** ([Album](#) \*album) const override  
*This method provides the basic match checking algorithm.*

## Protected Member Functions inherited from [Digikam::AlbumFilterModel](#)

- bool **filterAcceptsRow** (int source\_row, const QModelIndex &source\_parent) const override
- bool **lessThan** (const QModelIndex &left, const QModelIndex &right) const override
- [MatchResult](#) **matchResult** ([Album](#) \*album) const  
*Returns if the filter matches this album (same logic as filterAcceptsRow).*
- void **setSourceModel** (QAbstractItemModel \*const model) override  
*Use setSourceAlbumModel.*

## Protected Attributes

- bool **m\_filterChecked** = false
- bool **m\_filterPartiallyChecked** = false

## Protected Attributes inherited from Digikam::AlbumFilterModel

- `QPointer< AlbumFilterModel > m_chainedModel = nullptr`
- `FilterBehavior m_filterBehavior = FullFiltering`
- `QObject * m_parent = nullptr`
- `SearchTextSettings m_settings`

## Additional Inherited Members

## Public Types inherited from Digikam::AlbumFilterModel

- enum `FilterBehavior` { `SimpleFiltering` , `FullFiltering` , `StrictFiltering` }
- enum `MatchResult` { `NoMatch = 0` , `DirectMatch` , `ParentMatch` , `ChildMatch` , `SpecialMatch` }

## Public Slots inherited from Digikam::AlbumFilterModel

- void `setSearchTextSettings` (const `SearchTextSettings` &settings)  
*Accepts new settings used for filtering and applies them to the model.*

## Signals inherited from Digikam::AlbumFilterModel

- void `hasSearchResult` (bool hasResult)  
*Indicates whether the newly applied filter results in a search result or not.*
- void `searchTextSettingsAboutToChange` (bool searched, bool willSearch)  
*This signal indicates that a new SearchTextSettings arrived and is about to be applied to the model.*
- void `searchTextSettingsChanged` (bool wasSearching, bool searched)  
*Indicates that new search text settings were applied.*
- void `signalFilterChanged` ()  
*Indicates that a new filter was applied to the model.*

## Static Public Member Functions inherited from Digikam::AlbumFilterModel

- `template<typename T >`  
`static int compareByOrder` (const T &a, const T &b, Qt::SortOrder sortOrder)
- `static int compareByOrder` (int compareResult, Qt::SortOrder sortOrder)  
*Takes a typical result from a compare method (0 is equal, -1 is less than, 1 is greater than) and applies the given sort order to it.*
- `template<typename T >`  
`static int compareValue` (const T &a, const T &b)  
*Returns the usual compare result of -1, 0, or 1 for lessThan, equals and greaterThan.*

## Protected Slots inherited from Digikam::AlbumFilterModel

- void `slotAlbumRenamed` (`Album` \*album)
- void `slotAlbumsHaveBeenUpdated` (int type)

## 9.171.1 Member Function Documentation

### 9.171.1.1 isFiltering()

```
bool Digikam::CheckableAlbumFilterModel::isFiltering ( ) const [override], [virtual]
```

#### Returns

`true` if the current selected filter could result in any filtering without checking if this really happens.

Reimplemented from [Digikam::AlbumFilterModel](#).

Reimplemented in [Digikam::SearchFilterModel](#), and [Digikam::TagPropertiesFilterModel](#).

### 9.171.1.2 matches()

```
bool Digikam::CheckableAlbumFilterModel::matches (
    Album * album ) const [override], [protected], [virtual]
```

Return true if this single album matches the current criteria. This method can be overridden to provide custom filtering.

#### Parameters

<i>album</i>	the album to tell if it matches the filter criteria or not.
--------------	---

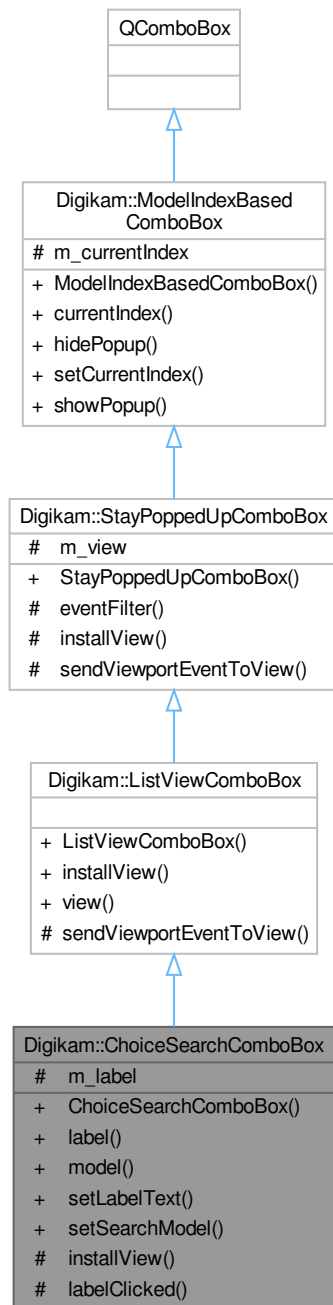
Reimplemented from [Digikam::AlbumFilterModel](#).

Reimplemented in [Digikam::SearchFilterModel](#), [Digikam::TagPropertiesFilterModel](#), and [Digikam::TagsManagerFilterModel](#).



## 9.172 Digikam::ChoiceSearchComboBox Class Reference

Inheritance diagram for Digikam::ChoiceSearchComboBox:



### Signals

- void **checkStateChanged** ()

## Public Member Functions

- [ChoiceSearchComboBox](#) (QWidget \*const parent=nullptr)  
*A combo box for entering a choice of values.*
- [DSqueezedClickLabel](#) \* **label** () const
- [ChoiceSearchModel](#) \* **model** () const
- void **setLabelText** (const QString &text)  
*Updates the text on the line edit area.*
- void **setSearchModel** ([ChoiceSearchModel](#) \*model)  
*Sets the model and initializes the widget.*

## Public Member Functions inherited from [Digikam::ListViewComboBox](#)

- [ListViewComboBox](#) (QWidget \*parent=nullptr)  
*This class provides an implementation of a [StayPoppedUpComboBox](#) with a [QListView](#).*
- [QListView](#) \* **view** () const  
*Returns the [QTreeView](#) of this class.*

## Public Member Functions inherited from [Digikam::StayPoppedUpComboBox](#)

- [StayPoppedUpComboBox](#) (QWidget \*const parent=nullptr)  
*This class provides an abstract [QComboBox](#) with a custom view (which is created by implementing subclasses) instead of the usual [QListView](#).*

## Public Member Functions inherited from [Digikam::ModelIndexBasedComboBox](#)

- [ModelIndexBasedComboBox](#) (QWidget \*const parent=nullptr)  
*[QComboBox](#) has a current index based on a single integer.*
- [QModelIndex](#) **currentIndex** () const
- void **hidePopup** () override
- void **setCurrentIndex** (const [QModelIndex](#) &index)
- void **showPopup** () override

## Protected Slots

- void **labelClicked** ()

## Protected Member Functions

- void **installView** ([QAbstractItemView](#) \*view=nullptr) override  
*Replace the standard combo box list view with a [QTreeView](#).*

## Protected Member Functions inherited from [Digikam::ListViewComboBox](#)

- void **sendViewportEventToView** ([QEvent](#) \*e) override  
*Implement in subclass: Send the given event to the [viewportEvent\(\)](#) method of `m_view`.*

## Protected Member Functions inherited from [Digikam::StayPoppedUpComboBox](#)

- bool **eventFilter** (QObject \*watched, QEvent \*event) override
- void **installView** (QAbstractItemView \*view)

*Replace the standard combo box list view with the given view.*

## Protected Attributes

- [DSqueezedClickLabel](#) \* **m\_label** = nullptr

## Protected Attributes inherited from [Digikam::StayPoppedUpComboBox](#)

- QAbstractItemView \* **m\_view** = nullptr

## Protected Attributes inherited from [Digikam::ModelIndexBasedComboBox](#)

- QPersistentModelIndex **m\_currentIndex**

## 9.172.1 Constructor & Destructor Documentation

### 9.172.1.1 ChoiceSearchComboBox()

```
Digikam::ChoiceSearchComboBox::ChoiceSearchComboBox (
    QWidget *const parent = nullptr ) [explicit]
```

Operates on a [ChoiceSearchModel](#). After constructing the object, call `setModel` with your model.

## 9.172.2 Member Function Documentation

### 9.172.2.1 installView()

```
void Digikam::ChoiceSearchComboBox::installView (
    QAbstractItemView * view = nullptr ) [override], [protected], [virtual]
```

Call this after installing an appropriate model.

Reimplemented from [Digikam::ListViewComboBox](#).

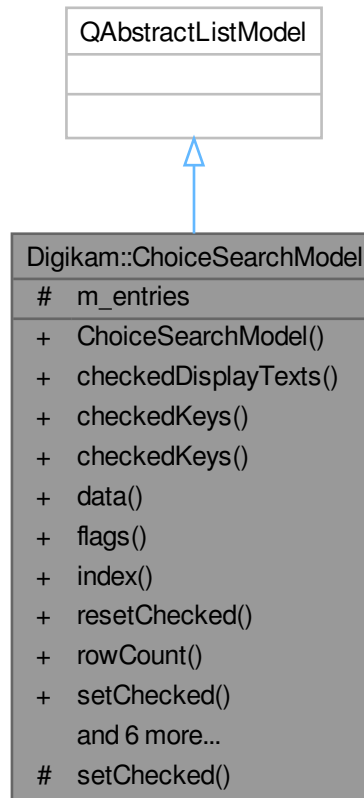
### 9.172.2.2 setSearchModel()

```
void Digikam::ChoiceSearchComboBox::setSearchModel (
    ChoiceSearchModel * model )
```

Can only be called once for a widget.

## 9.173 Digikam::ChoiceSearchModel Class Reference

Inheritance diagram for Digikam::ChoiceSearchModel:



### Classes

- class [Entry](#)

### Public Types

- enum `CustomRoles` { `IdRole = Qt::UserRole` }

### Signals

- void `checkStateChanged` (const `QVariant` &key, bool isChecked)

## Public Member Functions

- **ChoicesearchModel** (QObject \*const parent=nullptr)
- QStringList **checkedDisplayTexts** () const  
*Returns the display text of all entries that are selected.*
- QVariantList **checkedKeys** () const  
*Returns the keys of all entries that are selected (checked).*
- template<typename T >  
QList< T > **checkedKeys** () const  
*Returns the keys of all entries that are selected (checked), converted to a list of the template type.*
- QVariant **data** (const QModelIndex &index, int role) const override
- Qt::ItemFlags **flags** (const QModelIndex &index) const override
- QModelIndex **index** (int row, int column=0, const QModelIndex &parent=QModelIndex()) const override
- void **resetChecked** ()  
*Sets all entries to unchecked.*
- int **rowCount** (const QModelIndex &parent) const override
- template<typename T >  
void **setChecked** (const QList< T > &keys, bool checked=true)  
*Sets the check state of all the entries whose key is found in the list to checked.*
- template<typename T >  
void **setChecked** (const T &key, bool checked=true)  
*Sets the check state of the entry with given key.*
- template<typename T >  
void **setChecked** (const T &value, SearchXml::Relation relation)  
*Sets the check state of all entries.*
- void **setChoice** (const QMap< int, QString > &data)  
*Sets the data from the given map, with integer keys and QString user displayable value.*
- void **setChoice** (const QStringList &data)  
*Sets the data from the given list, taking every first entry as the key, every second as the user displayable value.*
- void **setChoice** (const QVariantList &data)  
*Sets the data from the given list, taking every first entry as the key, every second as the user displayable value.*
- bool **setData** (const QModelIndex &index, const QVariant &value, int role) override

## Protected Member Functions

- void **setChecked** (int index, bool checked)

## Protected Attributes

- QList< [Entry](#) > **m\_entries**

## 9.173.1 Member Function Documentation

### 9.173.1.1 checkedKeys()

```
template<typename T >
QList< T > Digikam::ChoicesearchModel::checkedKeys ( ) const
```

Supported for Int and QString types.

### 9.173.1.2 setChecked()

```
template<typename T >
void Digikam::ChoiceSearchModel::setChecked (
    const T & value,
    SearchXml::Relation relation )
```

The check state is determined by the key of an entry, the relation, and a constant value. Think of "Set to checked if key is less than 5". Supported for Int and QString types.

### 9.173.1.3 setChoice()

```
void Digikam::ChoiceSearchModel::setChoice (
    const QVariantList & data )
```

Ensure that the QVariants' type is correct (identical for all even entries, QString for all odd entries).

## 9.174 Digikam::ChoiceSearchModel::Entry Class Reference

### Public Member Functions

- **Entry** (const QVariant &key, const QString &userDisplay)
- bool **operator==** (const [Entry](#) &other) const
- bool **operator==** (const QVariant &other) const

### Public Attributes

- bool **m\_checkState** = false
- QString **m\_display**
- QVariant **m\_key**

## 9.174.1 Member Function Documentation

### 9.174.1.1 operator==( )

```
bool Digikam::ChoiceSearchModel::Entry::operator== (
    const QVariant & other ) const
```

## 9.175 Digikam::CIETongueWidget Class Reference

Inheritance diagram for Digikam::CIETongueWidget:



### Public Member Functions

- **CIETongueWidget** (int w, int h, QWidget \*const parent=nullptr, cmsHPROFILE hMonitor=nullptr)
- void **loadingFailed** ()
- void **loadingStarted** ()
- bool **setProfileData** (const QByteArray &profileData=QByteArray())
- bool **setProfileFromFile** (const QUrl &file=QUrl())
- void **uncalibratedColor** ()

### Protected Member Functions

- QRgb **colorByCoord** (double x, double y)
- void **drawLabels** ()
- void **drawSmallEllipse** (LPcmsCIExyY xyY, BYTE r, BYTE g, BYTE b, int sz)
- void **drawTongueAxis** ()
- void **drawTongueGrid** ()
- void **fillTongue** ()
- int **grids** (double val) const
- void **outlineTongue** ()
- void **paintEvent** (QPaintEvent \*) override
- void **resizeEvent** (QResizeEvent \*event) override

## 9.176 Digikam::ClickDragReleaseItem Class Reference

Inheritance diagram for Digikam::ClickDragReleaseItem:



### Signals

- void **cancelled** ()
- void **finished** (const QRectF &rect)
- void **moving** (const QRectF &rect)
- void **started** (const QPointF &pos)

*Signals are emitted at click, drag and release event.*



**Public Member Functions**

- **ClickDragReleaseItem** (QGraphicsItem \*const parent)
- QRectF **boundingRect** () const override
- void **paint** (QPainter \*, const QStyleOptionGraphicsItem \*, QWidget \*) override

**Protected Member Functions**

- void **hoverMoveEvent** (QGraphicsSceneHoverEvent \*) override
- void **keyPressEvent** (QKeyEvent \*) override
- void **mouseDoubleClickEvent** (QGraphicsSceneMouseEvent \*) override
- void **mouseMoveEvent** (QGraphicsSceneMouseEvent \*) override
- void **mousePressEvent** (QGraphicsSceneMouseEvent \*) override
  - 1) *Press - Drag - Release: mousePressEvent, PressedState -> mouseMoveEvent over threshold, PressDragState -> mouseReleaseEvent, finished*
  - 2) *Click - Move - Click: mousePressEvent, PressedState -> mouseReleaseEvent, ClickedMoveState -> hoverMoveEvent -> mouseReleaseEvent, finished*
- void **mouseReleaseEvent** (QGraphicsSceneMouseEvent \*) override

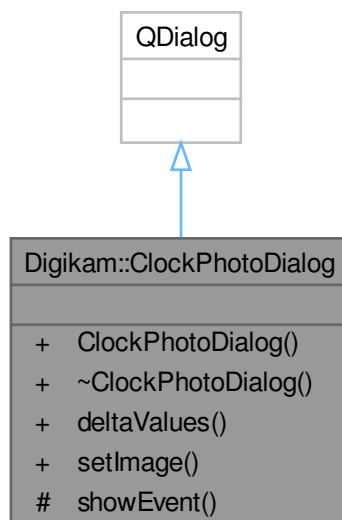
**9.176.1 Member Function Documentation****9.176.1.1 started**

```
void Digikam::ClickDragReleaseItem::started (
    const QPointF & pos ) [signal]
```

Reported positions are in scene coordinates. A drag is reported only if the mouse was moved a certain threshold. A release is reported after every press.

**9.177 Digikam::ClockPhotoDialog Class Reference**

Inheritance diagram for Digikam::ClockPhotoDialog:



## Public Member Functions

- **ClockPhotoDialog** (QWidget \*const parent, const QUrl &defaultUrl)
- **DeltaTime deltaValues** () const
- bool **setImage** (const QUrl &)

*Try to load the photo specified by the QUrl, and set the datetime widget to the photo time.*

## Protected Member Functions

- void **showEvent** (QShowEvent \*) override

## 9.177.1 Member Function Documentation

### 9.177.1.1 setImage()

```
bool Digikam::ClockPhotoDialog::setImage (
    const QUrl & imageFile )
```

Return true on success, or false if either the photo can't be read or the datetime information can't be read.

## 9.178 Digikam::CMat Struct Reference

[CMat](#):

### Public Attributes

- double \* **center**  
*Points to element with index (0, 0)*
- double \* **data**  
*Contents of matrix.*
- int **radius**  
*Radius of the matrix.*
- int **row\_stride**  
*Size of one row = 2 \* radius + 1.*

### 9.178.1 Detailed Description

Centered matrix. This is a square matrix where the indices range from [-radius, radius]. The matrix contains (2 \* radius + 1) \*\* 2 elements.

## 9.179 Digikam::CollectionImageChangeset Class Reference

### Public Types

- enum **Operation** {  
**Unknown** , **Added** , **Removed** , **RemovedAll** ,  
**Deleted** , **RemovedDeleted** , **Moved** , **Copied** }

## Public Member Functions

- [CollectionImageChangeset](#) ()=default  
An [CollectionImageChangeset](#) covers adding and removing an image to/from the collection.
- [CollectionImageChangeset](#) (const QList< qlonglong > &ids, const QList< int > &albums, [Operation](#) operation)
- [CollectionImageChangeset](#) (const QList< qlonglong > &ids, int album, [Operation](#) operation)
- [CollectionImageChangeset](#) (qlonglong id, int album, [Operation](#) operation)
- QList< int > [albums](#) () const
- bool [containsAlbum](#) (int id) const
- bool [containsImage](#) (qlonglong id) const
- QList< qlonglong > [ids](#) () const  
Specification of this changeset.
- [Operation](#) [operation](#) () const
- [CollectionImageChangeset](#) & [operator](#)<< (const [CollectionImageChangeset](#) &other)  
Combines two [CollectionImageChangesets](#).
- [CollectionImageChangeset](#) & [operator](#)<< (const QDBusArgument &argument)
- const [CollectionImageChangeset](#) & [operator](#)>> (QDBusArgument &argument) const

## 9.179.1 Member Enumeration Documentation

### 9.179.1.1 Operation

```
enum Digikam::CollectionImageChangeset::Operation
```

#### Enumerator

Added	"Added" indicates that images have been added to albums.
Removed	"Removed" indicates that an image has been removed from the given album, and has possibly set a status of Removed and a null <a href="#">Album</a> (though this can already have changed to valid values), but the image-specific tables have not been removed.
RemovedAll	"RemovedAll" indicates that for all entries in the specified album, the "Removed" operation has been carried out. This is equivalent to a "Removed" changesets with all image ids in the list, but for RemovedAll, the list may not be explicitly given (may be empty).
Deleted	"Deleted" indicates that the image-specific tables have been removed from the database. While "Removed" means all data is still there, though possibly not accessible from an album, this means all data has been irreversibly deleted.
RemovedDeleted	Special combination: Images which has the "Removed" status have now been "Delete"d. A changeset with Removed or RemovedAll is guaranteed to have been sent anytime before. Image ids nor albums ids may or may be not available in any combination.
Moved	Images have been moved. This is extra information; a Removed and then an Added changeset are guaranteed to be sent subsequently. <a href="#">Album</a> is the source album.
Copied	Images have been copied. This is extra information; an Added changeset is guaranteed to be sent subsequently. <a href="#">Album</a> is the source album.

## 9.179.2 Constructor & Destructor Documentation

### 9.179.2.1 CollectionImageChangeset()

```
Digikam::CollectionImageChangeset::CollectionImageChangeset ( ) [default]
```

It is described by a list of affected image ids, a list of affected albums, and an operation. Special Case "Removed↔All": If all images have been removed from an album, operation is RemovedAll, the album list contains the (now empty) albums, `ids()` is empty, but `containsImage()` always returns true. Special Case "RemovedDeleted": Images with the "Removed" status are now irreversibly deleted. `ids()` and/or `albums()` may be empty (this means information is not available).

### 9.179.3 Member Function Documentation

#### 9.179.3.1 `ids()`

```
QList< qlonglong > Digikam::CollectionImageChangeset::ids ( ) const
```

All special cases where the returned list may be empty are noted above. The lists are valid unless such a case is explicitly mentioned.

#### 9.179.3.2 `operator<<()`

```
CollectionImageChangeset & Digikam::CollectionImageChangeset::operator<< (
    const CollectionImageChangeset & other )
```

The operations shall not differ between the two sets; the operation is set to Unknown if it differs. This is especially not suitable for RemovedAll changesets.

## 9.180 Digikam::CollectionLocation Class Reference

### Public Types

- enum `CaseSensitivity` { `UnknownCaseSensitivity` , `CaseInsensitive` , `CaseSensitive` }
- enum `Status` { `LocationNull` , `LocationAvailable` , `LocationHidden` , `LocationUnavailable` , `LocationDeleted` }
- enum `Type` { `Undefined` = 0 , `VolumeHardWired` = 1 , `VolumeRemovable` = 2 , `Network` = 3 }

### Public Member Functions

- `QString albumRootPath ()` const  
*The current file system path leading to this album root.*
- `Qt::CaseSensitivity asQtCaseSensitivity ()` const  
*Return as Qt case sensitivity enum of location.*
- `CaseSensitivity caseSensitivity ()` const  
*The case sensitivity of location.*
- `size_t hash ()` const
- `int id ()` const  
*The id uniquely identifying this collection.*
- `bool isAvailable ()` const
- `bool isNull ()` const
- `QString label ()` const  
*A user-visible, optional label.*
- `Status status ()` const  
*The current status.*
- `Type type ()` const  
*The type of location.*

## Public Attributes

- QString **identifier**

## Protected Attributes

- CaseSensitivity **m\_caseSensitivity** = UnknownCaseSensitivity
- int **m\_id** = -1
- QString **m\_label**
- QString **m\_path**
- Status **m\_status** = LocationNull
- Type **m\_type** = VolumeHardWired

## 9.180.1 Member Enumeration Documentation

### 9.180.1.1 CaseSensitivity

```
enum Digikam::CollectionLocation::CaseSensitivity
```

#### Enumerator

UnknownCaseSensitivity	The location has an unknown case sensitivity.
CaseInsensitive	The location is case insensitive.
CaseSensitive	The location is case sensitive.

### 9.180.1.2 Status

```
enum Digikam::CollectionLocation::Status
```

#### Enumerator

LocationNull	An invalid status. A location has this status if it is not valid, and it had this status before its creation (for oldStatus information)
LocationAvailable	The location if available. This is the most common status.
LocationHidden	The location is explicitly hidden. This gives no information if the location was available were it not hidden.
LocationUnavailable	The location is currently not available. (Harddisk unplugged, CD not in drive, network fs not mounted etc.) It may become available any time.
LocationDeleted	An invalid status. A location object acquires this status if it has been deleted. The object then does no longer point to an existing location.

### 9.180.1.3 Type

```
enum Digikam::CollectionLocation::Type
```

## Enumerator

Undefined	The location is undefined. Keep values constant.
VolumeHardWired	The location is located on a storage device that is built-in without frequent removal: Hard-disk inside the machine.
VolumeRemovable	The location is located on a storage device that can be removed from the local machine, and is expected to be removed. USB stick, USB hard-disk, CD, DVD
Network	The location is available via a network file system. The availability depends on the network connection.

## 9.180.2 Member Function Documentation

### 9.180.2.1 albumRootPath()

```
QString Digikam::CollectionLocation::albumRootPath ( ) const
```

Only guaranteed to be valid for location with status Available.

### 9.180.2.2 asQtCaseSensitivity()

```
Qt::CaseSensitivity Digikam::CollectionLocation::asQtCaseSensitivity ( ) const
```

For unknown, it is assumed to be Qt::CaseSensitive.

### 9.180.2.3 caseSensitivity()

```
CollectionLocation::CaseSensitivity Digikam::CollectionLocation::caseSensitivity ( ) const
```

See above for possible values.

### 9.180.2.4 status()

```
CollectionLocation::Status Digikam::CollectionLocation::status ( ) const
```

See above for possible values.

### 9.180.2.5 type()

```
CollectionLocation::Type Digikam::CollectionLocation::type ( ) const
```

See above for possible values.

## 9.181 Digikam::CollectionManager Class Reference

Inheritance diagram for Digikam::CollectionManager:



### Public Types

- enum `LocationCheckResult` { `LocationInvalidCheck`, `LocationAllRight`, `LocationHasProblems`, `LocationNotAllowed` }

## Signals

- void **triggerUpdateVolumesList** ()

## Public Member Functions

- void **refresh** ()  
*Clears all locations and re-reads the lists of collection locations.*
- void **setWatchDisabled** ()  
*Disables the collection watch.*

## Static Public Member Functions

- static void **cleanUp** ()
- static **CollectionManager** \* **instance** ()

## Operations on Albums

- class **Private**
- class **CoreDbWatch**
- class **CoreDbAccess**
- QStringList **allAvailableAlbumRootPaths** ()  
*Returns a list of the paths of all currently available root paths.*
- QString **albumRootPath** (int id)  
*Returns the album root path with the given id.*
- QString **albumRootLabel** (int id)  
*Returns the album root label with the given id.*
- QUrl **albumRoot** (const QUrl &fileUrl)  
*For a given path, the part of the path that forms the album root is returned, ending without a slash.*
- QString **albumRootPath** (const QUrl &fileUrl)
- QString **albumRootPath** (const QString &filePath)
- bool **isAlbumRoot** (const QUrl &fileUrl)  
*Returns true if the given path forms an album root.*
- bool **isAlbumRoot** (const QString &filePath)  
*The file path should not end with the directory slash.*
- QString **album** (const QUrl &fileUrl)  
*Returns the album part of the given file path, i.e.*
- QString **album** (const QString &filePath)
- QString **album** (const **CollectionLocation** &location, const QUrl &fileUrl)
- QString **album** (const **CollectionLocation** &location, const QString &filePath)
- QUrl **oneAlbumRoot** ()  
*Returns just one album root, out of the list of available location, the one that is most suitable to serve as a default, e.g.*
- QString **oneAlbumRootPath** ()



## Operations on Collection Location

- [CollectionLocation addLocation](#) (const QUrl &fileUrl, const QString &label=QString())  
*Add the given file system location as new collection location.*
- [CollectionLocation addNetworkLocation](#) (const QUrl &fileUrl, const QString &label=QString())
- [CollectionLocation refreshLocation](#) (const [CollectionLocation](#) &location, int newType, const QStringList &pathList, const QString &label=QString())
- [LocationCheckResult checkLocation](#) (const QUrl &fileUrl, QList< [CollectionLocation](#) > &assumeDeleted, QString \*message=nullptr, QString \*suggestedMessageIconName=nullptr)  
*Analyzes the given file path.*
- [LocationCheckResult checkNetworkLocation](#) (const QUrl &fileUrl, QList< [CollectionLocation](#) > &assumeDeleted, QString \*message=nullptr, QString \*suggestedMessageIconName=nullptr)
- void [removeLocation](#) (const [CollectionLocation](#) &location)  
*Removes the given location.*
- void [setLabel](#) (const [CollectionLocation](#) &location, const QString &label)  
*Sets the label of the given location.*
- void [changeType](#) (const [CollectionLocation](#) &location, int type)  
*Changes the [CollectionLocation::Type](#) of the given location.*
- QList< [CollectionLocation](#) > [checkHardWiredLocations](#) ()  
*Checks the locations of type [HardWired](#).*
- void [migrationCandidates](#) (const [CollectionLocation](#) &disappearedLocation, QString \*const technicalDescription, QStringList \*const candidateIdentifiers, QStringList \*const candidateDescriptions)  
*For a given disappeared location (retrieved from [checkHardWiredLocations\(\)](#)) retrieve a user-presentable technical description (excluding the [CollectionLocation](#)'s label) and a list of identifiers and corresponding user presentable strings of candidates to where the given location may have been moved.*
- void [migrateToVolume](#) (const [CollectionLocation](#) &location, const QString &identifier)  
*Migrates the existing collection to a new volume, identified by an internal identifier as returned by [checkHardWiredLocations\(\)](#).*
- QList< [CollectionLocation](#) > [allLocations](#) ()  
*Returns a list of all [CollectionLocations](#) stored in the database.*
- QList< [CollectionLocation](#) > [allAvailableLocations](#) ()  
*Returns a list of all currently available [CollectionLocations](#).*
- [CollectionLocation locationForAlbumRootId](#) (int id)  
*Returns the location for the given album root id.*
- [CollectionLocation locationForAlbumRoot](#) (const QUrl &fileUrl)  
*Returns the [CollectionLocation](#) that contains the given album root.*
- [CollectionLocation locationForAlbumRootPath](#) (const QString &albumRootPath)
- [CollectionLocation locationForUrl](#) (const QUrl &fileUrl)  
*Returns the [CollectionLocation](#) that contains the given path.*
- [CollectionLocation locationForPath](#) (const QString &filePath)
- void [locationStatusChanged](#) (const [CollectionLocation](#) &location, int oldStatus)  
*Emitted when the status of a collection location changed.*
- void [locationPropertiesChanged](#) (const [CollectionLocation](#) &location)  
*Emitted when the label of a collection location is changed.*

### 9.181.1 Member Enumeration Documentation

#### 9.181.1.1 LocationCheckResult

```
enum Digikam::CollectionManager::LocationCheckResult
```

## Enumerator

LocationInvalidCheck	The check did not succeed, status unknown.
LocationAllRight	All right. The accompanying message may be empty.
LocationHasProblems	Location can be added, but the user should be aware of a problem.
LocationNotAllowed	Adding the location will fail (e.g. there is already a location for the path)

## 9.181.2 Member Function Documentation

### 9.181.2.1 addLocation()

```
CollectionLocation Digikam::CollectionManager::addLocation (
    const QUrl & fileUrl,
    const QString & label = QString() )
```

Type and availability will be detected. On failure returns null. This would be the case if the given url is already contained in another collection location. You may pass an optional user-visible label that will be stored in the database. The label has no further meaning and can be freely chosen.

[CollectionLocation](#) objects returned are simple data containers. If the corresponding location is returned, the data is still safe to access, but does not represent anything. Therefore, do not store returned objects, but prefer to retrieve them freshly.

### 9.181.2.2 album()

```
QString Digikam::CollectionManager::album (
    const QUrl & fileUrl )
```

the album root path at the beginning is removed and the second part, starting with "/", ending without a slash, is returned. Example: "/media/fotos/Paris 2007" gives "/Paris 2007" Returns a null QString if the file path is not located in an album root. Returns "/" if the file path is an album root. Note that trailing slashes are removed in the return value, regardless if there was one or not. Note that you have to feed a path/url pointing to a directory. File names cannot be recognized as such by this method, and will be treated as a directory.

### 9.181.2.3 albumRoot()

```
QUrl Digikam::CollectionManager::albumRoot (
    const QUrl & fileUrl )
```

Example: "/media/fotos/Paris 2007" gives "/media/fotos". Only available (or hidden, but available) album roots are guaranteed to be found.

### 9.181.2.4 albumRootLabel()

```
QString Digikam::CollectionManager::albumRootLabel (
    int id )
```

Returns a null QString if the root path does not exist or is not available.

### 9.181.2.5 albumRootPath()

```
QString Digikam::CollectionManager::albumRootPath (
    int id )
```

Returns a null QString if the root path does not exist or is not available.

### 9.181.2.6 checkHardWiredLocations()

```
QList< CollectionLocation > Digikam::CollectionManager::checkHardWiredLocations ( )
```

If one of these is not available currently, it is added to the list of disappeared locations. This case may happen if a file system is changed, a backup restored or other actions taken that change the UUID, although the data may still be available and mounted. If there are hard-wired volumes available which are candidates for a newly appeared volume (in fact those that do not contain any collections currently), they are added to the map, identifier -> i18n'ed user presentable description. The identifier can be used for changeVolume.

### 9.181.2.7 checkLocation()

```
CollectionManager::LocationCheckResult Digikam::CollectionManager::checkLocation (
    const QUrl & fileUrl,
    QList< CollectionLocation > & assumeDeleted,
    QString * message = nullptr,
    QString * suggestedMessageIconName = nullptr )
```

Creates an info message describing the result of identification or possible problems. The text is i18n'ed and can be presented to the user. The returned result enum describes the test result.

### 9.181.2.8 isAlbumRoot() [1/2]

```
bool Digikam::CollectionManager::isAlbumRoot (
    const QString & filePath )
```

Using [CoreDbUrl](#)'s method is fine.

### 9.181.2.9 isAlbumRoot() [2/2]

```
bool Digikam::CollectionManager::isAlbumRoot (
    const QUrl & fileUrl )
```

It will return false if the path is a path below an album root, or if the path does not belong to an album root. Example: "/media/fotos/Paris 2007" is an album with album root "/media/fotos". "/media/fotos" returns true, "/media/fotos/Paris 2007" and "/media" return false. Only available (or hidden, but available) album roots are guaranteed to be found.

### 9.181.2.10 locationForAlbumRoot()

```
CollectionLocation Digikam::CollectionManager::locationForAlbumRoot (
    const QUrl & fileUrl )
```

The path must be an album root with [isAlbumRoot\(\)](#) == true. Returns 0 if no collection location matches. Only available (or hidden, but available) locations are guaranteed to be found.

### 9.181.2.11 locationForUrl()

```
CollectionLocation Digikam::CollectionManager::locationForUrl (
    const QUrl & fileUrl )
```

Equivalent to calling `locationForAlbumRoot(albumRoot(fileUrl))`. Only available (or hidden, but available) locations are guaranteed to be found.

### 9.181.2.12 locationStatusChanged

```
void Digikam::CollectionManager::locationStatusChanged (
    const CollectionLocation & location,
    int oldStatus ) [signal]
```

This means that the location became available, hidden or unavailable.

An added location will change its status after addition, from Null to Available, Hidden or Unavailable.

A removed location will change its status to Deleted during the removal; in this case, you shall not use the object passed with this signal with any method of [CollectionManager](#).

The second signal argument is of type [CollectionLocation::Status](#) and describes the status before the state change occurred

### 9.181.2.13 migrateToVolume()

```
void Digikam::CollectionManager::migrateToVolume (
    const CollectionLocation & location,
    const QString & identifier )
```

Use this *only* to react to changes like those detailed for `checkHardWiredLocations`; the actual data pointed to shall be unchanged.

### 9.181.2.14 oneAlbumRoot()

```
QUrl Digikam::CollectionManager::oneAlbumRoot ( )
```

to suggest as default place when the user wants to add files.

### 9.181.2.15 refresh()

```
void Digikam::CollectionManager::refresh ( )
```

Enables the watch.

### 9.181.2.16 removeLocation()

```
void Digikam::CollectionManager::removeLocation (
    const CollectionLocation & location )
```

This means that all images contained on the location will be removed from the database, all tags will be lost.

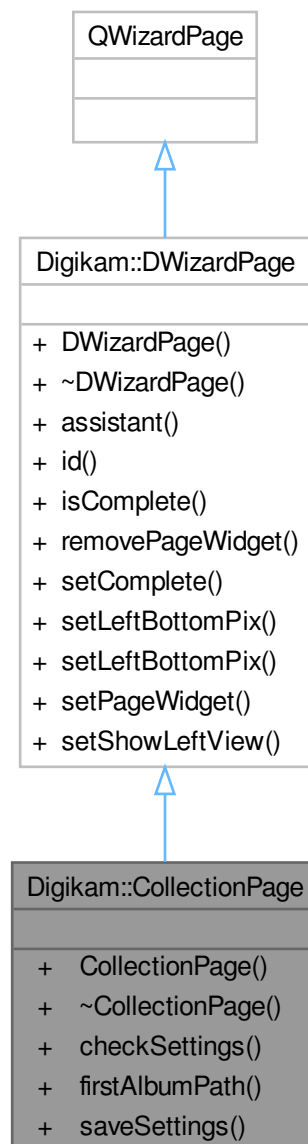
### 9.181.2.17 setWatchDisabled()

```
void Digikam::CollectionManager::setWatchDisabled ( )
```

It will be reenabled as soon as [refresh\(\)](#) is called or any other action triggered.

## 9.182 Digikam::CollectionPage Class Reference

Inheritance diagram for Digikam::CollectionPage:



**Public Member Functions**

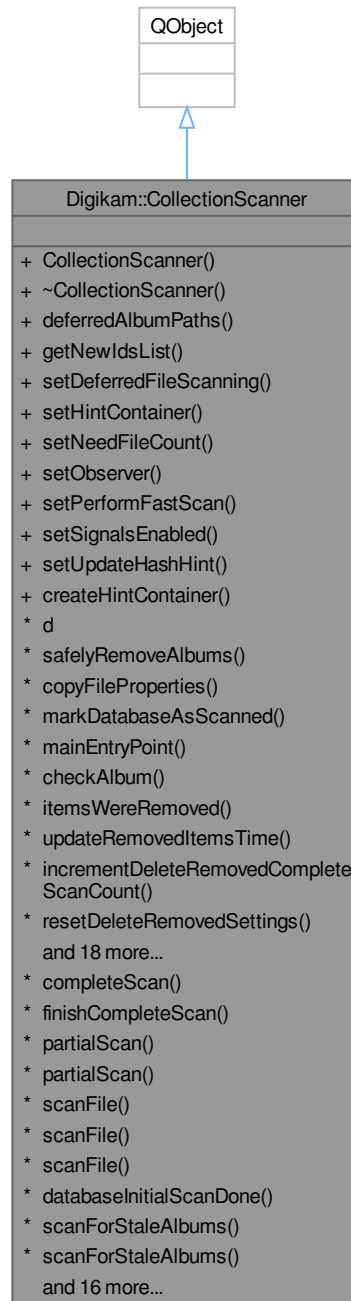
- **CollectionPage** (QWizard \*const dlg)
- bool **checkSettings** ()
- QString **firstAlbumPath** () const
- void **saveSettings** ()

**Public Member Functions inherited from [Digikam::DWizardPage](#)**

- **DWizardPage** (QWizard \*const dlg, const QString &title)
- QWizard \* **assistant** () const
- int **id** () const
- bool **isComplete** () const override
- void **removePageWidget** (QWidget \*const w)
- void **setComplete** (bool b)
- void **setLeftBottomPix** (const QIcon &icon)
- void **setLeftBottomPix** (const QPixmap &pix)
- void **setPageWidget** (QWidget \*const w)
- void **setShowLeftView** (bool v)

## 9.183 Digikam::CollectionScanner Class Reference

Inheritance diagram for Digikam::CollectionScanner:



### Public Types

- enum `FileScanMode` { `NormalScan` , `ModifiedScan` , `Rescan` , `CleanScan` }

## Public Member Functions

- QStringList **deferredAlbumPaths** () const
- QList< qlonglong > **getNewIdsList** () const  
*Returns item ids from new detected items.*
- void **setDeferredFileScanning** (bool defer)
- void **setHintContainer** ([CollectionScannerHintContainer](#) \*const container)
- void **setNeedFileCount** (bool on)  
*Call this to enable emitting the total files to scan (for progress info) before a complete collection scan.*
- void **setObserver** ([CollectionScannerObserver](#) \*const observer)  
*Set an observer to be able to cancel a running scan.*
- void **setPerformFastScan** (bool on)  
*Call this to disable fast scan with album date check.*
- void **setSignalsEnabled** (bool on)  
*Call this to enable the progress info signals.*
- void **setUpdateHashHint** (bool hint=true)

## Static Public Member Functions

- static [CollectionScannerHintContainer](#) \* **createHintContainer** ()  
*Hints give the scanner additional info about things that happened in the past carried out by higher level which the collection scanner cannot know.*

## Scan utilities

- void **safelyRemoveAlbums** (const QList< int > &albumIds)  
*Prepare the given albums to be removed, typically by setting the albums as orphan and removing all entries from the albums.*
- static void **copyFileProperties** (const [ItemInfo](#) &source, const [ItemInfo](#) &dest)  
*When a file is derived from another file, typically through editing, copy all relevant attributes from source file to the new file.*
- void **markDatabaseAsScanned** ()
- void **mainEntryPoint** (bool complete)
- int **checkAlbum** (const [CollectionLocation](#) &location, const QString &album)
- void **itemsWereRemoved** (const QList< qlonglong > &removedIds)
- void **updateRemovedItemsTime** ()
- void **incrementDeleteRemovedCompleteScanCount** ()
- void **resetDeleteRemovedSettings** ()
- bool **checkDeleteRemoved** ()
- void **loadNameFilters** ()
- int **countItemsInFolder** (const QString &path)
- [DatabaseItem::Category](#) **category** (const [QFileInfo](#) &info)
- void **totalFilesToScan** (int count)  
*Emitted once in scanAlbums(), the scan() methods, and updateItemsWithoutDate().*
- void **startScanningAlbumRoot** (const QString &albumRoot)  
*Notifies the begin of the scanning of the specified album root, album, of stale files, or of the whole collection (after stale files)*
- void **startScanningAlbum** (const QString &albumRoot, const QString &album)
- void **startScanningForStaleAlbums** ()
- void **startScanningAlbumRoots** ()
- void **startCompleteScan** ()
- void **signalScannedNewImage** (const [QFileInfo](#) &info)



- void [finishedScanningAlbumRoot](#) (const QString &albumRoot)  
*Emitted when the scanning has finished.*
- void **finishedScanningAlbum** (const QString &albumRoot, const QString &album, int filesScanned)
- void **finishedScanningForStaleAlbums** ()
- void **finishedCompleteScan** ()
- void [scannedFiles](#) (int filesScanned)  
*Emitted between startScanningAlbum and finishedScanningAlbum.*
- void **cancelled** ()  
*Emitted when the observer told to cancel the scan.*

### Scan operations

- void [completeScan](#) ()  
*Carries out a full scan on all available parts of the collection.*
- void [finishCompleteScan](#) (const QStringList &albumPaths)  
*If you enable deferred file scanning for a [completeScan\(\)](#), new files will not be scanned.*
- void [partialScan](#) (const QString &filePath)  
*Carries out a partial scan on the specified path of the collection.*
- void **partialScan** (const QString &albumRoot, const QString &album)  
*Same procedure as above, but albumRoot and album is provided.*
- qlonglong [scanFile](#) (const QString &filePath, [FileScanMode](#) mode=[ModifiedScan](#))  
*The given file will be scanned according to the given mode.*
- qlonglong [scanFile](#) (const QString &albumRoot, const QString &album, const QString &fileName, [FileScanMode](#) mode=[ModifiedScan](#))  
*Same procedure as above, but albumRoot and album is provided.*
- void **scanFile** (const [ItemInfo](#) &info, [FileScanMode](#) mode=[ModifiedScan](#))  
*The given file represented by the [ItemInfo](#) will be scanned according to mode.*
- static bool [databaseInitialScanDone](#) ()  
*Returns if the initial scan of the database has been done.*
- void **scanForStaleAlbums** (const QList< [CollectionLocation](#) > &locations)
- void **scanForStaleAlbums** (const QList< int > &locationIdsToScan)
- void **scanAlbumRoot** (const [CollectionLocation](#) &location)
- void **scanAlbum** (const [CollectionLocation](#) &location, const QString &album, bool checkDate=false)
- void **scanExistingFile** (const QFileInfo &fi, qlonglong id)
- void **scanFileNormal** (const QFileInfo &info, const [ItemScanInfo](#) &scanInfo, bool checkSidecar=true, const QFileInfo \*const sidecarInfo=nullptr)
- void **scanModifiedFile** (const QFileInfo &info, const [ItemScanInfo](#) &scanInfo)
- void **scanFileUpdateHashReuseThumbnail** (const QFileInfo &fi, const [ItemScanInfo](#) &scanInfo, bool fileWasEdited)
- void **cleanScanFile** (const QFileInfo &info, const [ItemScanInfo](#) &scanInfo)
- void **rescanFile** (const QFileInfo &info, const [ItemScanInfo](#) &scanInfo)
- void **completeScanCleanupPart** ()
- void **completeHistoryScanning** ()
- void **finishHistoryScanning** ()
- void **historyScanningStage2** (const QList< qlonglong > &ids)
- void **historyScanningStage3** (const QList< qlonglong > &ids)
- qlonglong [scanFile](#) (const QFileInfo &fi, int albumId, qlonglong id, [FileScanMode](#) mode)
- qlonglong [scanNewFile](#) (const QFileInfo &info, int albumId)
- qlonglong [scanNewFileFullScan](#) (const QFileInfo &info, int albumId)

## 9.183.1 Member Enumeration Documentation

### 9.183.1.1 FileScanMode

enum [Digikam::CollectionScanner::FileScanMode](#)

## Enumerator

NormalScan	The file will be scanned like it is done for any usual scan. If it was not modified, no further action is taken. If the file is not known yet, it will be fully scanned, or, if an identical file is found, this data will be copied.
ModifiedScan	The file will scanned like a modified file. Only a selected portion of the metadata will be updated into the database. If the file is not known yet, it will be fully scanned, or, if an identical file is found, this data will be copied.
Rescan	The file will be scanned like a completely new file. The complete metadata is re-read into the database. No search for identical files will be done.
CleanScan	This is the same as Rescan but the database metadata will be cleaned up if the corresponding metadata write option is enabled.

## 9.183.2 Member Function Documentation

### 9.183.2.1 completeScan()

```
void Digikam::CollectionScanner::completeScan ( )
```

Only a full scan can finally remove deleted files from the database, only a full scan will mark the database as scanned. The database will be locked while running (Note: this is not done for partialScans).

### 9.183.2.2 createHintContainer()

```
CollectionScannerHintContainer * Digikam::CollectionScanner::createHintContainer ( ) [static]
```

They allow to carry out optimizations. Record hints in a container, and provide the container to the collection scanner. The Container set in setHintContainer must be one created by createContainer.

### 9.183.2.3 databaseInitialScanDone()

```
bool Digikam::CollectionScanner::databaseInitialScanDone ( ) [static]
```

This is the first complete scan after creation of a new database file (or update requiring a rescan)

### 9.183.2.4 finishCompleteScan()

```
void Digikam::CollectionScanner::finishCompleteScan (
    const QStringList & albumPaths )
```

The relevant albums are available from deferredAlbumPaths() when [completeScan\(\)](#) has finished. You need to call [finishCompleteScan\(\)](#) afterwards with the list to get the same complete scan than undeferred [completeScan\(\)](#).

### 9.183.2.5 finishedScanningAlbumRoot

```
void Digikam::CollectionScanner::finishedScanningAlbumRoot (
    const QString & albumRoot ) [signal]
```

Note that start/finishScanningAlbum may be emitted recursively.

### 9.183.2.6 partialScan()

```
void Digikam::CollectionScanner::partialScan (
    const QString & filePath )
```

The includes scanning for new files + albums and updating modified file data. Files no longer found in the specified path however are not completely removed, but only marked as removed. They will be removed only after a complete scan.

### 9.183.2.7 scanFile() [1/2]

```
qulonglong Digikam::CollectionScanner::scanFile (
    const QString & albumRoot,
    const QString & album,
    const QString & fileName,
    FileScanMode mode = ModifiedScan )
```

If you already have this info it need not be retrieved. Returns the image id of the file, or -1 on failure.

### 9.183.2.8 scanFile() [2/2]

```
qulonglong Digikam::CollectionScanner::scanFile (
    const QString & filePath,
    FileScanMode mode = ModifiedScan )
```

Returns the image id of the file.

### 9.183.2.9 scannedFiles

```
void Digikam::CollectionScanner::scannedFiles (
    int filesScanned ) [signal]
```

In between these two signals, the sum of filesScanned of all sent signals equals the one reported by finished↔ ScanningAlbum()

### 9.183.2.10 setNeedFileCount()

```
void Digikam::CollectionScanner::setNeedFileCount (
    bool on )
```

Default is off. If on, setSignalEnabled() must be on to take effect.

### 9.183.2.11 setPerformFastScan()

```
void Digikam::CollectionScanner::setPerformFastScan (
    bool on )
```

Default is on.

### 9.183.2.12 setSignalsEnabled()

```
void Digikam::CollectionScanner::setSignalsEnabled (
    bool on )
```

Default is off.

### 9.183.2.13 totalFilesToScan

```
void Digikam::CollectionScanner::totalFilesToScan (
    int count ) [signal]
```

Gives the number of the files that need to be scanned.

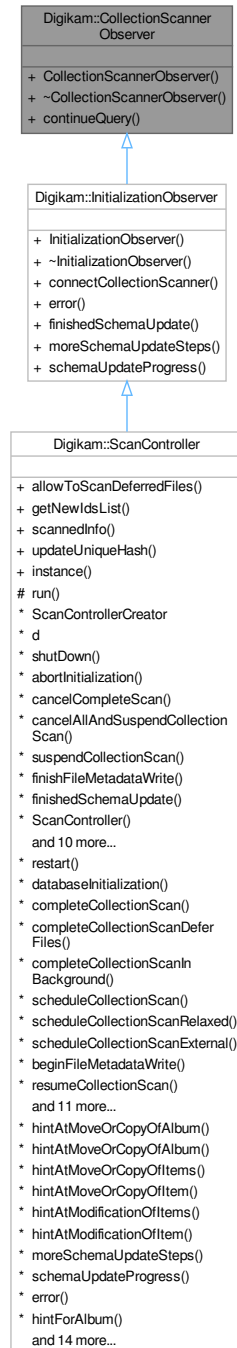
## 9.184 Digikam::CollectionScannerHintContainer Class Reference

### Public Member Functions

- **CollectionScannerHintContainer** ()=default  
*Note: All methods of this class must be thread-safe.*
- virtual void **clear** ()=0
- virtual void **recordHint** (const [ItemMetadataAdjustmentHint](#) &hints)=0
- virtual void **recordHints** (const QList< [AlbumCopyMoveHint](#) > &hints)=0
- virtual void **recordHints** (const QList< [ItemChangeHint](#) > &hints)=0
- virtual void **recordHints** (const QList< [ItemCopyMoveHint](#) > &hints)=0

## 9.185 Digikam::CollectionScannerObserver Class Reference

Inheritance diagram for Digikam::CollectionScannerObserver:

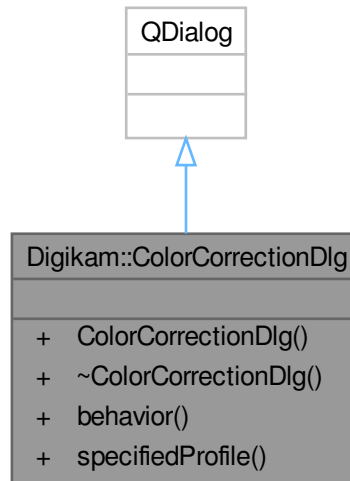


### Public Member Functions

- virtual bool **continueQuery** ()=0

## 9.186 Digikam::ColorCorrectionDlg Class Reference

Inheritance diagram for Digikam::ColorCorrectionDlg:



### Public Types

- enum **Mode** { **ProfileMismatch** , **MissingProfile** , **UncalibratedColor** }

### Public Member Functions

- **ColorCorrectionDlg** (Mode mode, const [DImg](#) &preview, const QString &file, QWidget \*const parent=nullptr)
- ICCSettingsContainer::Behavior **behavior** () const
- [IccProfile](#) **specifiedProfile** () const

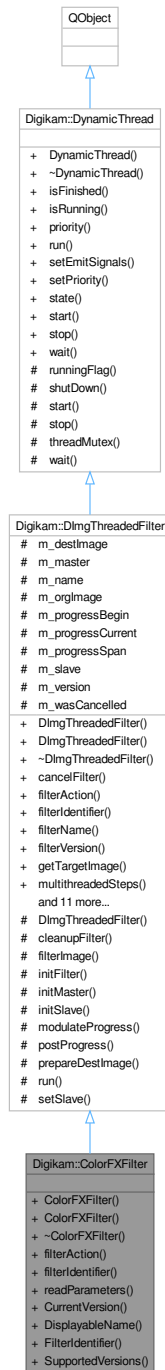
## 9.187 Digikam::ColorFXContainer Class Reference

### Public Attributes

- int **colorFXType** = 0  
*ColorFXFilter::Solarize.*
- int **intensity** = 100
- int **iterations** = 2
- int **level** = 0
- QString **path**

## 9.188 Digikam::ColorFXFilter Class Reference

Inheritance diagram for Digikam::ColorFXFilter:



### Public Types

- enum **ColorFXFilterTypes** {
  - Solarize** = 0 , **Vivid** , **Neon** , **FindEdges** , **Lut3D** }

## Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

## Public Member Functions

- **ColorFXFilter** ([DImg](#) \*const orgImage, [QObject](#) \*const parent, const [ColorFXContainer](#) &settings=[ColorFXContainer](#)())
- **ColorFXFilter** ([QObject](#) \*const parent=nullptr)
- [FilterAction](#) filterAction () override
 

*Returns the action description corresponding to currently set options.*
- [QString](#) filterIdentifier () const override
 

*Return the identifier for this filter in the image history.*
- void [readParameters](#) (const [FilterAction](#) &action) override

## Public Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImg](#) \*const orgImage, [QObject](#) \*const parent, const [QString](#) &name=[QString](#)())
 

*Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter](#) ([QObject](#) \*const parent=nullptr, const [QString](#) &name=[QString](#)())
 

*Constructs a filter without argument.*
- virtual void [cancelFilter](#) ()
 

*Cancel the threaded computation.*
- const [QString](#) & [filterName](#) ()
- int [filterVersion](#) () const
- [DImg](#) [getTargetImage](#) ()
- [QList](#)< int > [multithreadedSteps](#) (int stop, int start=0) const
 

*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool [parametersSuccessfullyRead](#) () const
 

*Optional: error handling for readParameters.*
- virtual [QString](#) [readParametersError](#) (const [FilterAction](#) &actionThatFailed) const
- void [setFilterName](#) (const [QString](#) &name)
- void [setFilterVersion](#) (int version)
 

*Replaying a filter action: Set the filter version.*
- void [setOriginalImage](#) (const [DImg](#) &orgImage)
- void [setupAndStartDirectly](#) (const [DImg](#) &orgImage, [DImgThreadedFilter](#) \*const master, int progress←Begin=0, int progressEnd=100)
 

*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void [setupFilter](#) (const [DImg](#) &orgImage)
 

*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void [startFilter](#) ()
 

*Start the threaded computation.*
- virtual void [startFilterDirectly](#) ()
 

*Start computation of this filter, directly in this thread.*
- virtual [QList](#)< int > [supportedVersions](#) () const



## Public Member Functions inherited from Digikam::DynamicThread

- [DynamicThread](#) (QObject \*const parent=nullptr)
 

*This class extends QRunnable, so you have to reimplement virtual void [run\(\)](#).*
- [~DynamicThread](#) () override
 

*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool [isFinished](#) () const
- bool [isRunning](#) () const
- QThread::Priority [priority](#) () const
- void [setEmitSignals](#) (bool emitThem)
- void [setPriority](#) (QThread::Priority priority)
 

*Sets the priority for this dynamic thread.*
- State [state](#) () const

## Static Public Member Functions

- static int [CurrentVersion](#) ()
- static QString [DisplayableName](#) ()
- static QString [FilterIdentifier](#) ()
- static QList< int > [SupportedVersions](#) ()

## Additional Inherited Members

## Public Slots inherited from Digikam::DynamicThread

- void [start](#) ()
- void [stop](#) ()
 

*Stop computation, sets the running flag to false.*
- void [wait](#) ()
 

*Waits until the thread finishes.*

## Signals inherited from Digikam::DImgThreadedFilter

- void [finished](#) (bool success)
 

*Emitted when the computation has completed.*
- void [progress](#) (int progress)
 

*Emitted when progress info from the calculation is available.*
- void [started](#) ()
 

*This signal is emitted when image data is available and the computation has started.*

## Signals inherited from Digikam::DynamicThread

- void [finished](#) ()
- void [starting](#) ()
 

*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from [Digikam::DImgThreadedFilter](#)

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.188.1 Member Function Documentation

### 9.188.1.1 filterAction()

`FilterAction` Digikam::ColorFXFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.188.1.2 filterIdentifier()

`QString` Digikam::ColorFXFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

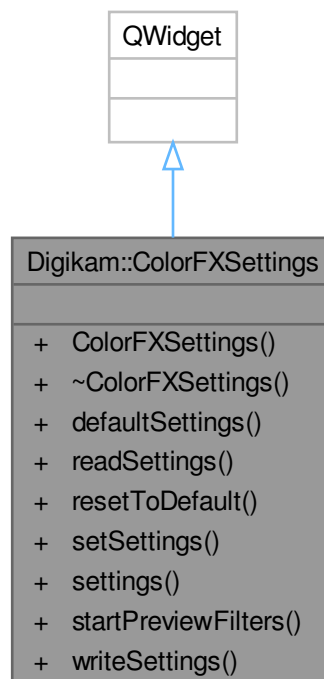
### 9.188.1.3 readParameters()

```
void Digikam::ColorFXFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

## 9.189 Digikam::ColorFXSettings Class Reference

Inheritance diagram for Digikam::ColorFXSettings:



## Signals

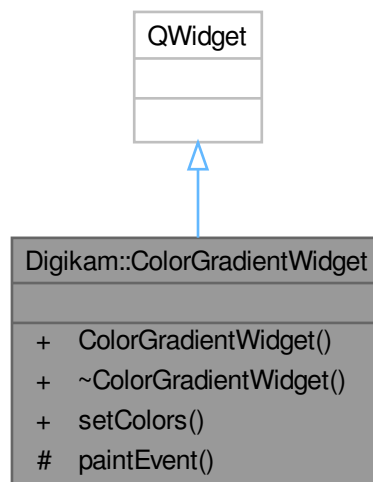
- void **signalSettingsChanged** ()

## Public Member Functions

- **ColorFXSettings** (QWidget \*const parent, bool useGenericImg=true)
- **ColorFXContainer defaultSettings** () const
- void **readSettings** (const KConfigGroup &group)
- void **resetToDefault** ()
- void **setSettings** (const **ColorFXContainer** &settings)
- **ColorFXContainer settings** () const
- void **startPreviewFilters** ()
- void **writeSettings** (KConfigGroup &group)

## 9.190 Digikam::ColorGradientWidget Class Reference

Inheritance diagram for Digikam::ColorGradientWidget:



## Public Member Functions

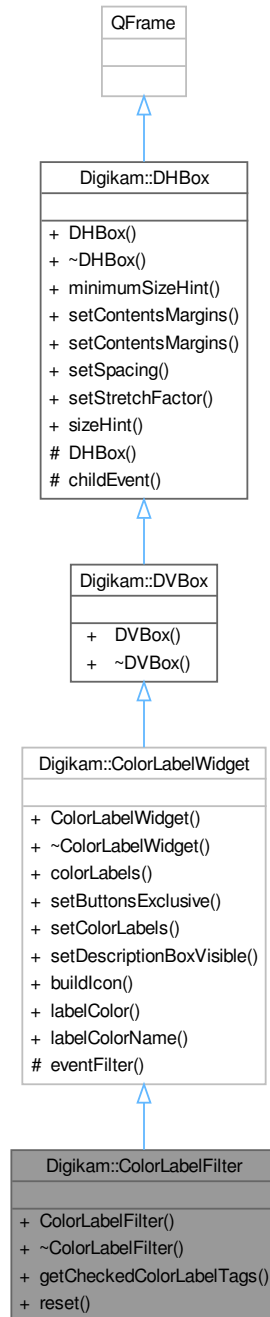
- **ColorGradientWidget** (Qt::Orientation orientation, int size, QWidget \*const parent=nullptr)
- void **setColors** (const QColor &col1, const QColor &col2)

## Protected Member Functions

- void **paintEvent** (QPaintEvent \*) override

## 9.191 Digikam::ColorLabelFilter Class Reference

Inheritance diagram for Digikam::ColorLabelFilter:



### Signals

- void **signalColorLabelSelectionChanged** (const QList< ColorLabel > &)

## Signals inherited from [Digikam::ColorLabelWidget](#)

- void **signalColorLabelChanged** (int)

## Public Member Functions

- **ColorLabelFilter** (QWidget \*const parent=nullptr)
- QList< [TAlbum](#) \* > **getCheckedColorLabelTags** ()
- void **reset** ()

## Public Member Functions inherited from [Digikam::ColorLabelWidget](#)

- **ColorLabelWidget** (QWidget \*const parent=nullptr)
- QList< ColorLabel > **colorLabels** () const  
*Return the list of Color Label buttons turned on or an empty list of none.*
- void **setButtonsExclusive** (bool b)  
*Set all Color Label buttons exclusive or not.*
- void **setColorLabels** (const QList< ColorLabel > &list)  
*Turn on Color Label buttons using list.*
- void **setDescriptionBoxVisible** (bool b)  
*Show or not on the bottom view the description of label with shortcuts.*

## Public Member Functions inherited from [Digikam::DVBox](#)

- **DVBox** (QWidget \*const parent=nullptr)

## Public Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (QWidget \*const parent=nullptr)
- QSize **minimumSizeHint** () const override
- void **setContentMargins** (const QMargins &margins)
- void **setContentMargins** (int left, int top, int right, int bottom)
- void **setSpacing** (int space)
- void **setStretchFactor** (QWidget \*const widget, int stretch)
- QSize **sizeHint** () const override

## Additional Inherited Members

## Static Public Member Functions inherited from [Digikam::ColorLabelWidget](#)

- static QIcon **buildIcon** (ColorLabel label, int size=12)
- static QColor **labelColor** (ColorLabel label)
- static QString **labelColorName** (ColorLabel label)

## Protected Member Functions inherited from [Digikam::ColorLabelWidget](#)

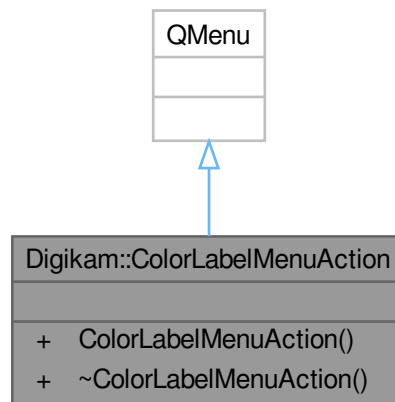
- bool **eventFilter** (QObject \*obj, QEvent \*ev) override

## Protected Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (bool vertical, QWidget \*const parent)
- void **childEvent** (QChildEvent \*e) override

## 9.192 Digikam::ColorLabelMenuAction Class Reference

Inheritance diagram for Digikam::ColorLabelMenuAction:



### Signals

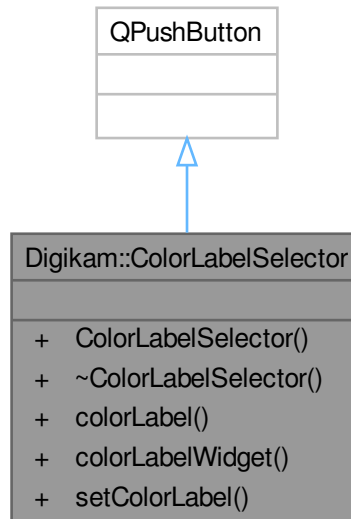
- void **signalColorLabelChanged** (int)

### Public Member Functions

- **ColorLabelMenuAction** (QMenu \*const parent=nullptr)

## 9.193 Digikam::ColorLabelSelector Class Reference

Inheritance diagram for Digikam::ColorLabelSelector:



### Signals

- void **signalColorLabelChanged** (int)

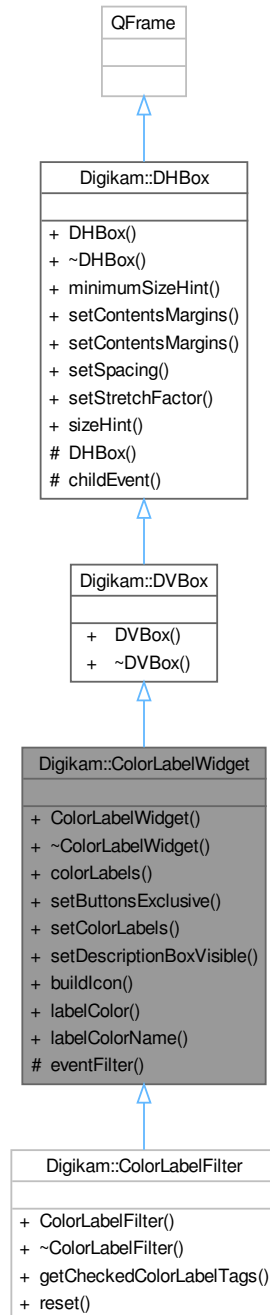
### Public Member Functions

- **ColorLabelSelector** (QWidget \*const parent=nullptr)
- ColorLabel **colorLabel** ()
- [ColorLabelWidget](#) \* **colorLabelWidget** () const
- void **setColorLabel** (ColorLabel label)



## 9.194 Digikam::ColorLabelWidget Class Reference

Inheritance diagram for Digikam::ColorLabelWidget:



### Signals

- void **signalColorLabelChanged** (int)

## Public Member Functions

- **ColorLabelWidget** (QWidget \*const parent=nullptr)
- QList< ColorLabel > **colorLabels** () const  
*Return the list of Color Label buttons turned on or an empty list of none.*
- void **setButtonsExclusive** (bool b)  
*Set all Color Label buttons exclusive or not.*
- void **setColorLabels** (const QList< ColorLabel > &list)  
*Turn on Color Label buttons using list.*
- void **setDescriptionBoxVisible** (bool b)  
*Show or not on the bottom view the description of label with shortcuts.*

## Public Member Functions inherited from [Digikam::DVBox](#)

- **DVBox** (QWidget \*const parent=nullptr)

## Public Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (QWidget \*const parent=nullptr)
- QSize **minimumSizeHint** () const override
- void **setContentsMargins** (const QMargins &margins)
- void **setContentsMargins** (int left, int top, int right, int bottom)
- void **setSpacing** (int space)
- void **setStretchFactor** (QWidget \*const widget, int stretch)
- QSize **sizeHint** () const override

## Static Public Member Functions

- static QIcon **buildIcon** (ColorLabel label, int size=12)
- static QColor **labelColor** (ColorLabel label)
- static QString **labelColorName** (ColorLabel label)

## Protected Member Functions

- bool **eventFilter** (QObject \*obj, QEvent \*ev) override

## Protected Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (bool vertical, QWidget \*const parent)
- void **childEvent** (QChildEvent \*e) override

## 9.194.1 Member Function Documentation

### 9.194.1.1 setButtonsExclusive()

```
void Digikam::ColorLabelWidget::setButtonsExclusive (
    bool b )
```

Default is true as only one can be selected. Non-exclusive mode is dedicated for Advanced Search tool.

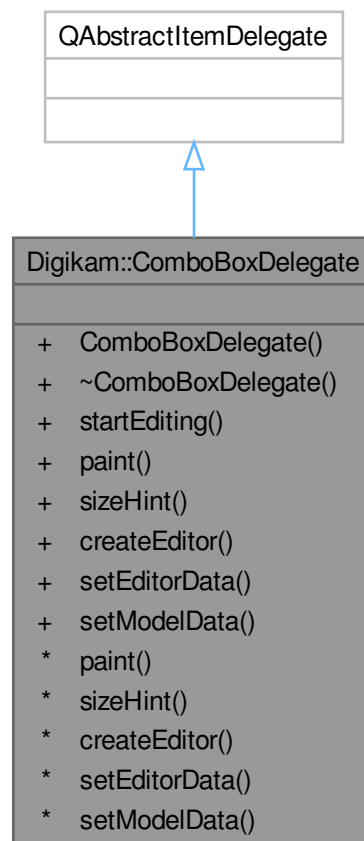
## 9.194.1.2 setColorLabels()

```
void Digikam::ColorLabelWidget::setColorLabels (
    const QList< ColorLabel > & list )
```

Pass an empty list to clear all selection.

## 9.195 Digikam::ComboBoxDelegate Class Reference

Inheritance diagram for Digikam::ComboBoxDelegate:



## Public Member Functions

- **ComboBoxDelegate** (`DItemsList *const`, `const QMap< int, QString > &`)
- void **startEditing** (`QTreeWidgetItem *`, `int`)

*Whenever an element needs to be edited, this method should be called.*

- void **paint** (QPainter \*, const QStyleOptionViewItem &, const QModelIndex &) const override  
*Overloaded functions to provide the delegate functionality.*
- QSize **sizeHint** (const QStyleOptionViewItem &, const QModelIndex &) const override
- QWidget \* **createEditor** (QWidget \*, const QStyleOptionViewItem &, const QModelIndex &) const override
- void **setEditorData** (QWidget \*, const QModelIndex &) const override
- void **setModelData** (QWidget \*, QAbstractItemModel \*, const QModelIndex &) const override

## 9.195.1 Member Function Documentation

### 9.195.1.1 startEditing()

```
void Digikam::ComboBoxDelegate::startEditing (
    QTreeWidgetItem * item,
    int column )
```

It's actually a hack to prevent the item text shining through whenever editing occurs.

## 9.196 Digikam::CommentInfo Class Reference

### Public Member Functions

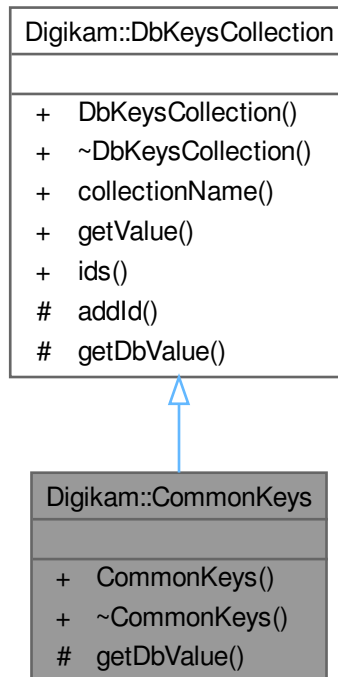
- bool **isNull** () const

### Public Attributes

- QString **author**
- QString **comment**
- QDateTime **date**
- int **id** = -1
- qlonglong **imageId** = -1
- QString **language**
- DatabaseComment::Type **type** = DatabaseComment::UndefinedType

## 9.197 Digikam::CommonKeys Class Reference

Inheritance diagram for Digikam::CommonKeys:



### Protected Member Functions

- `QString` `getDbValue` (const `QString` &key, `ParseSettings` &settings) override  
*Abstract method for retrieving the value from the database for the given key.*

### Protected Member Functions inherited from [Digikam::DbKeysCollection](#)

- void `addId` (const `QString` &id, const `QString` &description)  
*Add an ID to the key collection.*

### Additional Inherited Members

### Public Member Functions inherited from [Digikam::DbKeysCollection](#)

- `DbKeysCollection` (const `QString` &n)  
*Default constructor.*
- `QString` `collectionName` () const  
*Get the name of the DbKeysCollection.*
- `QString` `getValue` (const `QString` &key, `ParseSettings` &settings)  
*Get a value from the database.*
- `DbKeyIdsMap` `ids` () const  
*Get all IDs associated with this key collection.*

## 9.197.1 Member Function Documentation

### 9.197.1.1 getDbValue()

```
QString Digikam::CommonKeys::getDbValue (
    const QString & key,
    ParseSettings & settings ) [override], [protected], [virtual]
```

This method has to be implemented by all child classes. It is called by the [getValue\(\)](#) method.

#### Parameters

<i>key</i>	the key representing the value in the database
<i>settings</i>	the ParseSettings object holding all relevant information about the image.

#### Returns

the value of the given database key

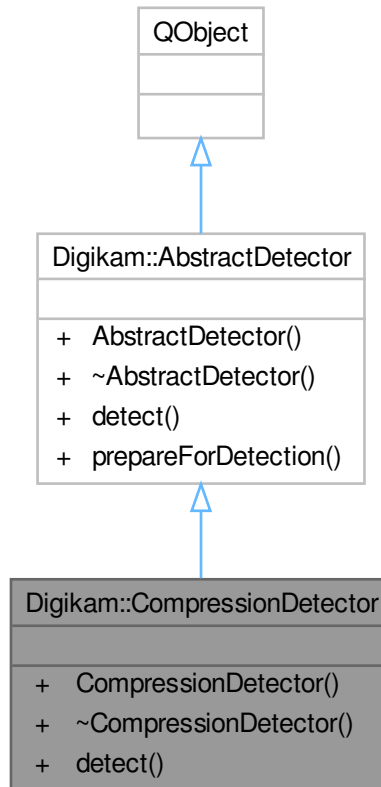
#### See also

[DbKeysCollection::getValue\(\)](#)

Implements [Digikam::DbKeysCollection](#).

## 9.198 Digikam::CompressionDetector Class Reference

Inheritance diagram for Digikam::CompressionDetector:



### Public Member Functions

- float [detect](#) (const cv::Mat &image) const override

### Public Member Functions inherited from [Digikam::AbstractDetector](#)

- **AbstractDetector** (QObject \*const parent=nullptr)

### Additional Inherited Members

### Static Public Member Functions inherited from [Digikam::AbstractDetector](#)

- static cv::Mat **prepareForDetection** (const [DImg](#) &inputImage)

*NOTE: Maybe this function will move to `read_image()` of `imagequalityparser` in case all detectors of IQS use `cv::Mat`.*

## 9.198.1 Member Function Documentation

### 9.198.1.1 detect()

```
float Digikam::CompressionDetector::detect (
    const cv::Mat & image ) const [override], [virtual]
```

Implements [Digikam::AbstractDetector](#).

## 9.199 Digikam::ContentAwareContainer Class Reference

### Public Types

- enum **EnergyFunction** {  
    **GradientNorm** = 0 , **SumOfAbsoluteValues** , **XAbsoluteValue** , **LumaGradientNorm** ,  
    **LumaSumOfAbsoluteValues** , **LumaXAbsoluteValue** }

### Public Attributes

- EnergyFunction **func** = GradientNorm
- uint **height** = 0
- QImage **mask**
- bool **preserve\_skin\_tones** = false
- Qt::Orientation **resize\_order** = Qt::Horizontal
- double **rigidity** = 0.0
- int **side\_switch\_freq** = 4
- int **step** = 1
- uint **width** = 0



## 9.200 Digikam::ContentAwareFilter Class Reference

Inheritance diagram for Digikam::ContentAwareFilter:



### Public Member Functions

- **ContentAwareFilter** (`Dimg *const orgImage`, `QObject *const parent=nullptr`, `const ContentAwareContainer &settings=ContentAwareContainer()`)

- **ContentAwareFilter** (QObject \*const parent=nullptr)
- **FilterAction filterAction** () override  
*Returns the action description corresponding to currently set options.*
- QString **filterIdentifier** () const override  
*Return the identifier for this filter in the image history.*
- void **progressCallback** (int progress)
- void **readParameters** (const **FilterAction** &action) override

## Public Member Functions inherited from **Digikam::DImgThreadedFilter**

- **DImgThreadedFilter** (DImg \*const orgImage, QObject \*const parent, const QString &name=QString())  
*Constructs a filter with all arguments (ready to use).*
- **DImgThreadedFilter** (QObject \*const parent=nullptr, const QString &name=QString())  
*Constructs a filter without argument.*
- const QString & **filterName** ()
- int **filterVersion** () const
- **DImg getTargetImage** ()
- QList< int > **multithreadedSteps** (int stop, int start=0) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool **parametersSuccessfullyRead** () const  
*Optional: error handling for readParameters.*
- virtual QString **readParametersError** (const **FilterAction** &actionThatFailed) const
- void **setFilterName** (const QString &name)
- void **setFilterVersion** (int version)  
*Replaying a filter action: Set the filter version.*
- void **setOriginalImage** (const **DImg** &orgImage)
- void **setupAndStartDirectly** (const **DImg** &orgImage, **DImgThreadedFilter** \*const master, int progress←Begin=0, int progressEnd=100)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void **setupFilter** (const **DImg** &orgImage)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void **startFilter** ()  
*Start the threaded computation.*
- virtual void **startFilterDirectly** ()  
*Start computation of this filter, directly in this thread.*
- virtual QList< int > **supportedVersions** () const

## Public Member Functions inherited from **Digikam::DynamicThread**

- **DynamicThread** (QObject \*const parent=nullptr)  
*This class extends QRunnable, so you have to reimplement virtual void [run\(\)](#).*
- **~DynamicThread** () override  
*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool **isFinished** () const
- bool **isRunning** () const
- QThread::Priority **priority** () const
- void **setEmitSignals** (bool emitThem)
- void **setPriority** (QThread::Priority priority)  
*Sets the priority for this dynamic thread.*
- State **state** () const

### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from [Digikam::DImgThreadedFilter](#)

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.200.1 Member Function Documentation

### 9.200.1.1 filterAction()

`FilterAction` Digikam::ContentAwareFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.200.1.2 filterIdentifier()

`QString` Digikam::ContentAwareFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.200.1.3 readParameters()

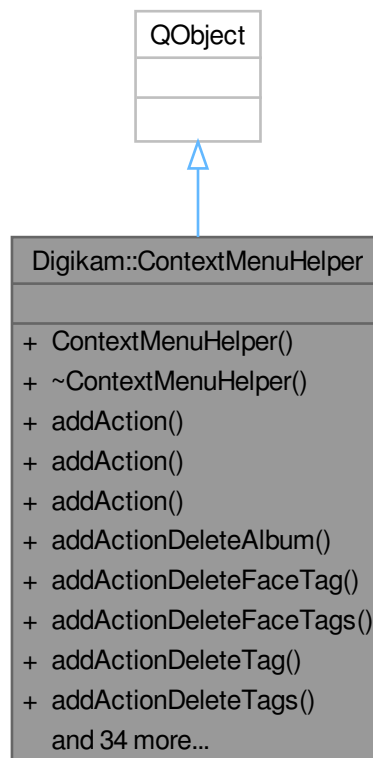
`void` Digikam::ContentAwareFilter::readParameters (   
     const `FilterAction` & *action* ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

## 9.201 Digikam::ContextMenuHelper Class Reference

A helper class to add actions and special menus to the context menu.

Inheritance diagram for Digikam::ContextMenuHelper:



## Public Types

- typedef const QList< qlonglong > **imagelds**

## Signals

- void **signalAddNewTagFromABCMenu** (const QString &)
- void **signalAddToExistingQueue** (int)
- void **signalAssignColorLabel** (int)
- void **signalAssignPickLabel** (int)
- void **signalAssignRating** (int)
- void **signalAssignTag** (int)
- void **signalCreateGroup** ()
- void **signalCreateGroupByFilename** ()
- void **signalCreateGroupByTime** ()
- void **signalCreateGroupByTimelapse** ()
- void **signalGotoAlbum** (const ItemInfo &)
- void **signalGotoDate** (const ItemInfo &)
- void **signalGotoTag** (int)
- void **signalPopupTagsView** ()
- void **signalRemoveFromGroup** ()
- void **signalRemoveTag** (int)
- void **signalSetThumbnail** (const ItemInfo &)
- void **signalUngroup** ()

## Public Member Functions

- [ContextMenuHelper](#) (QMenu \*const parent)  
*Constructs the helper class.*
- void [addAction](#) (const QString &name, bool addDisabled=false)  
*Add an action from the actionCollection.*
- void [addAction](#) (QAction \*const action, bool addDisabled=false)  
*Add a temporary action.*
- void [addAction](#) (QAction \*const action, QObject \*const recv, const char \*const slot, bool addDisabled=false)  
*Add a temporary action and assign it to a custom slot.*
- void [addActionDeleteAlbum](#) ([AlbumModificationHelper](#) \*const helper, [PAAlbum](#) \*const album)
- void [addActionDeleteFaceTag](#) ([TagModificationHelper](#) \*const helper, [TAAlbum](#) \*const tag)  
*Add action to delete tags from people sidebar.*
- void [addActionDeleteFaceTags](#) ([TagModificationHelper](#) \*const helper, const QList< [TAAlbum](#) \* > &tags)
- void [addActionDeleteTag](#) ([TagModificationHelper](#) \*const helper, [TAAlbum](#) \*const tag)
- void [addActionDeleteTags](#) ([TagModificationHelper](#) \*const helper, const QList< [TAAlbum](#) \* > &tags)
- void [addActionEditAlbum](#) ([AlbumModificationHelper](#) \*const helper, [PAAlbum](#) \*const album)
- void [addActionEditTag](#) ([TagModificationHelper](#) \*const helper, [TAAlbum](#) \*const tag)
- void [addActionNewAlbum](#) ([AlbumModificationHelper](#) \*const helper, [PAAlbum](#) \*const parentAlbum=nullptr)  
*Add actions to add, remove or edit a tag.*
- void [addActionNewTag](#) ([TagModificationHelper](#) \*const helper, [TAAlbum](#) \*const parentTag=nullptr)  
*Add actions to add, remove or edit a tag.*
- void [addActionRenameAlbum](#) ([AlbumModificationHelper](#) \*const helper, [PAAlbum](#) \*const album)
- void [addActionResetAlbumIcon](#) ([AlbumModificationHelper](#) \*const helper, [PAAlbum](#) \*const album)
- void [addActionTagsToFaceTags](#) ([TagModificationHelper](#) \*const helper, const QList< [TAAlbum](#) \* > &tags)
- void [addActionTagToFaceTag](#) ([TagModificationHelper](#) \*const helper, [TAAlbum](#) \*const tag)  
*Add action to set tags as face tags.*

- void [addAlbumCheckUncheckActions](#) ([Album](#) \*const album)  
*Add a Select and Deselect menu to check and uncheck albums.*
- void [addAssignTagsMenu](#) (const [imageIds](#) &ids)  
*Add "Assign Tags" menu.*
- void [addCreateTagFromAddressbookMenu](#) ()  
*Add a menu to create new tags from addressbook entries.*
- void [addExportMenu](#) ()  
*Add Export Webservices actions menu.*
- void [addGotoMenu](#) (const [imageIds](#) &ids)  
*Add the Goto menu.*
- void [addGroupActions](#) (const [imageIds](#) &ids)
- void [addGroupMenu](#) (const [imageIds](#) &ids, const [QList](#)< [QAction](#) \* > &extraMenuItems=[QList](#)< [QAction](#) \* >())  
*Add a "Group" menu.*
- void [addImportMenu](#) ()  
*Add Import Webservices actions menu.*
- void [addIQSAction](#) ([QObject](#) \*const recv, const char \*const slot)  
*Add the standard Image Quality Sorter action and connect it to the appropriate slot.*
- void [addLabelsAction](#) ()  
*Add "Pick/Color/Rating Labels" action.*
- void [addOpenAndNavigateActions](#) (const [imageIds](#) &ids, bool lightTable=false)  
*Add section for main views for opening and moving/going to albums.*
- void [addQueueManagerMenu](#) ()  
*Add Queue Manager actions menu.*
- void [addRemoveAllTags](#) (const [imageIds](#) &ids)  
*Add "Remove all Tags" action.*
- void [addRemoveTagsMenu](#) (const [imageIds](#) &ids)  
*Add "Remove Tags" menu.*
- void [addSeparator](#) ()  
*Add a separator to the context menu.*
- void [addServicesMenu](#) (const [QList](#)< [QUrl](#) > &selectedItems)  
*Add the services menu to the menu.*
- void [addStandardActionCopy](#) ([QObject](#) \*const recv, const char \*const slot)  
*Add the standard copy action and connect it to the appropriate slot.*
- void [addStandardActionCut](#) ([QObject](#) \*const recv, const char \*const slot)  
*Add the standard cut action and connect it to the appropriate slot.*
- void [addStandardActionItemDelete](#) ([QObject](#) \*const recv, const char \*const slot, int quantity=1)  
*Add the standard delete action and connect it to the appropriate slot.*
- void [addStandardActionLightTable](#) ()  
*Add the lighttable action to the menu.*
- void [addStandardActionPaste](#) ([QObject](#) \*const recv, const char \*const slot)  
*Add the standard paste action and connect it to the appropriate slot.*
- void [addStandardActionThumbnail](#) (const [imageIds](#) &ids, [Album](#) \*const album)  
*Add the thumbnail action to the menu.*
- void [addSubMenu](#) ([QMenu](#) \*subMenu)  
*Add a submenu to the parent context menu.*
- [QAction](#) \* [exec](#) (const [QPoint](#) &pos, [QAction](#) \*const at=nullptr)  
*Execute the registered parent menu and evaluate the triggered actions.*
- void [setAlbumModel](#) ([AbstractCheckableAlbumModel](#) \*const model)  
*Set an album model.*
- void [setItemFilterModel](#) ([ItemFilterModel](#) \*const model)  
*Set a filter model.*

### 9.201.1 Detailed Description

The ContextMenuHelper class helps adding commonly used actions and menus. Use this class to add

- actions from the actionCollection
- standard actions (copy, paste, delete)
- temporary actions
- predefined special actions
- predefined submenus to the menu.

All [addAction\(\)](#) methods take a special parameter 'addDisabled'. This parameter controls if disabled actions are added to the menu. Normally adding disabled actions is turned off, to clean up the menu and make it more readable.

If the ContextMenuHelper class is used, you need to call its own [exec\(\)](#) method, instead the one from the parent menu. This way signals from special menus can be emitted and connected to the appropriate slots.

### 9.201.2 Constructor & Destructor Documentation

#### 9.201.2.1 ContextMenuHelper()

```
Digikam::ContextMenuHelper::ContextMenuHelper (
    QMenu *const parent ) [explicit]
```

##### Parameters

<i>parent</i>	the menu the helper class is linked to
---------------	--

### 9.201.3 Member Function Documentation

#### 9.201.3.1 addAction() [1/3]

```
void Digikam::ContextMenuHelper::addAction (
    const QString & name,
    bool addDisabled = false )
```

This method adds actions from the actionCollection. The actionCollection can be set in the constructor of the [ContextMenuHelper](#) class.

##### Parameters

<i>name</i>	the name of the action in the actionCollection
<i>addDisabled</i>	if set, disabled actions are added to the menu



**9.201.3.2 addAction()** [2/3]

```
void Digikam::ContextMenuHelper::addAction (
    QAction *const action,
    bool addDisabled = false )
```

Sometimes it is necessary to define actions that only exist in the current context menu content. Use this method to add such an action.

**Parameters**

<i>action</i>	the action to add
<i>addDisabled</i>	if set, disabled actions are added to the menu

**9.201.3.3 addAction()** [3/3]

```
void Digikam::ContextMenuHelper::addAction (
    QAction *const action,
    QObject *const recv,
    const char *const slot,
    bool addDisabled = false )
```

Use this method if you want to add a temporary action and immediately connect it to the receiving slot.

**Parameters**

<i>action</i>	the action to add
<i>recv</i>	the receiver of the triggered action
<i>slot</i>	the slot to connect the triggered action to
<i>addDisabled</i>	if set, disabled actions are added to the menu

**9.201.3.4 addActionNewAlbum()**

```
void Digikam::ContextMenuHelper::addActionNewAlbum (
    AlbumModificationHelper *const helper,
    PAlbum *const parentAlbum = nullptr )
```

The tag modification helper is used to execute the action. You must set the parent tag to use on modification helper.

**9.201.3.5 addActionNewTag()**

```
void Digikam::ContextMenuHelper::addActionNewTag (
    TagModificationHelper *const helper,
    TAlbum *const parentTag = nullptr )
```

The tag modification helper is used to execute the action. You must set the parent tag to use on modification helper.

### 9.201.3.6 addAlbumCheckUncheckActions()

```
void Digikam::ContextMenuHelper::addAlbumCheckUncheckActions (
    Album *const album )
```

Note: Call setAlbumModel before, or this will have no effect.

### 9.201.3.7 addAssignTagsMenu()

```
void Digikam::ContextMenuHelper::addAssignTagsMenu (
    const imageIds & ids )
```

This menu will provide a list of all tags available so that they can be assigned to the current selected items.

To make this menu work, you need to run [exec\(\)](#) from this class, otherwise the signals are not emitted and you will not be able to react on triggered actions from this menu. Make sure to connect the signals to the appropriate slots in the context menu handling method.

#### Parameters

<i>ids</i>	the selected items
------------	--------------------

#### See also

[exec\(\)](#)  
[signalAssignTag\(\)](#)

### 9.201.3.8 addGotoMenu()

```
void Digikam::ContextMenuHelper::addGotoMenu (
    const imageIds & ids )
```

This menu will provide the following actions for the given item:

- Goto [Album](#)
- Goto Date
- Goto Tag To make this menu work, you need to run [exec\(\)](#) from this class, otherwise the signals are not emitted and you will not be able to react on triggered actions from this menu. Make sure to connect the signals to the appropriate slots in the context menu handling method.

#### Parameters

<i>ids</i>	the list of selected items
------------	----------------------------

#### See also

[exec\(\)](#)  
[signalGotoAlbum\(\)](#) [signalGotoDate\(\)](#) [signalGotoTag\(\)](#)

TODO:tags to be ported to multiple selection

### 9.201.3.9 addGroupMenu()

```
void Digikam::ContextMenuHelper::addGroupMenu (
    const imageIds & ids,
    const QList< QAction * > & extraMenuItems = QList<QAction*>() )
```

This menu will provide actions open, close, add to, remove from, or split a group.

addGroupActions will add the actions as a flat list, not in a submenu. Note: Call setItemFilterModel before to have Open/Close group actions.

### 9.201.3.10 addIQSAction()

```
void Digikam::ContextMenuHelper::addIQSAction (
    QObject *const rcv,
    const char *const slot )
```

#### Parameters

<i>rcv</i>	the receiver of the triggered action
<i>slot</i>	the slot to connect the triggered action to

### 9.201.3.11 addLabelsAction()

```
void Digikam::ContextMenuHelper::addLabelsAction ( )
```

This action will provide methods to assign pick/color/rating labels to the currently selected items.

To make this menu work, you need to run [exec\(\)](#) from this class, otherwise the signals are not emitted and you will not be able to react on triggered actions from this menu. Make sure to connect the signals to the appropriate slots in the context menu handling method.

#### See also

- [exec\(\)](#)
- [signalAssignPickLabel\(\)](#)
- [signalAssignColorLabel\(\)](#)
- [signalAssignRating\(\)](#)

### 9.201.3.12 addOpenAndNavigateActions()

```
void Digikam::ContextMenuHelper::addOpenAndNavigateActions (
    const imageIds & ids,
    bool lightTable = false )
```

This is a convenience function to ensure consistent menus and reduce code duplication.

## Parameters

<i>ids</i>	the list of selected items
<i>lightTable</i>	for the light table

**9.201.3.13 addRemoveAllTags()**

```
void Digikam::ContextMenuHelper::addRemoveAllTags (
    const imageIds & ids )
```

Removes all tags from the selected item ids except face tags.

## Parameters

<i>ids</i>	the selected items
------------	--------------------

**9.201.3.14 addRemoveTagsMenu()**

```
void Digikam::ContextMenuHelper::addRemoveTagsMenu (
    const imageIds & ids )
```

This menu will provide a list of all tags assigned to the current items. Actions triggered in here will remove the selected tag from the items.

To make this menu work, you need to run [exec\(\)](#) from this class, otherwise the signals are not emitted and you will not be able to react on triggered actions from this menu. Make sure to connect the signals to the appropriate slots in the context menu handling method.

## Parameters

<i>ids</i>	the selected items
------------	--------------------

## See also

[exec\(\)](#)  
[signalRemoveTag\(\)](#)

**9.201.3.15 addServicesMenu()**

```
void Digikam::ContextMenuHelper::addServicesMenu (
    const QList< QUrl > & selectedItems )
```

The services menu is used to open the selected items in a different application. It will query the item for registered services and provide them in a submenu. The menu will be titled "Open With...".

## Parameters

<i>selectedItems</i>	the list of selected items
----------------------	----------------------------

**9.201.3.16 addStandardActionCopy()**

```
void Digikam::ContextMenuHelper::addStandardActionCopy (
    QObject *const recv,
    const char *const slot )
```

**Parameters**

<i>recv</i>	the receiver of the triggered action
<i>slot</i>	the slot to connect the triggered action to

**9.201.3.17 addStandardActionCut()**

```
void Digikam::ContextMenuHelper::addStandardActionCut (
    QObject *const recv,
    const char *const slot )
```

**Parameters**

<i>recv</i>	the receiver of the triggered action
<i>slot</i>	the slot to connect the triggered action to

**9.201.3.18 addStandardActionItemDelete()**

```
void Digikam::ContextMenuHelper::addStandardActionItemDelete (
    QObject *const recv,
    const char *const slot,
    int quantity = 1 )
```

**Parameters**

<i>recv</i>	the receiver of the triggered action
<i>slot</i>	the slot to connect the triggered action to
<i>quantity</i>	the number of the files that should be deleted. This parameter is used for the action name and is normally used when deleting more then one item.

**9.201.3.19 addStandardActionLightTable()**

```
void Digikam::ContextMenuHelper::addStandardActionLightTable ( )
```

Do not use [addAction\(\)](#) to add the lighttable action, because we need to handle special cases here. Depending on whether the lighttable window has already been created and filled with items, we set different actions.

**9.201.3.20 addStandardActionPaste()**

```
void Digikam::ContextMenuHelper::addStandardActionPaste (
    QObject *const recv,
    const char *const slot )
```

## Parameters

<i>recv</i>	the receiver of the triggered action
<i>slot</i>	the slot to connect the triggered action to

**9.201.3.21 addStandardActionThumbnail()**

```
void Digikam::ContextMenuHelper::addStandardActionThumbnail (
    const imageIds & ids,
    Album *const album )
```

Do not use [addAction\(\)](#) to add the thumbnail action, because we need to handle special cases here. Depending on whether the current view is album or icon view, we set different actions.

## Parameters

<i>ids</i>	the selected items in the current view
<i>album</i>	the current album the AlbumIconView is displaying

**9.201.3.22 addSubMenu()**

```
void Digikam::ContextMenuHelper::addSubMenu (
    QMenu * subMenu )
```

## Parameters

<i>subMenu</i>	the submenu to be added
----------------	-------------------------

**9.201.3.23 exec()**

```
QAction * Digikam::ContextMenuHelper::exec (
    const QPoint & pos,
    QAction *const at = nullptr )
```

Always use this method instead the one from the parent menu. It will ensure that the signals are emitted and special cases are handled.

## Parameters

<i>pos</i>	position of the triggered action in the registered menu
<i>at</i>	the action that should be at the position <i>pos</i>

## Returns

the triggered action

### 9.201.3.24 setAlbumModel()

```
void Digikam::ContextMenuHelper::setAlbumModel (
    AbstractCheckableAlbumModel *const model )
```

The check/uncheck actions will operate directly on the model.

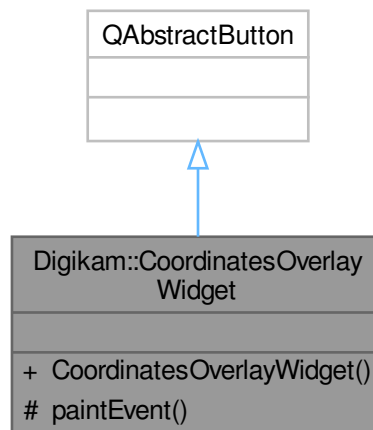
### 9.201.3.25 setItemFilterModel()

```
void Digikam::ContextMenuHelper::setItemFilterModel (
    ItemFilterModel *const model )
```

Some of the group actions will operate directly on the model.

## 9.202 Digikam::CoordinatesOverlayWidget Class Reference

Inheritance diagram for Digikam::CoordinatesOverlayWidget:



### Public Member Functions

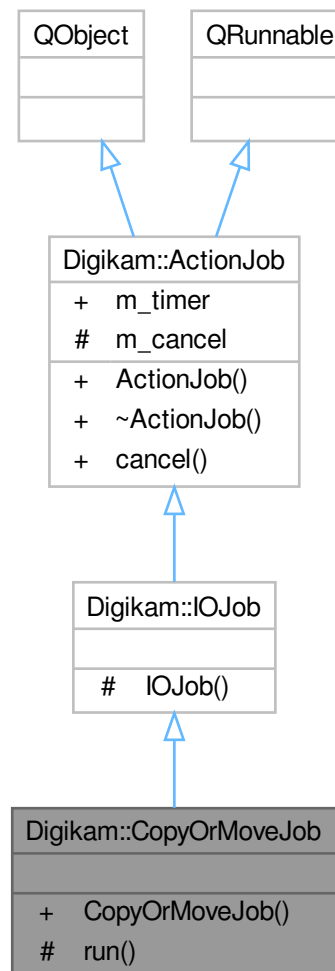
- **CoordinatesOverlayWidget** (`QWidget *const parent=nullptr`)

### Protected Member Functions

- void **paintEvent** (`QPaintEvent *`) override

## 9.203 Digikam::CopyOrMoveJob Class Reference

Inheritance diagram for Digikam::CopyOrMoveJob:



### Public Member Functions

- **CopyOrMoveJob** ([IOJobData](#) \*const data)

### Public Member Functions inherited from [Digikam::ActionJob](#)

- **ActionJob** ([QObject](#) \*const parent=nullptr)  
*Constructor which delegate deletion of [QRunnable](#) instance to [ActionThreadBase](#), not [QThreadPool](#).*
- **~ActionJob** () override  
*Re-implement destructor in you implementation.*



### Protected Member Functions

- void **run** () override

### Additional Inherited Members

### Public Slots inherited from [Digikam::ActionJob](#)

- void **cancel** ()  
*Call this method to cancel job.*

### Signals inherited from [Digikam::IOJob](#)

- void **signalError** (const QString &errMsg)
- void **signalOneProcessed** (const QUrl &url)

### Signals inherited from [Digikam::ActionJob](#)

- void **signalDone** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job is done.*
- void **signalProgress** (int)  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job progress.*
- void **signalStarted** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager that job is started.*

### Public Attributes inherited from [Digikam::ActionJob](#)

- QElapsedTimer **m\_timer**  
*Timer to determine the running time of the job.*

### Protected Attributes inherited from [Digikam::ActionJob](#)

- bool **m\_cancel** = false  
*You can use this boolean in your implementation to know if job must be canceled.*

## 9.204 Digikam::CopyrightInfo Class Reference

### Public Member Functions

- bool **isNull** () const

### Public Attributes

- QString **extraValue**
- qlonglong **id** = -1
- QString **property**
- QString **value**

## 9.205 Digikam::CoreDB Class Reference

### Public Types

- enum **CopyrightPropertyUnique** { **PropertyUnique** , **PropertyExtraValueUnique** , **PropertyNoConstraint** }
- enum **ItemSortOrder** { **NoItemSorting** , **ByItemName** , **ByItemPath** , **ByItemDate** , **ByItemRating** }

### Public Member Functions

- **CoreDB** ([CoreDbBackend](#) \*const backend)  
*Constructor.*
- **~CoreDB** ()  
*Destructor.*
- int [addAlbum](#) (int albumRootId, const QString &relativePath, const QString &caption, const QDate &date, const QString &collection) const  
*Add a new album to the database with the given attributes.*
- int [addAlbumRoot](#) ([CollectionLocation::Type](#) type, const QString &identifier, const QString &specificPath, const QString &label) const  
*Add a new album to the database with the given attributes.*
- void [addImageMetadata](#) (qulonglong imageID, const QVariantList &infos, DatabaseFields::ImageMetadata fields=DatabaseFields::ImageMetadataAll)  
*Add (or replace) the ImageMetadata of the specified item.*
- void [addImageRelation](#) (const [ImageRelation](#) &relation)
- void [addImageRelation](#) (qulonglong subjectId, qulonglong objectId, DatabaseRelation::Type type)  
*Adds an image relation entry.*
- void [addImageRelations](#) (const QList< qulonglong > &subjectIds, const QList< qulonglong > &objectIds, DatabaseRelation::Type type)  
*This method requires two lists of same size and will add list1[0]->list2[0],...,list1[n]->list2[n].*
- void [addImageTagProperty](#) (const [ImageTagProperty](#) &property)
- void [addImageTagProperty](#) (qulonglong imageId, int tagId, const QString &property, const QString &value)  
*Adds a tag property.*
- qulonglong [addItem](#) (int albumID, const QString &name, DatabaseItem::Status status, DatabaseItem::Category category, const QDateTime &modificationDate, qulonglong fileSize, const QString &uniqueHash) const  
*Put a new item in the database or replace an existing one.*
- void [addItemInformation](#) (qulonglong imageID, const QVariantList &infos, DatabaseFields::ItemInformation fields=DatabaseFields::ItemInformationAll)  
*Add (or replace) the ItemInformation of the specified item.*
- void [addItemPosition](#) (qulonglong imageID, const QVariantList &infos, DatabaseFields::ItemPositions fields=DatabaseFields::ItemPositionsAll)  
*Add (or replace) the ItemPosition of the specified item.*
- void [addItemTag](#) (int albumID, const QString &name, int tagID)  
*Add a tag for the item.*
- void [addItemTag](#) (qulonglong imageID, int tagID, bool newTag=false)  
*Add a tag for the item.*
- int [addSearch](#) (DatabaseSearch::Type type, const QString &name, const QString &query) const  
*Add a new search to the database with the given attributes.*
- int [addTag](#) (int parentTagID, const QString &name, const QString &iconKDE, qulonglong iconID) const  
*Adds a new tag to the database with given name, icon and parent id.*

- void **addTagProperty** (const [TagProperty](#) &property)
- void **addTagProperty** (int tagId, const QString &property, const QString &value)
 

*Adds a tag property.*
- void **addTagsToItems** (const QList< qlonglong > &imageIDs, const QList< int > &tagIDs)
 

*Add each tag of a list of tags to each member of a list of items.*
- int **addToDownloadHistory** (const QString &identifier, const QString &name, qlonglong fileSize, const QDateTime &date) const
 

*Add the specified fingerprint to the download history table.*
- void **addVideoMetadata** (qlonglong imageID, const QVariantList &infos, DatabaseFields::VideoMetadata fields=DatabaseFields::VideoMetadataAll)
 

*Add (or replace) the VideoMetadata of the specified item.*
- void **changeImageComment** (int commentId, qlonglong imageID, const QVariantList &infos, DatabaseFields::ItemComments fields=DatabaseFields::ItemCommentsAll)
 

*Changes the properties of a comment.*
- void **changeImageMetadata** (qlonglong imageID, const QVariantList &infos, DatabaseFields::ImageMetadata fields=DatabaseFields::ImageMetadataAll)
 

*Change the indicated fields of the image information for the specified item.*
- void **changeItemInformation** (qlonglong imageID, const QVariantList &infos, DatabaseFields::ItemInformation fields=DatabaseFields::ItemInformationAll)
 

*Change the indicated fields of the image information for the specified item.*
- void **changeItemPosition** (qlonglong imageID, const QVariantList &infos, DatabaseFields::ItemPositions fields=DatabaseFields::ItemPositionsAll)
 

*Change the indicated fields of the image information for the specified item.*
- void **changeVideoMetadata** (qlonglong imageID, const QVariantList &infos, DatabaseFields::VideoMetadata fields=DatabaseFields::VideoMetadataAll)
 

*Change the indicated fields of the video information for the specified item.*
- bool **copyAlbumProperties** (int srcAlbumID, int dstAlbumID) const
 

*Copy the properties of the given srcAlbum to the dstAlbum.*
- void **copyImageAttributes** (qlonglong srcId, qlonglong destId)
 

*Copies all image-specific information, in all tables, from image srcId to destId.*
- void **copyImageProperties** (qlonglong srcId, qlonglong destId)
 

*Copies all entries in the ImageProperties table.*
- void **copyImageTags** (qlonglong srcId, qlonglong destId)
 

*Copies all entries in the ImageTags table.*
- qlonglong **copyItem** (int srcAlbumID, const QString &srcName, int dstAlbumID, const QString &dstName)
 

*Copy the attributes of an item to a different item.*
- QUuid **databaseUuid** ()
 

*Returns a UUID for the database file.*
- void **deleteAlbum** (int albumID)
 

*Deletes an album from the database.*
- void **deleteAlbumRoot** (int rootId)
 

*Deletes an album root from the database.*
- void **deleteItem** (int albumID, const QString &file)
 

*Deletes an item from the database.*
- void **deleteItem** (qlonglong imageId)
 

*Deletes an item from the database if it does not belong to an album.*
- void **deleteObsoleteItem** (qlonglong imageId)
 

*Deletes an item from the database without checking the album.*
- void **deleteRemovedItems** ()
 

*Delete all items from the database that are marked as removed.*
- void **deleteSearch** (int searchID)
 

*Delete a search from the database.*

- void **deleteSearches** (DatabaseSearch::Type type)
  - Delete all search with the given type.*
- void **deleteStaleAlbums** ()
  - Deletes albums from the database that were previously removed with [makeStaleAlbum\(\)](#)*
- void **deleteTag** (int tagID)
  - Deletes a tag from the database.*
- QList< qlonglong > **findByNameAndCreationDate** (const QString &fileName, const QDateTime &creationDate) const
  - Returns all items with the given file name and creation date.*
- qlonglong **findImageId** (int albumID, const QString &name, DatabaseItem::Status status, DatabaseItem::Category category, qlonglong fileSize, const QString &uniqueHash) const
  - Find the imageId fitting to the information given for the item.*
- int **findInDownloadHistory** (const QString &identifier, const QString &name, qlonglong fileSize, const QDateTime &date) const
  - Search for the specified fingerprint in the download history table.*
- QList< int > **getAlbumAndSubalbumsForPath** (int albumRootId, const QString &relativePath) const
  - Find out the album ids for a given relative path, including the subalbums.*
- QDate **getAlbumAverageDate** (int albumID) const
  - Returns the average date of all images for that album.*
- int **getAlbumForPath** (int albumRootId, const QString &relativePath, bool create=true) const
  - Find out the album for a given folder.*
- QDate **getAlbumHighestDate** (int albumID) const
  - Returns the highest/newest date of all images for that album.*
- QDate **getAlbumLowestDate** (int albumID) const
  - Returns the lowest/oldest date of all images for that album.*
- QDateTime **getAlbumModificationDate** (int albumID) const
  - Returns the QDateTime of the album modification date.*
- QMap< QString, QDateTime > **getAlbumModificationMap** (int albumRootId) const
  - Returns a QMap with relative path and the album modification date.*
- QString **getAlbumRelativePath** (int albumID) const
  - Given an albumid, this returns the relative path for that album (the path below the album root, starting with a slash)*
- int **getAlbumRootId** (int albumID) const
  - Given an albumid, this returns the album root id for that album.*
- QList< AlbumRootInfo > **getAlbumRoots** () const
  - Returns all albums and their attributes in the database.*
- QList< AlbumShortInfo > **getAlbumShortInfos** () const
  - Returns all albums in the database with their albumRoot and ID, ordered by id.*
- QList< int > **getAlbumsOnAlbumRoot** (int albumRootId) const
  - Find out all album ids of a given album root.*
- QVariantList **getAllCreationDates** () const
  - Returns a QVariantList of creationDate of all items.*
- QStringList **getAllImagePropertiesByName** (const QString &property) const
- QList< qlonglong > **getAllItems** () const
  - Returns all ids of items in images table.*
- QHash< qlonglong, QPair< int, int > > **getAllItemsWithAlbum** () const
  - Returns all ids of items with album ids in images table.*
- QString **getDatabaseEncoding** () const
  - Returns database encoding.*
- QStringList **getDirtyOrMissingFaceImageUrls** () const
  - Returns a list of all images where the Faces have either not been detected yet, or is outdated because the file is identified as changed since the generation of the fingerprint.*

- void [getFilterSettings](#) (QStringList \*imageFilter, QStringList \*videoFilter, QStringList \*audioFilter)
 

*Get the settings for the file name filters of this database.*
- qlonglong [getFirstItemWithFaceTag](#) (int tagId) const
 

*Returns the first item that has a confirmed face with the tag.*
- QMap< QString, int > [getFormatStatistics](#) () const
 

*Returns a QMap<QString,int> of ItemInformation.format corresponding to count of items with that format.*
- QMap< QString, int > [getFormatStatistics](#) (DatabaseItem::Category category) const
- QList< [ItemScanInfo](#) > [getIdenticalFiles](#) (const QString &uniqueHash, qlonglong fileSize, qlonglong sourceId=-1) const
- QList< [ItemScanInfo](#) > [getIdenticalFiles](#) (qlonglong id) const
 

*Find items that are, with reasonable certainty, identical to the file pointed to by id.*
- void [getIgnoreDirectoryFilterSettings](#) (QStringList \*ignoreDirectoryFilter)
- qlonglong [getImageId](#) (int albumId, const QString &name) const
 

*Get the imageId of the item.*
- QList< qlonglong > [getImageIds](#) (DatabaseItem::Status status) const
 

*Returns all image ids with the given status.*
- QList< qlonglong > [getImageIds](#) (DatabaseItem::Status status, DatabaseItem::Category category) const
 

*Returns all image ids with the given status and category.*
- QList< qlonglong > [getImageIds](#) (int albumId, const QString &name, DatabaseItem::Status status) const
 

*Get the imageId fitting to the information given for the item.*
- QList< qlonglong > [getImageIds](#) (int albumId, DatabaseItem::Status status, bool scanned=true) const
 

*Get the imageId fitting to the information given for the item.*
- QList< QVariant > [getImageIdsFromArea](#) (qreal lat1, qreal lat2, qreal lng1, qreal lng2, int sortMode, const QString &sortBy) const
- QVariantList [getImageMetadata](#) (qlonglong imageId, DatabaseFields::ImageMetadata metadata← Fields=DatabaseFields::ImageMetadataAll) const
 

*Read image metadata.*
- QString [getImageProperty](#) (qlonglong imageId, const QString &property) const
 

*Returns the property with the specified name for the specified image.*
- QVariantList [getImageFields](#) (qlonglong imageId, DatabaseFields::Images imageFields) const
 

*Returns the requested fields from the Images table.*
- QVector< QList< qlonglong > > [getImageRelatedFrom](#) (const QList< qlonglong > &subjectIds, DatabaseRelation::Type type=DatabaseRelation::UndefinedType) const
- QList< qlonglong > [getImageRelatedFrom](#) (qlonglong subjectId, DatabaseRelation::Type type=Database← Relation::UndefinedType) const
 

*Retrieves all images that the given image is related to (retrieves objects, given image is subject) If type is given, filters by type, otherwise returns all types.*
- QVector< QList< qlonglong > > [getImageRelatingTo](#) (const QList< qlonglong > &objectIds, Database← Relation::Type type=DatabaseRelation::UndefinedType) const
- QList< qlonglong > [getImageRelatingTo](#) (qlonglong objectId, DatabaseRelation::Type type=Database← Relation::UndefinedType) const
 

*Retrieves all images that relate to the given image (retrieves subject, given image is object) If type is given, filters by type, otherwise returns all types.*
- QList< qlonglong > [getImageWithTagProperty](#) (int tagId, const QString &property) const
 

*Returns all image ids that are associated to the tag with the given property.*
- QList< qlonglong > [getImageWithProperty](#) (const QString &property) const
 

*Returns all image ids that are associated to the given property.*
- QList< [ImageTagProperty](#) > [getImageTagProperties](#) (qlonglong imageId, int tagId=-1) const
 

*Get the properties for the given image/tag pair.*
- QString [getImageUuid](#) (qlonglong imageId) const
 

*Retrieves the image UUID.*
- int [getItemAlbum](#) (qlonglong imageId) const

- Find the album of an item.*

  - QList< [CommentInfo](#) > **getItemComments** (qlonglong imageID) const  
*Retrieves all available comments for the specified item.*
  - QList< int > **getItemCommonTagIDs** (const QList< qlonglong > &imageIDList) const  
*Given a set of items (identified by their IDs), get a list of ID of all common tags.*
  - QList< [CopyrightInfo](#) > **getItemCopyright** (qlonglong imageID, const QString &property=QString()) const  
*Returns the copyright properties of the specified image.*
  - qlonglong **getItemFromAlbum** (int albumID, const QString &fileName) const  
*Returns the id of the item with the given filename in the album with the given id.*
  - [ImageHistoryEntry](#) **getItemHistory** (qlonglong imageID) const  
*Retrieves the history entry for the given image.*
  - QMap< qlonglong, QString > **getItemIDsAndURLsInAlbum** (int albumID) const  
*Given a albumID, get a map of ids and urls of all items in the album.*
  - QList< qlonglong > **getItemIDsInAlbum** (int albumID) const  
*Given a albumID, get a list of ids of all items in the album.*
  - QList< qlonglong > **getItemIDsInTag** (int tagID, bool recursive=false) const  
*Given a tagID, get a list of ids of all items in the tag.*
  - QVariantList **getItemInformation** (qlonglong imageID, DatabaseFields::ItemInformation infoFields=DatabaseFields::ItemInformationAll) const  
*Read image information.*
  - QString **getItemName** (qlonglong imageID) const  
*Retrieve the name of the item.*
  - QStringList **getItemNamesInAlbum** (int albumID, bool recursive=false) const  
*Returns all items for a given albumid.*
  - QVariantList **getItemPosition** (qlonglong imageID, DatabaseFields::ItemPositions positionFields=DatabaseFields::ItemPositionsAll) const  
*Read image metadata.*
  - QVariantList **getItemPositions** (const QList< qlonglong > &imageIDs, DatabaseFields::ItemPositions fields) const
  - [ItemScanInfo](#) **getItemScanInfo** (qlonglong imageID) const  
*Get scan info from the image ID.*
  - QList< [ItemScanInfo](#) > **getItemScanInfos** (int albumID) const  
*Returns an [ItemScanInfo](#) object for each item in the album with the specified album id.*
  - QList< qlonglong > **getItemsForUuid** (const QString &uuid) const  
*Retrieves the images with the given UUID.*
  - [ItemShortInfo](#) **getItemShortInfo** (int albumRootId, const QString &relativePath, const QString &name) const  
*Get item and album if from albumRootId, album path and file name.*
  - [ItemShortInfo](#) **getItemShortInfo** (qlonglong imageID) const  
*Get item and album info from the image ID.*
  - QVector< QList< int > > **getItemsTagIDs** (const QList< qlonglong > &imageIDs) const  
*For a list of items, return the tag ids associated with the item.*
  - QStringList **getItemsURLsWithTag** (int tagId) const  
*Returns a list of all images where tagId is assigned Return item URLs.*
  - QList< int > **getItemTagIDs** (qlonglong imageID) const  
*Get a list of IDs of all the tags for the item.*
  - QStringList **getItemTagNames** (qlonglong imageID) const  
*Get a list of names of all the tags for the item.*
  - QStringList **getItemURLsInAlbum** (int albumID, ItemSortOrder order=NoItemSorting) const  
*Given a albumID, get a list of the url of all items in the album.*
  - QStringList **getItemURLsInTag** (int tagID, bool recursive=false) const  
*Given a tagid, get a list of the url of all items in the tag.*



- QStringList **getListFromImageMetadata** (DatabaseFields::ImageMetadata field) const  
*Return a list from a field from imageMetadata.*
- QPair< int, int > **getNumberOfAllItemsAndAlbums** (int albumID) const  
*Returns the QPair<int, int> of all items (first) and albums (second) as a counter in the album.*
- QHash< int, int > **getNumberOfImagesInAlbums** () const  
*Returns a QHash<int, int> of album id -> count of items in the album.*
- QHash< int, int > **getNumberOfImagesInTagProperties** (const QString &property) const  
*Returns a QHash<int, int> of tag id -> count of items with the given tag property.*
- int **getNumberOfImagesInTagProperties** (int tagId, const QString &property) const  
*Returns the count of images that have a tag property for the given tag.*
- QHash< int, int > **getNumberOfImagesInTags** () const  
*Returns a QHash<int, int> of tag id -> count of items with the tag.*
- int **getNumberOfItemsInAlbum** (int albumID) const  
*Returns the number of items in the album.*
- QList< qlonglong > **getObsoleteItemIds** () const  
*Get obsolete item ids.*
- QList< qlonglong > **getOneRelatedImageEach** (const QList< qlonglong > &ids, DatabaseRelation::Type type=DatabaseRelation::UndefinedType) const  
*For each of the given ids, find one single related image (direction does not matter).*
- QList< int > **getRecentlyAssignedTags** () const  
*Get a list of recently assigned tags (only last 6 tags are listed)*
- QList< qlonglong > **getRelatedImagesToByType** (DatabaseRelation::Type type) const  
*Retrieves all images that related to (retrieves objects) by given type.*
- QList< QPair< qlonglong, qlonglong > > **getRelationCloud** (qlonglong imageId, DatabaseRelation::Type type=DatabaseRelation::UndefinedType) const  
*For the given image id, retrieves all relations of all related images: Each pair (a,b) means "a is related to b".*
- SearchInfo **getSearchInfo** (int searchId) const  
*Get information about the specified search.*
- QString **getSearchQuery** (int searchId) const  
*Get the query for the search specified by its id.*
- QString **getSetting** (const QString &keyword) const  
*This function returns the value which is stored in the database (table Settings).*
- QList< int > **getTagIdsWithProperties** (qlonglong imageId) const  
*Get all tagIds for which ImageTagProperties exist for the given image.*
- TagInfo **getTagInfo** (int tagId) const
- QList< TagProperty > **getTagProperties** () const  
*Returns the list of all tag properties (ordered by tag id, then property).*
- QList< TagProperty > **getTagProperties** (const QString &property) const  
*Returns the list of tag properties with the given attribute.*
- QList< TagProperty > **getTagProperties** (int tagID) const  
*Returns the list of tag properties of the given tag.*
- QList< TagShortInfo > **getTagShortInfos** () const  
*Returns all tags in the database with their parent id and name, ordered by id.*
- QList< int > **getTagsWithProperty** (const QString &property) const  
*Returns a list of tag ids with the specified property.*
- int **getUniqueHashVersion** () const  
*Returns the version used for the unique hash in this database.*
- void **getUserFilterSettings** (QString \*imageFilterString, QString \*videoFilterString, QString \*audioFilterString)  
*Returns the user-configurable filter settings.*
- void **getUserIgnoreDirectoryFilterSettings** (QString \*ignoreDirectoryFilterString)

- QVariantList [getVideoMetadata](#) (qlonglong imageID, DatabaseFields::VideoMetadata metadata← Fields=DatabaseFields::VideoMetadataAll) const  
*Read video metadata.*
- bool **hasImageHistory** (qlonglong imageid) const  
*Returns true if the image has a history stored in DB If not, it returns false.*
- bool **hasImagesRelatedFrom** (qlonglong subjectId, DatabaseRelation::Type type=DatabaseRelation::← UndefinedType) const
- bool **hasImagesRelatingTo** (qlonglong objectId, DatabaseRelation::Type type=DatabaseRelation::← UndefinedType) const
- bool **hasTags** (const QList< qlonglong > &imageIDList) const  
*Given a set of items (identified by their IDs), this will see if any of the items has a tag.*
- bool **integrityCheck** () const  
*Returns true if the integrity of the database is preserved.*
- void **makeStaleAlbum** (int albumID)  
*Makes the album a stale entry by setting the albumRoot to 0.*
- void **migrateAlbumRoot** (int rootId, const QString &identifier)  
*Migrates a given album root to a new disk location.*
- void **moveItem** (int srcAlbumID, const QString &srcName, int dstAlbumID, const QString &dstName)  
*Move the attributes of an item to a different item.*
- void **removeAllImageComments** (qlonglong imageID)  
*Remove all [ItemComments](#).*
- void **removeAllImageProperties** (qlonglong imageID)
- QList< qlonglong > **removeAllImageRelationsFrom** (qlonglong subjectId, DatabaseRelation::Type type) const
- QList< qlonglong > **removeAllImageRelationsTo** (qlonglong objectId, DatabaseRelation::Type type) const
- void **removeAllItemCopyrightProperties** (qlonglong imageID)  
*Removes all copyright properties for the given image id.*
- void **removeImageComment** (int commentId, qlonglong imageID)  
*Remove the specified entry in [ItemComments](#).*
- void **removeImageProperty** (qlonglong imageID, const QString &property)
- void **removeImagePropertyByName** (const QString &property)
- void **removeImageRelation** (const [ImageRelation](#) &relation)
- void **removeImageRelation** (qlonglong subjectId, qlonglong objectId, DatabaseRelation::Type type)  
*Removes image relations.*
- void **removeImageTagProperties** (qlonglong imageid, int tagId=-1, const QString &property=QString(), const QString &value=QString())  
*Removes properties for the given tag.*
- void **removeItemAllTags** (qlonglong imageID, const QList< int > &currentTagIds)  
*Remove all tags for the item.*
- void **removeItemCopyrightProperties** (qlonglong imageID, const QString &property=QString(), const QString &extraValue=QString(), const QString &value=QString())  
*Removes copyright properties for the given image id.*
- void **removeItemPosition** (qlonglong imageid)  
*Remove the entry in [ItemPositions](#) for the given image.*
- void **removeItemPositionAltitude** (qlonglong imageid)  
*Remove the altitude in [ItemPositions](#) for the given image.*
- void **removeItems** (const QList< qlonglong > &itemIDs, const QList< int > &albumIDs=QList< int >())  
*Marks all items in the list as removed, resets their dirids.*
- void **removeItemsFromAlbum** (int albumID, const QList< qlonglong > &ids\_forInformation=QList< qlonglong >())  
*Marks all items in the specified album as removed, resets their dirids.*



- void [removeItemsPermanently](#) (const QList< qlonglong > &itemIDs, const QList< int > &albumIDs=QList< int >())  
*Marks all items in the list as obsolete, resets their dirids.*
- void [removeItemTag](#) (qlonglong imageID, int tagID)  
*Remove a specific tag for the item.*
- void [removeTagProperties](#) (int tagId, const QString &property=QString(), const QString &value=QString())  
*Removes properties for the given tag.*
- void [removeTagsFromItems](#) (const QList< qlonglong > &imageIDs, const QList< int > &tagIDs)  
*Remove each tag from a list of tags from a each member of a list of items.*
- void [renameAlbum](#) (int albumID, int newAlbumRootId, const QString &newRelativePath)  
*Give an existing album a new relativePath and a newAlbumRootId.*
- void [renameItem](#) (qlonglong imageID, const QString &newName)  
*Rename the item.*
- AlbumInfo::List [scanAlbums](#) () const  
*Returns all albums and their attributes in the database.*
- SearchInfo::List [scanSearches](#) () const  
*Returns all searches from the database.*
- TagInfo::List [scanTags](#) () const  
*Returns all tags and their attributes in the database.*
- void [setAlbumCaption](#) (int albumID, const QString &caption)  
*Set a caption for the album.*
- void [setAlbumCategory](#) (int albumID, const QString &category)  
*Set a category for the album.*
- void [setAlbumDate](#) (int albumID, const QDate &date)  
*Set a date for the album.*
- void [setAlbumIcon](#) (int albumID, qlonglong iconID)  
*Set the icon for the album.*
- void [setAlbumModificationDate](#) (int albumID, const QDateTime &modificationDate)  
*Set the modification date time for the album.*
- void [setAlbumRootCaseSensitivity](#) (int rootId, [CollectionLocation::CaseSensitivity](#) caseSensitivity)  
*Sets the case sensitivity of the specified album root to a new value.*
- void [setAlbumRootLabel](#) (int rootId, const QString &newLabel)  
*Changes the label of the specified album root.*
- void [setAlbumRootPath](#) (int rootId, const QString &newPath)  
*Changes the specificPath of the specified album root.*
- void [setAlbumRootType](#) (int rootId, [CollectionLocation::Type](#) newType)  
*Sets the type of the specified album root to a new value.*
- void [setFilterSettings](#) (const QStringList &imageFilter, const QStringList &videoFilter, const QStringList &audioFilter)  
*Sets the main filter settings of the database.*
- void [setIgnoreDirectoryFilterSettings](#) (const QStringList &ignoreDirectoryFilter)
- int [setImageComment](#) (qlonglong imageID, const QString &comment, DatabaseComment::Type type, const QString &language=QString(), const QString &author=QString(), const QDateTime &date=QDateTime()) const  
*Sets the comments for the image.*
- void [setImageProperty](#) (qlonglong imageID, const QString &property, const QString &value)  
*Sets the property with the given name for the given image to the specified value.*
- void [setImageUuid](#) (qlonglong imageId, const QString &uuid)
- void [setItemAlbum](#) (qlonglong imageID, qlonglong albumId)  
*Updates the album field for the item.*

- void **setItemCopyrightProperty** (qulonglong imageID, const QString &property, const QString &value, const QString &extraValue=QString(), CopyrightPropertyUnique uniqueness=PropertyUnique)
 

*Sets the property with the given name for the given image to the specified value and extraValue.*
- void **setItemHistory** (qulonglong imageID, const QString &history)
 

*Changes (adds or updates) the image history.*
- void **setItemManualOrder** (qulonglong imageID, qulonglong value)
 

*Updates the manualOrder field for the item.*
- void **setItemModificationDate** (qulonglong imageID, const QDateTime &modificationDate)
 

*Updates the modification date field for the item.*
- void **setItemStatus** (qulonglong imageID, DatabaseItem::Status status)
 

*Updates the status field for the item.*
- void **setSetting** (const QString &keyword, const QString &value)
 

*This adds a keyword-value combination to the database Settings table if the keyword already exists, the value will be replaced with the new value.*
- void **setTagIcon** (int tagID, const QString &iconKDE, qulonglong iconID)
 

*Set the icon for the tag.*
- void **setTagName** (int tagID, const QString &name)
 

*Set a new name for the tag.*
- void **setTagParentID** (int tagID, int newParentTagID)
 

*Set the parent tagid for the tag.*
- void **setUniqueHashVersion** (int version)
- void **setUserFilterSettings** (const QStringList &imageFilter, const QStringList &videoFilter, const QStringList &audioFilter)
 

*Sets the user-configurable filter settings.*
- void **setUserIgnoreDirectoryFilterSettings** (const QStringList &ignoreDirectoryFilters)
- void **updateItem** (qulonglong imageID, DatabaseItem::Category category, const QDateTime &modificationDate, qulonglong fileSize, const QString &uniqueHash)
 

*Update the fields of the Images table that have changed when the file has been modified on disk.*
- void **updateSearch** (int searchID, DatabaseSearch::Type type, const QString &name, const QString &query)
 

*Updates Search with new attributes.*
- void **vacuum** ()
 

*Shrinks the database.*

### Static Public Member Functions

- static void **addBoundValuePlaceholders** (QString &query, int count)
- static QStringList **imageCommentsFieldList** (DatabaseFields::ItemComments fields)
- static QStringList **imageInformationFieldList** (DatabaseFields::ItemImageInformation fields)
- static QStringList **imageMetadataFieldList** (DatabaseFields::ImageMetadata fields)
- static QStringList **imagePositionsFieldList** (DatabaseFields::ItemPositions fields)
- static QStringList **imagesFieldList** (DatabaseFields::Images fields)
- static QStringList **videoMetadataFieldList** (DatabaseFields::VideoMetadata fields)

### Protected Member Functions

- QVector< QList< qulonglong > > **getRelatedImages** (QList< qulonglong > ids, bool fromOrTo, DatabaseRelation::Type type, bool boolean) const
- QList< qulonglong > **getRelatedImages** (qulonglong id, bool fromOrTo, DatabaseRelation::Type type, bool boolean) const

**Friends**

- class **Digikam::CoreDbAccess**

**9.205.1 Member Function Documentation****9.205.1.1 addAlbum()**

```
int Digikam::CoreDB::addAlbum (
    int albumRootId,
    const QString & relativePath,
    const QString & caption,
    const QDate & date,
    const QString & collection ) const
```

**Parameters**

<i>albumRootId</i>	id of the album root of the new album
<i>relativePath</i>	url of the album
<i>caption</i>	the album caption
<i>date</i>	the date for the album
<i>collection</i>	the album collection

**Returns**

the id of the album added or -1 if it failed

**9.205.1.2 addAlbumRoot()**

```
int Digikam::CoreDB::addAlbumRoot (
    CollectionLocation::Type type,
    const QString & identifier,
    const QString & specificPath,
    const QString & label ) const
```

**Parameters**

<i>type</i>	The type of the album root
<i>identifier</i>	The album root identifier
<i>specificPath</i>	The path specific to volume
<i>label</i>	An (optional) user-visible label

**Returns**

the album root id of the newly created root

### 9.205.1.3 addImageMetadata()

```
void Digikam::CoreDB::addImageMetadata (
    qlonglong imageID,
    const QVariantList & infos,
    DatabaseFields::ImageMetadata fields = DatabaseFields::ImageMetadataAll )
```

If there is already an entry, it will be discarded. The QVariantList shall have at most 16 entries, of types as defined in the DBSCHEMA and in metadatainfo.h, in this order:

0) String make 1) String model 2) String lens 3) Double aperture 4) Double focalLength 5) Double focalLength35 6) Double exposureTime 7) Int exposureProgram 8) Int exposureMode 9) Int sensitivity 10) Int flash 11) Int WhiteBalance 12) Int WhiteBalanceColorTemperature 13) Int meteringMode 14) Double subjectDistance 15) Double subjectDistanceCategory

#### Note

: you can leave out entries from this list. Indicate the values that you have passed in the ImageMetadata flag in the third parameters.

### 9.205.1.4 addImageTagProperty()

```
void Digikam::CoreDB::addImageTagProperty (
    qlonglong imageId,
    int tagId,
    const QString & property,
    const QString & value )
```

Note that this never replaces existing entries. It is also all right to add multiple entries for a tag with the same property. To replace an existing entry, remove the entry before.

### 9.205.1.5 addItem()

```
qlonglong Digikam::CoreDB::addItem (
    int albumID,
    const QString & name,
    DatabaseItem::Status status,
    DatabaseItem::Category category,
    const QDateTime & modificationDate,
    qlonglong fileSize,
    const QString & uniqueHash ) const
```

#### Returns

the id of item added or -1 if it fails

### 9.205.1.6 addItemInformation()

```
void Digikam::CoreDB::addItemInformation (
    qlonglong imageID,
    const QVariantList & infos,
    DatabaseFields::ItemInformation fields = DatabaseFields::ItemInformationAll )
```

If there is already an entry, it will be discarded. The QVariantList shall have 9 entries, of types in this order:

0) Int rating 1) DateTime\* creationDate 2) DateTime\* digitizationDate 3) Int orientation 4) Int width 5) Int height 6) String format 7) Int colorDepth 8) Int colorModel

#### Note

: you can provide the date also as a string in the format Qt::IsoDate. You can leave out entries from this list, which will then be filled with null values. Indicate the values that you have passed in the ItemInformation flag in the third parameters.

### 9.205.1.7 addItemPosition()

```
void Digikam::CoreDB::addItemPosition (
    qlonglong imageID,
    const QVariantList & infos,
    DatabaseFields::ItemPositions fields = DatabaseFields::ItemPositionsAll )
```

If there is already an entry, it will be discarded. The QVariantList shall have at most 10 entries, of types in this order:

0) String Latitude 1) Double LatitudeNumber 2) String Longitude 3) Double LongitudeNumber 4) Double Altitude 5) Double Orientation 6) Double Tilt 7) Double Roll 8) Double Accuracy 9) String Description

#### Note

: you can leave out entries from this list. Indicate the values that you have passed in the [ItemInfo](#) flag in the third parameters.

### 9.205.1.8 addItemTag() [1/2]

```
void Digikam::CoreDB::addItemTag (
    int albumID,
    const QString & name,
    int tagID )
```

#### Parameters

<i>albumID</i>	the albumID of the item
<i>name</i>	the name of the item
<i>tagID</i>	the tagID for the tag

**9.205.1.9 addItemTag()** [2/2]

```
void Digikam::CoreDB::addItemTag (
    qlonglong imageID,
    int tagID,
    bool newTag = false )
```

**Parameters**

<i>imageID</i>	the ID of the item
<i>tagID</i>	the tagID for the tag
<i>newTag</i>	add to last assigned tag list

**9.205.1.10 addSearch()**

```
int Digikam::CoreDB::addSearch (
    DatabaseSearch::Type type,
    const QString & name,
    const QString & query ) const
```

**Parameters**

<i>type</i>	search type
<i>name</i>	name of the search
<i>query</i>	search query to use

**Returns**

the id of the album added or -1 if it failed

**9.205.1.11 addTag()**

```
int Digikam::CoreDB::addTag (
    int parentTagID,
    const QString & name,
    const QString & iconKDE,
    qlonglong iconID ) const
```

**Parameters**

<i>parentTagID</i>	the id of the tag which will become the new tags parent
<i>name</i>	the name of the tag
<i>iconKDE</i>	the name of the icon file (this is filename which kde iconloader can load up)
<i>iconID</i>	the id of the icon file Note: if the iconKDE parameter is empty, then the iconID parameter is used

**Returns**

the id of the tag added or -1 if it failed

### 9.205.1.12 addTagProperty()

```
void Digikam::CoreDB::addTagProperty (
    int tagId,
    const QString & property,
    const QString & value )
```

Note that this never replaces existing entries. It is also all right to add multiple entries for a tag with the same property. To replace an existing entry, remove the entry before.

### 9.205.1.13 addToDownloadHistory()

```
int Digikam::CoreDB::addToDownloadHistory (
    const QString & identifier,
    const QString & name,
    qlonglong fileSize,
    const QDateTime & date ) const
```

Returns the id of the entry.

### 9.205.1.14 addVideoMetadata()

```
void Digikam::CoreDB::addVideoMetadata (
    qlonglong imageID,
    const QVariantList & infos,
    DatabaseFields::VideoMetadata fields = DatabaseFields::VideoMetadataAll )
```

If there is already an entry, it will be discarded. The QVariantList shall have 8 entries, of types in this order:

- 0) String AspectRatio
- 1) String AudioBitRate
- 2) String AudioChannelType
- 3) String AudioCodec
- 4) String Duration
- 5) String FrameRate
- 6) String VideoCodec

#### Note

: you can leave out entries from this list, which will then be filled with null values. Indicate the values that you have passed in the VideoMetadata flag in the third parameters.

### 9.205.1.15 changelImageComment()

```
void Digikam::CoreDB::changeImageComment (
    int commentId,
    qlonglong imageID,
    const QVariantList & infos,
    DatabaseFields::ItemComments fields = DatabaseFields::ItemCommentsAll )
```

The QVariantList shall have at most 5 entries, of types in this order:

- 0) Int Type
- 1) String Language
- 2) String Author
- 3) DateTime Date
- 4) String Comment

#### 9.205.1.16 `changeImageMetadata()`

```
void Digikam::CoreDB::changeImageMetadata (
    qlonglong imageID,
    const QVariantList & infos,
    DatabaseFields::ImageMetadata fields = DatabaseFields::ImageMetadataAll )
```

This method does nothing if the item does not yet have an entry in the ItemInformation table. The parameters are as for the method above.

#### 9.205.1.17 `changeItemInformation()`

```
void Digikam::CoreDB::changeItemInformation (
    qlonglong imageID,
    const QVariantList & infos,
    DatabaseFields::ItemInformation fields = DatabaseFields::ItemInformationAll )
```

Fields not indicated by the fields parameter will not be touched. This method does nothing if the item does not yet have an entry in the ItemInformation table. The parameters are as for the method above.

#### 9.205.1.18 `changeItemPosition()`

```
void Digikam::CoreDB::changeItemPosition (
    qlonglong imageID,
    const QVariantList & infos,
    DatabaseFields::ItemPositions fields = DatabaseFields::ItemPositionsAll )
```

This method does nothing if the item does not yet have an entry in the ItemInformation table. The parameters are as for the method above.

#### 9.205.1.19 `changeVideoMetadata()`

```
void Digikam::CoreDB::changeVideoMetadata (
    qlonglong imageID,
    const QVariantList & infos,
    DatabaseFields::VideoMetadata fields = DatabaseFields::VideoMetadataAll )
```

This method does nothing if the item does not yet have an entry in the ItemInformation table. The parameters are as for the method above.

#### 9.205.1.20 `copyAlbumProperties()`

```
bool Digikam::CoreDB::copyAlbumProperties (
    int srcAlbumID,
    int dstAlbumID ) const
```

Both albums must exist.

#### Returns

true if the operations succeeds



**9.205.1.21 copyItem()**

```
qlonglong Digikam::CoreDB::copyItem (
    int srcAlbumID,
    const QString & srcName,
    int dstAlbumID,
    const QString & dstName )
```

Useful when say a file is copied. The operation fails (returns -1) if src and dest are identical.

**Parameters**

<i>srcAlbumID</i>	the id of the source album
<i>dstAlbumID</i>	the id of the destination album
<i>srcName</i>	the name of the source file
<i>dstName</i>	the name of the destination file

**Returns**

the id of item added or -1 if it fails

**9.205.1.22 databaseUuid()**

```
QUuid Digikam::CoreDB::databaseUuid ( )
```

This UUID is kept stable over schema updates.

**9.205.1.23 deleteAlbum()**

```
void Digikam::CoreDB::deleteAlbum (
    int albumID )
```

This will not delete the subalbums of the album.

**Parameters**

<i>albumID</i>	the id of the album
----------------	---------------------

**9.205.1.24 deleteAlbumRoot()**

```
void Digikam::CoreDB::deleteAlbumRoot (
    int rootId )
```

**Parameters**

<i>rootId</i>	the id of the album root
---------------	--------------------------

**9.205.1.25 deleteItem() [1/2]**

```
void Digikam::CoreDB::deleteItem (
    int albumID,
    const QString & file )
```

**Parameters**

<i>albumID</i>	The id of the album.
<i>file</i>	The filename of the file to delete.

**9.205.1.26 deleteItem() [2/2]**

```
void Digikam::CoreDB::deleteItem (
    qulonglong imageId )
```

This method can only be used if the album of the image is null!

**Parameters**

<i>image↔ Id</i>	The id of the image.
----------------------	----------------------

**9.205.1.27 deleteObsoleteItem()**

```
void Digikam::CoreDB::deleteObsoleteItem (
    qulonglong imageId )
```

**Parameters**

<i>image↔ Id</i>	The id of the image.
----------------------	----------------------

**9.205.1.28 deleteRemovedItems()**

```
void Digikam::CoreDB::deleteRemovedItems ( )
```

**Warning**

: Use with care!

**9.205.1.29 deleteSearch()**

```
void Digikam::CoreDB::deleteSearch (
    int searchID )
```

## Parameters

<i>searchID</i>	the id of the search
-----------------	----------------------

**9.205.1.30 deleteTag()**

```
void Digikam::CoreDB::deleteTag (
    int tagID )
```

This will not delete the subtags of the tag.

## Parameters

<i>tagID</i>	the id of the tag
--------------	-------------------

**9.205.1.31 findImageId()**

```
qulonglong Digikam::CoreDB::findImageId (
    int albumID,
    const QString & name,
    DatabaseItem::Status status,
    DatabaseItem::Category category,
    qulonglong fileSize,
    const QString & uniqueHash ) const
```

## Parameters

<i>albumID</i>	the albumID of the item (-1 means null)
<i>name</i>	the name of the item
<i>status</i>	the status of the item
<i>category</i>	the category of the item
<i>fileSize</i>	the file size
<i>uniqueHash</i>	the unique hash

## Returns

the ImageId for the item, or -1 if no matching or more than one infos were found.

**9.205.1.32 findInDownloadHistory()**

```
int Digikam::CoreDB::findInDownloadHistory (
    const QString & identifier,
    const QString & name,
    qulonglong fileSize,
    const QDateTime & date ) const
```

Returns the id of the entry, or -1 if not found.

**9.205.1.33 getAlbumAndSubalbumsForPath()**

```
QList< int > Digikam::CoreDB::getAlbumAndSubalbumsForPath (
    int albumRootId,
    const QString & relativePath ) const
```

**Parameters**

<i>album↔ RootId</i>	id of the album root of the album
<i>relativePath</i>	The path for which you want the albumIDs relative to the album root

**Returns**

a list of album ids. The list is empty if no albums are found.

**9.205.1.34 getAlbumAverageDate()**

```
QDate Digikam::CoreDB::getAlbumAverageDate (
    int albumID ) const
```

**Parameters**

<i>albumID</i>	the id of the album to calculate
----------------	----------------------------------

**Returns**

the date.

**9.205.1.35 getAlbumForPath()**

```
int Digikam::CoreDB::getAlbumForPath (
    int albumRootId,
    const QString & relativePath,
    bool create = true ) const
```

**Parameters**

<i>album↔ RootId</i>	id of the album root of the album
<i>relativePath</i>	The relative path for which you want the albumID relative to the album root
<i>create</i>	If true, an album is newly created if it does not yet exist. If false, -1 is returned if no album exists.

**Returns**

The albumID for that folder, or -1 if it does not exist and create is false.

**9.205.1.36 getAlbumHighestDate()**

```
QDate Digikam::CoreDB::getAlbumHighestDate (
    int albumID ) const
```

**Parameters**

<i>albumID</i>	the id of the album to calculate
----------------	----------------------------------

**Returns**

the date.

**9.205.1.37 getAlbumLowestDate()**

```
QDate Digikam::CoreDB::getAlbumLowestDate (
    int albumID ) const
```

**Parameters**

<i>albumID</i>	the id of the album to calculate
----------------	----------------------------------

**Returns**

the date.

**9.205.1.38 getAlbumModificationDate()**

```
QDateTime Digikam::CoreDB::getAlbumModificationDate (
    int albumID ) const
```

**Parameters**

<i>albumID</i>	the id of the album
----------------	---------------------

**9.205.1.39 getAlbumModificationMap()**

```
QMap< QString, QDateTime > Digikam::CoreDB::getAlbumModificationMap (
    int albumRootId ) const
```

**Parameters**

<i>album↔ RootId</i>	id of the album root of the album
--------------------------	-----------------------------------

#### 9.205.1.40 `getAlbumRelativePath()`

```
QString Digikam::CoreDB::getAlbumRelativePath (
    int albumID ) const
```

##### Parameters

<i>albumID</i>	the id of the album
----------------	---------------------

##### Returns

the url of the album

#### 9.205.1.41 `getAlbumRootId()`

```
int Digikam::CoreDB::getAlbumRootId (
    int albumID ) const
```

##### Parameters

<i>albumID</i>	the id of the albumdb
----------------	-----------------------

##### Returns

the id of the album root of this album

#### 9.205.1.42 `getAlbumRoots()`

```
QList< AlbumRootInfo > Digikam::CoreDB::getAlbumRoots ( ) const
```

##### Returns

a list of albums and their attributes

#### 9.205.1.43 `getAlbumsOnAlbumRoot()`

```
QList< int > Digikam::CoreDB::getAlbumsOnAlbumRoot (
    int albumRootId ) const
```

##### Returns

a list of album ids.

#### 9.205.1.44 `getAllItemsWithAlbum()`

```
QHash< qlonglong, QPair< int, int > > Digikam::CoreDB::getAllItemsWithAlbum ( ) const
```

QPair.first == albumRootID QPair.second == albumID

**9.205.1.45** `getDatabaseEncoding()`

```
QString Digikam::CoreDB::getDatabaseEncoding ( ) const
```

For SQLite should UTF-8. For MySQL like UTF8MB4.

**9.205.1.46** `getDirtyOrMissingFacelImageUrls()`

```
QStringList Digikam::CoreDB::getDirtyOrMissingFaceImageUrls ( ) const
```

Return image ids or item URLs.

**9.205.1.47** `getFilterSettings()`

```
void Digikam::CoreDB::getFilterSettings (
    QStringList * imageFilter,
    QStringList * videoFilter,
    QStringList * audioFilter )
```

Returns a list with lowercase suffixes only, no wildcards added ("png", not "\*.png") Returned is a joint result of main and user settings. If you are not interested in a specific value, pass 0.

**9.205.1.48** `getIdenticalFiles()`

```
QList< ItemScanInfo > Digikam::CoreDB::getIdenticalFiles (
    qlonglong id ) const
```

Criteria: Unique Hash, file size and album non-null. The first variant will not return an [ItemScanInfo](#) for id. The second allows to pass one id as source id for exclusion from the list. If this is -1, no id is excluded.

**9.205.1.49** `getImageId()`

```
qlonglong Digikam::CoreDB::getImageId (
    int albumID,
    const QString & name ) const
```

**Parameters**

<i>albumID</i>	the albumID of the item
<i>name</i>	the name of the item

**Returns**

the ImageId for the item, or -1 if it does not exist

**9.205.1.50** `getImageIds()` [1/4]

```
QList< qlonglong > Digikam::CoreDB::getImageIds (
```

```
DatabaseItem::Status status ) const
```

#### Parameters

<i>status</i>	The status.
---------------	-------------

#### Returns

The ids of the images that have the given status.

#### 9.205.1.51 getImagelds() [2/4]

```
QList< qlonglong > Digikam::CoreDB::getImageIds (
    DatabaseItem::Status status,
    DatabaseItem::Category category ) const
```

#### Parameters

<i>status</i>	The status.
<i>category</i>	The category.

#### Returns

The ids of the images that have the given status.

#### 9.205.1.52 getImagelds() [3/4]

```
QList< qlonglong > Digikam::CoreDB::getImageIds (
    int albumID,
    const QString & name,
    DatabaseItem::Status status ) const
```

#### Parameters

<i>albumID</i>	the albumID of the item (-1 means NULL)
<i>name</i>	the name of the item
<i>status</i>	the status of the item

#### Returns

the Imagelds for the item, or an empty list if there are no matching entries.

#### 9.205.1.53 getImagelds() [4/4]

```
QList< qlonglong > Digikam::CoreDB::getImageIds (
    int albumID,
    DatabaseItem::Status status,
    bool scanned = true ) const
```



## Parameters

<i>albumID</i>	the albumID of the item (-1 means NULL)
<i>status</i>	the status of the item
<i>scanned</i>	return scanned/unscanned items

## Returns

the ImageIds for the item, or an empty list if there are no matching entries.

**9.205.1.54 getImageMetadata()**

```
QVariantList Digikam::CoreDB::getImageMetadata (
    qlonglong imageID,
    DatabaseFields::ImageMetadata metadataFields = DatabaseFields::ImageMetadataAll )
const
```

Parameters as above.

**9.205.1.55 getImagesFields()**

```
QVariantList Digikam::CoreDB::getImagesFields (
    qlonglong imageID,
    DatabaseFields::Images imagesFields ) const
```

Choose the fields with the mask. The fields will be returned in the following order and type: 0) Int Album 1) String Name 2) Int Status 3) Int Category 4) DateTime ModificationDate 5) int FileSize 6) String uniqueHash

**9.205.1.56 getImagesRelatedFrom()**

```
QList< qlonglong > Digikam::CoreDB::getImagesRelatedFrom (
    qlonglong subjectId,
    DatabaseRelation::Type type = DatabaseRelation::UndefinedType ) const
```

"Get images related to from this"

**9.205.1.57 getImagesRelatingTo()**

```
QList< qlonglong > Digikam::CoreDB::getImagesRelatingTo (
    qlonglong objectId,
    DatabaseRelation::Type type = DatabaseRelation::UndefinedType ) const
```

"Get images this image is relating to"

**9.205.1.58 getImageTagProperties()**

```
QList< ImageTagProperty > Digikam::CoreDB::getImageTagProperties (
    qlonglong imageId,
    int tagId = -1 ) const
```

If the tagID is -1, returns the ImageTagProperties for all tagIDs of the given image.

**9.205.1.59 getItemAlbum()**

```
int Digikam::CoreDB::getItemAlbum (
    qlonglong imageID ) const
```

**Parameters**

<i>imageID</i>	The ID of the item
----------------	--------------------

**Returns**

The ID of the [PAlbum](#) of the item, or -1 if not found

**9.205.1.60 getItemCommonTagIDs()**

```
QList< int > Digikam::CoreDB::getItemCommonTagIDs (
    const QList< qlonglong > & imageIDList ) const
```

**Parameters**

<i>imageIDList</i>	a list of IDs of the items
--------------------	----------------------------

**Returns**

the list of common IDs of the given items

**9.205.1.61 getItemCopyright()**

```
QList< CopyrightInfo > Digikam::CoreDB::getItemCopyright (
    qlonglong imageID,
    const QString & property = QString() ) const
```

If property is not null, only the given property is returned.

**9.205.1.62 getItemFromAlbum()**

```
qlonglong Digikam::CoreDB::getItemFromAlbum (
    int albumID,
    const QString & fileName ) const
```

**Parameters**

<i>albumID</i>	The albumId in which we search the item.
<i>fileName</i>	The name of the item file.

**Returns**

The item id or -1 if not existent.

**9.205.1.63 getItemIDsAndURLsInAlbum()**

```
QMap< qlonglong, QString > Digikam::CoreDB::getItemIDsAndURLsInAlbum (
```

```
int albumID ) const
```

**Note**

: Uses the [CollectionManager](#)

**Parameters**

<i>albumID</i>	the id of the album
----------------	---------------------

**Returns**

a map of ids and urls for the items in the album. The urls are the absolute path of the items

**9.205.1.64 getItemIDsInAlbum()**

```
QList< qlonglong > Digikam::CoreDB::getItemIDsInAlbum (
    int albumID ) const
```

**Parameters**

<i>albumID</i>	the id of the album
----------------	---------------------

**Returns**

a list of ids for the items in the album.

**9.205.1.65 getItemIDsInTag()**

```
QList< qlonglong > Digikam::CoreDB::getItemIDsInTag (
    int tagID,
    bool recursive = false ) const
```

**Parameters**

<i>tagID</i>	the id of the tag
<i>recursive</i>	perform a recursive folder hierarchy parsing

**Returns**

a list of ids for the items in the tag.

**9.205.1.66 getItemInformation()**

```
QVariantList Digikam::CoreDB::getItemInformation (
    qlonglong imageID,
```

```

        DatabaseFields::ItemInformation infoFields = DatabaseFields::ItemInformationAll )
const

```

Parameters as above.

### 9.205.1.67 getItemName()

```

QString Digikam::CoreDB::getItemName (
    qlonglong imageID ) const

```

#### Parameters

<i>imageID</i>	The ID of the item
----------------	--------------------

#### Returns

The name of the item, or a null string if not found

### 9.205.1.68 getItemNamesInAlbum()

```

QStringList Digikam::CoreDB::getItemNamesInAlbum (
    int albumID,
    bool recursive = false ) const

```

This is used to verify if all items on disk are consistent with the database in the [CollectionScanner](#) class.

#### Parameters

<i>albumID</i>	The albumID for which you want all items.
<i>recursive</i>	perform a recursive folder hierarchy parsing

#### Returns

It returns a QStringList with the filenames.

### 9.205.1.69 getItemPosition()

```

QVariantList Digikam::CoreDB::getItemPosition (
    qlonglong imageID,
    DatabaseFields::ItemPositions positionFields = DatabaseFields::ItemPositionsAll )
const

```

Parameters as above.

### 9.205.1.70 getItemsTagIDs()

```

QVector< QList< int > > Digikam::CoreDB::getItemsTagIDs (
    const QList< qlonglong > & imageIds ) const

```

Amounts to calling getItemTagIDs for each id in imageIds, but is optimized.

### 9.205.1.71 getItemTagIDs()

```
QList< int > Digikam::CoreDB::getItemTagIDs (
    qlonglong imageID ) const
```

#### Parameters

<i>imageID</i>	the ID of the item
----------------	--------------------

#### Returns

the list of IDs of all tags for the item

### 9.205.1.72 getItemTagNames()

```
QStringList Digikam::CoreDB::getItemTagNames (
    qlonglong imageID ) const
```

#### Parameters

<i>imageID</i>	the ID of the item
----------------	--------------------

#### Returns

the list of names of all tags for the item

### 9.205.1.73 getItemURLsInAlbum()

```
QStringList Digikam::CoreDB::getItemURLsInAlbum (
    int albumID,
    ItemSortOrder order = NoItemSorting ) const
```

#### Note

: Uses the [CollectionManager](#)

#### Parameters

<i>albumID</i>	the id of the album
<i>order</i>	order for the returned items to use

#### Returns

a list of urls for the items in the album. The urls are the absolute path of the items

### 9.205.1.74 getItemURLsInTag()

```
QStringList Digikam::CoreDB::getItemURLsInTag (
    int tagID,
    bool recursive = false ) const
```

#### Note

: Uses the [CollectionManager](#)

#### Parameters

<i>tagID</i>	the id of the tag
<i>recursive</i>	perform a recursive folder hierarchy parsing

#### Returns

a list of urls for the items in the tag. The urls are the absolute path of the items

### 9.205.1.75 getNumberOfAllItemsAndAlbums()

```
QPair< int, int > Digikam::CoreDB::getNumberOfAllItemsAndAlbums (
    int albumID ) const
```

#### Parameters

<i>albumID</i>	the id of the album
----------------	---------------------

### 9.205.1.76 getNumberOfItemsInAlbum()

```
int Digikam::CoreDB::getNumberOfItemsInAlbum (
    int albumID ) const
```

#### Parameters

<i>albumID</i>	the id of the album
----------------	---------------------

### 9.205.1.77 getOneRelatedImageEach()

```
QList< qlonglong > Digikam::CoreDB::getOneRelatedImageEach (
    const QList< qlonglong > & ids,
    DatabaseRelation::Type type = DatabaseRelation::UndefinedType ) const
```

Ids are unique in the returned list, and do not correspond by index to the given list.

**9.205.1.78 getRecentlyAssignedTags()**

```
QList< int > Digikam::CoreDB::getRecentlyAssignedTags ( ) const
```

**Returns**

the list of recently assigned tags

**9.205.1.79 getRelationCloud()**

```
QList< QPair< qlonglong, qlonglong > > Digikam::CoreDB::getRelationCloud (
    qlonglong imageId,
    DatabaseRelation::Type type = DatabaseRelation::UndefinedType ) const
```

Each a and b in the list will have a direct or indirect relation to the initial *imageId*. If *type* is given, filters by type, otherwise returns all types.

**9.205.1.80 getSetting()**

```
QString Digikam::CoreDB::getSetting (
    const QString & keyword ) const
```

**Parameters**

<i>keyword</i>	The keyword for which the value has to be returned.
----------------	---

**Returns**

The values which belongs to the keyword, or a null string if no value is set.

**9.205.1.81 getTagsWithProperty()**

```
QList< int > Digikam::CoreDB::getTagsWithProperty (
    const QString & property ) const
```

**9.205.1.82 getUniqueHashVersion()**

```
int Digikam::CoreDB::getUniqueHashVersion ( ) const
```

The value is cached.

**9.205.1.83 getUserFilterSettings()**

```
void Digikam::CoreDB::getUserFilterSettings (
    QString * imageFilterString,
    QString * videoFilterString,
    QString * audioFilterString )
```

If you are not interested in a specific value, pass 0.

#### 9.205.1.84 `getVideoMetadata()`

```
QVariantList Digikam::CoreDB::getVideoMetadata (
    qlonglong imageID,
    DatabaseFields::VideoMetadata metadataFields = DatabaseFields::VideoMetadataAll )
const
```

Parameters as above.

#### 9.205.1.85 `hasTags()`

```
bool Digikam::CoreDB::hasTags (
    const QList< qlonglong > & imageIDList ) const
```

##### Parameters

<i>imageIDList</i>	a list of IDs of the items
--------------------	----------------------------

##### Returns

true if at least one of the items has a tag

#### 9.205.1.86 `makeStaleAlbum()`

```
void Digikam::CoreDB::makeStaleAlbum (
    int albumID )
```

Emits the same changeset as [deleteAlbum\(\)](#)

#### 9.205.1.87 `migrateAlbumRoot()`

```
void Digikam::CoreDB::migrateAlbumRoot (
    int rootId,
    const QString & identifier )
```

This only changes the values in the AlbumRoots table. It is expected that this merely reflects underlying partition changes, still pointing to the same data.

#### 9.205.1.88 `moveItem()`

```
void Digikam::CoreDB::moveItem (
    int srcAlbumID,
    const QString & srcName,
    int dstAlbumID,
    const QString & dstName )
```

Useful when say a file is renamed



## Parameters

<i>srcAlbumID</i>	the id of the source album
<i>dstAlbumID</i>	the id of the destination album
<i>srcName</i>	the name of the source file
<i>dstName</i>	the name of the destination file

**9.205.1.89 removeImageRelation()**

```
void Digikam::CoreDB::removeImageRelation (
    qlonglong subjectId,
    qlonglong objectId,
    DatabaseRelation::Type type )
```

The batch methods return all removed partners.

**9.205.1.90 removeImageTagProperties()**

```
void Digikam::CoreDB::removeImageTagProperties (
    qlonglong imageId,
    int tagId = -1,
    const QString & property = QString(),
    const QString & value = QString() )
```

If the value is given, removes only the entries with the given property/value pair. If only property is given, removes all properties with the given name. If property is null, removes all properties for the given tag. If tagId is -1, removes all image tag properties for the given image.

## Note

: After the first parameter you give as a wildcard, the following will be ignored and taken as wildcard as well.

**9.205.1.91 removeItemAllTags()**

```
void Digikam::CoreDB::removeItemAllTags (
    qlonglong imageID,
    const QList< int > & currentTagIds )
```

## Parameters

<i>imageID</i>	the ID of the item
<i>currentTagIds</i>	the current tags ids assigned to the item

**9.205.1.92 removeItemCopyrightProperties()**

```
void Digikam::CoreDB::removeItemCopyrightProperties (
    qlonglong imageID,
```

```
const QString & property = QString(),
const QString & extraValue = QString(),
const QString & value = QString() )
```

All values after the first null value, in order of parameters, are treated as wild cards (you can give value as wildcard; value and extraValue; or property, extraValue and value).

#### Warning

: extraValue is ordered before value in this method! Take a care to the parameter order.

#### 9.205.1.93 removeItems()

```
void Digikam::CoreDB::removeItems (
    const QList< qlonglong > & itemIDs,
    const QList< int > & albumIDs = QList<int>() )
```

The items can later be removed by [deleteRemovedItems\(\)](#).

#### Parameters

<i>itemIDs</i>	a list of item IDs to be marked
<i>albumIDs</i>	this parameter is purely informational. it shall contain the albums that the items are removed from.

#### 9.205.1.94 removeItemsFromAlbum()

```
void Digikam::CoreDB::removeItemsFromAlbum (
    int albumID,
    const QList< qlonglong > & ids_forInformation = QList<qlonglong>() )
```

The album can be deleted afterwards without removing the entries for the items, which can later be removed by [deleteRemovedItems\(\)](#).

#### Parameters

<i>albumID</i>	The id of the album
<i>ids_forInformation</i>	Fully optional: The image ids in the album, if you know them anyway. This parameter is only used for distributing the change notification.

#### 9.205.1.95 removeItemsPermanently()

```
void Digikam::CoreDB::removeItemsPermanently (
    const QList< qlonglong > & itemIDs,
    const QList< int > & albumIDs = QList<int>() )
```

The items can later be removed by [deleteRemovedItems\(\)](#).

## Parameters

<i>itemIDs</i>	a list of item IDs to be marked
<i>albumIDs</i>	this parameter is purely informational. it shall contain the albums that the items are removed from.

**9.205.1.96 removeItemTag()**

```
void Digikam::CoreDB::removeItemTag (
    qlonglong imageID,
    int tagID )
```

## Parameters

<i>imageID</i>	the ID of the item
<i>tagID</i>	the tagID for the tag

**9.205.1.97 removeTagProperties()**

```
void Digikam::CoreDB::removeTagProperties (
    int tagId,
    const QString & property = QString(),
    const QString & value = QString() )
```

If the value is given, removes only the entries with the given property/value pair. If only property is given, removes all properties with the given name. If property is null, removes all properties for the given tag.

**9.205.1.98 renameItem()**

```
void Digikam::CoreDB::renameItem (
    qlonglong imageID,
    const QString & newName )
```

Note: we not use here [ImageChangeset](#).

**9.205.1.99 scanAlbums()**

```
AlbumInfo::List Digikam::CoreDB::scanAlbums ( ) const
```

## Returns

a list of albums and their attributes

**9.205.1.100 scanSearches()**

```
SearchInfo::List Digikam::CoreDB::scanSearches ( ) const
```

## Returns

a list of searches from the database

**9.205.1.101 scanTags()**

```
TagInfo::List Digikam::CoreDB::scanTags ( ) const
```

**Returns**

a list of tags and their attributes

**9.205.1.102 setAlbumCaption()**

```
void Digikam::CoreDB::setAlbumCaption (
    int albumID,
    const QString & caption )
```

**Parameters**

<i>albumID</i>	the id of the album
<i>caption</i>	the new caption for the album

**9.205.1.103 setAlbumCategory()**

```
void Digikam::CoreDB::setAlbumCategory (
    int albumID,
    const QString & category )
```

**Parameters**

<i>albumID</i>	the id of the album
<i>category</i>	the new category for the album

**9.205.1.104 setAlbumDate()**

```
void Digikam::CoreDB::setAlbumDate (
    int albumID,
    const QDate & date )
```

**Parameters**

<i>albumID</i>	the id of the album
<i>date</i>	the date for the album

**9.205.1.105 setAlbumIcon()**

```
void Digikam::CoreDB::setAlbumIcon (
    int albumID,
    qlonglong iconID )
```

## Parameters

<i>albumID</i>	the id of the album
<i>iconID</i>	the id of the icon file

**9.205.1.106 setAlbumModificationDate()**

```
void Digikam::CoreDB::setAlbumModificationDate (
    int albumID,
    const QDateTime & modificationDate )
```

## Parameters

<i>albumID</i>	the id of the album
<i>modificationDate</i>	the modification date time for the album

**9.205.1.107 setAlbumRootLabel()**

```
void Digikam::CoreDB::setAlbumRootLabel (
    int rootId,
    const QString & newLabel )
```

## Parameters

<i>rootId</i>	the id of the album root
<i>newLabel</i>	new label for the album root

**9.205.1.108 setAlbumRootPath()**

```
void Digikam::CoreDB::setAlbumRootPath (
    int rootId,
    const QString & newPath )
```

## Parameters

<i>rootId</i>	the id of the album root
<i>newPath</i>	new path for the album root

**9.205.1.109 setFilterSettings()**

```
void Digikam::CoreDB::setFilterSettings (
    const QStringList & imageFilter,
    const QStringList & videoFilter,
    const QStringList & audioFilter )
```

Should only be called at schema update.

**9.205.1.110 setImageComment()**

```
int Digikam::CoreDB::setImageComment (
    qlonglong imageID,
    const QString & comment,
    DatabaseComment::Type type,
    const QString & language = QString(),
    const QString & author = QString(),
    const QDateTime & date = QDateTime() ) const
```

A comment for the image with the same source, language and author will be overwritten.

**Parameters**

<i>imageID</i>	The imageID of the image
<i>comment</i>	The comment string
<i>type</i>	The type of the comment
<i>language</i>	Information about the language of the comment. A null string shall be used if language information is not available from the source, or if the comment is in the default language.
<i>author</i>	Optional information about the author who wrote the comment. If not supported by the source, pass a null string.
<i>date</i>	Optional information about the date when the comment was written. If not supported by the source, pass a null string.

**Returns**

the comment ID of the comment

**9.205.1.111 setItemAlbum()**

```
void Digikam::CoreDB::setItemAlbum (
    qlonglong imageID,
    qlonglong albumId )
```

**Note**

: Do not use this to move the item. This function only has the purpose to reuse image infos for restored images from trash.

**9.205.1.112 setItemStatus()**

```
void Digikam::CoreDB::setItemStatus (
    qlonglong imageID,
    DatabaseItem::Status status )
```

**Note**

: Do not use this to set to the Removed status, see [removeItems\(\)](#).

**9.205.1.113 setSetting()**

```
void Digikam::CoreDB::setSetting (
    const QString & keyword,
    const QString & value )
```

## Parameters

<i>keyword</i>	The keyword
<i>value</i>	The value

**9.205.1.114 setTagIcon()**

```
void Digikam::CoreDB::setTagIcon (
    int tagID,
    const QString & iconKDE,
    qlonglong iconID )
```

## Parameters

<i>tagID</i>	the id of the tag
<i>iconKDE</i>	the filename for the kde icon file
<i>iconID</i>	the id of the icon file Note: Only one of the iconKDE or iconID parameters is used. if the iconKDE parameter is empty, then the iconID parameter is used

**9.205.1.115 setTagName()**

```
void Digikam::CoreDB::setTagName (
    int tagID,
    const QString & name )
```

## Parameters

<i>tagID</i>	the id of the tag
<i>name</i>	the new name for the tag

**9.205.1.116 setTagParentID()**

```
void Digikam::CoreDB::setTagParentID (
    int tagID,
    int newParentTagID )
```

This is equivalent to reparenting the tag

## Parameters

<i>tagID</i>	the id of the tag
<i>newParentTagID</i>	the new parentid for the tag

**9.205.1.117 setUserFilterSettings()**

```
void Digikam::CoreDB::setUserFilterSettings (
```

```

const QStringList & imageFilter,
const QStringList & videoFilter,
const QStringList & audioFilter )

```

The lists shall be as specified for `getFilterSettings`. They may include entries starting with "-", which indicates that this format shall be removed from the list, if it is included in the main settings list.

### 9.205.1.118 updateItem()

```

void Digikam::CoreDB::updateItem (
    qlonglong imageID,
    DatabaseItem::Category category,
    const QDateTime & modificationDate,
    qlonglong fileSize,
    const QString & uniqueHash )

```

#### Parameters

<i>imageID</i>	the image that has been modified
<i>category</i>	the image category that has been modified
<i>modificationDate</i>	the image time-stamp that has been modified
<i>fileSize</i>	the image file size that has been modified
<i>uniqueHash</i>	the image hash that has been modified

### 9.205.1.119 updateSearch()

```

void Digikam::CoreDB::updateSearch (
    int searchID,
    DatabaseSearch::Type type,
    const QString & name,
    const QString & query )

```

#### Parameters

<i>searchID</i>	the id of the search
<i>type</i>	type of the search
<i>name</i>	name of the search
<i>query</i>	database query of the search

## 9.206 Digikam::CoreDbAccess Class Reference

The [CoreDbAccess](#) provides access to the database: Create an instance of this class on the stack to retrieve a pointer to the database.

#### Public Types

- enum **ApplicationStatus** { **MainApplication** , **DatabaseSlave** }



## Public Member Functions

- [CoreDbAccess](#) ()  
*Create a [CoreDbAccess](#) object for the default database.*
- [CoreDbBackend](#) \* **backend** () const  
*Retrieve a pointer to the database backend.*
- [CoreDB](#) \* **db** () const  
*Retrieve a pointer to the album database.*
- QString **lastError** ()  
*Returns the error message for the last error that occurred, or a null QString of no error occurred.*
- void **setLastError** (const QString &error)  
*Set the "last error" message.*

## Static Public Member Functions

- static bool **checkReadyForUse** ([InitializationObserver](#) \*const observer=nullptr)  
*Method to one-time initialize a database when new parameters have been set: Make sure that the database is open, that the schema has properly been initialized.*
- static void **cleanUpDatabase** ()  
*Clean up the database access.*
- static [CoreDbWatch](#) \* **databaseWatch** ()  
*Return the [CoreDbWatch](#).*
- static void **initDbEngineErrorHandler** ([DbEngineErrorHandler](#) \*const errorHandler)  
*Setup the errors handler instance.*
- static [DbEngineParameters](#) **parameters** ()  
*Return the default parameters.*
- static void **setParameters** (const [DbEngineParameters](#) &parameters)  
*Set the default parameters.*
- static void **setParameters** (const [DbEngineParameters](#) &parameters, ApplicationStatus status)

## Friends

- class **CoreDbAccessUnlock**

### 9.206.1 Detailed Description

While you hold an instance of [CoreDbAccess](#), the database access is locked for other threads, but *not* for other processes. This is due to the fact that while databases allow concurrent access (of course), their client libs may not be thread-safe.

When initializing your application, you need to call two methods:

- in a not-yet-multithreaded context, you need to call [setParameters](#)
- to make sure that the database is available and the schema is properly initialized, call [checkReadyForUse\(\)](#)

## 9.206.2 Constructor & Destructor Documentation

### 9.206.2.1 CoreDbAccess()

```
Digikam::CoreDbAccess::CoreDbAccess ( )
```

Note that when initializing your app, `setParameters` need to be called (in a not-yet-multithreaded context) for this to work. If the database is not yet opened, it will be opened. The schema will not be checked, use `checkReadyForUse()` for a full opening process including schema update and error messages.

## 9.206.3 Member Function Documentation

### 9.206.3.1 checkReadyForUse()

```
bool Digikam::CoreDbAccess::checkReadyForUse (
    InitializationObserver *const observer = nullptr ) [static]
```

If the parameters were not changed, this method has no effect.

#### Returns

if the database is ready for use

### 9.206.3.2 cleanUpDatabase()

```
void Digikam::CoreDbAccess::cleanUpDatabase ( ) [static]
```

When this function has been called, the access can be restored by calling `setParameters`. Construction a database access object otherwise after calling this method will crash.

### 9.206.3.3 setLastError()

```
void Digikam::CoreDbAccess::setLastError (
    const QString & error )
```

This method is not for public use.

### 9.206.3.4 setParameters()

```
void Digikam::CoreDbAccess::setParameters (
    const DbEngineParameters & parameters ) [static]
```

Call this function at least once in the starting phase of your application, when no other threads will yet access the database, to initialize `DatabaseAccess`. After this initial call, it is thread-safe to call this function again. In a subsequent call, if the parameters are identical, nothing is done. If the parameters change, the current database will be closed. When parameters have been set or changed, the new one will be opened on-demand, i.e. when the first `CoreDbAccess` object is constructed.

## 9.207 Digikam::CoreDbAccessUnlock Class Reference

### Public Member Functions

- [CoreDbAccessUnlock](#) ()

*Acquire an object of this class if you want to assure that the [CoreDbAccess](#) is not held during the lifetime of the object.*

- [CoreDbAccessUnlock](#) ([CoreDbAccess](#) \*const access)

### 9.207.1 Constructor & Destructor Documentation

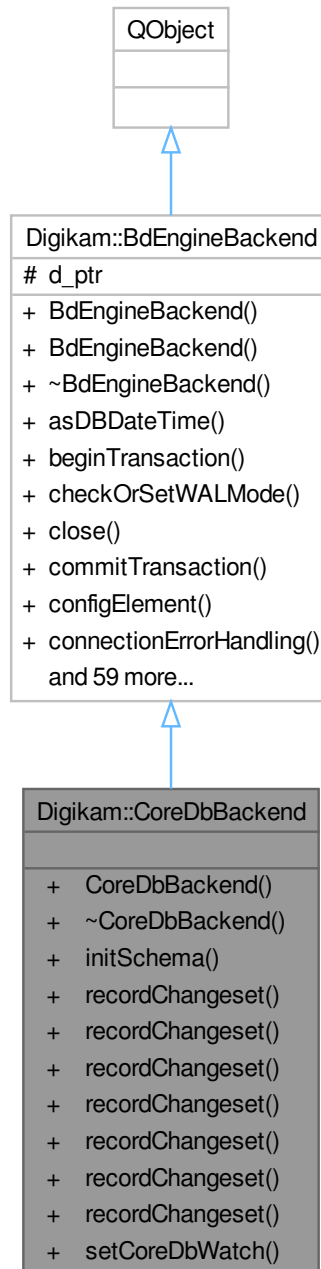
#### 9.207.1.1 CoreDbAccessUnlock()

Digikam::CoreDbAccessUnlock::CoreDbAccessUnlock ( )

At creation, the lock is obtained shortly, then all locks are released. At destruction, all locks are acquired again. If you need to access any locked structures during lifetime, acquire a new [CoreDbAccess](#).

## 9.208 Digikam::CoreDbBackend Class Reference

Inheritance diagram for Digikam::CoreDbBackend:



### Public Member Functions

- **CoreDbBackend** ([DbEngineLocking](#) \*const locking, const QString &backendName=QLatin1String("digikam← Database-"))

- bool **initSchema** ([CoreDbSchemaUpdater](#) \*updater)  
*Initialize the database schema to the current version, carry out upgrades if necessary.*
- void **recordChangeset** (const [AlbumChangeset](#) &changeset)
- void **recordChangeset** (const [AlbumRootChangeset](#) &changeset)
- void **recordChangeset** (const [CollectionImageChangeset](#) &changeset)
- void **recordChangeset** (const [ImageChangeset](#) &changeset)  
*Notify all listeners of the changeset.*
- void **recordChangeset** (const [ImageTagChangeset](#) &changeset)
- void **recordChangeset** (const [SearchChangeset](#) &changeset)
- void **recordChangeset** (const [TagChangeset](#) &changeset)
- void **setCoreDbWatch** ([CoreDbWatch](#) \*watch)  
*Sets the global database watch.*

## Public Member Functions inherited from [Digikam::BdEngineBackend](#)

- [BdEngineBackend](#) (const QString &backendName, [DbEngineLocking](#) \*const locking)  
*Creates a database backend.*
- **BdEngineBackend** (const QString &backendName, [DbEngineLocking](#) \*const locking, [BdEngineBackend](#)←Private &dd)
- QDateTime **asDBDateTime** (const QDateTime &dateTime) const  
*Depending on the database backend return a local or UTC date format.*
- [BdEngineBackend::QueryState](#) **beginTransaction** ()  
*Begin a database transaction.*
- bool **checkOrSetWALMode** ()  
*Check or set WAL mode for SQLite database if enabled in settings.*
- void **close** ()  
*Close the database connection.*
- [BdEngineBackend::QueryState](#) **commitTransaction** ()  
*Commit the current database transaction.*
- [DbEngineConfigSettings](#) **configElement** () const  
*Return config read from XML, corresponding to this backend's database type.*
- bool **connectionErrorHandling** (int retries)  
*Called when an attempted connection to the database failed.*
- [DbEngineSqlQuery](#) **copyQuery** (const [DbEngineSqlQuery](#) &old)  
*Creates a faithful copy of the passed query, with the current db connection.*
- DbType **databaseType** () const  
*Return the database type.*
- bool **exec** ([DbEngineSqlQuery](#) &query)  
*Calls exec/execBatch on the query, and handles debug output if something went wrong.*
- bool **execBatch** ([DbEngineSqlQuery](#) &query)
- [QueryState](#) **execDBAction** (const [DbEngineAction](#) &action, const QMap< QString, QVariant > &bindingMap, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)  
*Performs the database action on the current database.*
- [QueryState](#) **execDBAction** (const [DbEngineAction](#) &action, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)  
*Performs the database action on the current database.*
- [QueryState](#) **execDBAction** (const QString &action, const QMap< QString, QVariant > &bindingMap, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execDBAction** (const QString &action, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)

- QSqlQuery [execDBActionQuery](#) (const [DbEngineAction](#) &action, const QMap< QString, QVariant > &bindingMap)
  - Performs the database action on the current database.*
- QSqlQuery **execDBActionQuery** (const QString &action, const QMap< QString, QVariant > &bindingMap)
- [QueryState](#) [execDirectSql](#) (const QString &query)
  - Calls exec on the query, and handles debug output if something went wrong.*
- [QueryState](#) [execDirectSqlWithResult](#) (const QString &query, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
  - Calls exec on the query, and handles debug output if something went wrong.*
- [DbEngineSqlQuery](#) [execQuery](#) (const QString &sql)
  - Executes the statement and returns the query object.*
- [DbEngineSqlQuery](#) **execQuery** (const QString &sql, const QList< QVariant > &boundValues)
- [DbEngineSqlQuery](#) **execQuery** (const QString &sql, const QMap< QString, QVariant > &bindingMap)
  - Method which accept a hashmap with key, values which are used for named binding.*
- [DbEngineSqlQuery](#) **execQuery** (const QString &sql, const QVariant &boundValue1)
- [DbEngineSqlQuery](#) **execQuery** (const QString &sql, const QVariant &boundValue1, const QVariant &boundValue2)
- [DbEngineSqlQuery](#) **execQuery** (const QString &sql, const QVariant &boundValue1, const QVariant &boundValue2, const QVariant &boundValue3)
- [DbEngineSqlQuery](#) **execQuery** (const QString &sql, const QVariant &boundValue1, const QVariant &boundValue2, const QVariant &boundValue3, const QVariant &boundValue4)
- void **execQuery** ([DbEngineSqlQuery](#) &preparedQuery, const QList< QVariant > &boundValues)
- void **execQuery** ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &boundValue1)
  - Binds the values and executes the prepared query.*
- void **execQuery** ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &boundValue1, const QVariant &boundValue2)
- void **execQuery** ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &boundValue1, const QVariant &boundValue2, const QVariant &boundValue3)
- void **execQuery** ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &boundValue1, const QVariant &boundValue2, const QVariant &boundValue3, const QVariant &boundValue4)
- [QueryState](#) **execSql** (const QString &sql, const QList< QVariant > &boundValues, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) [execSql](#) (const QString &sql, const QMap< QString, QVariant > &bindingMap, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
  - Method which accepts a map for named binding.*
- [QueryState](#) **execSql** (const QString &sql, const QVariant &boundValue1, const QVariant &boundValue2, const QVariant &boundValue3, const QVariant &boundValue4, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** (const QString &sql, const QVariant &boundValue1, const QVariant &boundValue2, const QVariant &boundValue3, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** (const QString &sql, const QVariant &boundValue1, const QVariant &boundValue2, const QVariant &boundValue2, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** (const QString &sql, const QVariant &boundValue1, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) [execSql](#) (const QString &sql, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
  - Executes the SQL statement, and write the returned data into the values list.*
- [QueryState](#) **execSql** ([DbEngineSqlQuery](#) &preparedQuery, const QList< QVariant > &boundValues, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &boundValue1, const QVariant &boundValue2, const QVariant &boundValue3, const QVariant &boundValue4, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)

- **QueryState execSql** ([DbEngineSqlQuery](#) &preparedQuery, const [QVariant](#) &boundValue1, const [QVariant](#) &boundValue2, const [QVariant](#) &boundValue3, [QList](#)< [QVariant](#) > \*const values=nullptr, [QVariant](#) \*const lastInsertId=nullptr)
- **QueryState execSql** ([DbEngineSqlQuery](#) &preparedQuery, const [QVariant](#) &boundValue1, const [QVariant](#) &boundValue2, [QList](#)< [QVariant](#) > \*const values=nullptr, [QVariant](#) \*const lastInsertId=nullptr)
- **QueryState execSql** ([DbEngineSqlQuery](#) &preparedQuery, const [QVariant](#) &boundValue1, [QList](#)< [QVariant](#) > \*const values=nullptr, [QVariant](#) \*const lastInsertId=nullptr)
- **QueryState execSql** ([DbEngineSqlQuery](#) &preparedQuery, [QList](#)< [QVariant](#) > \*const values=nullptr, [QVariant](#) \*const lastInsertId=nullptr)
- **QueryState execUpsertDBAction** (const [DbEngineAction](#) &action, const [QVariant](#) &id, const [QStringList](#) &fieldNames, const [QList](#)< [QVariant](#) > &values)
 

*Performs a special DBAction that is usually needed to "INSERT or UPDATE" entries in a table.*
- **QueryState execUpsertDBAction** (const [QString](#) &action, const [QVariant](#) &id, const [QStringList](#) &fieldNames, const [QList](#)< [QVariant](#) > &values)
- **DbEngineAction getDBAction** (const [QString](#) &actionName) const
 

*Returns a database action with name, specified in actionName, for the current database.*
- **DbEngineSqlQuery getQuery** ()
 

*Creates an empty query object waiting for the statement.*
- **QueryState handleQueryResult** ([DbEngineSqlQuery](#) &query, [QList](#)< [QVariant](#) > \*const values, [QVariant](#) \*const lastInsertId)
 

*Checks if there was a connection error.*
- bool **isCompatible** (const [DbEngineParameters](#) &parameters)
 

*Checks if the parameters can be used for this database backend.*
- bool **isInTransaction** () const
 

*Returns if the database is in a different thread in a transaction.*
- bool **isOpen** () const
- bool **isReady** () const
- [QString](#) **lastError** ()
 

*Returns a description of the last error that occurred on this database.*
- [QStringError](#) **lastSQLError** ()
 

*Returns the last error that occurred on this database.*
- int **maximumBoundValues** () const
 

*Returns the maximum number of bound parameters allowed per query.*
- bool **open** (const [DbEngineParameters](#) &parameters)
 

*Open the database connection.*
- **DbEngineSqlQuery prepareQuery** (const [QString](#) &sql)
 

*Creates a query object prepared with the statement, waiting for bound values.*
- bool **queryErrorHandling** ([DbEngineSqlQuery](#) &query, int retries)
 

*Called with a failed query.*
- [QList](#)< [QVariant](#) > **readToList** ([DbEngineSqlQuery](#) &query)
 

*Reads data of returned result set into a list which is returned.*
- void **rollbackTransaction** ()
 

*Rollback the current database transaction.*
- void **setDbEngineErrorHandler** ([DbEngineErrorHandler](#) \*const handler)
 

*Add a DbEngineErrorHandler.*
- void **setForeignKeyChecks** (bool check)
 

*Enables or disables FOREIGN\_KEY\_CHECKS for the database.*
- **Status status** () const
 

*Returns the current status of the database backend.*
- [QStringList](#) **tables** ()
 

*Returns a list with the names of tables in the database.*
- bool **transactionErrorHandling** (const [QStringError](#) &lastError, int retries)

## Additional Inherited Members

### Public Types inherited from [Digikam::BdEngineBackend](#)

- enum **DbType** { [SQLite](#) , [MySQL](#) }
- enum **QueryOperationStatus** { [ExecuteNormal](#) , [Wait](#) , [AbortQueries](#) }
- enum [QueryStateEnum](#) { [NoErrors](#) , [SQLError](#) , [ConnectionError](#) }
- enum [Status](#) { [Unavailable](#) , [Open](#) , [OpenSchemaChecked](#) }

### Protected Attributes inherited from [Digikam::BdEngineBackend](#)

- `BdEngineBackendPrivate *const d_ptr = nullptr`

## 9.208.1 Member Function Documentation

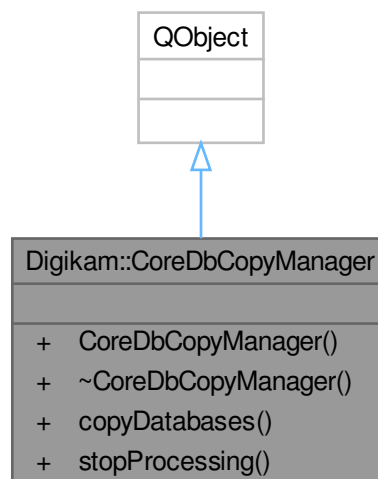
### 9.208.1.1 `initSchema()`

```
bool Digikam::CoreDbBackend::initSchema (
    CoreDbSchemaUpdater * updater )
```

Shall only be called from the thread that called `open()`.

## 9.209 [Digikam::CoreDbCopyManager](#) Class Reference

Inheritance diagram for [Digikam::CoreDbCopyManager](#):





### Public Types

- enum **FinishStates** { **success** , **failed** , **canceled** }

### Public Slots

- void **stopProcessing** ()

### Signals

- void **finished** (int finishState, const QString &errorMsg)
- void **smallStepStarted** (int currValue, int maxValue)
- void **stepStarted** (const QString &stepName)

### Public Member Functions

- void **copyDatabases** (const [DbEngineParameters](#) &fromDBParameters, const [DbEngineParameters](#) &toDBParameters)

## 9.210 Digikam::CoreDbDownloadHistory Class Reference

### Static Public Member Functions

- static void **setDownloaded** (const QString &identifier, const QString &name, qlonglong fileSize, const QDateTime &date)  
*Sets the status of the item to Downloaded.*
- static [CamItemInfo::DownloadStatus](#) **status** (const QString &identifier, const QString &name, qlonglong fileSize, const QDateTime &date)  
*Queries the status of a download item that is uniquely described by the four parameters.*

### 9.210.1 Member Function Documentation

#### 9.210.1.1 status()

```
CamItemInfo::DownloadStatus Digikam::CoreDbDownloadHistory::status (
    const QString & identifier,
    const QString & name,
    qlonglong fileSize,
    const QDateTime & date ) [static]
```

The identifier is recommended to be an MD5 hash of properties describing the camera, if available, and the directory path (though you are free to use all four parameters as you want)

## 9.211 Digikam::CoreDbNameFilter Class Reference

### Public Member Functions

- [CoreDbNameFilter](#) (const QString &filter)  
*Creates a name filter object with the given filter string.*
- bool **matches** (const QString &name)  
*Returns if the specified name matches this filter.*

### Protected Attributes

- QList< QRegularExpression > **m\_filterList**

### 9.211.1 Constructor & Destructor Documentation

#### 9.211.1.1 CoreDbNameFilter()

```
Digikam::CoreDbNameFilter::CoreDbNameFilter (
    const QString & filter ) [explicit]
```

The string is a list of text parts of which one needs to match (file suffixes), separated by ';' characters.

## 9.212 Digikam::CoreDbOperationGroup Class Reference

When you intend to execute a number of write operations to the database, group them while holding a [CoreDbOperationGroup](#).

### Public Member Functions

- **CoreDbOperationGroup** ()  
*Retrieve a [CoreDbAccess](#) object each time when constructing and destructing.*
- **CoreDbOperationGroup** ([CoreDbAccess](#) \*const access)  
*Use an existing [CoreDbAccess](#) object, which must live as long as this object exists.*
- void **allowLift** ()  
*Allows to [lift\(\)](#).*
- void **lift** ()  
*This will - if a transaction is held - commit the transaction and acquire a new one.*
- void **resetTime** ()  
*Resets to 0 the time used by [allowLift\(\)](#)*
- void **setMaximumTime** (int msec)

### 9.212.1 Detailed Description

For some database systems (SQLite), keeping a transaction across write operations occurring in short time results in enormous speedup (800x). For system that do not need this optimization, this class is a no-op.

## 9.212.2 Member Function Documentation

### 9.212.2.1 allowLift()

```
void Digikam::CoreDbOperationGroup::allowLift ( )
```

The transaction will be lifted if the time set by `setMaximumTime()` has expired.

### 9.212.2.2 lift()

```
void Digikam::CoreDbOperationGroup::lift ( )
```

This may improve concurrent access.

## 9.213 Digikam::CoreDbPrivilegesChecker Class Reference

### Public Member Functions

- **CoreDbPrivilegesChecker** (const [DbEngineParameters](#) &parameters)
- bool **checkPriv** ([CoreDbBackend](#) &dbBackend, const QString &dbName)
- bool **checkPrivileges** (QStringList &insufficientRights)

## 9.214 Digikam::CoreDbSchemaUpdater Class Reference

### Public Member Functions

- **CoreDbSchemaUpdater** ([CoreDB](#) \*const albumDB, [CoreDbBackend](#) \*const backend, const [DbEngineParameters](#) &parameters)
- const QString **getLastErrorMessage** ()
- void **setCoreDbAccess** ([CoreDbAccess](#) \*const dbAccess)
- void **setObserver** ([InitializationObserver](#) \*const observer)
- bool **update** ()
- bool **updateUniqueHash** ()

### Static Public Member Functions

- static int **filterSettingsVersion** ()
- static bool **isUniqueHashUpToDate** ()
- static int **schemaVersion** ()
- static int **uniqueHashVersion** ()

## 9.215 Digikam::CoreDbTransaction Class Reference

Convenience class: You can create a [CoreDbTransaction](#) object for a scope for which you want to declare a database commit.

### Public Member Functions

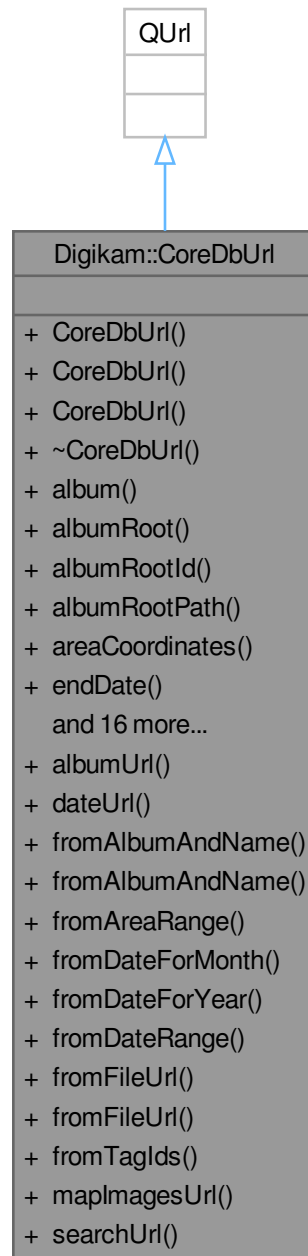
- **CoreDbTransaction** ()  
*Retrieve a [CoreDbAccess](#) object each time when constructing and destructing.*
- **CoreDbTransaction** ([CoreDbAccess](#) \*const access)  
*Use an existing [CoreDbAccess](#) object, which must live as long as this object exists.*

### 9.215.1 Detailed Description

Equivalent to calling `beginTransaction` and `commitTransaction` on the album db.

## 9.216 Digikam::CoreDbUrl Class Reference

Inheritance diagram for Digikam::CoreDbUrl:



### Public Member Functions

- **CoreDbUrl** ()=default  
*Create an invalid database URL.*

- **CoreDbUrl** (const [CoreDbUrl](#) &url)
- **CoreDbUrl** (const [QUrl](#) &digikamUrl)
  - Create a [CoreDbUrl](#) object from a [QUrl](#), to retrieve the information stored.*
- [QString](#) **album** () const
  - Returns the album: This is the directory hierarchy below the album root.*
- [QUrl](#) **albumRoot** () const
  - The following methods are only applicable for a certain protocol each.*
- int **albumRootId** () const
  - Returns the album root id.*
- [QString](#) **albumRootPath** () const
  - Returns the album root path of the file or album referenced by this URL In the example above, this is "/media/fotos".*
- bool [areaCoordinates](#) (double \*lat1, double \*lat2, double \*lon1, double \*lon2) const
  - MapImages URL.*
- [QDate](#) **endDate** () const
  - Return the referenced end date (excluded from the referenced span)*
- [QUrl](#) **fileUrl** () const
  - Converts this digikamalbums:// URL to a [file://](#) URL.*
- bool [isAlbumUrl](#) () const
  - These test for the protocol of this URL.*
- bool [isDateUrl](#) () const
- bool [isMapImagesUrl](#) () const
- bool [isSearchUrl](#) () const
- bool [isTagUrl](#) () const
- [QString](#) **name** () const
  - Returns the file name.*
- [CoreDbUrl](#) & **operator=** (const [CoreDbUrl](#) &url)
- [CoreDbUrl](#) & **operator=** (const [QUrl](#) &digikamalbumsUrl)
- bool **operator==** (const [QUrl](#) &digikamalbumsUrl) const
- [DbEngineParameters](#) **parameters** () const
  - Returns the [DbEngineParameters](#) stored in this URL.*
- int [searchId](#) () const
  - Search URL.*
- void **setParameters** (const [DbEngineParameters](#) &parameters)
  - Change the database parameters stored in this URL Applicable to all protocols.*
- [QDate](#) **startDate** () const
  - Date URL.*
- int [tagId](#) () const
  - Tag URL.*
- [QList](#)< int > **tagIds** () const
  - Returns the tag ids of all tags in the tag path of this tag, the topmost tag in the hierarchy first.*

### Static Public Member Functions

- static [CoreDbUrl](#) **albumUrl** (const [DbEngineParameters](#) &parameters=[CoreDbAccess::parameters](#)())
  - Create an empty digikamalbums:/ url.*
- static [CoreDbUrl](#) **dateUrl** (const [DbEngineParameters](#) &parameters=[CoreDbAccess::parameters](#)())
  - Create an empty digikamdates:/ url.*
- static [CoreDbUrl](#) **fromAlbumAndName** (const [QString](#) &name, const [QString](#) &album, const [QUrl](#) &albumRoot, const [DbEngineParameters](#) &parameters=[CoreDbAccess::parameters](#)())
- static [CoreDbUrl](#) **fromAlbumAndName** (const [QString](#) &name, const [QString](#) &album, const [QUrl](#) &albumRoot, int albumRootId, const [DbEngineParameters](#) &parameters=[CoreDbAccess::parameters](#)())

Create a `digikamalbums:/` url from an album name and an image in this album.

- static `CoreDbUrl fromAreaRange` (const qreal lat1, const qreal lng1, const qreal lat2, const qreal lng2, const `DbEngineParameters` &parameters=`CoreDbAccess::parameters()`)
- static `CoreDbUrl fromDateForMonth` (const QDate &date, const `DbEngineParameters` &parameters=`CoreDbAccess::parameters()`)

Create a `digikamdates:/` url for the month of the given date.

- static `CoreDbUrl fromDateForYear` (const QDate &date, const `DbEngineParameters` &parameters=`CoreDbAccess::parameters()`)

Create a `digikamdates:/` url for the year of the given date.

- static `CoreDbUrl fromDateRange` (const QDate &startDate, const QDate &endDate, const `DbEngineParameters` &parameters=`CoreDbAccess::parameters()`)

Create a `digikamdates:/` url for a specified time span which begin with the start date (inclusive) and ends before the end date (exclusive).

- static `CoreDbUrl fromFileUrl` (const QUrl &fileUrl, const QUrl &albumRoot, const `DbEngineParameters` &parameters=`CoreDbAccess::parameters()`)
- static `CoreDbUrl fromFileUrl` (const QUrl &fileUrl, const QUrl &albumRoot, int albumRootId, const `DbEngineParameters` &parameters=`CoreDbAccess::parameters()`)

This class shall facilitate the usage of `digikamalbums:/`, `digikamtags:/`, `digikamdates:/` and `digikamsearch:` URLs.

- static `CoreDbUrl fromTagIds` (const QList< int > &tagIds, const `DbEngineParameters` &parameters=`CoreDbAccess::parameters()`)

Create a `digikamtags:/` url from a list of tag IDs, where this list is the tag hierarchy of the referenced tag, with the topmost parent first, and the tag last in the list.

- static `CoreDbUrl mapImagesUrl` (const `DbEngineParameters` &parameters=`CoreDbAccess::parameters()`)

Create an empty `digikammapimages:/` url.

- static `CoreDbUrl searchUrl` (int searchId, const `DbEngineParameters` &parameters=`CoreDbAccess::parameters()`)

Create a `digikamsearch:` URL for the search with the given id.

## 9.216.1 Member Function Documentation

### 9.216.1.1 album()

```
QString Digikam::CoreDbUrl::album ( ) const
```

In the example above, the album is `"/Summer 2007"`

### 9.216.1.2 albumRoot()

```
QUrl Digikam::CoreDbUrl::albumRoot ( ) const
```

If the URL has another protocol, the return value of these methods is undefined. `Album` URL Returns the album root URL of the file or album referenced by this URL In the example above, this is `"file:///media/fotos"`

### 9.216.1.3 areaCoordinates()

```
bool Digikam::CoreDbUrl::areaCoordinates (
    double * lat1,
    double * lat2,
    double * lon1,
    double * lon2 ) const
```

Returns the coordinates surrounding the map area. Returns true if the string to number conversion was ok.

#### 9.216.1.4 fromAlbumAndName()

```
CoreDbUrl Digikam::CoreDbUrl::fromAlbumAndName (
    const QString & name,
    const QString & album,
    const QUrl & albumRoot,
    int albumRootId,
    const DbEngineParameters & parameters = CoreDbAccess::parameters() ) [static]
```

If name is empty, the album is referenced. Other parameters as above.

#### 9.216.1.5 fromDateForMonth()

```
CoreDbUrl Digikam::CoreDbUrl::fromDateForMonth (
    const QDate & date,
    const DbEngineParameters & parameters = CoreDbAccess::parameters() ) [static]
```

(The whole month of the given date will included in the referenced time span)

#### 9.216.1.6 fromDateForYear()

```
CoreDbUrl Digikam::CoreDbUrl::fromDateForYear (
    const QDate & date,
    const DbEngineParameters & parameters = CoreDbAccess::parameters() ) [static]
```

(The whole year of the given date will included in the referenced time span)

#### 9.216.1.7 fromDateRange()

```
CoreDbUrl Digikam::CoreDbUrl::fromDateRange (
    const QDate & startDate,
    const QDate & endDate,
    const DbEngineParameters & parameters = CoreDbAccess::parameters() ) [static]
```

To cover the whole year of 1984, you would pass 1/1/1984 and 1/1/1985.

#### 9.216.1.8 fromFileUrl()

```
CoreDbUrl Digikam::CoreDbUrl::fromFileUrl (
    const QUrl & fileUrl,
    const QUrl & albumRoot,
    int albumRootId,
    const DbEngineParameters & parameters = CoreDbAccess::parameters() ) [static]
```

It provides functions to set and get the parameters stored in such a URL. (with the exception of the search parameters in a search URL, which are out of the scope of this class.) Create a digikamalbums:/ URL from a `file://` URL. The file URL can point to a file or a directory (an album in this case). The additional information stored in the URL need to be supplied as well:

- The album root in which the entity pointed to is stored. This is the left part of the file URL. (if the file is `"/media/fotos/Summer 2007/001.jpg"`, the album root may be `"/media/fotos"`)
- The parameters of the database that is referenced



### 9.216.1.9 fromTagIds()

```
CoreDbUrl Digikam::CoreDbUrl::fromTagIds (
    const QList< int > & tagIds,
    const DbEngineParameters & parameters = CoreDbAccess::parameters() ) [static]
```

An empty list references the root tag.

### 9.216.1.10 isAlbumUrl()

```
bool Digikam::CoreDbUrl::isAlbumUrl ( ) const
```

The protocol string is of course available via protocol().

### 9.216.1.11 name()

```
QString Digikam::CoreDbUrl::name ( ) const
```

In the example above, this is "001.jpg"

### 9.216.1.12 parameters()

```
DbEngineParameters Digikam::CoreDbUrl::parameters ( ) const
```

Applicable to all protocols.

### 9.216.1.13 searchId()

```
int Digikam::CoreDbUrl::searchId ( ) const
```

Return the id of the search.

### 9.216.1.14 startDate()

```
QDate Digikam::CoreDbUrl::startDate ( ) const
```

Return the referenced start date (included in the referenced span)

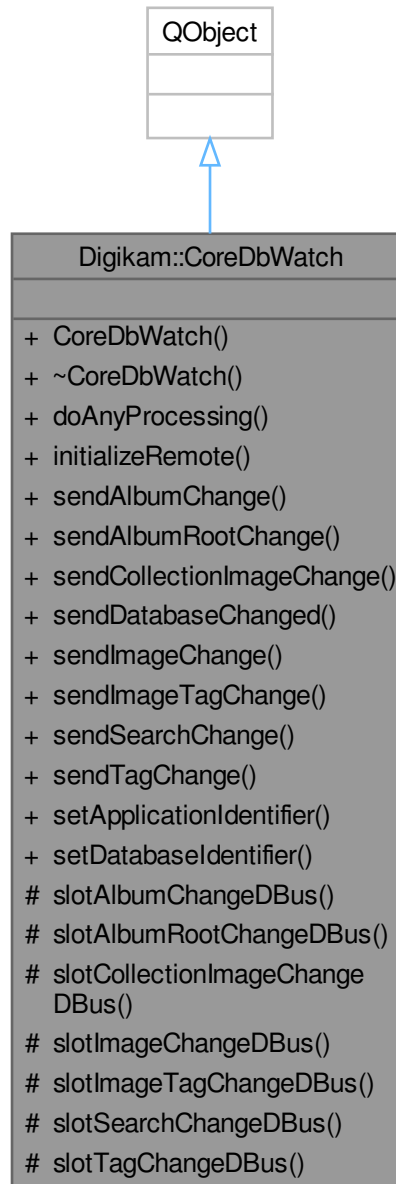
### 9.216.1.15 tagId()

```
int Digikam::CoreDbUrl::tagId ( ) const
```

Returns the tag ID, or -1 if the root tag is referenced

## 9.217 Digikam::CoreDbWatch Class Reference

Inheritance diagram for Digikam::CoreDbWatch:



### Public Types

- enum **DatabaseMode** { **DatabaseMaster** , **DatabaseSlave** }

## Signals

- void **albumChange** (const [AlbumChangeset](#) &changeset)
- void **albumRootChange** (const [AlbumRootChangeset](#) &changeset)
- void **collectionImageChange** (const [CollectionImageChangeset](#) &changeset)
- void **databaseChanged** ()
  - Retrieve the *CoreDbWatch* object from *CoreDbAccess::databaseWatch()*.
- void **imageChange** (const [ImageChangeset](#) &changeset)
  - *Notifies of changes in the database.*
- void **imageTagChange** (const [ImageTagChangeset](#) &changeset)
- void **searchChange** (const [SearchChangeset](#) &changeset)
- void **signalAlbumChangeDBus** (const QString &databaseIdentifier, const QString &applicationIdentifier, const [Digikam::AlbumChangeset](#) &changeset)
- void **signalAlbumRootChangeDBus** (const QString &databaseIdentifier, const QString &applicationIdentifier, const [Digikam::AlbumRootChangeset](#) &changeset)
- void **signalCollectionImageChangeDBus** (const QString &databaseIdentifier, const QString &applicationIdentifier, const [Digikam::CollectionImageChangeset](#) &changeset)
- void **signalImageChangeDBus** (const QString &databaseIdentifier, const QString &applicationIdentifier, const [Digikam::ImageChangeset](#) &changeset)
  - *DBus signals, for internal use.*
- void **signalImageTagChangeDBus** (const QString &databaseIdentifier, const QString &applicationIdentifier, const [Digikam::ImageTagChangeset](#) &changeset)
- void **signalSearchChangeDBus** (const QString &databaseIdentifier, const QString &applicationIdentifier, const [Digikam::SearchChangeset](#) &changeset)
- void **signalTagChangeDBus** (const QString &databaseIdentifier, const QString &applicationIdentifier, const [Digikam::TagChangeset](#) &changeset)
- void **tagChange** (const [TagChangeset](#) &changeset)

## Public Member Functions

- void **doAnyProcessing** ()
- void **initializeRemote** (DatabaseMode mode)
- void **sendAlbumChange** (const [AlbumChangeset](#) &changeset)
- void **sendAlbumRootChange** (const [AlbumRootChangeset](#) &changeset)
- void **sendCollectionImageChange** (const [CollectionImageChangeset](#) &changeset)
- void **sendDatabaseChanged** ()
  - *library-internal signal-trigger methods*
- void **sendImageChange** (const [ImageChangeset](#) &changeset)
- void **sendImageTagChange** (const [ImageTagChangeset](#) &changeset)
- void **sendSearchChange** (const [SearchChangeset](#) &changeset)
- void **sendTagChange** (const [TagChangeset](#) &changeset)
- void **setApplicationIdentifier** (const QString &identifier)
- void **setDatabaseIdentifier** (const QString &identifier)

## Protected Slots

- void **slotAlbumChangeDBus** (const QString &databaseIdentifier, const QString &applicationIdentifier, const [Digikam::AlbumChangeset](#) &changeset)
- void **slotAlbumRootChangeDBus** (const QString &databaseIdentifier, const QString &applicationIdentifier, const [Digikam::AlbumRootChangeset](#) &changeset)
- void **slotCollectionImageChangeDBus** (const QString &databaseIdentifier, const QString &applicationIdentifier, const [Digikam::CollectionImageChangeset](#) &changeset)

- void **slotImageChangeDBus** (const QString &databaseIdentifier, const QString &applicationIdentifier, const [Digikam::ImageChangeset](#) &changeset)  
*DBus slots, for internal use.*
- void **slotImageTagChangeDBus** (const QString &databaseIdentifier, const QString &applicationIdentifier, const [Digikam::ImageTagChangeset](#) &changeset)
- void **slotSearchChangeDBus** (const QString &databaseIdentifier, const QString &applicationIdentifier, const [Digikam::SearchChangeset](#) &changeset)
- void **slotTagChangeDBus** (const QString &databaseIdentifier, const QString &applicationIdentifier, const [Digikam::TagChangeset](#) &changeset)

## 9.217.1 Member Function Documentation

### 9.217.1.1 databaseChanged

```
void Digikam::CoreDbWatch::databaseChanged ( ) [signal]
```

This does not describe a change of the contents of a table; rather, it signals that a new database has been loaded. That means all cached content has to be discarded.

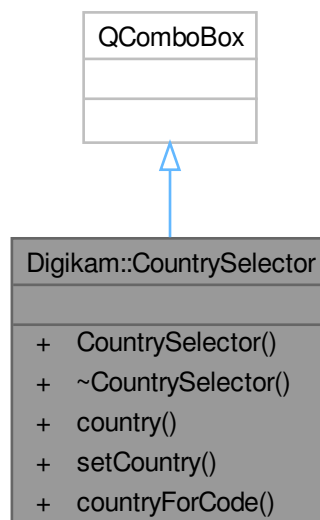
### 9.217.1.2 imageChange

```
void Digikam::CoreDbWatch::imageChange (
    const ImageChangeset & changeset ) [signal]
```

Connect to the set of signals that you are interested in.

## 9.218 Digikam::CountrySelector Class Reference

Inheritance diagram for Digikam::CountrySelector:



**Public Member Functions**

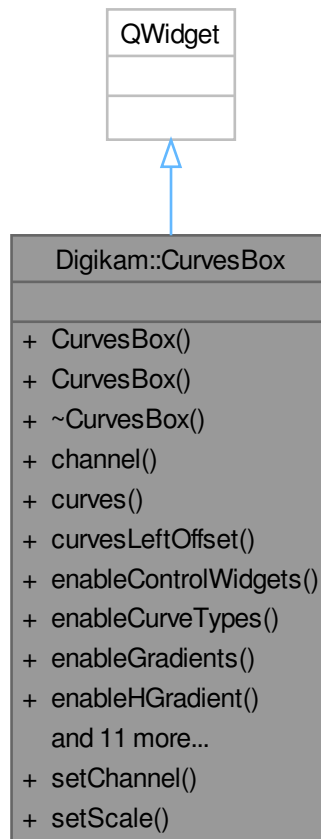
- **CountrySelector** (QWidget \*const parent)
- bool **country** (QString &countryCode, QString &countryName) const
- void **setCountry** (const QString &countryCode)

**Static Public Member Functions**

- static QString **countryForCode** (const QString &countryCode)

**9.219 Digikam::CurvesBox Class Reference**

Inheritance diagram for Digikam::CurvesBox:

**Public Types**

- enum **ColorPicker** { **NoPicker** = -1 , **BlackTonal** = 0 , **GrayTonal** , **WhiteTonal** }
- enum **CurvesDrawingType** { **SmoothDrawing** = 0 , **FreeDrawing** }

### Public Slots

- void **setChannel** (ChannelType channel)
- void **setScale** ([HistogramScale](#) scale)

### Signals

- void **signalChannelReset** (int)
- void **signalCurvesChanged** ()
- void **signalCurveTypeChanged** (int)
- void **signalPickerChanged** (int)

### Public Member Functions

- **CurvesBox** (int w, int h, const [DImg](#) &img, QWidget \*const parent=nullptr, bool readOnly=false)
- **CurvesBox** (int w, int h, QWidget \*const parent=nullptr, bool readOnly=false)
- ChannelType **channel** () const
- [ImageCurves](#) \* **curves** () const
- int **curvesLeftOffset** () const
- void **enableControlWidgets** (bool enable)
- void **enableCurveTypes** (bool enable)
- void **enableGradients** (bool enable)
- void **enableHGradient** (bool enable)
- void **enablePickers** (bool enable)
- void **enableResetButton** (bool enable)
- void **enableVGradient** (bool enable)
- int **picker** () const
- void **readCurveSettings** (KConfigGroup &group, const QString &prefix)
- void **reset** ()
- void **resetChannel** (int channel)
- void **resetChannels** ()
- void **resetPickers** ()
- void **setCurveGuide** (const [DColor](#) &color)
- void **writeCurveSettings** (KConfigGroup &group, const QString &prefix)

## 9.220 Digikam::CurvesContainer Class Reference

### Public Member Functions

- [CurvesContainer](#) ()=default  
*Provides a convenient storage for a curve.*
- **CurvesContainer** (int type, bool sixteenBit)
- void **initialize** ()  
*Fills the values with a linear curve suitable for type and sixteenBit parameters.*
- bool **isEmpty** () const  
*An empty container is interpreted as a linear curve.*
- bool **isStoredLosslessly** () const  
*Serialize from and to [FilterAction](#).*
- bool **operator==** (const [CurvesContainer](#) &other) const
- void **writeToFilterAction** ([FilterAction](#) &action, const QString &prefix=QString()) const

## Static Public Member Functions

- static [CurvesContainer](#) **fromFilterAction** (const [FilterAction](#) &action, const QString &prefix=QString())

## Public Attributes

- int **curvesType** = [ImageCurves::CURVE\\_SMOOTH](#)  
*Smooth : QPolygon have size of 18 points.*
- bool **sixteenBit** = false
- QPolygon **values** [ColorChannels]

## 9.220.1 Constructor & Destructor Documentation

### 9.220.1.1 CurvesContainer()

```
Digikam::CurvesContainer::CurvesContainer ( ) [default]
```

Initially, the values are empty. Call [initialize\(\)](#) before adjusting values manually.

## 9.220.2 Member Function Documentation

### 9.220.2.1 isEmpty()

```
bool Digikam::CurvesContainer::isEmpty ( ) const
```

A non-empty container can also be linear; test for [isLinear\(\)](#) of the resulting [ImageCurves](#). Note: If an [ImageCurves](#) is linear, it will return an empty container.

### 9.220.2.2 isStoredLosslessly()

```
bool Digikam::CurvesContainer::isStoredLosslessly ( ) const
```

[isStoredLosslessly](#) returns false if the curve cannot be losslessly stored in XML because it would be too large (free 16 bit). It is then lossily compressed.

## 9.220.3 Member Data Documentation

### 9.220.3.1 curvesType

```
int Digikam::CurvesContainer::curvesType = ImageCurves::CURVE\_SMOOTH
```

Free : QPolygon have size of 255 or 65535 values.

## 9.221 Digikam::CurvesFilter Class Reference

Inheritance diagram for Digikam::CurvesFilter:



### Public Member Functions

- **CurvesFilter** (const [CurvesContainer](#) &settings, [DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, [DImg](#) &destImage, int progressBegin=0, int progressEnd=100)



- **CurvesFilter** ([DImg](#) \*const orgImage, [QObject](#) \*const parent=nullptr, const [CurvesContainer](#) &settings=[CurvesContainer](#)())
- **CurvesFilter** ([QObject](#) \*const parent=nullptr)
- [FilterAction](#) **filterAction** () override
 

*Returns the action description corresponding to currently set options.*
- [QString](#) **filterIdentifier** () const override
 

*Return the identifier for this filter in the image history.*
- void [readParameters](#) (const [FilterAction](#) &action) override

## Public Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImg](#) \*const orgImage, [QObject](#) \*const parent, const [QString](#) &name=[QString](#)())
 

*Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter](#) ([QObject](#) \*const parent=nullptr, const [QString](#) &name=[QString](#)())
 

*Constructs a filter without argument.*
- virtual void **cancelFilter** ()
 

*Cancel the threaded computation.*
- const [QString](#) & **filterName** ()
- int **filterVersion** () const
- [DImg](#) **getTargetImage** ()
- [QList](#)< int > **multithreadedSteps** (int stop, int start=0) const
 

*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool **parametersSuccessfullyRead** () const
 

*Optional: error handling for readParameters.*
- virtual [QString](#) **readParametersError** (const [FilterAction](#) &actionThatFailed) const
- void **setFilterName** (const [QString](#) &name)
- void **setFilterVersion** (int version)
 

*Replaying a filter action: Set the filter version.*
- void **setOriginalImage** (const [DImg](#) &orgImage)
- void **setupAndStartDirectly** (const [DImg](#) &orgImage, [DImgThreadedFilter](#) \*const master, int progress←Begin=0, int progressEnd=100)
 

*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void **setupFilter** (const [DImg](#) &orgImage)
 

*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void **startFilter** ()
 

*Start the threaded computation.*
- virtual void **startFilterDirectly** ()
 

*Start computation of this filter, directly in this thread.*
- virtual [QList](#)< int > **supportedVersions** () const

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread](#) ([QObject](#) \*const parent=nullptr)
 

*This class extends [QRunnable](#), so you have to reimplement virtual void [run\(\)](#).*
- **~DynamicThread** () override
 

*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool **isFinished** () const
- bool **isRunning** () const
- [QThread::Priority](#) **priority** () const
- void **setEmitSignals** (bool emitThem)
- void **setPriority** ([QThread::Priority](#) priority)
 

*Sets the priority for this dynamic thread.*
- State **state** () const

### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from Digikam::DImgThreadedFilter

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from Digikam::DynamicThread

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from Digikam::DImgThreadedFilter

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.221.1 Member Function Documentation

### 9.221.1.1 filterAction()

`FilterAction` Digikam::CurvesFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.221.1.2 filterIdentifier()

`QString` Digikam::CurvesFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

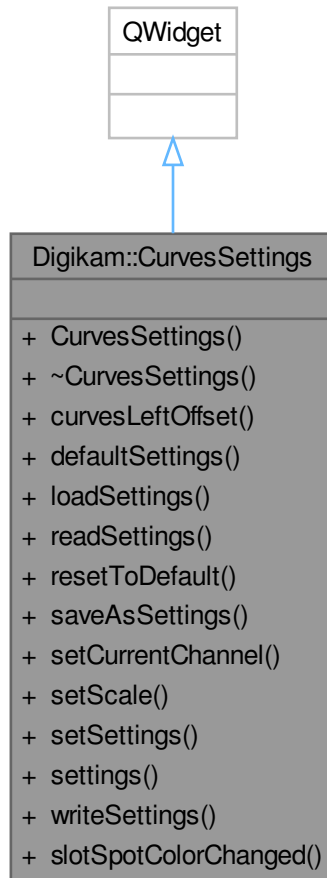
### 9.221.1.3 readParameters()

```
void Digikam::CurvesFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

## 9.222 Digikam::CurvesSettings Class Reference

Inheritance diagram for Digikam::CurvesSettings:



### Public Slots

- void `slotSpotColorChanged` (const [Digikam::DColor](#) &color)

### Signals

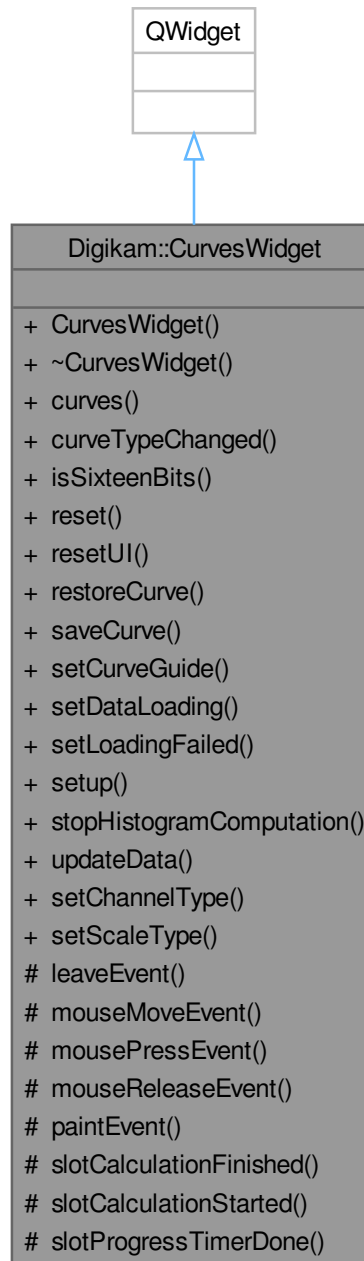
- void `signalChannelReset` (int)
- void `signalPickerChanged` (int)
- void `signalSettingsChanged` ()
- void `signalSpotColorChanged` ()

## Public Member Functions

- **CurvesSettings** (QWidget \*const parent, [DImg](#) \*const img)
- int **curvesLeftOffset** () const
- [CurvesContainer](#) **defaultSettings** () const
- void **loadSettings** ()
- void **readSettings** (KConfigGroup &group)
- void **resetToDefault** ()
- void **saveAsSettings** ()
- void **setCurrentChannel** (ChannelType channel)
- void **setScale** ([HistogramScale](#) type)
- void **setSettings** (const [CurvesContainer](#) &settings)
- [CurvesContainer](#) **settings** () const
- void **writeSettings** (KConfigGroup &group)

## 9.223 Digikam::CurvesWidget Class Reference

Inheritance diagram for Digikam::CurvesWidget:



### Public Slots

- void **setChannelType** (ChannelType channel)
- void **setScaleType** ([HistogramScale](#) scale)

## Signals

- void **signalCurvesChanged** ()
- void **signalHistogramComputationDone** ()
- void **signalHistogramComputationFailed** ()
- void **signalMouseMoved** (int x, int y)

## Public Member Functions

- **CurvesWidget** (int w, int h, QWidget \*const parent, bool readOnly=false)
- **ImageCurves** \* **curves** () const
- void **curveTypeChanged** ()
- bool **isSixteenBits** () const
- void **reset** ()
  - *Resets the ui including the user specified curve.*
- void **resetUI** ()
  - *Resets only the ui and keeps the curve.*
- void **restoreCurve** (const KConfigGroup &group, const QString &prefix)
  - *Restores the curve tfrom the given group with prefix as a prefix for the curve point config entries.*
- void **saveCurve** (KConfigGroup &group, const QString &prefix)
  - *Saves the currently created curve to the given group with prefix as a prefix for the curve point config entries.*
- void **setCurveGuide** (const QColor &color)
- void **setDataLoading** ()
- void **setLoadingFailed** ()
- void **setup** (int w, int h, bool readOnly)
- void **stopHistogramComputation** ()
  - *Stop current histogram computations.*
- void **updateData** (const QImage &img)
  - *Updates the image data the curve should be used for.*

## Protected Slots

- void **slotCalculationFinished** (bool success)
- void **slotCalculationStarted** ()
- void **slotProgressTimerDone** ()

## Protected Member Functions

- void **leaveEvent** (QEvent \*) override
- void **mouseMoveEvent** (QMouseEvent \*) override
- void **mousePressEvent** (QMouseEvent \*) override
- void **mouseReleaseEvent** (QMouseEvent \*) override
- void **paintEvent** (QPaintEvent \*) override

## 9.223.1 Member Function Documentation

### 9.223.1.1 restoreCurve()

```
void Digikam::CurvesWidget::restoreCurve (
    const KConfigGroup & group,
    const QString & prefix )
```



## Parameters

<i>group</i>	the group to restore the curve from
<i>prefix</i>	the prefix prepended to the point numbers in the config

**9.223.1.2 saveCurve()**

```
void Digikam::CurvesWidget::saveCurve (
    KConfigGroup & group,
    const QString & prefix )
```

## Parameters

<i>group</i>	the group to save the curve to
<i>prefix</i>	the prefix prepended to the point numbers in the config

**9.223.1.3 updateData()**

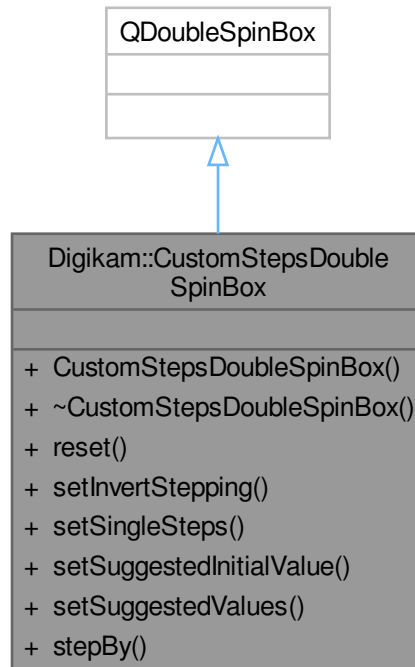
```
void Digikam::CurvesWidget::updateData (
    const DImg & img )
```

## Parameters

<i>img</i>	image data
------------	------------

## 9.224 Digikam::CustomStepsDoubleSpinBox Class Reference

Inheritance diagram for Digikam::CustomStepsDoubleSpinBox:



### Public Member Functions

- **CustomStepsDoubleSpinBox** (QWidget \*const parent=nullptr)
  - This is a normal QDoubleSpinBox which allows to customize the stepping behavior, for cases where linear steps are not applicable.*
- void **reset** ()
  - Resets to minimum value.*
- void **setInvertStepping** (bool invert)
- void **setSingleSteps** (double smaller, double larger)
  - Allows to set to different default single steps, for the range below m\_values, the other for above.*
- void **setSuggestedInitialValue** (double initialValue)
  - Sets the value that should be set as first value when first moving away from the minimum value.*
- void **setSuggestedValues** (const QList< double > &values)
  - Set a list of values that are usually applicable for the type of data of the combo box.*
- void **stepBy** (int steps) override

## 9.224.1 Member Function Documentation

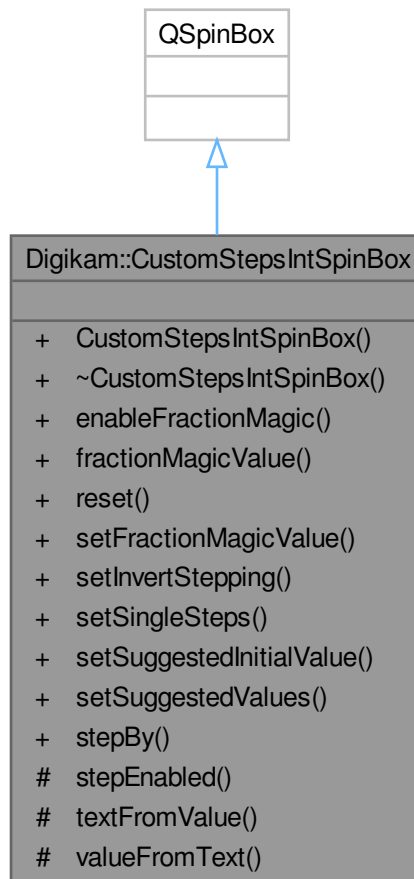
### 9.224.1.1 setSuggestedValues()

```
void Digikam::CustomStepsDoubleSpinBox::setSuggestedValues (
    const QList< double > & values )
```

The user can still type in any other value. Boundaries are not touched. Up or below the min and max values of the list given, default stepping is used.

## 9.225 Digikam::CustomStepsIntSpinBox Class Reference

Inheritance diagram for Digikam::CustomStepsIntSpinBox:



## Public Member Functions

- **CustomStepsIntSpinBox** (QWidget \*const parent=nullptr)  
*This is a normal QIntSpinBox which allows to customize the stepping behavior, for cases where linear steps are not applicable.*
- void **enableFractionMagic** (const QString &prefix)  
*Call this with a fraction prefix (like "1/") to enable magic handling of the value as fraction denominator.*
- double **fractionMagicValue** () const  
*value() and setValue() for fraction magic value.*
- void **reset** ()  
*Resets to minimum value.*
- void **setFractionMagicValue** (double value)
- void **setInvertStepping** (bool invert)
- void **setSingleSteps** (int smaller, int larger)  
*Allows to set to different default single steps, for the range below m\_values, the other for above.*
- void **setSuggestedInitialValue** (int initialValue)  
*Sets the value that should be set as first value when first moving away from the minimum value.*
- void **setSuggestedValues** (const QList< int > &values)  
*Set a list of values that are usually applicable for the type of data of the combo box.*
- void **stepBy** (int steps) override

## Protected Member Functions

- StepEnabled **stepEnabled** () const override
- QString **textFromValue** (int value) const override
- int **valueFromText** (const QString &text) const override

## 9.225.1 Member Function Documentation

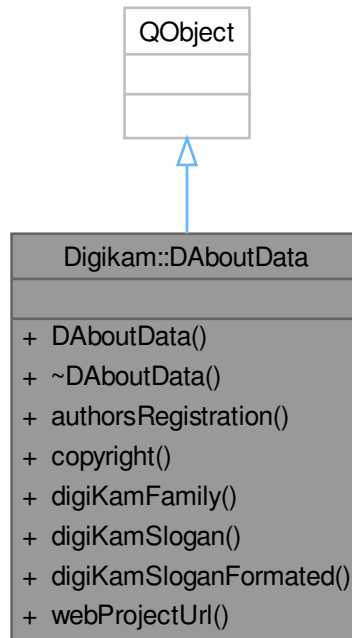
### 9.225.1.1 setSuggestedValues()

```
void Digikam::CustomStepsIntSpinBox::setSuggestedValues (
    const QList< int > & values )
```

The user can still type in any other value. Boundaries are not touched. Up or below the min and max values of the list given, default stepping is used.

## 9.226 Digikam::DAboutData Class Reference

Inheritance diagram for Digikam::DAboutData:



### Public Member Functions

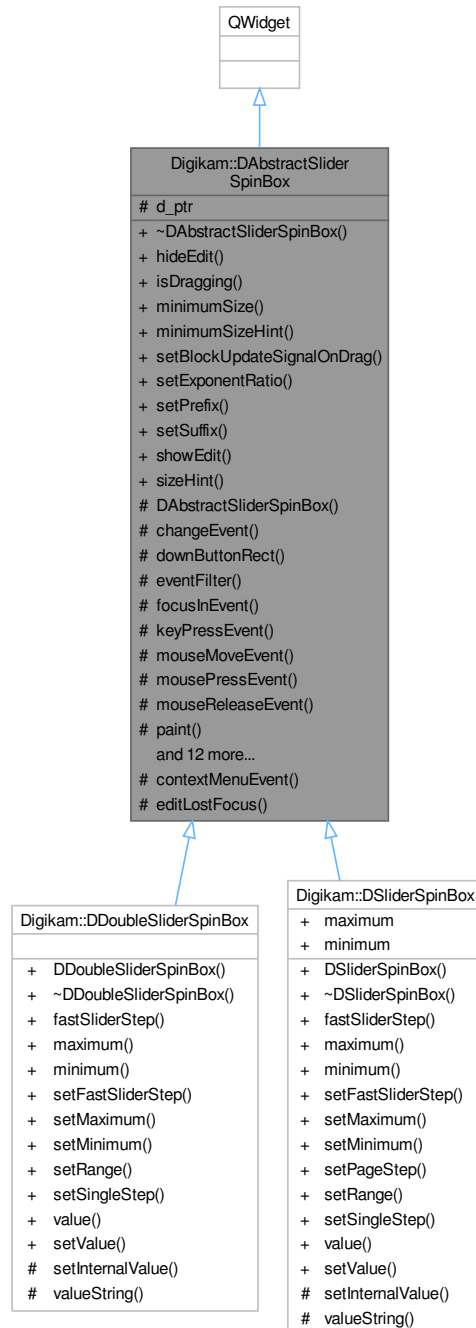
- `DAboutData` (`DXmlGuiWindow *const parent`)

### Static Public Member Functions

- static void `authorsRegistration` (`KAboutData &aboutData`)
- static const `QString copyright` ()
- static const `QString digiKamFamily` ()
- static const `QString digiKamSlogan` ()
- static const `QString digiKamSloganFormatted` ()
- static const `QUrl webProjectUrl` ()

## 9.227 Digikam::DAbstractSliderSpinBox Class Reference

Inheritance diagram for Digikam::DAbstractSliderSpinBox:



### Public Member Functions

- void `hideEdit ()`
- bool `isDragging ()` const

- virtual QSize **minimumSize** () const
- QSize **minimumSizeHint** () const override
- void **setBlockUpdateSignalOnDrag** (bool block)
  - If set to block, it informs inheriting classes that they shouldn't emit signals if the update comes from a mouse dragging the slider.*
- void **setExponentRatio** (double dbl)
- void **setPrefix** (const QString &prefix)
- void **setSuffix** (const QString &suffix)
- void **showEdit** ()
- QSize **sizeHint** () const override

### Protected Slots

- void **contextMenuEvent** (QContextMenuEvent \*event) override
- void **editLostFocus** ()

### Protected Member Functions

- **DAbstractSliderSpinBox** (QWidget \*const parent, DAbstractSliderSpinBoxPrivate \*const q)
- void **changeEvent** (QEvent \*e) override
- QRect **downButtonRect** (const QStyleOptionSpinBox &spinBoxOptions) const
- bool **eventFilter** (QObject \*recv, QEvent \*e) override
- void **focusInEvent** (QFocusEvent \*e) override
- void **keyPressEvent** (QKeyEvent \*e) override
- void **mouseMoveEvent** (QMouseEvent \*e) override
- void **mousePressEvent** (QMouseEvent \*e) override
- void **mouseReleaseEvent** (QMouseEvent \*e) override
- void **paint** (QPainter &painter)
- void **paintBreeze** (QPainter &painter)
- void **paintEvent** (QPaintEvent \*e) override
- void **paintFusion** (QPainter &painter)
- void **paintPlastique** (QPainter &painter)
- QStyleOptionProgressBar **progressBarOptions** () const
- QRect **progressRect** (const QStyleOptionSpinBox &spinBoxOptions) const
- virtual void **setInternalValue** (int value, bool blockUpdateSignal)=0
  - Sets the slider internal value.*
- QStyleOptionSpinBox **spinBoxOptions** () const
- QRect **upButtonRect** (const QStyleOptionSpinBox &spinBoxOptions) const
- int **valueForX** (int x, Qt::KeyboardModifiers modifiers=Qt::NoModifier) const
- virtual QString **valueString** () const =0
- void **wheelEvent** (QWheelEvent \*e) override

### Protected Attributes

- DAbstractSliderSpinBoxPrivate \*const **d\_ptr**

## 9.227.1 Member Function Documentation

### 9.227.1.1 setBlockUpdateSignalOnDrag()

```
void Digikam::DAbstractSliderSpinBox::setBlockUpdateSignalOnDrag (
    bool block )
```

Set this to true when dragging the slider and updates during the drag are not needed.

### 9.227.1.2 setInternalValue()

```
virtual void Digikam::DAbstractSliderSpinBox::setInternalValue (
    int value,
    bool blockUpdateSignal ) [protected], [pure virtual]
```

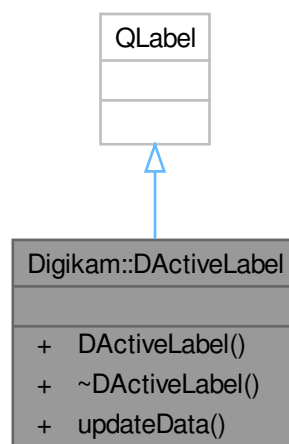
Inheriting classes should respect blockUpdateSignal so that, in specific cases, we have a performance improvement. See setIgnoreMouseMoveEvents.

Implemented in [Digikam::DSliderSpinBox](#), and [Digikam::DDoubleSliderSpinBox](#).

## 9.228 Digikam::DActiveLabel Class Reference

A widget to host an image into a label with an active url which can be open to default web browser using simple mouse click.

Inheritance diagram for Digikam::DActiveLabel:



### Public Member Functions

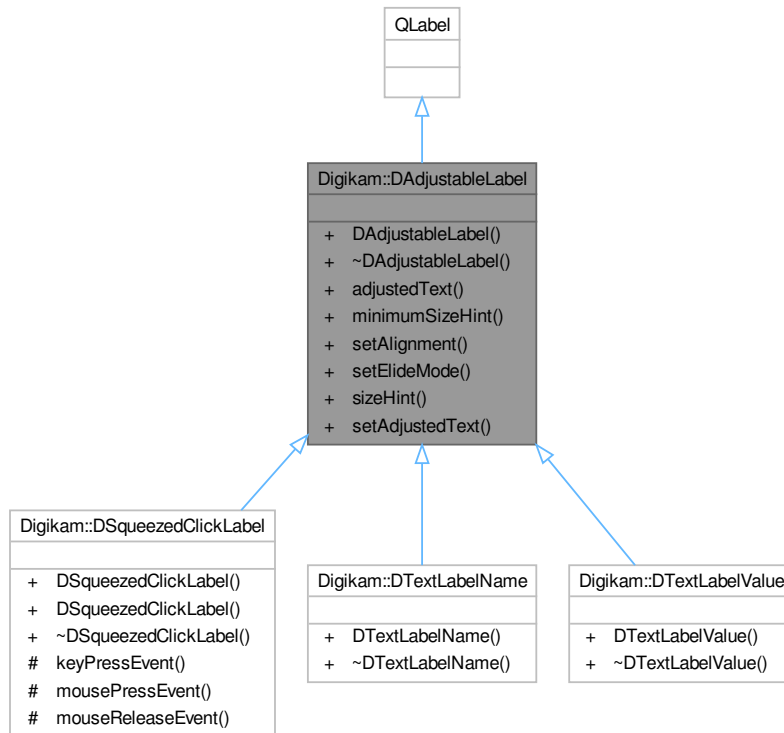
- **DActiveLabel** (const `QUrl` &url=`QUrl()`, const `QString` &imgPath=`QString()`, `QWidget` \*const parent=`nullptr`)
- void **updateData** (const `QUrl` &url, const `QImage` &img)



## 9.229 Digikam::DAdjustableLabel Class Reference

A label to show text adjusted to widget size.

Inheritance diagram for Digikam::DAdjustableLabel:



### Public Slots

- void **setAdjustedText** (const QString &text=QString())

### Public Member Functions

- **DAdjustableLabel** (QWidget \*const parent=nullptr)
- QString **adjustedText** () const
- QSize **minimumSizeHint** () const override
- void **setAlignment** (Qt::Alignment align)
- void **setElideMode** (Qt::TextElideMode mode)
- QSize **sizeHint** () const override

## 9.230 Digikam::DAAlbum Class Reference

A Date [Album](#) representation.

Inheritance diagram for Digikam::DAAlbum:



### Public Types

- enum `Range` { `Month = 0` , `Year` }

## Public Types inherited from Digikam::Album

- enum [Type](#) {  
[PHYSICAL](#) = 0 , [TAG](#) , [DATE](#) , [SEARCH](#) ,  
[FACE](#) }

## Public Member Functions

- **DAIbum** (const [QDate](#) &date, bool root=false, [Range](#) range=Month)
- [CoreDbUrl databaseUrl](#) () const override
- [QDate date](#) () const
- [Range range](#) () const

## Public Member Functions inherited from Digikam::Album

- [QList< int >](#) [childAlbumIds](#) (bool recursive=false)
- [AlbumList](#) [childAlbums](#) (bool recursive=false)
- [Album \\*](#) [childAtRow](#) (int row) const
- int [childCount](#) () const
- void \* [extraData](#) (const void \*const key) const  
*Retrieve the associated extra data associated with key.*
- [Album \\*](#) [firstChild](#) () const
- int [globalID](#) () const  
*An album ID is only unique among the set of all Albums of its Type.*
- int [id](#) () const  
*Each album has a ID uniquely identifying it in the set of Albums of a Type.*
- bool [isAncestorOf](#) ([Album \\*](#)const album) const
- bool [isRoot](#) () const
- bool [isTrashAlbum](#) () const
- bool [isUsedByLabelsTree](#) () const
- [Album \\*](#) [lastChild](#) () const
- [Album \\*](#) [next](#) () const
- [Album \\*](#) [parent](#) () const
- void **[prepareForDeletion](#)** ()  
*For secure deletion in an album model, call this function beforehand.*
- [Album \\*](#) [prev](#) () const
- void [removeExtraData](#) (const void \*const key)  
*Remove the associated extra data associated with key.*
- int [rowFromAlbum](#) () const
- void [setExtraData](#) (const void \*const key, void \*const value)  
*This allows to associate some "extra" data to a Album.*
- void [setUsedByLabelsTree](#) (bool isUsed)  
*Sets the property m\_usedByLabelsTree to true if the search album was created using the Colors and labels tree view.*
- [QString](#) [title](#) () const
- [Type](#) [type](#) () const

## Friends

- class **[AlbumManager](#)**

## Additional Inherited Members

### Static Public Member Functions inherited from [Digikam::Album](#)

- static int [globalID](#) ([Type type](#), int [id](#))  
*Produces the global id.*

### Protected Member Functions inherited from [Digikam::Album](#)

- **Album** ([Album::Type type](#), int [id](#), bool [root](#))  
*Constructor.*
- virtual [~Album](#) ()  
*Destructor.*
- void **clear** ()  
*Delete all child albums and also remove any associated extra data.*
- void **insertChild** ([Album \\*const child](#))
- void **removeChild** ([Album \\*const child](#))
- void **setParent** ([Album \\*const parent](#))
- void **setTitle** (const [QString &title](#))

## 9.230.1 Member Function Documentation

### 9.230.1.1 [databaseUrl\(\)](#)

```
CoreDbUrl Digikam::DAlbum::databaseUrl ( ) const [override], [virtual]
```

#### Returns

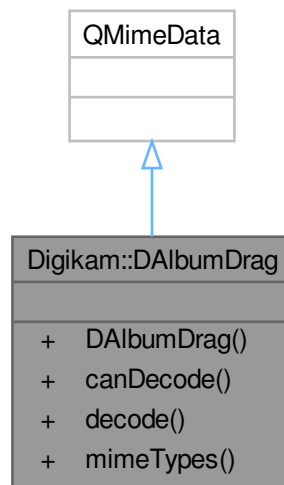
the kde url of the album

Implements [Digikam::Album](#).

## 9.231 [Digikam::DAlbumDrag](#) Class Reference

Provides a drag object for an album.

Inheritance diagram for Digikam::DAlbumDrag:



### Public Member Functions

- **DAlbumDrag** (const `QUrl` &databaseUrl, int albumid, const `QUrl` &fileUrl=`QUrl()`)

### Static Public Member Functions

- static bool **canDecode** (const `QMimeData` \*e)
- static bool **decode** (const `QMimeData` \*e, `QList`< `QUrl` > &urls, int &albumID)
- static `QStringList` **mimeTypees** ()

#### 9.231.1 Detailed Description

When an album is moved through drag'n'drop an object of this class is created.

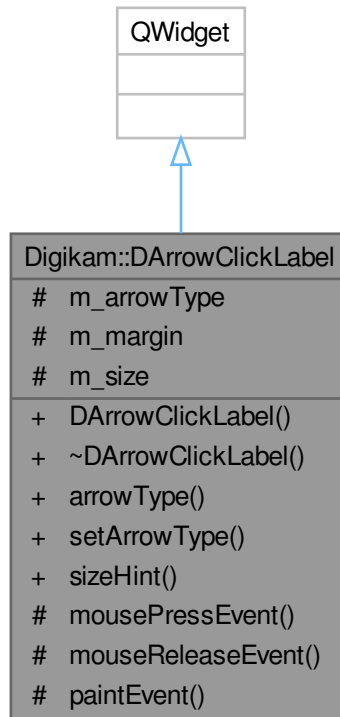
## 9.232 Digikam::DAlbumInfo Class Reference

### Public Member Functions

- **DAlbumInfo** (const `DInfoInterface::DInfoMap` &)
- `QString` **albumPath** () const
- `QString` **caption** () const
- `QDate` **date** () const
- `QString` **path** () const
- `QString` **title** () const

## 9.233 Digikam::DArrowClickLabel Class Reference

Inheritance diagram for Digikam::DArrowClickLabel:



### Signals

- void **leftClicked** ()

### Public Member Functions

- **DArrowClickLabel** (QWidget \*const parent=nullptr)
- Qt::ArrowType **arrowType** () const
- void **setArrowType** (Qt::ArrowType arrowType)
- QSize **sizeHint** () const override

### Protected Member Functions

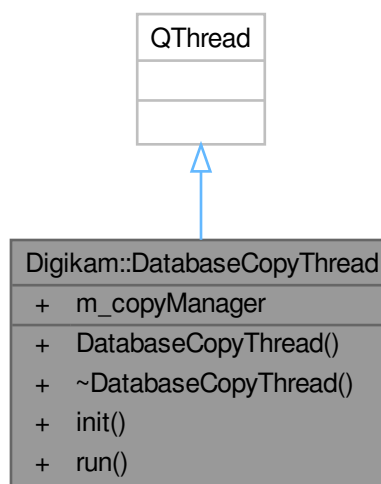
- void **mousePressEvent** (QMouseEvent \*event) override
- void **mouseReleaseEvent** (QMouseEvent \*event) override
- void **paintEvent** (QPaintEvent \*event) override

**Protected Attributes**

- Qt::ArrowType **m\_arrowType** = Qt::DownArrow
- int **m\_margin** = 2
- int **m\_size** = 8

**9.234 Digikam::DatabaseCopyThread Class Reference**

Inheritance diagram for Digikam::DatabaseCopyThread:

**Public Member Functions**

- **DatabaseCopyThread** (QWidget \*const parent)
- void **init** (const [DbEngineParameters](#) &fromDatabaseSettingsWidget, const [DbEngineParameters](#) &toDatabaseSettingsWidget)
- void **run** () override

**Public Attributes**

- [CoreDbCopyManager](#) **m\_copyManager**

**9.235 Digikam::DatabaseFields::DatabaseFieldsEnumIterator<FieldName > Class Template Reference**

You can iterate over each of the Enumerations defined above: `ImagesIterator`, `ImageMetadataIterator` etc.

**Public Member Functions**

- bool **atEnd** () const
- FieldName **operator\*** () const
- void **operator++** ()

**9.235.1 Detailed Description**

```
template<typename FieldName>
class Digikam::DatabaseFields::DatabaseFieldsEnumIterator< FieldName >
```

```
for (ImagesIterator it ; !it.atEnd() ; ++it) {...}
```

**9.236 Digikam::DatabaseFields::DatabaseFieldsEnumIteratorSetOnly< FieldName > Class Template Reference**

An iterator that iterates only over the flags which are set.

**Public Member Functions**

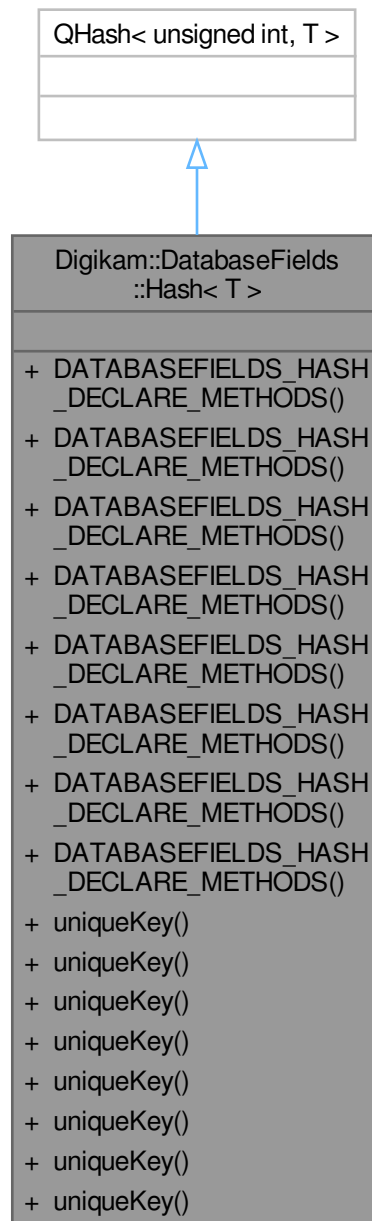
- **DatabaseFieldsEnumIteratorSetOnly** (const FieldName setValues)
- bool **atEnd** () const
- FieldName **operator\*** () const
- void **operator++** ()

**9.237 Digikam::DatabaseFields::FieldMetaInfo< FieldName > Class Template Reference****9.238 Digikam::DatabaseFields::Hash< T > Class Template Reference**

This class provides a hash on all DatabaseFields enums, allowing to use the enum values as independent keys.



Inheritance diagram for Digikam::DatabaseFields::Hash< T >:



### Public Member Functions

- **DATABASEFIELDS\_HASH\_DECLARE\_METHODS** (CustomEnum, uniqueKey)
- **DATABASEFIELDS\_HASH\_DECLARE\_METHODS** (ImageHistoryInfo, uniqueKey)
- **DATABASEFIELDS\_HASH\_DECLARE\_METHODS** (ImageMetadata, uniqueKey)
- **DATABASEFIELDS\_HASH\_DECLARE\_METHODS** (Images, uniqueKey)
- **DATABASEFIELDS\_HASH\_DECLARE\_METHODS** (ItemComments, uniqueKey)
- **DATABASEFIELDS\_HASH\_DECLARE\_METHODS** (ItemInformation, uniqueKey)
- **DATABASEFIELDS\_HASH\_DECLARE\_METHODS** (ItemPositions, uniqueKey)
- **DATABASEFIELDS\_HASH\_DECLARE\_METHODS** (VideoMetadata, uniqueKey)

### Static Public Member Functions

- static unsigned int **uniqueKey** (CustomEnum f)
- static unsigned int **uniqueKey** (ImageHistoryInfo f)
- static unsigned int **uniqueKey** (ImageMetadata f)
- static unsigned int **uniqueKey** (Images f)
- static unsigned int **uniqueKey** (ItemComments f)
- static unsigned int **uniqueKey** (ItemInformation f)
- static unsigned int **uniqueKey** (ItemPositions f)
- static unsigned int **uniqueKey** (VideoMetadata f)

### 9.238.1 Detailed Description

```
template<class T>
class Digikam::DatabaseFields::Hash< T >
```

You can use the class like a normal QHash with the value type defined by you, and as keys the members of the DatabaseFields enums. You can only use single enum members as keys, not or'ed numbers. You can use one custom enum, cast to DatabaseFields::CustomEnum, which can have at most 26 flag values (1 << 0 to 1 << 26). Pass this as the optional second template parameter.

### 9.239 Digikam::DatabaseFields::Set Class Reference

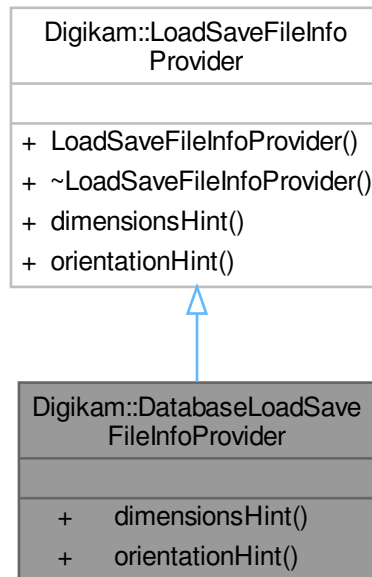
This class provides a set of all DatabaseFields enums, without resorting to a QSet.

#### Public Member Functions

- void **initialize** ()
- bool **operator&** (const [Set](#) &other)
- CustomEnum **operator&** (CustomEnum f) const
- [Set](#) & **operator<<** (const QDBusArgument &argument)
- CustomEnum & **operator=** (const CustomEnum &f)
- const [Set](#) & **operator>>** (QDBusArgument &argument) const
- CustomEnum **operator^** (CustomEnum f) const
- CustomEnum & **operator^=** (CustomEnum f)
- CustomEnum **operator|** (CustomEnum f) const
- CustomEnum & **operator|=** (CustomEnum f)
- [Set](#) & **setFields** (const [Set](#) &otherSet)

## 9.240 Digikam::DatabaseLoadSaveFileInfoProvider Class Reference

Inheritance diagram for Digikam::DatabaseLoadSaveFileInfoProvider:



### Public Member Functions

- `QSize` `dimensionsHint` (`const QString &path`) override  
*Gives a hint at the size of the image.*
- `int` `orientationHint` (`const QString &path`) override  
*Gives a hint at the orientation of the image.*

### 9.240.1 Member Function Documentation

#### 9.240.1.1 `dimensionsHint()`

```
QSize Digikam::DatabaseLoadSaveFileInfoProvider::dimensionsHint (
    const QString & path ) [override], [virtual]
```

This can be used to supersede the Exif information in the file.

Implements [Digikam::LoadSaveFileInfoProvider](#).

### 9.240.1.2 orientationHint()

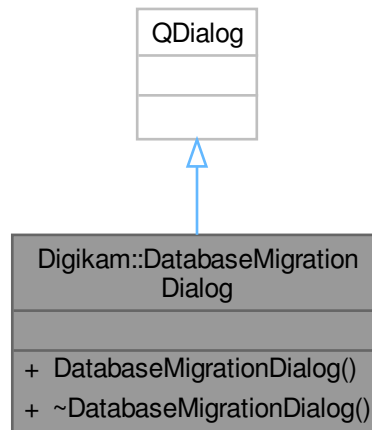
```
int Digikam::DatabaseLoadSaveFileInfoProvider::orientationHint (
    const QString & path ) [override], [virtual]
```

This can be used to supersede the Exif information in the file. Will not be used if DMetadata::ORIENTATION\_↔ UNSPECIFIED (default value)

Implements [Digikam::LoadSaveFileInfoProvider](#).

## 9.241 Digikam::DatabaseMigrationDialog Class Reference

Inheritance diagram for Digikam::DatabaseMigrationDialog:



### Public Member Functions

- **DatabaseMigrationDialog** (QWidget \*const parent)

## 9.242 Digikam::DatabaseOption Class Reference

Inheritance diagram for Digikam::DatabaseOption:



### Protected Member Functions

- `QString parseOperation (ParseSettings &settings, const QRegularExpressionMatch &match)` override  
*TODO: describe me.*

## Protected Member Functions inherited from [Digikam::Rule](#)

- bool [addToken](#) (const QString &id, const QString &description, const QString &actionName=QString())  
*add a token to the parser, every parser should at least assign one token object*
- void [setDescription](#) (const QString &desc)
- void [setIcon](#) (const QString &pixmap)
- void [setRegExp](#) (const QRegularExpression &regExp)
- void [setUseTokenMenu](#) (bool value)  
*If multiple tokens have been assigned to a rule, a menu will be created.*

## Additional Inherited Members

## Public Types inherited from [Digikam::Rule](#)

- enum [IconType](#) { [Action](#) = 0 , [Dialog](#) }

## Signals inherited from [Digikam::Rule](#)

- void [signalTokenTriggered](#) (const QString &)

## Public Member Functions inherited from [Digikam::Option](#)

- [Option](#) (const QString &name, const QString &description)
- [Option](#) (const QString &name, const QString &description, const QString &icon)

## Public Member Functions inherited from [Digikam::Rule](#)

- [Rule](#) (const QString &name)
- [Rule](#) (const QString &name, const QString &icon)
- QString [description](#) () const
- QPixmap [icon](#) (Rule::IconType type=Rule::Action) const
- bool [isValid](#) () const  
*Checks the validity of the parse object.*
- [ParseResults](#) [parse](#) ([ParseSettings](#) &settings)
- QRegularExpression & [regExp](#) () const  
*TODO: This is probably not needed anymore.*
- QPushButton \* [registerButton](#) (QWidget \*parent)  
*Register a button in the parent object.*
- QAction \* [registerMenu](#) (QMenu \*parent)  
*Register a menu action in the parent object.*
- virtual void [reset](#) ()  
*Resets the parser to its initial state.*
- TokenList & [tokens](#) () const
- bool [useTokenMenu](#) () const  
*Returns true if a token menu is used.*

## Static Public Member Functions inherited from [Digikam::Rule](#)

- static QString [escapeToken](#) (const QString &token)  
*Escape the token characters to make them work in regular expressions.*

## Protected Slots inherited from [Digikam::Rule](#)

- virtual void [slotTokenTriggered](#) (const QString &)

### 9.242.1 Member Function Documentation

#### 9.242.1.1 [parseOperation\(\)](#)

```
QString Digikam::DatabaseOption::parseOperation (
    ParseSettings & settings,
    const QRegularExpressionMatch & match ) [override], [protected], [virtual]
```

##### Parameters

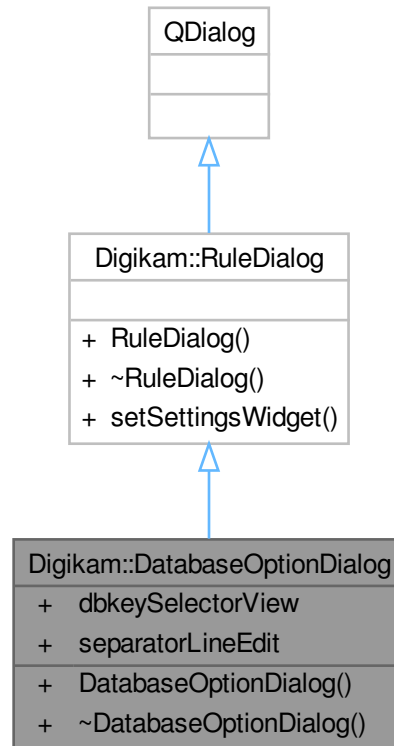
<i>settings</i>	contains settings
<i>match</i>	result of the regular expression match done in <a href="#">Option::parse()</a>

##### Returns

Implements [Digikam::Option](#).

## 9.243 Digikam::DatabaseOptionDialog Class Reference

Inheritance diagram for Digikam::DatabaseOptionDialog:



### Public Member Functions

- `DatabaseOptionDialog` (`Rule *const parent`)

### Public Member Functions inherited from `Digikam::RuleDialog`

- `RuleDialog` (`Rule *const parent`)
- void `setSettingsWidget` (`QWidget *const settingsWidget`)

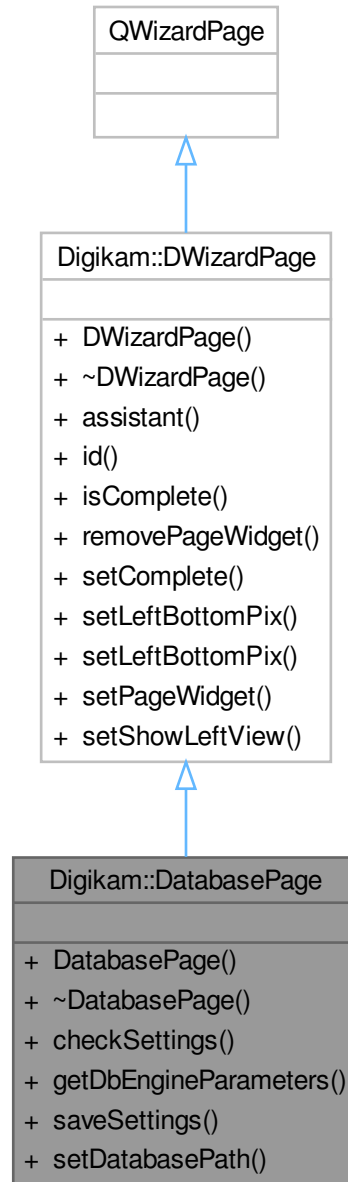
### Public Attributes

- `DbKeySelectorView * dbkeySelectorView = nullptr`
- `QLineEdit * separatorLineEdit = nullptr`



## 9.244 Digikam::DatabasePage Class Reference

Inheritance diagram for Digikam::DatabasePage:



### Public Member Functions

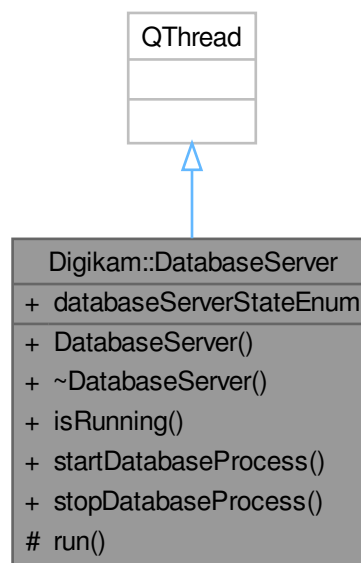
- `DatabasePage` (`QWizard *const dlg`)
- bool `checkSettings` ()
- `DbEngineParameters` `getDbEngineParameters` () const
- void `saveSettings` ()
- void `setDatabasePath` (const `QString &path`)

## Public Member Functions inherited from [Digikam::DWizardPage](#)

- **DWizardPage** (QWizard \*const dlg, const QString &title)
- QWizard \* **assistant** () const
- int **id** () const
- bool **isComplete** () const override
- void **removePageWidget** (QWidget \*const w)
- void **setComplete** (bool b)
- void **setLeftBottomPix** (const QIcon &icon)
- void **setLeftBottomPix** (const QPixmap &pix)
- void **setPageWidget** (QWidget \*const w)
- void **setShowLeftView** (bool v)

## 9.245 Digikam::DatabaseServer Class Reference

Inheritance diagram for Digikam::DatabaseServer:



### Public Types

- enum **DatabaseServerStateEnum** { **started** , **running** , **notRunning** , **stopped** }

### Signals

- void **done** ()

### Public Member Functions

- **DatabaseServer** (const [DbEngineParameters](#) &params, [DatabaseServerStarter](#) \*const parent=[DatabaseServerStarter::instance](#))
- bool **isRunning** () const  
*Returns true if the server process is running.*
- [DatabaseServerError](#) **startDatabaseProcess** ()  
*Starts the database management server.*
- void **stopDatabaseProcess** ()  
*Terminates the databaser server process.*

### Public Attributes

- DatabaseServerStateEnum **databaseServerStateEnum**

### Protected Member Functions

- void **run** () override

## 9.246 Digikam::DatabaseServerError Class Reference

### Public Types

- enum [DatabaseServerErrorEnum](#) { [NoErrors](#) = 0 , [NotSupported](#) , [StartError](#) }

### Public Member Functions

- **DatabaseServerError** (const [DatabaseServerError](#) &dbServerError)
- **DatabaseServerError** ([DatabaseServerErrorEnum](#) errorType=[NoErrors](#), const QString &errorText=QString())
- QString **getErrorText** () const
- [DatabaseServerErrorEnum](#) **getErrorType** () const
- void **setErrorText** (const QString &errorText)
- void **setErrorType** ([DatabaseServerErrorEnum](#) errorType)

### 9.246.1 Member Enumeration Documentation

#### 9.246.1.1 DatabaseServerErrorEnum

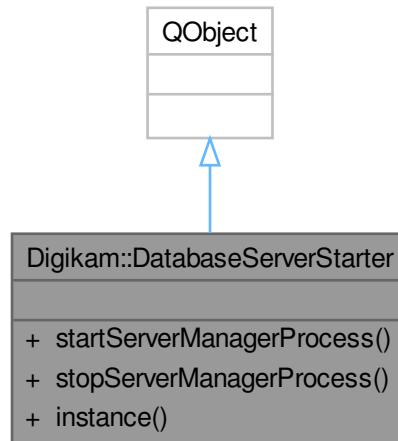
```
enum Digikam::DatabaseServerError::DatabaseServerErrorEnum
```

#### Enumerator

NoErrors	No errors occurred while starting the database server.
NotSupported	The requested database type is not supported.
StartError	A error has occurred while starting the database server executable.

## 9.247 Digikam::DatabaseServerStarter Class Reference

Inheritance diagram for Digikam::DatabaseServerStarter:



### Public Member Functions

- [DatabaseServerError](#) **startServerManagerProcess** (const [DbEngineParameters](#) &parameters) const
- void **stopServerManagerProcess** ()

### Static Public Member Functions

- static [DatabaseServerStarter](#) \* **instance** ()  
*Global instance of internal server starter.*

### Friends

- class **DatabaseServerStarterCreator**

## 9.247.1 Member Function Documentation

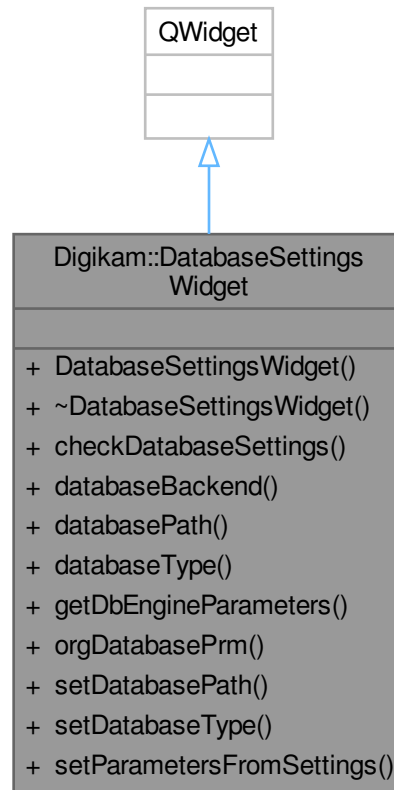
### 9.247.1.1 instance()

```
DatabaseServerStarter * Digikam::DatabaseServerStarter::instance ( ) [static]
```

All accessor methods are thread-safe.

## 9.248 Digikam::DatabaseSettingsWidget Class Reference

Inheritance diagram for Digikam::DatabaseSettingsWidget:



### Public Types

- enum **DatabaseType** { **SQLite** = 0 , **MysqlInternal** = 1 , **MysqlServer** = 2 }

### Public Member Functions

- **DatabaseSettingsWidget** (QWidget \*const parent=nullptr)
- bool [checkDatabaseSettings](#) ()
- *For SQLite or MysqlInternal, check properties of local path to store database files.*
- QString **databaseBackend** () const
- QString **databasePath** () const
- int **databaseType** () const
- [DbEngineParameters](#) **getDbEngineParameters** () const
- [DbEngineParameters](#) **orgDatabasePrm** () const
- void **setDatabasePath** (const QString &path)
- void **setDatabaseType** (int type)
- void **setParametersFromSettings** (const [ApplicationSettings](#) \*const settings, const bool &migration=false)

## 9.248.1 Member Function Documentation

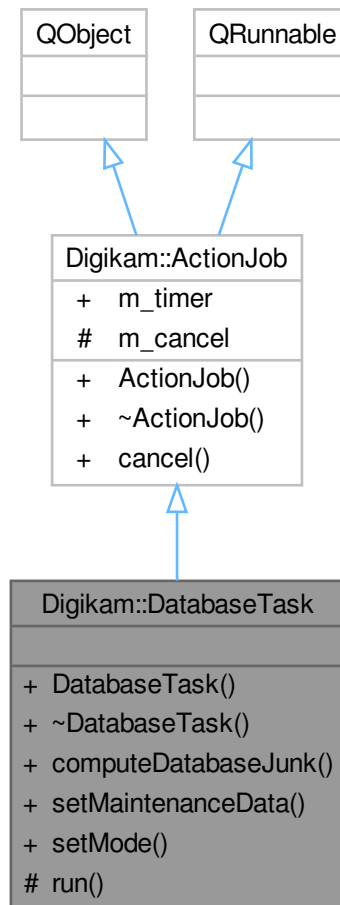
### 9.248.1.1 checkDatabaseSettings()

```
bool Digikam::DatabaseSettingsWidget::checkDatabaseSettings ( )
```

For MySQLServer, check the network connection and database names.

## 9.249 Digikam::DatabaseTask Class Reference

Inheritance diagram for Digikam::DatabaseTask:



### Public Types

- enum **Mode** {
  - Unknown** , **ComputeDatabaseJunk** , **CleanCoreDb** , **CleanThumbsDb** ,
  - CleanRecognitionDb** , **CleanSimilarityDb** , **ShrinkDatabases** }

## Signals

- void **signalAddItemsToProcess** (int count)  
*Signal to emit the count of additional items to process.*
- void **signalData** (const QList< qlonglong > &staleImageIds, const QList< int > &staleThumbIds, const QList< Identity > &staleIdentities, const QList< qlonglong > &staleSimilarityImageIds)
- void **signalFinished** ()
- void **signalFinished** (bool done, bool errorFree)

## Signals inherited from Digikam::ActionJob

- void **signalDone** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job is done.*
- void **signalProgress** (int)  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job progress.*
- void **signalStarted** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager that job is started.*

## Public Member Functions

- void **computeDatabaseJunk** (bool thumbsDb=false, bool facesDb=false, bool similarityDb=false)
- void **setMaintenanceData** ([MaintenanceData](#) \*const data=nullptr)
- void **setMode** (Mode mode)

## Public Member Functions inherited from Digikam::ActionJob

- **ActionJob** (QObject \*const parent=nullptr)  
*Constructor which delegate deletion of QRunnable instance to [ActionThreadBase](#), not QThreadPool.*
- **~ActionJob** () override  
*Re-implement destructor in you implementation.*

## Protected Member Functions

- void **run** () override

## Additional Inherited Members

## Public Slots inherited from Digikam::ActionJob

- void **cancel** ()  
*Call this method to cancel job.*

## Public Attributes inherited from Digikam::ActionJob

- QElapsedTimer **m\_timer**  
*Timer to determine the running time of the job.*

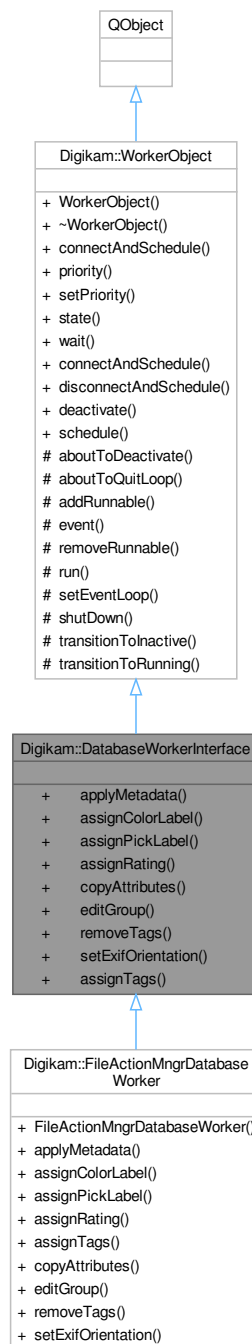
## Protected Attributes inherited from [Digikam::ActionJob](#)

- bool `m_cancel` = false

*You can use this boolean in your implementation to know if job must be canceled.*

## 9.250 Digikam::DatabaseWorkerInterface Class Reference

Inheritance diagram for Digikam::DatabaseWorkerInterface:





## Public Slots

- virtual void **assignTags** (const [FileActionItemInfoList](#) &, const [QList< int >](#) &)

## Public Slots inherited from [Digikam::WorkerObject](#)

- void **deactivate** ([DeactivatingMode](#) mode=[FlushSignals](#))  
*Quits execution of this worker object.*
- void **schedule** ()  
*Starts execution of this worker object: The object is moved to a thread and an event loop started, so that queued signals will be received.*

## Signals

- void **writeMetadata** ([FileActionItemInfoList](#) infos, int flag)
- void **writeMetadataToFiles** ([FileActionItemInfoList](#) infos)
- void **writeOrientationToFiles** ([FileActionItemInfoList](#) infos, int orientation)

## Signals inherited from [Digikam::WorkerObject](#)

- void **finished** ()
- void **started** ()

## Public Member Functions

- virtual void **applyMetadata** (const [FileActionItemInfoList](#) &, [DisjointMetadata](#) \*)
- virtual void **assignColorLabel** (const [FileActionItemInfoList](#) &, int)
- virtual void **assignPickLabel** (const [FileActionItemInfoList](#) &, int)
- virtual void **assignRating** (const [FileActionItemInfoList](#) &, int)
- virtual void **copyAttributes** (const [FileActionItemInfoList](#) &, const [QStringList](#) &)
- virtual void **editGroup** (int, const [ItemInfo](#) &, const [FileActionItemInfoList](#) &)
- virtual void **removeTags** (const [FileActionItemInfoList](#) &, const [QList< int >](#) &)
- virtual void **setExifOrientation** (const [FileActionItemInfoList](#) &, int)

## Public Member Functions inherited from [Digikam::WorkerObject](#)

- [WorkerObject](#) ()  
*Deriving from a worker object allows you to execute your slots in a thread.*
- bool **connectAndSchedule** (const [QObject](#) \*sender, const char \*signal, const char \*method, [Qt::](#)↔[ConnectionType](#) type=[Qt::AutoConnection](#)) const  
*You must normally call [schedule\(\)](#) to ensure that the object is active when you send a signal with work data.*
- [QThread::Priority](#) **priority** () const
- void **setPriority** ([QThread::Priority](#) priority)  
*Sets the priority for this dynamic thread.*
- State **state** () const
- void **wait** ()

## Additional Inherited Members

### Public Types inherited from [Digikam::WorkerObject](#)

- enum [DeactivatingMode](#) { [FlushSignals](#) , [KeepSignals](#) , [PhaseOut](#) }
- enum [State](#) { [Inactive](#) , [Scheduled](#) , [Running](#) , [Deactivating](#) }

### Static Public Member Functions inherited from [Digikam::WorkerObject](#)

- static bool **connectAndSchedule** (const QObject \*sender, const char \*signal, const [WorkerObject](#) \*receiver, const char \*method, Qt::ConnectionType type=Qt::AutoConnection)
- static bool **disconnectAndSchedule** (const QObject \*sender, const char \*signal, const [WorkerObject](#) \*receiver, const char \*method)

### Protected Member Functions inherited from [Digikam::WorkerObject](#)

- virtual void [aboutToDeactivate](#) ()  
*Called from [deactivate\(\)](#), typically from a different thread than the worker thread, possibly the UI thread.*
- virtual void [aboutToQuitLoop](#) ()  
*Called from within thread's event loop to quit processing.*
- void **addRunnable** (WorkerObjectRunnable \*loop)
- bool **event** (QEvent \*e) override
- void **removeRunnable** (WorkerObjectRunnable \*loop)
- void **run** ()
- void **setEventLoop** (QEventLoop \*loop)
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void **transitionToInactive** ()
- bool **transitionToRunning** ()

## 9.251 Digikam::DatabaseWriter Class Reference

Inheritance diagram for Digikam::DatabaseWriter:



### Public Slots

- void **process** (const `FacePipelineExtendedPackage::Ptr` &package)

## Public Slots inherited from [Digikam::WorkerObject](#)

- void **deactivate** ([DeactivatingMode](#) mode=[FlushSignals](#))  
*Quits execution of this worker object.*
- void **schedule** ()  
*Starts execution of this worker object: The object is moved to a thread and an event loop started, so that queued signals will be received.*

## Signals

- void **processed** (const [FacePipelineExtendedPackage::Ptr](#) &package)

## Signals inherited from [Digikam::WorkerObject](#)

- void **finished** ()
- void **started** ()

## Public Member Functions

- **DatabaseWriter** ([FacePipeline::WriteMode](#) wmode, [FacePipeline::Private](#) \*const dd)

## Public Member Functions inherited from [Digikam::WorkerObject](#)

- [WorkerObject](#) ()  
*Deriving from a worker object allows you to execute your slots in a thread.*
- bool **connectAndSchedule** (const [QObject](#) \*sender, const char \*signal, const char \*method, [Qt::](#)↔[ConnectionType](#) type=[Qt::AutoConnection](#)) const  
*You must normally call [schedule\(\)](#) to ensure that the object is active when you send a signal with work data.*
- [QThread::Priority](#) **priority** () const
- void **setPriority** ([QThread::Priority](#) priority)  
*Sets the priority for this dynamic thread.*
- State **state** () const
- void **wait** ()

## Protected Attributes

- [FacePipeline::Private](#) \*const **d** = nullptr
- [FacePipeline::WriteMode](#) **mode** = [FacePipeline::NormalWrite](#)
- [ThumbnailLoadThread](#) \* **thumbnailLoadThread** = nullptr

## Additional Inherited Members

## Public Types inherited from [Digikam::WorkerObject](#)

- enum [DeactivatingMode](#) { [FlushSignals](#) , [KeepSignals](#) , [PhaseOut](#) }
- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Static Public Member Functions inherited from [Digikam::WorkerObject](#)

- static bool **connectAndSchedule** (const QObject \*sender, const char \*signal, const [WorkerObject](#) \*receiver, const char \*method, Qt::ConnectionType type=Qt::AutoConnection)
- static bool **disconnectAndSchedule** (const QObject \*sender, const char \*signal, const [WorkerObject](#) \*receiver, const char \*method)

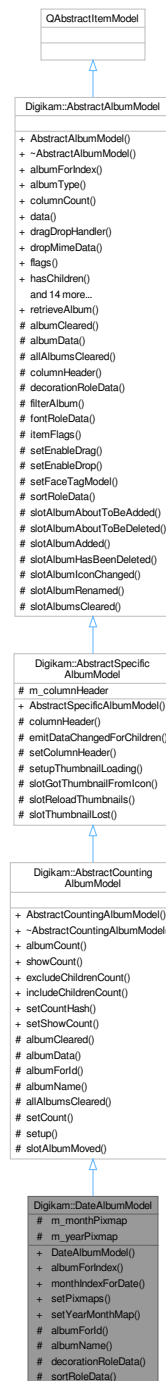
### Protected Member Functions inherited from [Digikam::WorkerObject](#)

- virtual void **aboutToDeactivate** ()  
*Called from [deactivate\(\)](#), typically from a different thread than the worker thread, possibly the UI thread.*
- virtual void **aboutToQuitLoop** ()  
*Called from within thread's event loop to quit processing.*
- void **addRunnable** (WorkerObjectRunnable \*loop)
- bool **event** (QEvent \*e) override
- void **removeRunnable** (WorkerObjectRunnable \*loop)
- void **run** ()
- void **setEventLoop** (QEventLoop \*loop)
- void **shutDown** ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void **transitionToInactive** ()
- bool **transitionToRunning** ()

## 9.252 Digikam::DateAlbumModel Class Reference

A model for date based albums.

Inheritance diagram for Digikam::DateAlbumModel:



## Public Slots

- void `setYearMonthMap` (const QMap< YearMonth, int > &yearMonthMap)

## Public Slots inherited from [Digikam::AbstractCountingAlbumModel](#)

- void `excludeChildrenCount` (const QModelIndex &index)

- *Displays only the count of the album, without adding child albums' counts.*
- void **includeChildrenCount** (const QModelIndex &index)
  - *Displays sum of the count of the album and child albums' counts.*
- void **setCountHash** (const QHash< int, int > &idCountHash)
  - *Enable displaying the count.*
- void **setShowCount** (bool show)
  - *Call to enable or disable showing the count. Default is false.*

### Public Member Functions

- **DateAlbumModel** (QObject \*const parent=nullptr)
  - *Constructor.*
- **DAlbum \* albumForIndex** (const QModelIndex &index) const
- QModelIndex **monthIndexForDate** (const QDate &date) const
  - *Finds an album index based on a date.*
- void **setPixmaps** (const QPixmap &forYearAlbums, const QPixmap &forMonthAlbums)
  - *Set pixmaps for the DecorationRole.*

### Public Member Functions inherited from [Digikam::AbstractCountingAlbumModel](#)

- **AbstractCountingAlbumModel** ([Album::Type](#) albumType, [Album](#) \*const rootAlbum, [RootAlbumBehavior](#) rootBehavior=[IncludeRootAlbum](#), QObject \*const parent=nullptr)
  - *Supports displaying a count alongside the album name in DisplayRole.*
- virtual int **albumCount** ([Album](#) \*album) const
  - *Returns the number of included items for this album.*
- bool **showCount** () const

### Public Member Functions inherited from [Digikam::AbstractSpecificAlbumModel](#)

- **AbstractSpecificAlbumModel** ([Album::Type](#) albumType, [Album](#) \*const rootAlbum, [RootAlbumBehavior](#) rootBehavior=[IncludeRootAlbum](#), QObject \*const parent=nullptr)
  - *Abstract base class, do not instantiate.*

### Public Member Functions inherited from [Digikam::AbstractAlbumModel](#)

- **AbstractAlbumModel** ([Album::Type](#) albumType, [Album](#) \*const rootAlbum, [RootAlbumBehavior](#) rootBehavior=[IncludeRootAlbum](#), QObject \*const parent=nullptr)
  - *Create an [AbstractAlbumModel](#) object for albums with the given type.*
- [Album](#) \* **albumForIndex** (const QModelIndex &index) const
  - *Returns the album object associated with the given model index.*
- [Album::Type](#) **albumType** () const
  - *Returns the [Album::Type](#) of the contained albums.*
- int **columnCount** (const QModelIndex &parent=QModelIndex()) const override
- QVariant **data** (const QModelIndex &index, int role=Qt::DisplayRole) const override
- [AlbumModelDragDropHandler](#) \* **dragDropHandler** () const
  - *Returns the drag drop handler, or 0 if none is installed.*
- bool **dropMimeData** (const QMimeData \*data, Qt::DropAction action, int row, int column, const QModelIndex &parent) override
- Qt::ItemFlags **flags** (const QModelIndex &index) const override

- bool **hasChildren** (const QModelIndex &parent=QModelIndex()) const override
- QVariant **headerData** (int section, Qt::Orientation orientation, int role=Qt::DisplayRole) const override
- QModelIndex **index** (int row, int column, const QModelIndex &parent=QModelIndex()) const override
- QModelIndex **indexForAlbum** (Album \*album) const  
*Return the QModelIndex for the given album, or an invalid index if the album is not contained in this model.*
- bool **isFaceTagModel** () const  
*Returns true if the album model a face tag model.*
- QMimeData \* **mimeData** (const QModelIndexList &indexes) const override
- QStringList **mimeTypes** () const override
- QModelIndex **parent** (const QModelIndex &index) const override
- Album \* **rootAlbum** () const
- RootAlbumBehavior **rootAlbumBehavior** () const  
*Returns the root album behavior set for this model.*
- QModelIndex **rootAlbumIndex** () const  
*Return the index corresponding to the root album.*
- int **rowCount** (const QModelIndex &parent=QModelIndex()) const override
- void **setDragDropHandler** (AlbumModelDragDropHandler \*handler)  
*Set a drag drop handler.*
- void **setDropIndex** (const QModelIndex &index)  
*Set current index from QDragMoveEvent.*
- Qt::DropActions **supportedDropActions** () const override

### Protected Member Functions

- Album \* **albumForId** (int id) const override  
*need to implement in subclass*
- QString **albumName** (Album \*a) const override  
*Can reimplement in subclass.*
- QVariant **decorationRoleData** (Album \*a) const override  
*For subclassing convenience: A part of the implementation of data()*
- QVariant **sortRoleData** (Album \*a) const override  
*For subclassing convenience: A part of the implementation of data()*

### Protected Member Functions inherited from [Digikam::AbstractCountingAlbumModel](#)

- void **albumCleared** (Album \*album) override  
*Notification when an entry is removed.*
- QVariant **albumData** (Album \*a, int role) const override  
*Reimplemented from parent classes.*
- void **allAlbumsCleared** () override  
*Notification when all entries are removed.*
- void **setCount** (Album \*album, int count)  
*If you do not use setCountHash, excludeChildrenCount and includeChildrenCount, you can set a count here.*
- void **setup** ()  
*Call this method in children class constructors to init signal/slots connections.*



## Protected Member Functions inherited from [Digikam::AbstractSpecificAlbumModel](#)

- QString [columnHeader](#) () const override  
*For subclassing convenience: A part of the implementation of headerData()*
- void **emitDataChangedForChildren** (Album \*album)
- void **setColumnHeader** (const QString &header)
- void **setupThumbnailLoading** ()  
*You need to call this from your constructor if you intend to load the thumbnail facilities of this class.*

## Protected Member Functions inherited from [Digikam::AbstractAlbumModel](#)

- virtual bool [filterAlbum](#) (Album \*album) const  
*Returns true for those and only those albums that shall be contained in this model.*
- virtual QVariant [fontRoleData](#) (Album \*a) const  
*For subclassing convenience: A part of the implementation of data()*
- virtual Qt::ItemFlags **itemFlags** (Album \*album) const  
*For subclassing convenience: A part of the implementation of itemFlags()*
- void [setEnableDrag](#) (bool enable)  
*Switch on drag and drop globally for all items.*
- void **setEnableDrop** (bool enable)
- void **setFaceTagModel** (bool enable)

## Protected Attributes

- QPixmap **m\_monthPixmap**
- QPixmap **m\_yearPixmap**

## Protected Attributes inherited from [Digikam::AbstractSpecificAlbumModel](#)

- QString **m\_columnHeader**

## Additional Inherited Members

## Public Types inherited from [Digikam::AbstractAlbumModel](#)

- enum [AlbumDataRole](#) {  
[AlbumTitleRole](#) = Qt::UserRole , [AlbumTypeRole](#) = Qt::UserRole + 1 , [AlbumPointerRole](#) = Qt::UserRole + 2  
, [AlbumIdRole](#) = Qt::UserRole + 3 ,  
[AlbumGlobalIdRole](#) = Qt::UserRole + 4 , [AlbumSortRole](#) = Qt::UserRole + 5 }
- enum [RootAlbumBehavior](#) { [IncludeRootAlbum](#) , [IgnoreRootAlbum](#) }  
*AbstractAlbumModel is the abstract base class for all models that present Album objects as managed by AlbumManager.*

## Signals inherited from [Digikam::AbstractCountingAlbumModel](#)

- void **signalUpdateAlbumCount** (Album \*album)

## Signals inherited from [Digikam::AbstractAlbumModel](#)

- void [rootAlbumAvailable](#) ()

*This is initialized once after creation, if the root album becomes available, if it was not already available at time of construction.*

## Static Public Member Functions inherited from [Digikam::AbstractAlbumModel](#)

- static [Album](#) \* [retrieveAlbum](#) (const [QModelIndex](#) &index)

*Returns the album represented by the index.*

## Protected Slots inherited from [Digikam::AbstractCountingAlbumModel](#)

- void [slotAlbumMoved](#) ([Album](#) \*album)

## Protected Slots inherited from [Digikam::AbstractSpecificAlbumModel](#)

- void [slotGotThumbnailFromIcon](#) ([Album](#) \*album, const [QPixmap](#) &thumbnail)
- void [slotReloadThumbnails](#) ()
- void [slotThumbnailLost](#) ([Album](#) \*album)

## Protected Slots inherited from [Digikam::AbstractAlbumModel](#)

- void [slotAlbumAboutToBeAdded](#) ([Album](#) \*album, [Album](#) \*parent, [Album](#) \*prev)
- void [slotAlbumAboutToBeDeleted](#) ([Album](#) \*album)
- void [slotAlbumAdded](#) ([Album](#) \*)
- void [slotAlbumHasBeenDeleted](#) ([Album](#) \*album)
- void [slotAlbumIconChanged](#) ([Album](#) \*album)
- void [slotAlbumRenamed](#) ([Album](#) \*album)
- void [slotAlbumsCleared](#) ()

## 9.252.1 Constructor & Destructor Documentation

### 9.252.1.1 [DateAlbumModel](#)()

```
Digikam::DateAlbumModel::DateAlbumModel (
    QObject *const parent = nullptr ) [explicit]
```

#### Parameters

<i>parent</i>	the parent for Qt's parent child mechanism
---------------	--

## 9.252.2 Member Function Documentation

### 9.252.2.1 albumForId()

```
Album * Digikam::DateAlbumModel::albumForId (
    int id ) const [override], [protected], [virtual]
```

Implements [Digikam::AbstractCountingAlbumModel](#).

### 9.252.2.2 albumName()

```
QString Digikam::DateAlbumModel::albumName (
    Album * a ) const [override], [protected], [virtual]
```

Reimplemented from [Digikam::AbstractCountingAlbumModel](#).

### 9.252.2.3 decorationRoleData()

```
QVariant Digikam::DateAlbumModel::decorationRoleData (
    Album * a ) const [override], [protected], [virtual]
```

Reimplemented from [Digikam::AbstractAlbumModel](#).

### 9.252.2.4 monthIndexForDate()

```
QModelIndex Digikam::DateAlbumModel::monthIndexForDate (
    const QDate & date ) const
```

The given date is therefore normalized to year-month-form. The day is ignored. This means the returned index always points to a month [DAAlbum](#).

#### Parameters

<i>date</i>	the date to search for (year and month)
-------------	---

#### Returns

model index corresponding to the album with the given date or an empty index if not found

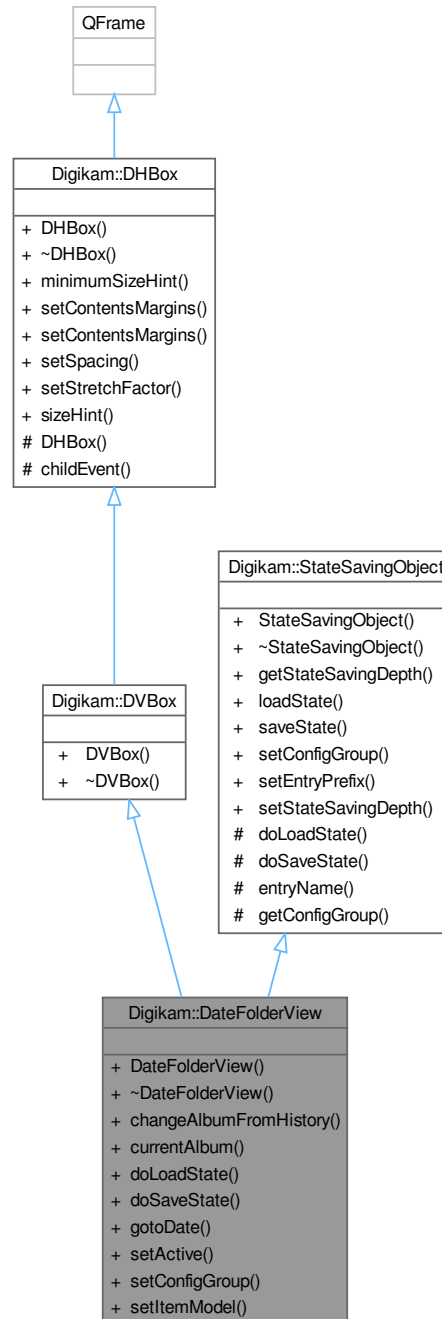
### 9.252.2.5 sortRoleData()

```
QVariant Digikam::DateAlbumModel::sortRoleData (
    Album * a ) const [override], [protected], [virtual]
```

Reimplemented from [Digikam::AbstractAlbumModel](#).

## 9.253 Digikam::DateFolderView Class Reference

Inheritance diagram for Digikam::DateFolderView:



### Public Member Functions

- **DateFolderView** (QWidget \*const parent, [DateAlbumModel](#) \*const dateAlbumModel)
- void **changeAlbumFromHistory** ([DAlbum](#) \*const album)

- [AlbumPointer](#)< [DAAlbum](#) > **currentAlbum** () const
- void [doLoadState](#) () override  
*Implement this hook method for state loading.*
- void [doSaveState](#) () override  
*Implement this hook method for state saving.*
- void **gotoDate** (const [QDate](#) &dt)
- void **setActive** (const bool val)
- void [setConfigGroup](#) (const [KConfigGroup](#) &group) override  
*Sets a dedicated config group that will be used to store and reload the state from.*
- void **setItemModel** ([ItemFilterModel](#) \*const model)

### Public Member Functions inherited from [Digikam::DVBox](#)

- [DVBox](#) ([QWidget](#) \*const parent=nullptr)

### Public Member Functions inherited from [Digikam::DHBox](#)

- [DHBox](#) ([QWidget](#) \*const parent=nullptr)
- [QSize](#) **minimumSizeHint** () const override
- void **setContentsMargins** (const [QMargins](#) &margins)
- void **setContentsMargins** (int left, int top, int right, int bottom)
- void **setSpacing** (int space)
- void **setStretchFactor** ([QWidget](#) \*const widget, int stretch)
- [QSize](#) **sizeHint** () const override

### Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) ([QObject](#) \*const host)  
*Constructor.*
- virtual **~StateSavingObject** ()  
*Destructor.*
- [StateSavingDepth](#) [getStateSavingDepth](#) () const  
*Returns the depth used for state saving or loading.*
- void **loadState** ()  
*Invokes loading the class' state.*
- void **saveState** ()  
*Invokes saving the class' state.*
- virtual void [setEntryPrefix](#) (const [QString](#) &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void [setStateSavingDepth](#) (const [StateSavingDepth](#) depth)  
*Sets the depth used for state saving or loading.*

### Additional Inherited Members

### Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }
- This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

## Protected Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (bool vertical, QWidget \*const parent)
- void **childEvent** (QChildEvent \*e) override

## Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- QString **entryName** (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup **getConfigGroup** () const  
*Returns the config group that must be used for state saving and loading.*

### 9.253.1 Member Function Documentation

#### 9.253.1.1 doLoadState()

```
void Digikam::DateFolderView::doLoadState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

#### 9.253.1.2 doSaveState()

```
void Digikam::DateFolderView::doSaveState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

#### 9.253.1.3 setConfigGroup()

```
void Digikam::DateFolderView::setConfigGroup (
    const KConfigGroup & group ) [override], [virtual]
```

If this method is not called, a group based on the object name is used.

You can re-implement this method to pass the group set here to child objects. Don't forget to call this method in your implementation.

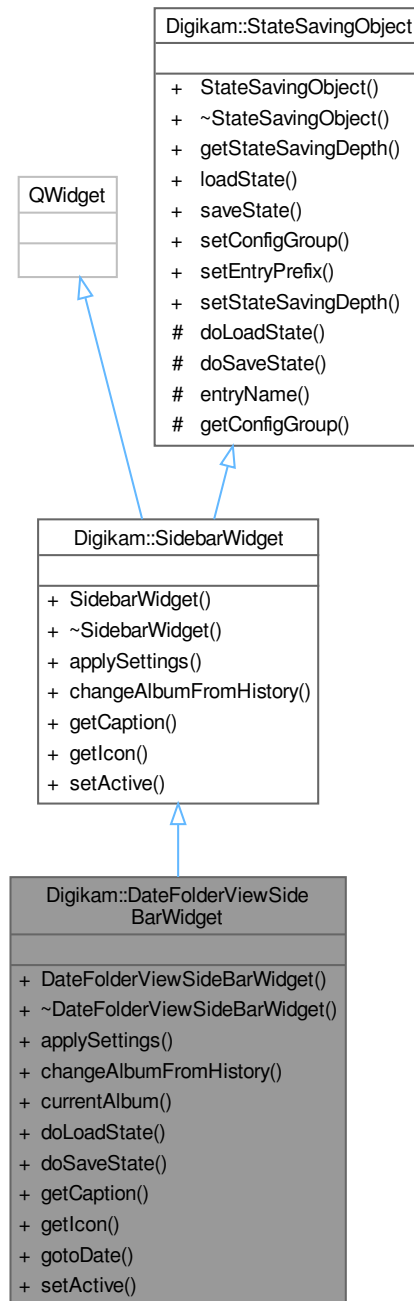
#### Parameters

<i>group</i>	config group to use for state saving and restoring
--------------	--

Reimplemented from [Digikam::StateSavingObject](#).

## 9.254 Digikam::DateFolderViewSideBarWidget Class Reference

Inheritance diagram for Digikam::DateFolderViewSideBarWidget:



### Public Member Functions

- **DateFolderViewSideBarWidget** (`QWidget *const parent`, `DateAlbumModel *const model`, `ItemAlbumFilterModel *const imageFilterModel`)

- void [applySettings](#) () override  
*This method is invoked when the application settings should be (re-) applied to this widget.*
- void [changeAlbumFromHistory](#) (const QList< Album \* > &album) override  
*This is called on this widget when the history requires to move back to the specified album.*
- [AlbumPointer](#)< DAAlbum > **currentAlbum** () const
- void [doLoadState](#) () override  
*Implement this hook method for state loading.*
- void [doSaveState](#) () override  
*Implement this hook method for state saving.*
- const QString [getCaption](#) () override  
*Must be implemented to return the title of this sidebar's tab.*
- const QIcon [getIcon](#) () override  
*Must be implemented and return the icon that shall be visible for this sidebar widget.*
- void **gotoDate** (const QDate &date)
- void [setActive](#) (bool active) override  
*This method is called if the visible sidebar widget is changed.*

### Public Member Functions inherited from [Digikam::SidebarWidget](#)

- [SidebarWidget](#) (QWidget \*const parent)  
*Constructor.*
- **~SidebarWidget** () override=default  
*Destructor.*

### Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (QObject \*const host)  
*Constructor.*
- virtual **~StateSavingObject** ()  
*Destructor.*
- [StateSavingDepth](#) [getStateSavingDepth](#) () const  
*Returns the depth used for state saving or loading.*
- void **loadState** ()  
*Invokes loading the class' state.*
- void **saveState** ()  
*Invokes saving the class' state.*
- virtual void [setConfigGroup](#) (const KConfigGroup &group)  
*Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void [setEntryPrefix](#) (const QString &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void [setStateSavingDepth](#) (const [StateSavingDepth](#) depth)  
*Sets the depth used for state saving or loading.*

### Additional Inherited Members

### Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }  
*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*



## Signals inherited from [Digikam::SidebarWidget](#)

- void **requestActiveTab** ([SidebarWidget](#) \*)  
*This signal can be emitted if this sidebar widget wants to be the one that is active.*
- void **signalNotificationError** (const QString &message, int type)  
*To dispatch error message to temporized pop-up notification widget hosted with icon-view.*

## Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- QString **entryName** (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup **getConfigGroup** () const  
*Returns the config group that must be used for state saving and loading.*

### 9.254.1 Member Function Documentation

#### 9.254.1.1 applySettings()

```
void Digikam::DateFolderViewSideBarWidget::applySettings ( ) [override], [virtual]
```

Implements [Digikam::SidebarWidget](#).

#### 9.254.1.2 changeAlbumFromHistory()

```
void Digikam::DateFolderViewSideBarWidget::changeAlbumFromHistory (
    const QList< Album * > & album ) [override], [virtual]
```

Implements [Digikam::SidebarWidget](#).

#### 9.254.1.3 doLoadState()

```
void Digikam::DateFolderViewSideBarWidget::doLoadState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

#### 9.254.1.4 doSaveState()

```
void Digikam::DateFolderViewSideBarWidget::doSaveState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

### 9.254.1.5 getCaption()

```
const QString Digikam::DateFolderViewSideBarWidget::getCaption ( ) [override], [virtual]
```

#### Returns

localized title string

Implements [Digikam::SidebarWidget](#).

### 9.254.1.6 getIcon()

```
const QIcon Digikam::DateFolderViewSideBarWidget::getIcon ( ) [override], [virtual]
```

#### Returns

pixmap icon

Implements [Digikam::SidebarWidget](#).

### 9.254.1.7 setActive()

```
void Digikam::DateFolderViewSideBarWidget::setActive (
    bool active ) [override], [virtual]
```

#### Parameters

<i>active</i>	if true, this widget is the new active widget, if false another widget is active
---------------	--

Implements [Digikam::SidebarWidget](#).

## 9.255 Digikam::DateFormat Class Reference

### Public Types

- typedef QPair< QString, QVariant > **DateFormatDescriptor**
- typedef QList< DateFormatDescriptor > **DateFormatMap**
- enum **Type** {
  - Standard** = 0 , **ISO** , **FullText** , **UnixTimeStamp** ,
  - Custom** }

### Public Member Functions

- QVariant **format** (const QString &identifier)
- QVariant **format** (Type type)
- QString **identifier** (Type type)
- DateFormatMap & **map** ()
- Type **type** (const QString &identifier)

## 9.256 Digikam::DateOption Class Reference

Inheritance diagram for Digikam::DateOption:



### Protected Member Functions

- `QString parseOperation (ParseSettings &settings, const QRegularExpressionMatch &match)` override  
*TODO: describe me.*

## Protected Member Functions inherited from [Digikam::Rule](#)

- bool [addToken](#) (const QString &id, const QString &description, const QString &actionName=QString())  
*add a token to the parser, every parser should at least assign one token object*
- void [setDescription](#) (const QString &desc)
- void [setIcon](#) (const QString &pixmap)
- void [setRegExp](#) (const QRegularExpression &regExp)
- void [setUseTokenMenu](#) (bool value)  
*If multiple tokens have been assigned to a rule, a menu will be created.*

## Additional Inherited Members

## Public Types inherited from [Digikam::Rule](#)

- enum [IconType](#) { [Action](#) = 0 , [Dialog](#) }

## Signals inherited from [Digikam::Rule](#)

- void [signalTokenTriggered](#) (const QString &)

## Public Member Functions inherited from [Digikam::Option](#)

- [Option](#) (const QString &name, const QString &description)
- [Option](#) (const QString &name, const QString &description, const QString &icon)

## Public Member Functions inherited from [Digikam::Rule](#)

- [Rule](#) (const QString &name)
- [Rule](#) (const QString &name, const QString &icon)
- QString [description](#) () const
- QPixmap [icon](#) (Rule::IconType type=Rule::Action) const
- bool [isValid](#) () const  
*Checks the validity of the parse object.*
- [ParseResults](#) [parse](#) ([ParseSettings](#) &settings)
- QRegularExpression & [regExp](#) () const  
*TODO: This is probably not needed anymore.*
- QPushButton \* [registerButton](#) (QWidget \*parent)  
*Register a button in the parent object.*
- QAction \* [registerMenu](#) (QMenu \*parent)  
*Register a menu action in the parent object.*
- virtual void [reset](#) ()  
*Resets the parser to its initial state.*
- TokenList & [tokens](#) () const
- bool [useTokenMenu](#) () const  
*Returns true if a token menu is used.*

## Static Public Member Functions inherited from [Digikam::Rule](#)

- static QString [escapeToken](#) (const QString &token)  
*Escape the token characters to make them work in regular expressions.*

## Protected Slots inherited from [Digikam::Rule](#)

- virtual void [slotTokenTriggered](#) (const QString &)

## 9.256.1 Member Function Documentation

### 9.256.1.1 [parseOperation\(\)](#)

```
QString Digikam::DateOption::parseOperation (
    ParseSettings & settings,
    const QRegularExpressionMatch & match ) [override], [protected], [virtual]
```

#### Parameters

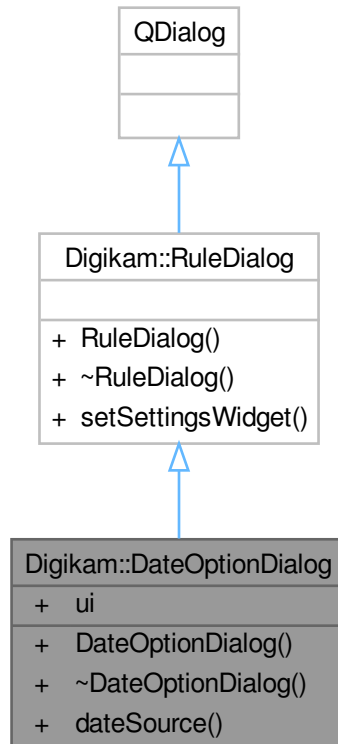
<i>settings</i>	contains settings
<i>match</i>	result of the regular expression match done in <a href="#">Option::parse()</a>

#### Returns

Implements [Digikam::Option](#).

## 9.257 Digikam::DateOptionDialog Class Reference

Inheritance diagram for Digikam::DateOptionDialog:



### Public Types

- enum `DateSource` { `FromImage = 0` , `CurrentDateTime` , `FixedDateTime` }

### Public Member Functions

- `DateOptionDialog` (`Rule` \*parent)
- `DateSource dateSource` () const

### Public Member Functions inherited from `Digikam::RuleDialog`

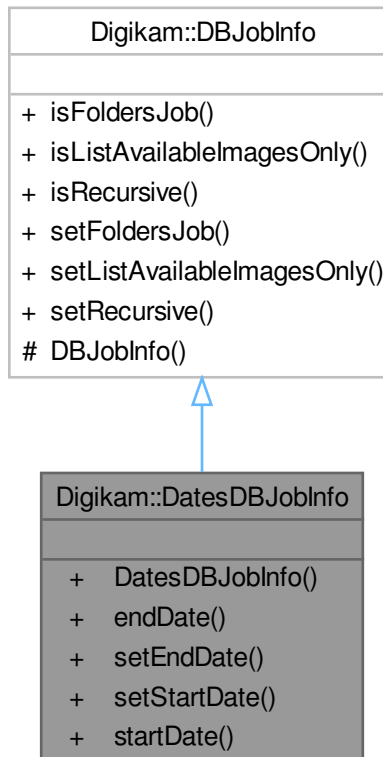
- `RuleDialog` (`Rule` \*const parent)
- void `setSettingsWidget` (`QWidget` \*const settingsWidget)

### Public Attributes

- `Ui::DateOptionDialogWidget` \*const `ui` = nullptr

## 9.258 Digikam::DatesDBJobInfo Class Reference

Inheritance diagram for Digikam::DatesDBJobInfo:



### Public Member Functions

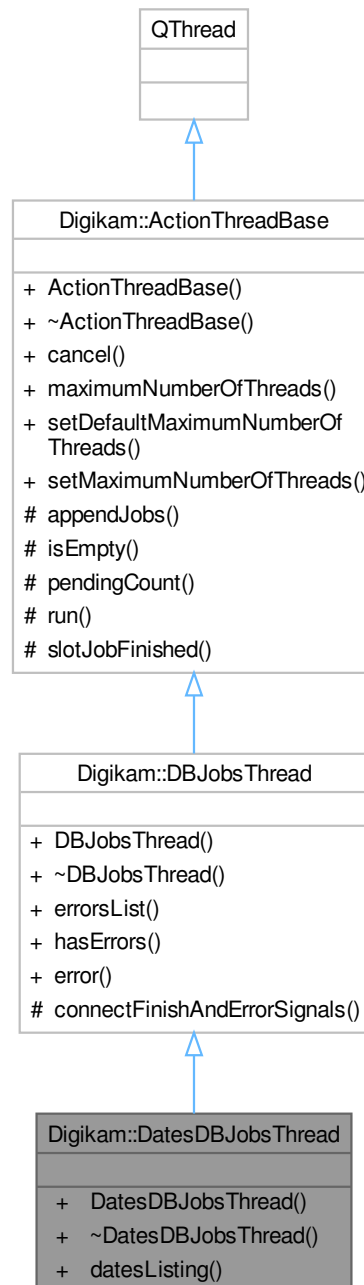
- `QDate endDate () const`
- `void setEndDate (const QDate &date)`
- `void setStartDate (const QDate &date)`
- `QDate startDate () const`

### Public Member Functions inherited from [Digikam::DBJobInfo](#)

- `bool isFoldersJob () const`
- `bool isListAvailableImagesOnly () const`
- `bool isRecursive () const`
- `void setFoldersJob ()`
- `void setListAvailableImagesOnly ()`
- `void setRecursive ()`

## 9.259 Digikam::DatesDBJobsThread Class Reference

Inheritance diagram for Digikam::DatesDBJobsThread:



### Signals

- void **foldersData** (const QHash< QDateTime, int > &)



## Signals inherited from [Digikam::DBJobsThread](#)

- void **data** (const QList< [ItemLISTERRecord](#) > &records)
- void **finished** ()

## Public Member Functions

- **DatesDBJobsThread** (QObject \*const parent)
- void [datesListing](#) (const [DatesDBJobInfo](#) &info)  
*Starts dates listing and scanning.*

## Public Member Functions inherited from [Digikam::DBJobsThread](#)

- **DBJobsThread** (QObject \*const parent)
- QList< QString > & [errorsList](#) ()  
*A method to get all errors reported from jobs.*
- bool [hasErrors](#) ()  
*hasErrors: a method to check for jobs errors*

## Public Member Functions inherited from [Digikam::ActionThreadBase](#)

- **ActionThreadBase** (QObject \*const parent=nullptr)
- void **cancel** (bool isCancel=true)  
*Cancel processing of current jobs under progress.*
- int **maximumNumberOfThreads** () const  
*Return the maximum number of threads used to parallelize collection of job processing.*
- void [setDefaultMaximumNumberOfThreads](#) ()  
*Reset maximum number of threads used to parallelize collection of job processing to max core detected on computer.*
- void **setMaximumNumberOfThreads** (int n)  
*Adjust maximum number of threads used to parallelize collection of job processing.*

## Additional Inherited Members

## Public Slots inherited from [Digikam::DBJobsThread](#)

- void [error](#) (const QString &errString)  
*Appends the error string to m\_errorsList.*

## Protected Slots inherited from [Digikam::ActionThreadBase](#)

- void [slotJobFinished](#) ()

## Protected Member Functions inherited from [Digikam::DBJobsThread](#)

- void [connectFinishAndErrorSignals](#) ([DBJob](#) \*const j)  
*Connects the signals of job to the signals of the thread.*

## Protected Member Functions inherited from [Digikam::ActionThreadBase](#)

- void [appendJobs](#) (const [ActionJobCollection](#) &jobs)  
*Append a collection of jobs to process into QThreadPool.*
- bool [isEmpty](#) () const  
*Return true if list of pending jobs to process is empty.*
- int [pendingCount](#) () const  
*Return the number of pending jobs to process.*
- void [run](#) () override  
*Main thread loop used to process jobs in todo list.*

### 9.259.1 Member Function Documentation

#### 9.259.1.1 [datesListing\(\)](#)

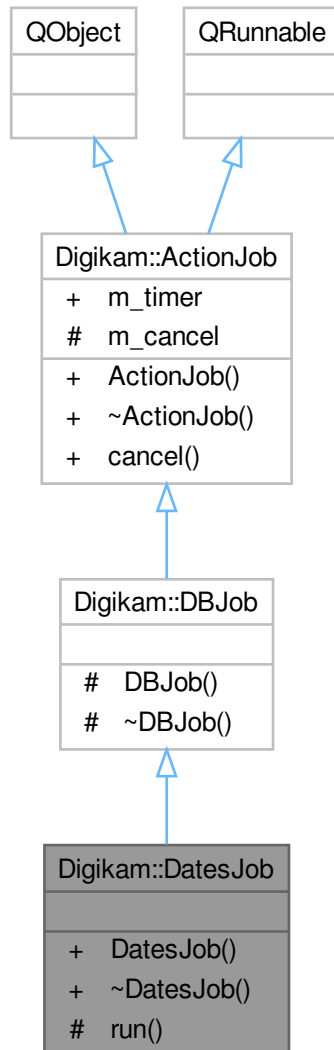
```
void Digikam::DatesDBJobsThread::datesListing (  
    const DatesDBJobInfo & info )
```

##### Parameters

<i>info</i>	represents the dates job info
-------------	-------------------------------

## 9.260 Digikam::DatesJob Class Reference

Inheritance diagram for Digikam::DatesJob:



### Signals

- void **foldersData** (const QHash< QDateTime, int > &datesStatMap)

### Signals inherited from [Digikam::DBJob](#)

- void **data** (const QList< [ItemListerRecord](#) > &records)
- void **error** (const QString &err)

## Signals inherited from [Digikam::ActionJob](#)

- void **signalDone** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job is done.*
- void **signalProgress** (int)  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job progress.*
- void **signalStarted** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager that job is started.*

## Public Member Functions

- **DatesJob** (const [DatesDBJobInfo](#) &jobInfo)

## Public Member Functions inherited from [Digikam::ActionJob](#)

- **ActionJob** (QObject \*const parent=nullptr)  
*Constructor which delegate deletion of QRunnable instance to [ActionThreadBase](#), not QThreadPool.*
- **~ActionJob** () override  
*Re-implement destructor in you implementation.*

## Protected Member Functions

- void **run** () override

## Additional Inherited Members

## Public Slots inherited from [Digikam::ActionJob](#)

- void **cancel** ()  
*Call this method to cancel job.*

## Public Attributes inherited from [Digikam::ActionJob](#)

- QElapsedTimer **m\_timer**  
*Timer to determine the running time of the job.*

## Protected Attributes inherited from [Digikam::ActionJob](#)

- bool **m\_cancel** = false  
*You can use this boolean in your implementation to know if job must be canceled.*

## 9.261 Digikam::DateTreeView Class Reference

Inheritance diagram for Digikam::DateTreeView:



### Public Slots

- void **setCurrentAlbum** (int dateId, bool selectInAlbumManager=true)
- void **setCurrentAlbums** (const QList< Album \* > &albums, bool selectInAlbumManager=true)

## Public Slots inherited from [Digikam::AbstractAlbumTreeView](#)

- void **adaptColumnsToContent** ()  
*Adapt the column sizes to the contents of the tree view.*
- void **expandEverything** (const QModelIndex &index)  
*Expands the complete tree under the given index.*
- void **scrollToSelectedAlbum** ()  
*Scrolls to the first selected album if there is one.*
- void **setCurrentAlbums** (const QList< Album \* > &albums, bool selectInAlbumManager=true)  
*Selects the given album.*
- void **setSearchTextSettings** (const SearchTextSettings &settings)
- void **slotCollapseAllNodes** ()  
*slotCollapseAllNodes - collapse all nodes without root node*
- void **slotCollapseNode** ()  
*slotCollapseNode - collapse recursively selected nodes*
- void **slotExpandNode** ()  
*slotExpandNode - expands recursively selected nodes*

## Public Member Functions

- **DateTreeView** (QWidget \*const parent=nullptr, Flags flags=DefaultFlags)
- **DAlbum \* albumForIndex** (const QModelIndex &index) const
- **DateAlbumModel \* albumModel** () const
- **DAlbum \* currentAlbum** () const
- void **setAlbumFilterModel** (AlbumFilterModel \*const filterModel)
- void **setAlbumModel** (DateAlbumModel \*const model)

## Public Member Functions inherited from [Digikam::AbstractCountingAlbumTreeView](#)

- **AbstractCountingAlbumTreeView** (QWidget \*const parent, Flags flags)

## Public Member Functions inherited from [Digikam::AbstractAlbumTreeView](#)

- **AbstractAlbumTreeView** (QWidget \*const parent, Flags flags)  
*Constructs an album tree view.*
- void **addContextMenuElement** (ContextMenuElement \*const element)
- **AlbumFilterModel \* albumFilterModel** () const
- **AbstractSpecificAlbumModel \* albumModel** () const
- QList< ContextMenuElement \* > **contextMenuElements** () const
- template<class A >  
QList< A \* > **currentAlbums** ()
- void **doLoadState** () override  
*Implements state loading for the album tree view in a somewhat clumsy procedure because the model may not be fully loaded when this method is called.*
- void **doSaveState** () override  
*Implement this hook method for state saving.*
- bool **expandMatches** (const QModelIndex &index)  
*Ensures that every current match is visible by expanding all parent entries.*
- QModelIndex **indexVisuallyAt** (const QPoint &p)  
*This is a combination of indexAt() checked with visualRect().*

- void **removeContextMenuElement** ([ContextMenuElement](#) \*const element)
- [QList](#)< [Album](#) \* > **selectedItems** ()
  - selectedItems()* -
- void **setAlbumManagerCurrentAlbum** (const bool setCurrentAlbum)
  - Some treeviews shall control the global current album kept by [AlbumManager](#).*
- void **setContextMenuIcon** (const [QPixmap](#) &pixmap)
  - Set the context menu title and icon.*
- void **setContextMenuTitle** (const [QString](#) &title)
- void **setEnabledContextMenu** (const bool enable)
  - Determines the global decision to show a popup menu or not.*
- void **setExpandNewCurrentItem** (const bool doThat)
  - Expand an item when making it the new current item.*
- void **setExpandOnSingleClick** (const bool doThat)
  - Enable expanding of tree items on single click on the item (default: off)*
- void **setSelectAlbumOnClick** (const bool selectOnClick)
  - Sets whether to select an album on click via the album manager or not.*
- void **setSelectOnContextMenu** (const bool select)
  - Sets whether to select the album under the mouse cursor on a context menu request (so that the album is shown using the album manager) or not.*
- bool **viewportEvent** ([QEvent](#) \*event) override
  - For internal use only.*

## Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) ([QObject](#) \*const host)
  - Constructor.*
- virtual **~StateSavingObject** ()
  - Destructor.*
- [StateSavingDepth](#) **getStateSavingDepth** () const
  - Returns the depth used for state saving or loading.*
- void **loadState** ()
  - Invokes loading the class' state.*
- void **saveState** ()
  - Invokes saving the class' state.*
- virtual void **setConfigGroup** (const [KConfigGroup](#) &group)
  - Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void **setEntryPrefix** (const [QString](#) &prefix)
  - Define a prefix that will be used for every entry in the config group.*
- void **setStateSavingDepth** (const [StateSavingDepth](#) depth)
  - Sets the depth used for state saving or loading.*

## Additional Inherited Members

## Public Types inherited from [Digikam::AbstractAlbumTreeView](#)

- enum [Flag](#) {
  - [CreateDefaultModel](#) , [CreateDefaultFilterModel](#) , [CreateDefaultDelegate](#) , [ShowCountAccordingToSettings](#) , [AlwaysShowInclusiveCounts](#) , **DefaultFlags** = [CreateDefaultFilterModel](#) | [CreateDefaultDelegate](#) | [ShowCountAccordingToSettings](#) }
- typedef [QFlags](#)< [Flag](#) > **Flags**

## Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }

*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

## Signals inherited from [Digikam::AbstractAlbumTreeView](#)

- void [currentAlbumChanged](#) ([Album](#) \*currentAlbum)  
*Emitted when the currently selected album changes.*
- void [selectedAlbumsChanged](#) (const [QList](#)< [Album](#) \* > &selectedAlbums)  
*Emitted when the current selection changes.*

## Protected Slots inherited from [Digikam::AbstractAlbumTreeView](#)

- void [albumSettingsChanged](#) ()
- void [slotCurrentChanged](#) ()
- virtual void [slotRootAlbumAvailable](#) ()  
*override if implemented behavior is not as intended*
- void [slotSearchTextSettingsAboutToChange](#) (bool searched, bool willSearch)
- void [slotSearchTextSettingsChanged](#) (bool wasSearching, bool searching)
- void [slotSelectionChanged](#) ()

## Protected Member Functions inherited from [Digikam::AbstractCountingAlbumTreeView](#)

- void [rowsInserted](#) (const [QModelIndex](#) &parent, int start, int end) override
- void [setAlbumFilterModel](#) ([AlbumFilterModel](#) \*const filterModel)
- void [setAlbumModel](#) ([AbstractCountingAlbumModel](#) \*const model)

## Protected Member Functions inherited from [Digikam::AbstractAlbumTreeView](#)

- virtual void [addCustomContextMenuActions](#) ([ContextMenuHelper](#) &cmh, [Album](#) \*album)  
*Hook method to add custom actions to the generated context menu.*
- virtual [QPixmap](#) [contextMenuIcon](#) () const  
*Hook method that can be implemented to return a special icon used for the context menu.*
- virtual [QString](#) [contextMenuTitle](#) () const  
*Hook method to implement that returns the title for the context menu.*
- void [dragEnterEvent](#) ([QDragEnterEvent](#) \*e) override
- void [dragLeaveEvent](#) ([QDragLeaveEvent](#) \*e) override
- void [dragMoveEvent](#) ([QDragMoveEvent](#) \*e) override
- void [dropEvent](#) ([QDropEvent](#) \*e) override
- virtual void [handleCustomContextMenuAction](#) ([QAction](#) \*action, const [AlbumPointer](#)< [Album](#) > &album)  
*Hook method to handle the custom context menu actions that were added with [addCustomContextMenuActions](#).*
- virtual void [middleButtonPressed](#) ([Album](#) \*a)
- void [mousePressEvent](#) ([QMouseEvent](#) \*e) override  
*Other helper methods.*
- virtual [QPixmap](#) [pixmapForDrag](#) (const [QStyleOptionViewItem](#) &option, [QList](#)< [QModelIndex](#) > indexes)  
*TODO: Move to delegate, when we have one.*
- void [rowsAboutToBeRemoved](#) (const [QModelIndex](#) &parent, int start, int end) override
- void [rowsInserted](#) (const [QModelIndex](#) &index, int start, int end) override
- void [setAlbumFilterModel](#) ([AlbumFilterModel](#) \*const filterModel)
- void [setAlbumModel](#) ([AbstractSpecificAlbumModel](#) \*const model)
- virtual bool [showContextMenuAt](#) ([QContextMenuEvent](#) \*event, [Album](#) \*albumForEvent)  
*Hook method to implement that determines if a context menu shall be displayed for the given event at the position coded in the event.*
- void [startDrag](#) ([Qt::DropActions](#) supportedActions) override



## Protected Member Functions inherited from [Digikam::StateSavingObject](#)

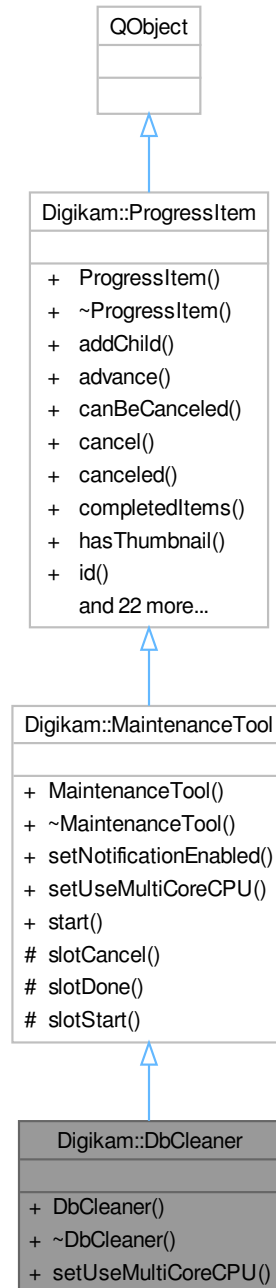
- QString [entryName](#) (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup [getConfigGroup](#) () const  
*Returns the config group that must be used for state saving and loading.*

## Protected Attributes inherited from [Digikam::AbstractAlbumTreeView](#)

- [AlbumFilterModel](#) \* [m\\_albumFilterModel](#) = nullptr
- [AbstractSpecificAlbumModel](#) \* [m\\_albumModel](#) = nullptr
- bool [m\\_checkOnMiddleClick](#) = false
- [AlbumModelDragDropHandler](#) \* [m\\_dragDropHandler](#) = nullptr
- Flags [m\\_flags](#) = DefaultFlags
- int [m\\_lastScrollBarValue](#) = 0
- bool [m\\_restoreCheckState](#) = false

## 9.262 Digikam::DbCleaner Class Reference

Inheritance diagram for Digikam::DbCleaner:



### Public Member Functions

- **DbCleaner** (bool cleanThumbsDb=false, bool cleanFacesDb=false, bool cleanSimilarityDb=false, bool shrinkDatabases=false, [ProgressItem](#) \*const parent=nullptr)
- void [setUseMultiCoreCPU](#) (bool b) override

*Re-implement this method if your tool is able to use multi-core CPU to process item in parallel.*

## Public Member Functions inherited from Digikam::MaintenanceTool

- **MaintenanceTool** (const QString &id, ProgressItem \*const parent=nullptr)
- void **setNotificationEnabled** (bool b)

*If true, show a notification message on desktop notification manager with time elapsed to run process.*

## Public Member Functions inherited from Digikam::ProgressItem

- **ProgressItem** (ProgressItem \*const parent, const QString &id, const QString &label, const QString &status, bool canBeCanceled, bool hasThumb)
- void **addChild** (ProgressItem \*const kiddo)
- bool **advance** (unsigned int v)

*Advance total items processed by n values and update percentage in progressbar.*

- bool **canBeCanceled** () const
  - void **cancel** ()
  - bool **canceled** () const
  - unsigned int **completedItems** () const
  - bool **hasThumbnail** () const
  - const QString & **id** () const
  - bool **incCompletedItems** (unsigned int v=1)
  - void **incTotalItems** (unsigned int v=1)
  - const QString & **label** () const
  - ProgressItem \* **parent** () const
  - unsigned int **progress** () const
  - void **removeChild** (ProgressItem \*const kiddo)
  - void **reset** ()
- Reset the progress value of this item to 0 and the status string to the empty string.*
- void **setComplete** ()
- Tell the item it has finished.*
- bool **setCompletedItems** (unsigned int v)
  - void **setLabel** (const QString &v)
  - void **setProgress** (unsigned int v)
- Set the progress (percentage of completion) value of this item.*
- void **setShowAtStart** (bool showAtStart)
- Set the property to pop-up item when it's added in progress manager.*
- void **setStatus** (const QString &v)
- Set the string to be used for showing this item's current status.*
- void **setThumbnail** (const QIcon &icon)
- Sets whether this item has a thumbnail.*
- void **setTotalItems** (unsigned int v)
  - void **setUsesBusyIndicator** (bool useBusyIndicator)
- Sets whether this item uses a busy indicator instead of real progress for its progress bar.*
- bool **showAtStart** () const
  - const QString & **status** () const
  - bool **totalCompleted** () const
  - unsigned int **totalItems** () const
  - void **updateProgress** ()
- Recalculate progress according to total/completed items and update.*
- bool **usesBusyIndicator** () const

## Additional Inherited Members

### Public Slots inherited from [Digikam::MaintenanceTool](#)

- void **start** ()

### Signals inherited from [Digikam::MaintenanceTool](#)

- void **signalCanceled** ()  
*Emit when process is canceled.*
- void **signalComplete** ()  
*Emit when process is done (not canceled).*

### Signals inherited from [Digikam::ProgressItem](#)

- void [progressItemAdded](#) ([ProgressItem](#) \*item)  
*Emitted when a new [ProgressItem](#) is added.*
- void [progressItemCanceled](#) ([ProgressItem](#) \*item)  
*Emitted when an item was canceled.*
- void **progressItemCanceledById** (const QString &id)
- void [progressItemCompleted](#) ([ProgressItem](#) \*item)  
*Emitted when a progress item was completed.*
- void [progressItemLabel](#) ([ProgressItem](#) \*item, const QString &label)  
*Emitted when the label of an item changed.*
- void [progressItemProgress](#) ([ProgressItem](#) \*item, unsigned int v)  
*Emitted when the progress value of an item changes.*
- void [progressItemStatus](#) ([ProgressItem](#) \*item, const QString &mess)  
*Emitted when the status message of an item changed.*
- void [progressItemThumbnail](#) ([ProgressItem](#) \*item, const QPixmap &thumb)  
*Emitted when the thumbnail data must be set in item.*
- void [progressItemUsesBusyIndicator](#) ([ProgressItem](#) \*item, bool value)  
*Emitted when the busy indicator state of an item changes.*

### Protected Slots inherited from [Digikam::MaintenanceTool](#)

- virtual void **slotCancel** ()
- virtual void **slotDone** ()
- virtual void **slotStart** ()

## 9.262.1 Member Function Documentation

### 9.262.1.1 `setUseMultiCoreCPU()`

```
void Digikam::DbCleaner::setUseMultiCoreCPU (
    bool ) [override], [virtual]
```

Reimplemented from [Digikam::MaintenanceTool](#).

## 9.263 Digikam::DbEngineAccess Class Reference

The [DbEngineAccess](#) class provides access to the database: Create an instance of this class on the stack to retrieve a pointer to the database.

### Static Public Member Functions

- static bool [checkReadyForUse](#) (QString &error)  
*Checks the availability of drivers.*

### 9.263.1 Member Function Documentation

#### 9.263.1.1 checkReadyForUse()

```
bool Digikam::DbEngineAccess::checkReadyForUse (  
    QString & error ) [static]
```

Must be used in children class. Return true if low level drivers are ready to use, else false with an error string of the problem.

## 9.264 Digikam::DbEngineAction Class Reference

### Public Attributes

- QList< [DbEngineActionElement](#) > **dbActionElements**
- QString **mode**
- QString **name**

## 9.265 Digikam::DbEngineActionElement Class Reference

### Public Attributes

- QString **mode**
- int **order** = 0
- QString **statement**

## 9.266 Digikam::DbEngineActionType Class Reference

The [DbEngineActionType](#) is used by the [BdEngineBackend](#) to wrap another data object within an sql statement and controls whether it should be used as field entry or as value (prepared to an sql statement with positional binding).

### Public Member Functions

- **DbEngineActionType** (const [DbEngineActionType](#) &actionType)
- QVariant **getActionValue** ()  
*Returns the wrapped object.*
- bool **isValue** () const  
*Returns true, if the entry is an value element.*
- void **setActionValue** (const QVariant &actionValue)  
*Sets the wrapped object.*
- void **setValue** (bool isValue)  
*Sets the DBAction mode: true, if the entry is an value element.*

### Static Public Member Functions

- static [DbEngineActionType](#) **fieldEntry** (const QVariant &actionValue)
- static [DbEngineActionType](#) **value** (const QVariant &value)

## 9.266.1 Member Function Documentation

### 9.266.1.1 isValue()

```
bool Digikam::DbEngineActionType::isValue ( ) const
```

Returns false, if the entry should be used as field entry.

### 9.266.1.2 setValue()

```
void Digikam::DbEngineActionType::setValue (
    bool isValue )
```

false, if the entry should be used as field entry.

## 9.267 Digikam::DbEngineConfig Class Reference

### Static Public Member Functions

- static bool **checkReadyForUse** ()
- static [DbEngineConfigSettings](#) **element** (const QString &databaseType)
- static QString **errorMessage** ()

## 9.268 Digikam::DbEngineConfigSettings Class Reference

### Public Attributes

- QString **connectOptions**
- QString **databaseID**
- QString **databaseName**
- QString **hostName**
- QString **password**
- QString **port**
- QMap< QString, [DbEngineAction](#) > **sqlStatements**
- QString **userName**

## 9.269 Digikam::DbEngineConfigSettingsLoader Class Reference

### Public Member Functions

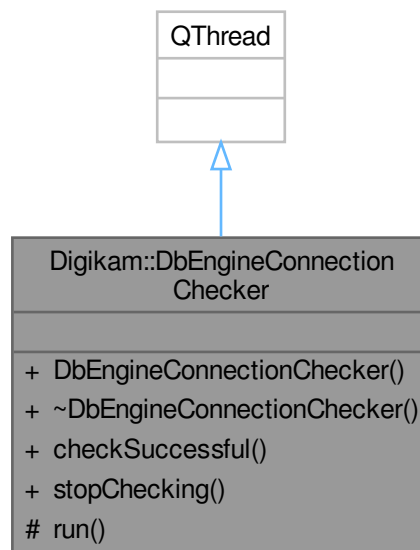
- **DbEngineConfigSettingsLoader** (const QString &filepath, int xmlVersion)
- bool **readConfig** (const QString &filepath, int xmlVersion)
- [DbEngineConfigSettings](#) **readDatabase** (const QDomElement &databaseElement)
- void **readDBActions** (const QDomElement &sqlStatementElements, [DbEngineConfigSettings](#) &configElement)

### Public Attributes

- QMap< QString, [DbEngineConfigSettings](#) > **databaseConfigs**
- QString **errorMessage**
- bool **isValid** = false

## 9.270 Digikam::DbEngineConnectionChecker Class Reference

Inheritance diagram for Digikam::DbEngineConnectionChecker:



### Public Slots

- void **stopChecking** ()

### Signals

- void **done** ()
- void **failedAttempt** ()

### Public Member Functions

- **DbEngineConnectionChecker** (const [DbEngineParameters](#) &parameters)
- bool **checkSuccessful** () const

### Protected Member Functions

- void **run** () override

## 9.271 Digikam::DbEngineErrorAnswer Class Reference

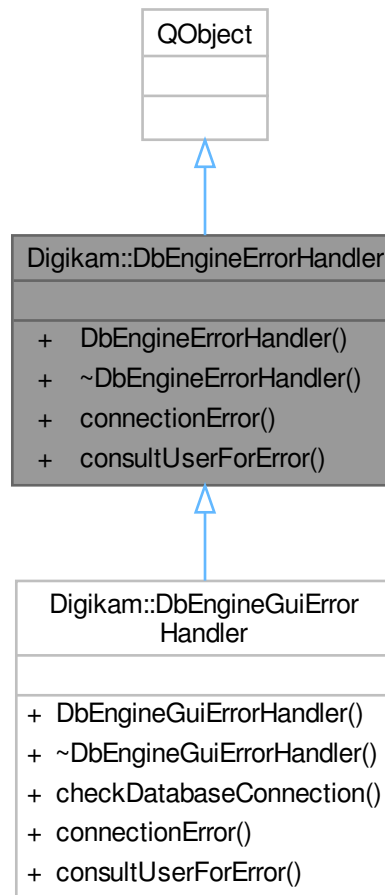
### Public Member Functions

- virtual void **connectionErrorAbortQueries** ()=0
- virtual void **connectionErrorContinueQueries** ()=0



## 9.272 Digikam::DbEngineErrorHandler Class Reference

Inheritance diagram for Digikam::DbEngineErrorHandler:



### Public Slots

- virtual void `connectionError` (`DbEngineErrorAnswer` \*answer, const QSqlError &error, const QString &query)=0

*In the situation of a connection error, all threads will be waiting with their queries and this method is called.*

- virtual void `consultUserForError` (`DbEngineErrorAnswer` \*answer, const QSqlError &error, const QString &query)=0

*In the situation of an error requiring user intervention or information, all threads will be waiting with their queries and this method is called.*

### 9.272.1 Member Function Documentation

#### 9.272.1.1 connectionError

```
virtual void Digikam::DbEngineErrorHandler::connectionError (
    DbEngineErrorAnswer * answer,
```

```
const QSqlError & error,
const QString & query ) [pure virtual], [slot]
```

This method can display an error dialog and try to repair the connection. It must then call either `connectionErrorContinueQueries()` or `connectionErrorAbortQueries()`. The method is guaranteed to be invoked in the UI thread.

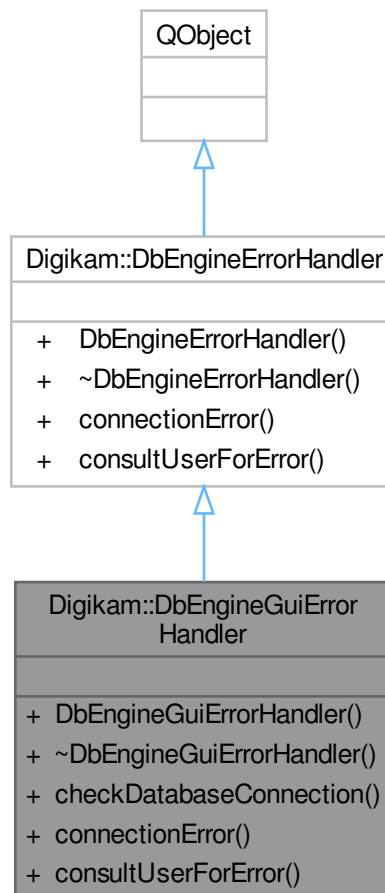
### 9.272.1.2 consultUserForError

```
virtual void Digikam::DbEngineErrorHandler::consultUserForError (
    DbEngineErrorAnswer * answer,
    const QSqlError & error,
    const QString & query ) [pure virtual], [slot]
```

This method can display an error dialog. It must then call either `connectionErrorContinueQueries()` or `connectionErrorAbortQueries()`. The method is guaranteed to be invoked in the UI thread.

## 9.273 Digikam::DbEngineGuiErrorHandler Class Reference

Inheritance diagram for Digikam::DbEngineGuiErrorHandler:



**Public Slots**

- void **connectionError** ([DbEngineErrorAnswer](#) \*answer, const QSqlError &error, const QString &query) override
- void **consultUserForError** ([DbEngineErrorAnswer](#) \*answer, const QSqlError &error, const QString &query) override

**Public Slots inherited from [Digikam::DbEngineErrorHandler](#)**

- virtual void **connectionError** ([DbEngineErrorAnswer](#) \*answer, const QSqlError &error, const QString &query)=0  
*In the situation of a connection error, all threads will be waiting with their queries and this method is called.*
- virtual void **consultUserForError** ([DbEngineErrorAnswer](#) \*answer, const QSqlError &error, const QString &query)=0  
*In the situation of an error requiring user intervention or information, all threads will be waiting with their queries and this method is called.*

**Public Member Functions**

- **DbEngineGuiErrorHandler** (const [DbEngineParameters](#) &parameters)
- bool **checkDatabaseConnection** ()

## 9.274 Digikam::DbEngineLocking Class Reference

**Public Attributes**

- int **lockCount** = 0  
*create a recursive mutex*
- QRecursiveMutex **mutex**

## 9.275 Digikam::DbEngineParameters Class Reference

This class encapsulates all parameters needed to establish a connection to a database (inspired by the API of Qt::Sql).

**Public Member Functions**

- **DbEngineParameters** (const QString &\_type, const QString &\_databaseNameCore, const QString &\_connectOptions=QString(), const QString &\_hostName=QString(), int \_port=-1, bool \_walMode=false, bool \_internalServer=false, const QString &\_userName=QString(), const QString &\_password=QString(), const QString &\_databaseNameThumbnails=QString(), const QString &\_databaseNameFace=QString(), const QString &\_databaseNameSimilarity=QString(), const QString &\_internalServerDBPath=QString(), const QString &\_internalServerMysqlInitCmd=QString(), const QString &\_internalServerMysqlAdminCmd=QString(), const QString &\_internalServerMysqlServerCmd=QString(), const QString &\_internalServerMysqlUpgradeCmd=QString())
- **DbEngineParameters** (const QUrl &url)  
*QUrl helpers.*
- [DbEngineParameters](#) **faceParameters** () const

- Replaces databaseName with databaseNameFace.*
- QString **getCoreDatabaseNameOrDir** () const
- QString **getFaceDatabaseNameOrDir** () const
- QString **getSimilarityDatabaseNameOrDir** () const
- QString **getThumbsDatabaseNameOrDir** () const
- QByteArray **hash** () const
  - Creates a unique hash of the values stored in this object.*
- void **insertInUrl** (QUrl &url) const
- QString **internalServerPath** () const
- bool **isMySQL** () const
- bool **isSQLite** () const
- bool **isValid** () const
  - Performs basic checks that the parameters are not empty and have the information required for the databaseType.*
- void **legacyAndDefaultChecks** (const QString &suggestedPath=QString())
- bool **operator!=** (const DbEngineParameters &other) const
- bool **operator==** (const DbEngineParameters &other) const
- void **readFromConfig** (const QString &configGroup=QString())
  - Read and write parameters from config.*
- void **removeLegacyConfig** ()
- void **setCoreDatabasePath** (const QString &folderOrFileOrName)
  - Use these methods if you set a file or a folder.*
- void **setFaceDatabasePath** (const QString &folderOrFileOrName)
- void **setInternalServerPath** (const QString &path)
  - For Mysql internal server: manage the database path to store database files.*
- void **setSimilarityDatabasePath** (const QString &folderOrFileOrName)
- void **setThumbsDatabasePath** (const QString &folderOrFileOrName)
- DbEngineParameters **similarityParameters** () const
  - Replaces databaseName with databaseNameFace.*
- QString **SQLiteDatabaseFile** () const
- DbEngineParameters **thumbnailParameters** () const
  - Replaces databaseName with databaseNameThumbnails.*
- void **writeToConfig** (const QString &configGroup=QString()) const

### Static Public Member Functions

- static QString **coreDatabaseDirectorySQLite** (const QString &path)
- static QString **coreDatabaseFileSQLite** (const QString &folderOrFile)
- static QString **defaultMysqlAdminCmd** ()
  - Return the default Mysql server administration name (Internal server only).*
- static QString **defaultMysqlInitCmd** ()
  - Return the default Mysql initialization command name (Internal server only).*
- static QString **defaultMysqlServerCmd** ()
  - Return the default Mysql server command name (Internal server only).*
- static QString **defaultMysqlUpgradeCmd** ()
  - Return the default Mysql upgrade command name (Internal server only).*
- static DbEngineParameters **defaultParameters** (const QString &databaseType)
  - Return a set of default parameters for the given type.*
- static QString **faceDatabaseDirectorySQLite** (const QString &path)
- static QString **faceDatabaseFileSQLite** (const QString &folderOrFile)
- static QString **MySQLDatabaseType** ()
- static DbEngineParameters **parametersForSQLite** (const QString &databaseFile)

Convenience methods to create a [DbEngineParameters](#) object for an SQLITE database specified by the local file path.

- static [DbEngineParameters](#) **parametersForSQLiteDefaultFile** (const QString &directory)
- static [DbEngineParameters](#) **parametersFromConfig** (const QString &configGroup=QString())
- static void **removeFromUrl** (QUrl &url)
- static QString **serverPrivatePath** ()

Return the hidden path from home directory to store private data used by internal Mysql server.

- static QString **similarityDatabaseDirectorySQLite** (const QString &path)
- static QString **similarityDatabaseFileSQLite** (const QString &folderOrFile)
- static QString **SQLiteDatabaseType** ()

Returns the databaseType designating the said database.

- static QString **thumbnailDatabaseDirectorySQLite** (const QString &path)
- static QString **thumbnailDatabaseFileSQLite** (const QString &folderOrFile)

## Public Attributes

- QString **connectOptions**
- QString **databaseNameCore**
- QString **databaseNameFace**
- QString **databaseNameSimilarity**
- QString **databaseNameThumbnails**
- QString **databaseType**
- QString **hostName**
- bool **internalServer** = false
- QString **internalServerDBPath**
- QString **internalServerMysqlAdminCmd**
- QString **internalServerMysqlInitCmd**

Settings stored in config file and used only with internal server at runtime to start server instance or init database tables.

- QString **internalServerMysqlServerCmd**
- QString **internalServerMysqlUpgradeCmd**
- QString **password**
- int **port** = -1
- QString **userName**
- bool **walMode** = false

### 9.275.1 Detailed Description

The values can be read from and written to a QUrl.

### 9.275.2 Member Function Documentation

#### 9.275.2.1 defaultParameters()

```
DbEngineParameters Digikam::DbEngineParameters::defaultParameters (
    const QString & databaseType ) [static]
```

For Mysql, it return internal server configuration.

### 9.275.2.2 `getCoreDatabaseNameOrDir()`

```
QString Digikam::DbEngineParameters::getCoreDatabaseNameOrDir ( ) const
```

#### Note

In case of SQLite, the database name typically is a file. For non-SQLite, this simply handle the database name.

### 9.275.2.3 `readFromConfig()`

```
void Digikam::DbEngineParameters::readFromConfig (
    const QString & configGroup = QString() )
```

You can specify the group, or use the default value.

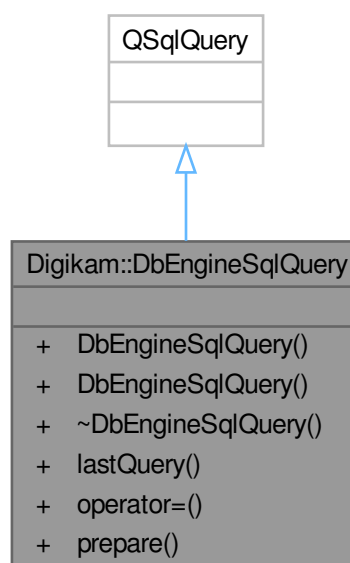
### 9.275.2.4 `SQLiteDatabaseType()`

```
QString Digikam::DbEngineParameters::SQLiteDatabaseType ( ) [static]
```

If you have a [DbEngineParameters](#) object already, you can use `isSQLite()` as well. These strings are identical to the driver identifiers in the Qt SQL module.

## 9.276 `Digikam::DbEngineSqlQuery` Class Reference

Inheritance diagram for `Digikam::DbEngineSqlQuery`:

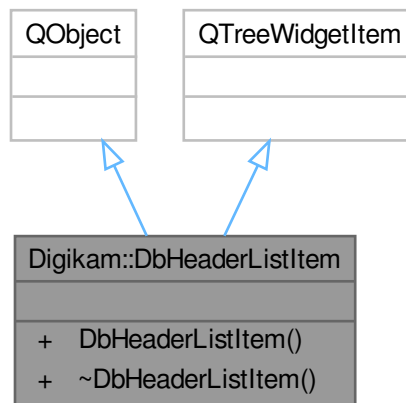


### Public Member Functions

- **DbEngineSqlQuery** (const QSqlDatabase &db)
- **DbEngineSqlQuery** (const QSqlQuery &other)
- QString **lastQuery** () const
- **DbEngineSqlQuery** & **operator=** (const **DbEngineSqlQuery** &other)
- bool **prepare** (const QString &query)

## 9.277 Digikam::DbHeaderListItem Class Reference

Inheritance diagram for Digikam::DbHeaderListItem:

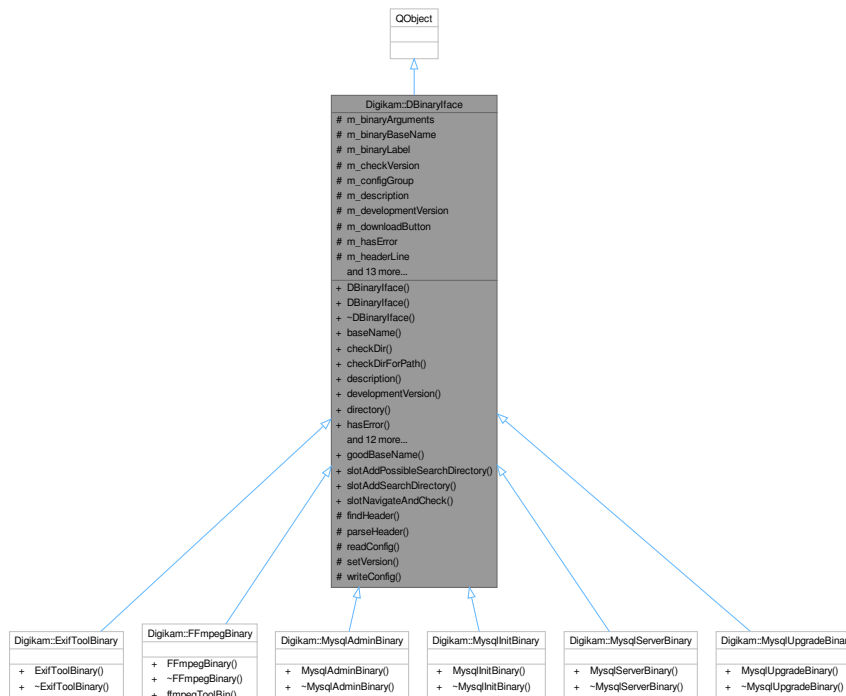


### Public Member Functions

- **DbHeaderListItem** (QTreeWidgetItem \*parent, const QString &key)

## 9.278 Digikam::DBinaryIface Class Reference

Inheritance diagram for Digikam::DBinaryIface:



### Public Slots

- virtual void **slotAddPossibleSearchDirectory** (const QString &dir)
- virtual void **slotAddSearchDirectory** (const QString &dir)
- virtual void **slotNavigateAndCheck** ()

### Signals

- void **signalBinaryValid** ()
- void **signalSearchDirectoryAdded** (const QString &dir)

### Public Member Functions

- **DBinaryIface** (const QString &binaryName, const QString &minimalVersion, const QString &header, const int headerLine, const QString &projectName, const QString &url, const QString &pluginName, const QStringList &args=QStringList(), const QString &desc=QString())
- **DBinaryIface** (const QString &binaryName, const QString &projectName, const QString &url, const QString &pluginName, const QStringList &args=QStringList(), const QString &desc=QString())
- virtual QString **baseName** () const
- virtual bool **checkDir** ()
- virtual bool **checkDirForPath** (const QString &path)
- const QString & **description** () const
- bool **developmentVersion** () const



- virtual QString **directory** () const
- bool **hasError** () const
- bool **isFound** () const
- bool **isValid** () const
- virtual QString **minimalVersion** () const
- virtual QString **path** () const
- virtual QString **path** (const QString &dir) const
- virtual QString **projectName** () const
- virtual bool **recheckDirectories** ()
- virtual void **setup** (const QString &prev=QString())
- virtual QUrl **url** () const
- const QString & **version** () const
- bool **versionIsRight** () const
- bool **versionIsRight** (const float) const

### Static Public Member Functions

- static QString **goodBaseName** (const QString &b)

### Protected Member Functions

- QString **findHeader** (const QStringList &output, const QString &header) const
- virtual bool **parseHeader** (const QString &output)
- virtual QString **readConfig** ()
- void **setVersion** (QString &version)
- virtual void **writeConfig** ()

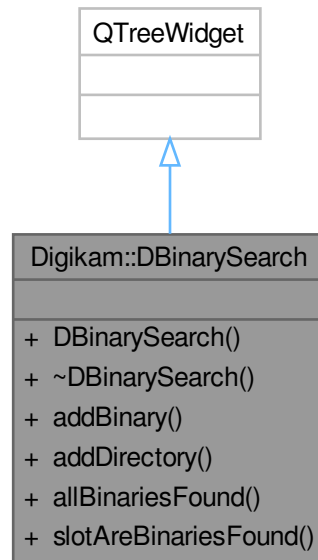
### Protected Attributes

- const QStringList **m\_binaryArguments**
- const QString **m\_binaryBaseName**
- QLabel \* **m\_binaryLabel** = nullptr
- const bool **m\_checkVersion**
- const QString **m\_configGroup**
- QString **m\_description**
- bool **m\_developmentVersion** = false
- QLabel \* **m\_downloadButton** = nullptr
- bool **m\_hasError** = false
- const int **m\_headerLine**
- const QString **m\_headerStarts**
- bool **m\_isFound** = false
- QLineEdit \* **m\_lineEdit** = nullptr
- const QString **m\_minimalVersion**
- QPushButton \* **m\_pathButton** = nullptr
- QString **m\_pathDir** = QLatin1String("")
- QFrame \* **m\_pathWidget** = nullptr
- const QString **m\_projectName**
- QSet< QString > **m\_searchPaths**
- QLabel \* **m\_statusIcon** = nullptr
- const QUrl **m\_url**
- QString **m\_version** = QLatin1String("")
- QLabel \* **m\_versionLabel** = nullptr

## 9.279 Digikam::DBinarySearch Class Reference

This class has nothing to do with a binary search, it is a widget to search for binaries.

Inheritance diagram for Digikam::DBinarySearch:



### Public Types

- enum `ColumnType` {  
**Status** = 0 , **Binary** , **Version** , **Button** ,  
**Link** }

### Public Slots

- void `slotAreBinariesFound ()`

### Signals

- void `signalAddDirectory` (const QString &dir)
- void `signalAddPossibleDirectory` (const QString &dir)
- void `signalBinariesFound` (bool)

### Public Member Functions

- `DBinarySearch` (QWidget \*const parent)
- void `addBinary` ([DBinaryIface](#) &binary)
- void `addDirectory` (const QString &dir)
- bool `allBinariesFound ()`

## 9.280 Digikam::DBInterface Class Reference

Inheritance diagram for Digikam::DBInterface:



### Public Slots

- void **slotDateTimeForUrl** (const QUrl &url, const QDateTime &dt, bool updModDate) override
- void **slotMetadataChangedForUrl** (const QUrl &url) override

## Public Member Functions

- **DBInterface** (QObject \*const parent, const QList< QUrl > &lst=QList< QUrl >(), const [OperationType](#) type=[UnspecifiedOps](#))
- QWidget \* [albumChooser](#) (QWidget \*const parent) const override  
*Albums chooser view methods (to use items from albums before to process).*
- [DAlbumIDs](#) [albumChooserItems](#) () const override
- [DInfoMap](#) [albumInfo](#) (int) const override
- QList< QUrl > [albumItems](#) ([Album](#) \*const album) const
- QList< QUrl > [albumItems](#) (int id) const override
- QList< QUrl > [albumsItems](#) (const [DAlbumIDs](#) &) const override
- QList< QUrl > [allAlbumItems](#) () const override
- QList< QUrl > [currentAlbumItems](#) () const override
- QList< [GPSItemContainer](#) \* > [currentGPSItems](#) () const override
- QList< QUrl > [currentSelectedItems](#) () const override  
*Low level items and albums methods.*
- QUrl [defaultUploadUrl](#) () const override  
*Url to upload new items without to use album selector.*
- void [deleteImage](#) (const QUrl &url) override  
*Manipulate with item.*
- [DInfoMap](#) [itemInfo](#) (const QUrl &) const override
- void [openSetupPage](#) (SetupPage page) override  
*Open configuration dialog page.*
- void [parseAlbumItemsRecursive](#) () override
- QMap< QString, QString > [passShortcutActionsToWidget](#) (QWidget \*const wdg) const override  
*Pass extra shortcut actions to widget and return prefixes of shortcuts.*
- void [setItemInfo](#) (const QUrl &, const [DInfoMap](#) &) override
- bool [supportAlbums](#) () const override
- QAbstractItemModel \* [tagFilterModel](#) () override  
*Return an instance of tag filter model if host application support this feature, else null pointer.*
- QUrl [uploadUrl](#) () const override
- QWidget \* [uploadWidget](#) (QWidget \*const parent) const override  
*Album selector view methods (to upload items from an external place).*

## Public Member Functions inherited from [Digikam::DInfoInterface](#)

- **DInfoInterface** (QObject \*const parent)
- Q\_SIGNAL void [signalAlbumItemsRecursiveCompleted](#) (const QList< QUrl > &imageList)
- Q\_SIGNAL void [signalSetupChanged](#) ()
- Q\_SIGNAL void [signalShortcutPressed](#) (const QString &shortcut, int val)
- virtual Q\_SLOT void [slotDateTimeForUrl](#) (const QUrl &url, const QDateTime &dt, bool updModDate)  
*Slot to call when date time stamp from item is changed.*
- virtual Q\_SLOT void [slotMetadataChangedForUrl](#) (const QUrl &url)  
*Slot to call when something in metadata from item is changed.*
- virtual QUrl [currentActiveItem](#) () const
- virtual void [setAlbumInfo](#) (int, const [DInfoMap](#) &) const
- Q\_SIGNAL void [signalLastItemUrl](#) (const QUrl &)
- Q\_SIGNAL void [signalAlbumChooserSelectionChanged](#) ()
- Q\_SIGNAL void [signalUploadUrlChanged](#) ()
- Q\_SIGNAL void [signalImportedImage](#) (const QUrl &)

## Additional Inherited Members

### Public Types inherited from [Digikam::DInfoInterface](#)

- typedef `QList< int >` **DAlbumIDs**  
*List of [Album](#) ids.*
- typedef `QMap< QString, QVariant >` **DInfoMap**  
*Map of properties name and value.*
- enum **SetupPage** { `ExifToolPage = 0` , `ImageQualityPage` }

### Public Attributes inherited from [Digikam::DInfoInterface](#)

- bool **forceAlbumSelection** = false

## 9.280.1 Member Function Documentation

### 9.280.1.1 albumChooser()

```
QWidget * Digikam::DBInfoIface::albumChooser (
    QWidget *const parent ) const [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

### 9.280.1.2 albumChooserItems()

```
DBInfoIface::DAlbumIDs Digikam::DBInfoIface::albumChooserItems ( ) const [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

### 9.280.1.3 albumInfo()

```
DBInfoIface::DInfoMap Digikam::DBInfoIface::albumInfo (
    int gid ) const [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

### 9.280.1.4 albumItems()

```
QList< QUrl > Digikam::DBInfoIface::albumItems (
    int id ) const [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

### 9.280.1.5 albumsItems()

```
QList< QUrl > Digikam::DBInfoIface::albumsItems (
    const DAlbumIDs & lst ) const [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

### 9.280.1.6 allAlbumItems()

```
QList< QUrl > Digikam::DBInfoIface::allAlbumItems ( ) const [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

### 9.280.1.7 currentAlbumItems()

```
QList< QUrl > Digikam::DBInfoIface::currentAlbumItems ( ) const [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

### 9.280.1.8 currentGPSItems()

```
QList< GPSItemContainer * > Digikam::DBInfoIface::currentGPSItems ( ) const [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

### 9.280.1.9 currentSelectedItems()

```
QList< QUrl > Digikam::DBInfoIface::currentSelectedItems ( ) const [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

### 9.280.1.10 defaultUploadUrl()

```
QUrl Digikam::DBInfoIface::defaultUploadUrl ( ) const [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

### 9.280.1.11 deleteImage()

```
void Digikam::DBInfoIface::deleteImage (
    const QUrl & url ) [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

### 9.280.1.12 itemInfo()

```
DBInfoIface::DInfoMap Digikam::DBInfoIface::itemInfo (
    const QUrl & url ) const [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

### 9.280.1.13 openSetupPage()

```
void Digikam::DBInfoIface::openSetupPage (
    SetupPage page ) [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

### 9.280.1.14 parseAlbumItemsRecursive()

```
void Digikam::DBInfoIface::parseAlbumItemsRecursive ( ) [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

### 9.280.1.15 passShortcutActionsToWidget()

```
QMap< QString, QString > Digikam::DBInfoIface::passShortcutActionsToWidget (
    QWidget *const ) const [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

### 9.280.1.16 setItemInfo()

```
void Digikam::DBInfoIface::setItemInfo (
    const QUrl & url,
    const DInfoMap & map ) [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

### 9.280.1.17 supportAlbums()

```
bool Digikam::DBInfoIface::supportAlbums ( ) const [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

### 9.280.1.18 tagFilterModel()

```
QAbstractItemModel * Digikam::DBInfoIface::tagFilterModel ( ) [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

### 9.280.1.19 uploadUrl()

```
QUrl Digikam::DBInfoIface::uploadUrl ( ) const [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

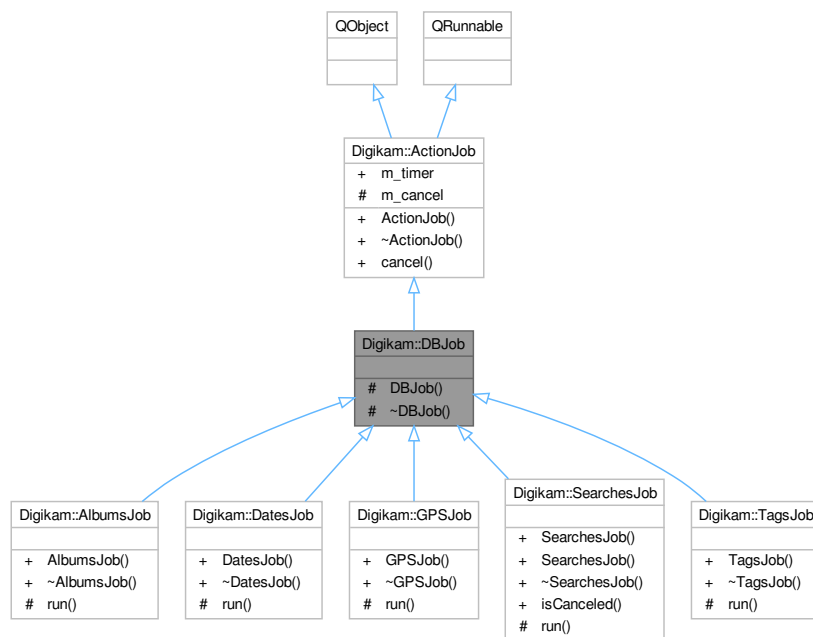
### 9.280.1.20 uploadWidget()

```
QWidget * Digikam::DBInfoIface::uploadWidget (
    QWidget *const parent ) const [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

## 9.281 Digikam::DBJob Class Reference

Inheritance diagram for Digikam::DBJob:



### Signals

- void **data** (const QList< [ItemLISTERRecord](#) > &records)
- void **error** (const QString &err)



## Signals inherited from [Digikam::ActionJob](#)

- void **signalDone** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job is done.*
- void **signalProgress** (int)  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job progress.*
- void **signalStarted** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager that job is started.*

## Additional Inherited Members

## Public Slots inherited from [Digikam::ActionJob](#)

- void **cancel** ()  
*Call this method to cancel job.*

## Public Member Functions inherited from [Digikam::ActionJob](#)

- **ActionJob** (QObject \*const parent=nullptr)  
*Constructor which delegate deletion of QRunnable instance to [ActionThreadBase](#), not QThreadPool.*
- **~ActionJob** () override  
*Re-implement destructor in you implementation.*

## Public Attributes inherited from [Digikam::ActionJob](#)

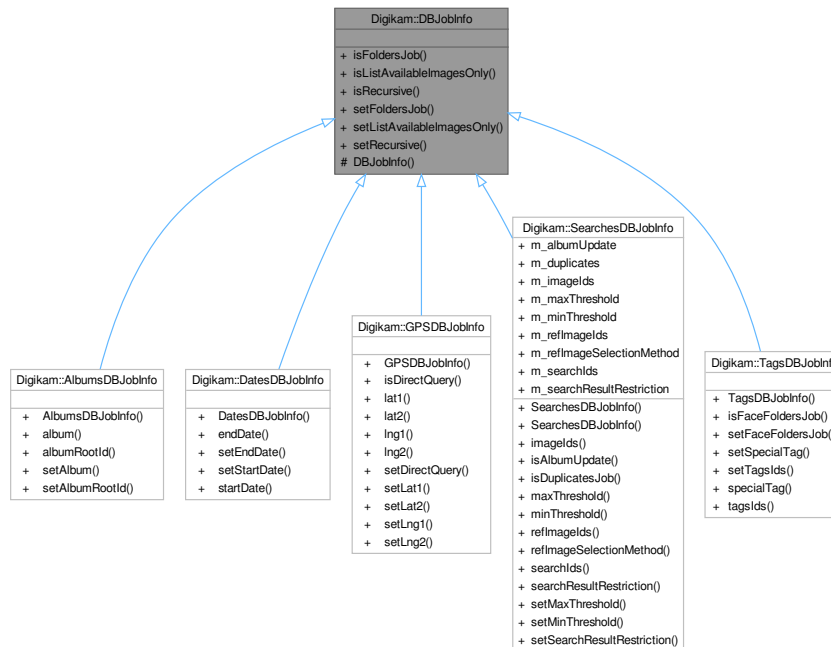
- QElapsedTimer **m\_timer**  
*Timer to determine the running time of the job.*

## Protected Attributes inherited from [Digikam::ActionJob](#)

- bool **m\_cancel** = false  
*You can use this boolean in your implementation to know if job must be canceled.*

## 9.282 Digikam::DBJobInfo Class Reference

Inheritance diagram for Digikam::DBJobInfo:

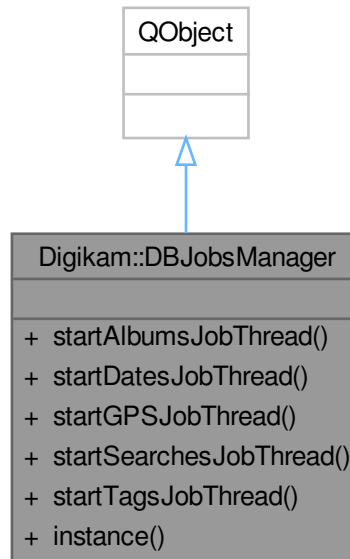


### Public Member Functions

- bool **isFoldersJob** () const
- bool **isListAvailableImagesOnly** () const
- bool **isRecursive** () const
- void **setFoldersJob** ()
- void **setListAvailableImagesOnly** ()
- void **setRecursive** ()

## 9.283 Digikam::DBJobsManager Class Reference

Inheritance diagram for Digikam::DBJobsManager:



### Public Member Functions

- [AlbumsDBJobsThread](#) \* [startAlbumsJobThread](#) (const [AlbumsDBJobInfo](#) &jInfo)  
*startAlbumsJobThread: creates and starts Albums Job Thread*
- [DatesDBJobsThread](#) \* [startDatesJobThread](#) (const [DatesDBJobInfo](#) &jInfo)  
*startDatesJobThread: creates and starts Dates Job Thread*
- [GPSDBJobsThread](#) \* [startGPSJobThread](#) (const [GPSDBJobInfo](#) &jInfo)  
*startGPSJobThread: creates and starts GPS Job Thread*
- [SearchesDBJobsThread](#) \* [startSearchesJobThread](#) (const [SearchesDBJobInfo](#) &jInfo)  
*startSearchesJobThread: creates and starts Searches Job Thread*
- [TagsDBJobsThread](#) \* [startTagsJobThread](#) (const [TagsDBJobInfo](#) &jInfo)  
*startTagsJobThread: creates and starts Tag Job Thread*

### Static Public Member Functions

- static [DBJobsManager](#) \* [instance](#) ()  
*instance: returns DBJobsManager singleton*

### Friends

- class [DBJobsManagerCreator](#)

## 9.283.1 Member Function Documentation

### 9.283.1.1 instance()

```
DBJobsManager * Digikam::DBJobsManager::instance ( ) [static]
```

#### Returns

[DBJobsManager](#) global instance

### 9.283.1.2 startAlbumsJobThread()

```
AlbumsDBJobsThread * Digikam::DBJobsManager::startAlbumsJobThread (
    const AlbumsDBJobInfo & jInfo )
```

#### Parameters

<i>jInfo</i>	holds job info about the DB job
--------------	---------------------------------

#### Returns

[AlbumsDBJobsThread](#) instance for signal/slot connection

### 9.283.1.3 startDatesJobThread()

```
DatesDBJobsThread * Digikam::DBJobsManager::startDatesJobThread (
    const DatesDBJobInfo & jInfo )
```

#### Parameters

<i>jInfo</i>	holds job info about the DB job
--------------	---------------------------------

#### Returns

[DatesDBJobsThread](#) instance for signal/slot connection

### 9.283.1.4 startGPSJobThread()

```
GPSDBJobsThread * Digikam::DBJobsManager::startGPSJobThread (
    const GPSDBJobInfo & jInfo )
```

#### Parameters

<i>jInfo</i>	holds job info about the DB job
--------------	---------------------------------

**Returns**

[GPSDBJobsThread](#) instance for signal/slot connection

**9.283.1.5 startSearchesJobThread()**

```
SearchesDBJobsThread * Digikam::DBJobsManager::startSearchesJobThread (
    const SearchesDBJobInfo & jInfo )
```

**Parameters**

<i>jInfo</i>	holds job info about the DB job
--------------	---------------------------------

**Returns**

[SearchesDBJobsThread](#) instance for signal/slot connection

**9.283.1.6 startTagsJobThread()**

```
TagsDBJobsThread * Digikam::DBJobsManager::startTagsJobThread (
    const TagsDBJobInfo & jInfo )
```

**Parameters**

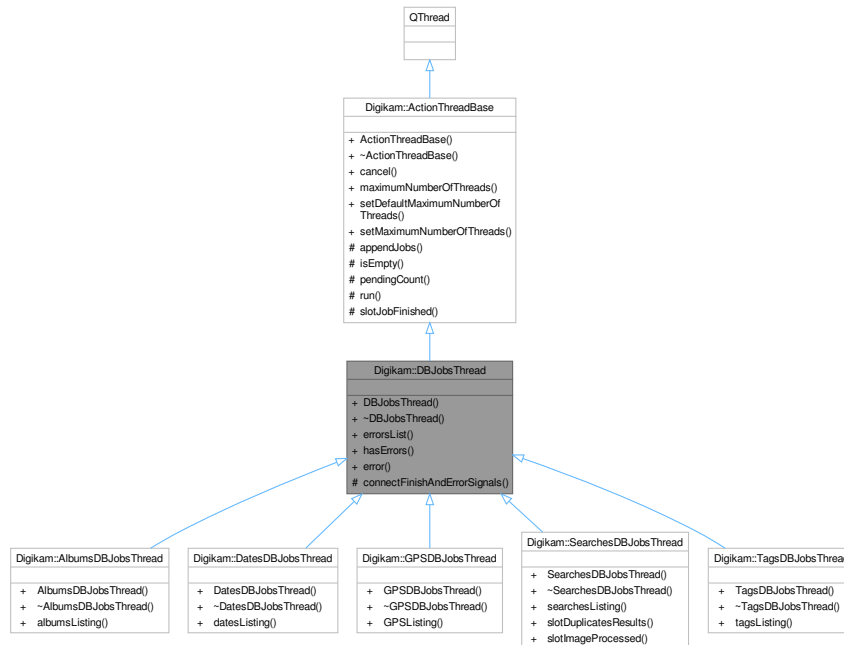
<i>jInfo</i>	holds job info about the DB job
--------------	---------------------------------

**Returns**

[TagsDBJobsThread](#) instance for signal/slot connection

## 9.284 Digikam::DBJobsThread Class Reference

Inheritance diagram for Digikam::DBJobsThread:



### Public Slots

- void `error` (const QString &errString)  
*Appends the error string to `m_errorsList`.*

### Signals

- void `data` (const QList< [ItemLISTERRecord](#) > &records)
- void `finished` ()

### Public Member Functions

- **DBJobsThread** (QObject \*const parent)
- QList< QString > & `errorsList` ()  
*A method to get all errors reported from jobs.*
- bool `hasErrors` ()  
*hasErrors: a method to check for jobs errors*

## Public Member Functions inherited from Digikam::ActionThreadBase

- **ActionThreadBase** (QObject \*const parent=nullptr)
- void **cancel** (bool isCancel=true)
 

*Cancel processing of current jobs under progress.*
- int **maximumNumberOfThreads** () const
 

*Return the maximum number of threads used to parallelize collection of job processing.*
- void **setDefaultMaximumNumberOfThreads** ()
 

*Reset maximum number of threads used to parallelize collection of job processing to max core detected on computer.*
- void **setMaximumNumberOfThreads** (int n)
 

*Adjust maximum number of threads used to parallelize collection of job processing.*

## Protected Member Functions

- void **connectFinishAndErrorSignals** (DBJob \*const j)
 

*Connects the signals of job to the signals of the thread.*

## Protected Member Functions inherited from Digikam::ActionThreadBase

- void **appendJobs** (const ActionJobCollection &jobs)
 

*Append a collection of jobs to process into QThreadPool.*
- bool **isEmpty** () const
 

*Return true if list of pending jobs to process is empty.*
- int **pendingCount** () const
 

*Return the number of pending jobs to process.*
- void **run** () override
 

*Main thread loop used to process jobs in todo list.*

## Additional Inherited Members

## Protected Slots inherited from Digikam::ActionThreadBase

- void **slotJobFinished** ()

## 9.284.1 Member Function Documentation

### 9.284.1.1 connectFinishAndErrorSignals()

```
void Digikam::DBJobsThread::connectFinishAndErrorSignals (
    DBJob *const j ) [protected]
```

#### Parameters

<i>j</i>	Job that wanted to be connected
----------	---------------------------------

**9.284.1.2 error**

```
void Digikam::DBJobsThread::error (
    const QString & errString ) [slot]
```

**Parameters**

<i>errString</i>	error string reported from the job
------------------	------------------------------------

**9.284.1.3 errorsList()**

```
QList< QString > & Digikam::DBJobsThread::errorsList ( )
```

**Returns**

String list with errors

**9.284.1.4 hasErrors()**

```
bool Digikam::DBJobsThread::hasErrors ( )
```

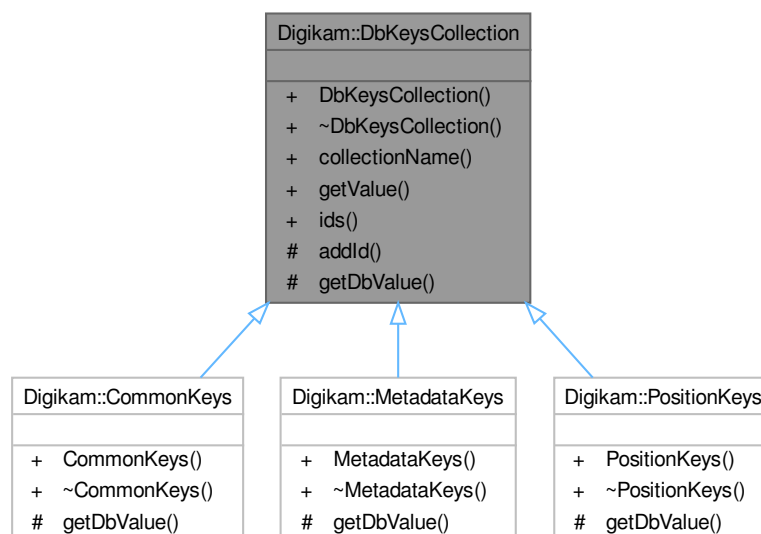
**Returns**

bool: true if the error list is not empty

**9.285 Digikam::DbKeysCollection Class Reference**

A class for managing / grouping database keys.

Inheritance diagram for Digikam::DbKeysCollection:





## Public Member Functions

- [DbKeysCollection](#) (const QString &n)  
*Default constructor.*
- QString [collectionName](#) () const  
*Get the name of the DbKeysCollection.*
- QString [getValue](#) (const QString &key, [ParseSettings](#) &settings)  
*Get a value from the database.*
- DbKeyIdsMap [ids](#) () const  
*Get all IDs associated with this key collection.*

## Protected Member Functions

- void [addId](#) (const QString &id, const QString &description)  
*Add an ID to the key collection.*
- virtual QString [getDbValue](#) (const QString &key, [ParseSettings](#) &settings)=0  
*Abstract method for retrieving the value from the database for the given key.*

## 9.285.1 Detailed Description

This class manages database keys and provides methods to get the appropriate value from the database.

## 9.285.2 Constructor & Destructor Documentation

### 9.285.2.1 DbKeysCollection()

```
Digikam::DbKeysCollection::DbKeysCollection (
    const QString & n ) [explicit]
```

#### Parameters

<i>n</i>	collection name
----------	-----------------

## 9.285.3 Member Function Documentation

### 9.285.3.1 addId()

```
void Digikam::DbKeysCollection::addId (
    const QString & id,
    const QString & description ) [protected]
```

#### Parameters

<i>id</i>	the id of the database key
<i>description</i>	a short description of the database key

**9.285.3.2 collectionName()**

```
QString Digikam::DbKeysCollection::collectionName ( ) const
```

**Returns**

the name of the collection

**9.285.3.3 getDbValue()**

```
virtual QString Digikam::DbKeysCollection::getDbValue (
    const QString & key,
    ParseSettings & settings ) [protected], [pure virtual]
```

This method has to be implemented by all child classes. It is called by the [getValue\(\)](#) method.

**Parameters**

<i>key</i>	the key representing the value in the database
<i>settings</i>	the ParseSettings object holding all relevant information about the image.

**Returns**

the value of the given database key

**See also**

[DbKeysCollection::getValue\(\)](#)

Implemented in [Digikam::CommonKeys](#), [Digikam::MetadataKeys](#), and [Digikam::PositionKeys](#).

**9.285.3.4 getValue()**

```
QString Digikam::DbKeysCollection::getValue (
    const QString & key,
    ParseSettings & settings )
```

**Parameters**

<i>key</i>	the key representing the value in the database
<i>settings</i>	the ParseSettings object holding all relevant information about the image.

**Returns**

the value of the given database key

### 9.285.3.5 ids()

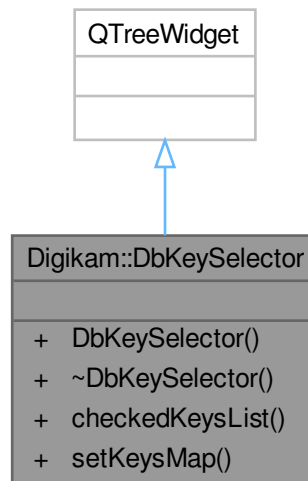
```
DbKeyIdMap Digikam::DbKeysCollection::ids ( ) const
```

#### Returns

a map of all associated ids and their description

## 9.286 Digikam::DbKeySelector Class Reference

Inheritance diagram for Digikam::DbKeySelector:

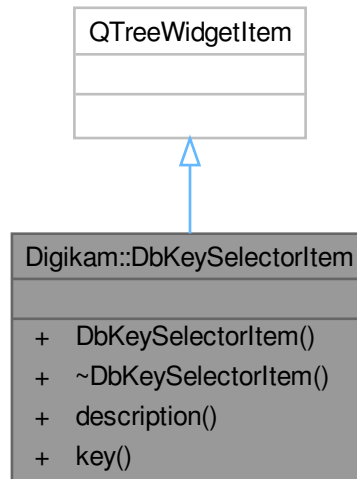


### Public Member Functions

- **DbKeySelector** (`QWidget *const parent`)
- `QStringList checkedKeysList` ()
- void **setKeysMap** (`const DbOptionKeysMap &map`)

## 9.287 Digikam::DbKeySelectorItem Class Reference

Inheritance diagram for Digikam::DbKeySelectorItem:

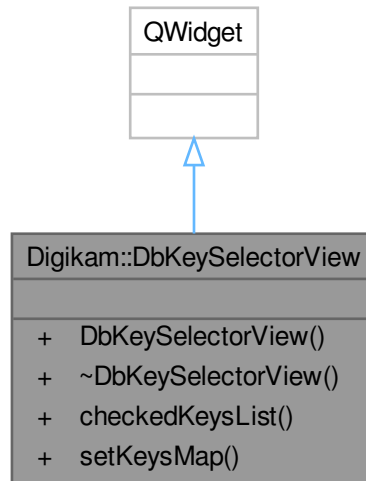


### Public Member Functions

- **DbKeySelectorItem** ([DbHeaderListItem](#) \*const parent, const QString &title, const QString &desc)
- QString **description** () const
- QString **key** () const

## 9.288 Digikam::DbKeySelectorView Class Reference

Inheritance diagram for Digikam::DbKeySelectorView:

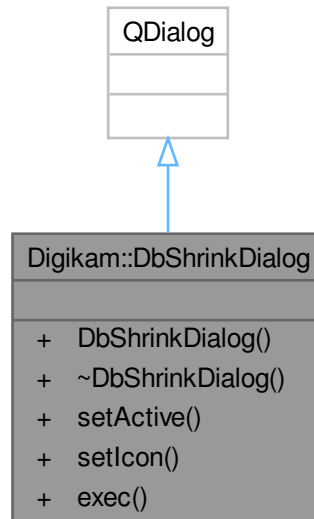


### Public Member Functions

- **DbKeySelectorView** (`QWidget *const parent`)
- `QStringList checkedKeysList () const`
- void **setKeysMap** (`const DbOptionKeysMap &map`)

## 9.289 Digikam::DbShrinkDialog Class Reference

Inheritance diagram for Digikam::DbShrinkDialog:



### Public Slots

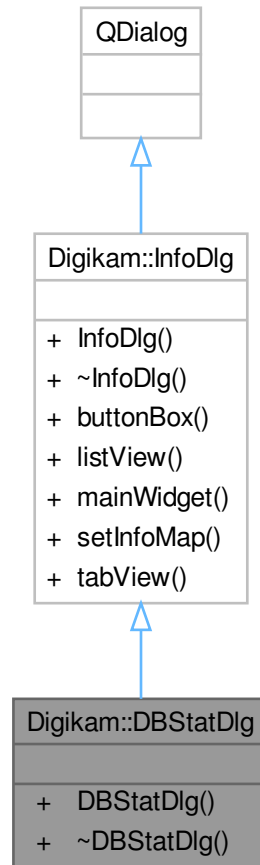
- int **exec** () override

### Public Member Functions

- **DbShrinkDialog** (QWidget \*const parent)
- void **setActive** (const int pos)
- void **setIcon** (const int pos, const QIcon &icon)

## 9.290 Digikam::DBStatDlg Class Reference

Inheritance diagram for Digikam::DBStatDlg:



### Public Member Functions

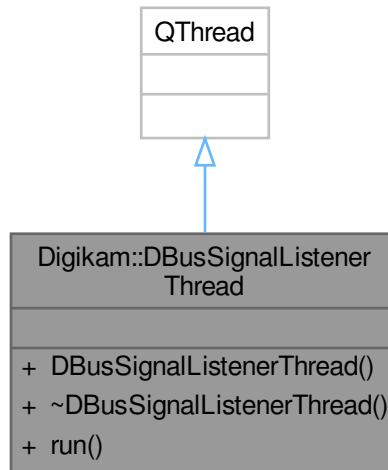
- **DBStatDlg** (QWidget \*const parent)

### Public Member Functions inherited from [Digikam::InfoDlg](#)

- **InfoDlg** (QWidget \*const parent)
- QDialogButtonBox \* **buttonBox** () const
- QTreeWidget \* **listView** () const
- QWidget \* **mainWidget** () const
- virtual void **setInfoMap** (const QMap< QString, QString > &list)
- QTabWidget \* **tableView** () const

## 9.291 Digikam::DBusSignalListenerThread Class Reference

Inheritance diagram for Digikam::DBusSignalListenerThread:



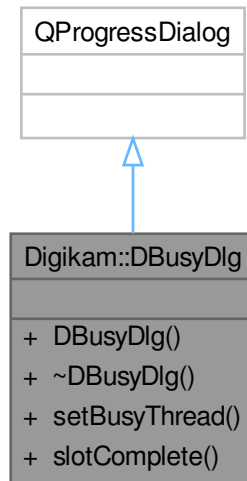
### Public Member Functions

- **DBusSignalListenerThread** ([CoreDbWatch](#) \*const qq, `CoreDbWatch::Private` \*const dd)
- void **run** () override



## 9.292 Digikam::DBusyDlg Class Reference

Inheritance diagram for Digikam::DBusyDlg:



### Public Slots

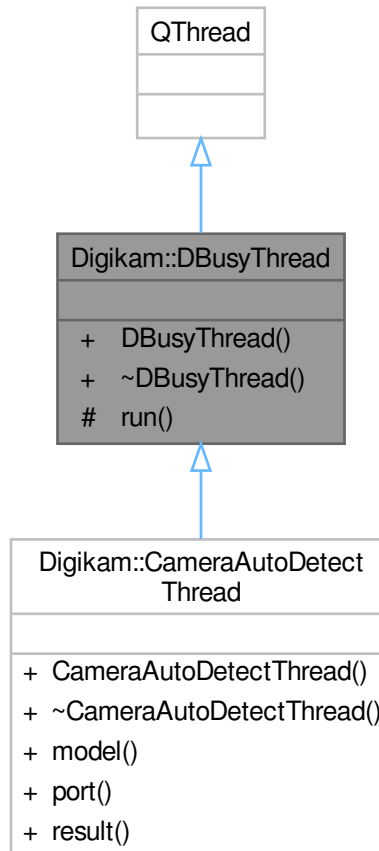
- void **slotComplete** ()

### Public Member Functions

- **DBusyDlg** (const QString &txt, QWidget \*const parent=nullptr)
- void **setBusyThread** ([DBusyThread](#) \*const thread)

## 9.293 Digikam::DBusyThread Class Reference

Inheritance diagram for Digikam::DBusyThread:



### Signals

- void `signalComplete ()`

### Public Member Functions

- `DBusyThread (QObject *const parent)`

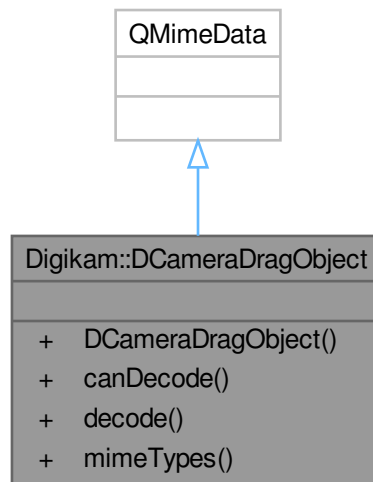
### Protected Member Functions

- void `run ()` override  
*Reimplement this method with your code to run in a separate thread.*

## 9.294 Digikam::DCameraDragObject Class Reference

Provides a drag object for a camera object.

Inheritance diagram for Digikam::DCameraDragObject:



### Public Member Functions

- `DCameraDragObject` (const [CameraType](#) &ctype)

### Static Public Member Functions

- static bool **canDecode** (const `QMimeData *e`)
- static bool **decode** (const `QMimeData *e`, [CameraType](#) &ctype)
- static `QStringList` **mimeTypes** ()

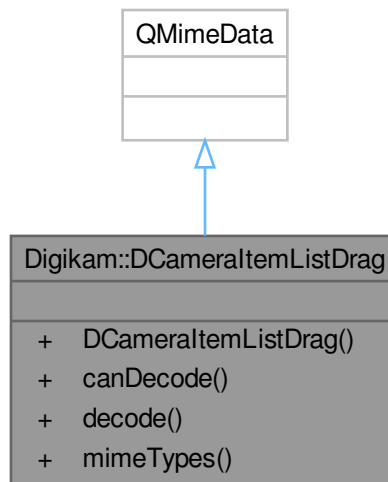
### 9.294.1 Detailed Description

When a camera object is moved through drag'n'drop an object of this class is created.

## 9.295 Digikam::DCameraltemListDrag Class Reference

Provides a drag object for a list of camera items.

Inheritance diagram for Digikam::DCameraltemListDrag:



### Public Member Functions

- **DCameraltemListDrag** (const QStringList &cameraltemPaths)

### Static Public Member Functions

- static bool **canDecode** (const QMimeData \*e)
- static bool **decode** (const QMimeData \*e, QStringList &cameraltemPaths)
- static QStringList **mimeTypes** ()

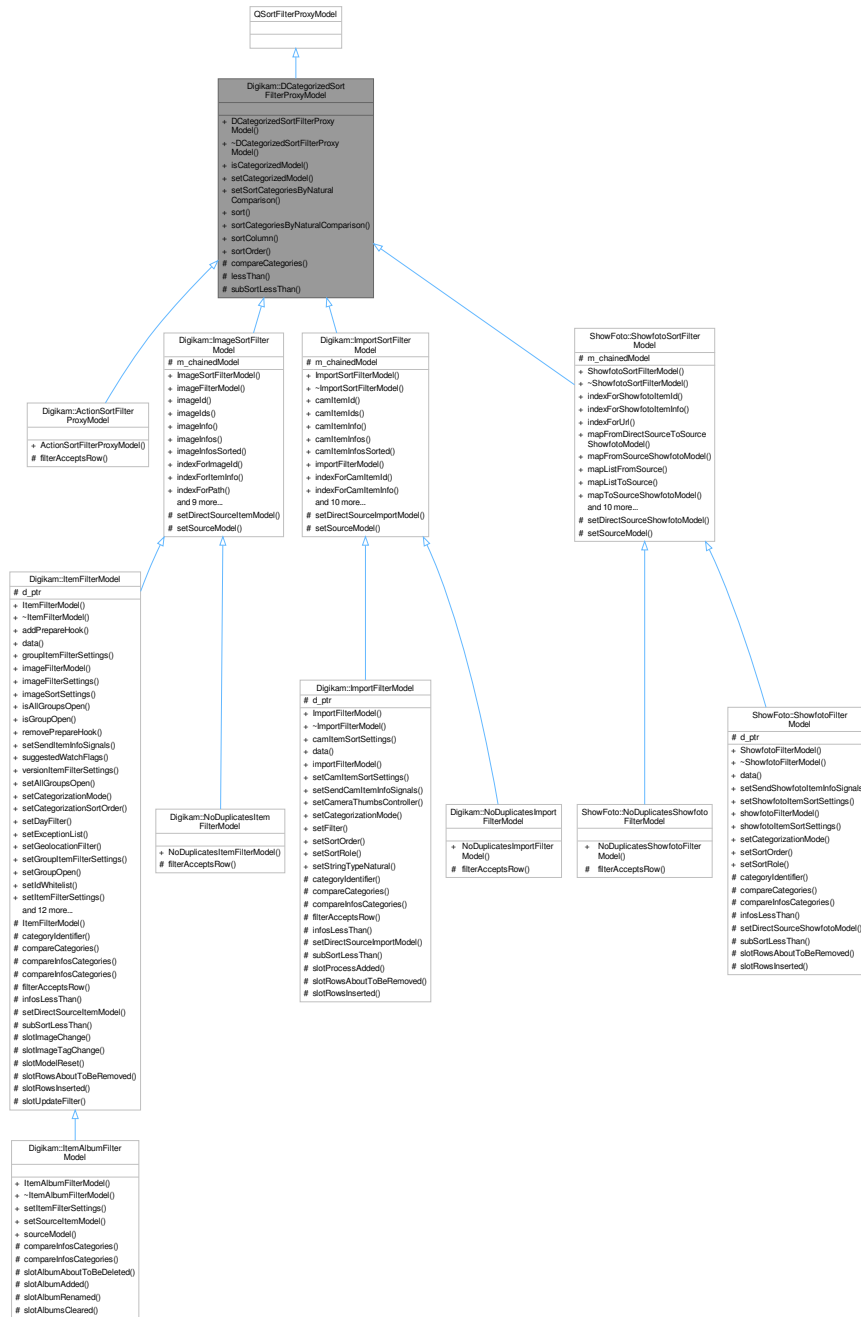
### 9.295.1 Detailed Description

When a camera item is moved through drag'n'drop an object of this class is created.

## 9.296 Digikam::DCategorizedSortFilterProxyModel Class Reference

This class lets you categorize a view.

Inheritance diagram for Digikam::DCategorizedSortFilterProxyModel:



### Public Types

- enum [AdditionalRoles](#) { [CategoryDisplayRole](#) = 0x17CE990A , [CategorySortRole](#) = 0x27857E60 }

## Public Member Functions

- **DCategorizedSortFilterProxyModel** (QObject \*const parent=nullptr)
- bool **isCategorizedModel** () const
- void **setCategorizedModel** (bool categorizedModel)
 

*Enables or disables the categorization feature.*
- void **setSortCategoriesByNaturalComparison** (bool **sortCategoriesByNaturalComparison**)
 

*Set if the sorting using CategorySortRole will use a natural comparison in the case that strings were returned.*
- void **sort** (int column, Qt::SortOrder order=Qt::AscendingOrder) override
 

*Overridden from QSortFilterProxyModel.*
- bool **sortCategoriesByNaturalComparison** () const
- int **sortColumn** () const
- Qt::SortOrder **sortOrder** () const

## Protected Member Functions

- virtual int **compareCategories** (const QModelIndex &left, const QModelIndex &right) const
 

*This method compares the category of the left index with the category of the right index.*
- bool **lessThan** (const QModelIndex &left, const QModelIndex &right) const override
 

*Overridden from QSortFilterProxyModel.*
- virtual bool **subSortLessThan** (const QModelIndex &left, const QModelIndex &right) const
 

*This method has a similar purpose as lessThan() has on QSortFilterProxyModel.*

### 9.296.1 Detailed Description

It is meant to be used along with [DCategorizedView](#) class.

In general terms all you need to do is to reimplement [subSortLessThan\(\)](#) and [compareCategories\(\)](#) methods. In order to make categorization work, you need to also call [setCategorizedModel\(\)](#) class to enable it, since the categorization is disabled by default.

### 9.296.2 Member Enumeration Documentation

#### 9.296.2.1 AdditionalRoles

```
enum Digikam::DCategorizedSortFilterProxyModel::AdditionalRoles
```

#### Enumerator

CategoryDisplayRole	This role is used for asking the category to a given index.  <b>Note</b> use printf "0x%08X\n" \$(((\$RANDOM*\$RANDOM)) to define additional roles.
CategorySortRole	This role is used for sorting categories. You can return a string or a long long value. Strings will be sorted alphabetically while long long will be sorted by their value. Please note that this value won't be shown on the view, is only for sorting purposes. What will be shown as "Category" on the view will be asked with the role CategoryDisplayRole.

## 9.296.3 Member Function Documentation

### 9.296.3.1 compareCategories()

```
int Digikam::DCategorizedSortFilterProxyModel::compareCategories (
    const QModelIndex & left,
    const QModelIndex & right ) const [protected], [virtual]
```

Internally and if not reimplemented, this method will ask for `left` and `right` models for role `CategorySortRole`. In order to correctly sort categories, the `data()` method of the model should return a `qulonglong` (or numeric) value, or a `QString` object. `QString` objects will be sorted with `QString::localeAwareCompare` if `sortCategoriesByNaturalComparison()` is true.

#### Note

Please have present that: `QString(QChar(QChar::ObjectReplacementCharacter)) > QString(QChar(QChar::ReplacementCharacter)) > [ all possible strings ] > QString();`

This means that `QString()` will be sorted the first one, while `QString(QChar(QChar::ObjectReplacementCharacter))` and `QString(QChar(QChar::ReplacementCharacter))` will be sorted in last position.

#### Warning

Please note that `data()` method of the model should return always information of the same type. If you return a `QString` for an index, you should return always `QStrings` for all indexes for role `CategorySortRole` in order to correctly sort categories. You can't mix by returning a `QString` for one index, and a `qulonglong` for other.

#### Note

If you need a more complex layout, you will have to reimplement this method.

#### Returns

A negative value if the category of `left` should be placed before the category of `right`. 0 if `left` and `right` are on the same category, and a positive value if the category of `left` should be placed after the category of `right`.

Reimplemented in [Digikam::ItemFilterModel](#), [ShowFoto::ShowfotoFilterModel](#), and [Digikam::ImportFilterModel](#).

### 9.296.3.2 isCategorizedModel()

```
bool Digikam::DCategorizedSortFilterProxyModel::isCategorizedModel ( ) const
```

#### Returns

whether the model is categorized or not. Disabled by default.

### 9.296.3.3 lessThan()

```
bool Digikam::DCategorizedSortFilterProxyModel::lessThan (
    const QModelIndex & left,
    const QModelIndex & right ) const [override], [protected]
```

If you are subclassing [DCategorizedSortFilterProxyModel](#), you will probably not need to reimplement this method.

It calls [compareCategories\(\)](#) to sort by category. If the both items are in the same category (i.e. [compareCategories](#) returns 0), then [subSortLessThan](#) is called.

#### Returns

Returns true if the item `left` is less than the item `right` when sorting.

#### Warning

You usually won't need to reimplement this method when subclassing from [DCategorizedSortFilterProxyModel](#).

### 9.296.3.4 setCategorizedModel()

```
void Digikam::DCategorizedSortFilterProxyModel::setCategorizedModel (
    bool categorizedModel )
```

#### Parameters

<i>categorizedModel</i>	whether to enable or disable the categorization feature.
-------------------------	--

### 9.296.3.5 setSortCategoriesByNaturalComparison()

```
void Digikam::DCategorizedSortFilterProxyModel::setSortCategoriesByNaturalComparison (
    bool sortCategoriesByNaturalComparison )
```

If enabled, `QCollator` will be used for sorting.

#### Parameters

<i>sortCategoriesByNaturalComparison</i>	whether to sort using a natural comparison or not.
--	--

### 9.296.3.6 sort()

```
void Digikam::DCategorizedSortFilterProxyModel::sort (
    int column,
    Qt::SortOrder order = Qt::AscendingOrder ) [override]
```

Sorts the source model using `column` for the given `order`.



### 9.296.3.7 sortCategoriesByNaturalComparison()

```
bool Digikam::DCategorizedSortFilterProxyModel::sortCategoriesByNaturalComparison ( ) const
```

#### Returns

whether it is being used a natural comparison for sorting. Enabled by default.

### 9.296.3.8 sortColumn()

```
int Digikam::DCategorizedSortFilterProxyModel::sortColumn ( ) const
```

#### Returns

the column being used for sorting.

### 9.296.3.9 sortOrder()

```
Qt::SortOrder Digikam::DCategorizedSortFilterProxyModel::sortOrder ( ) const
```

#### Returns

the sort order being used for sorting.

### 9.296.3.10 subSortLessThan()

```
bool Digikam::DCategorizedSortFilterProxyModel::subSortLessThan (
    const QModelIndex & left,
    const QModelIndex & right ) const [protected], [virtual]
```

It is used for sorting items that are in the same category.

#### Returns

Returns true if the item `left` is less than the item `right` when sorting.

Reimplemented in [Digikam::ItemFilterModel](#), [ShowFoto::ShowfotoFilterModel](#), and [Digikam::ImportFilterModel](#).



## Public Member Functions

- **DCategorizedView** (QWidget \*const parent=nullptr)
- virtual QModelIndexList **categorizedIndexesIn** (const QRect &rect) const  
*This method will return all indexes whose visual rect intersects rect.*
- virtual QModelIndex **categoryAt** (const QPoint &point) const  
*This method will return the first index of the category in the region of which point is found.*
- **DCategoryDrawer** \* **categoryDrawer** () const
- virtual QItemSelectionRange **categoryRange** (const QModelIndex &index) const  
*This method returns the range of indexes contained in the category in which index is sorted.*
- virtual QRect **categoryVisualRect** (const QModelIndex &index) const  
*This method will return the visual rect of the header of the category in which index is sorted.*
- QModelIndex **indexAt** (const QPoint &point) const override
- void **setCategoryDrawer** (DCategoryDrawer \*categoryDrawer)
- void **setDrawDraggedItems** (bool drawDraggedItems)  
*Switch on drawing of dragged items.*
- void **setGridSize** (const QSize &size)
- void **setModel** (QAbstractItemModel \*model) override
- QRect **visualRect** (const QModelIndex &index) const override

## Protected Slots

- void **currentChanged** (const QModelIndex &current, const QModelIndex &previous) override
- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- virtual void **rowsInsertedArtificial** (const QModelIndex &parent, int start, int end)
- virtual void **rowsRemoved** (const QModelIndex &parent, int start, int end)
- virtual void **slotLayoutChanged** ()
- void **updateGeometries** () override

## Protected Member Functions

- void **dragLeaveEvent** (QDragLeaveEvent \*event) override
- void **dragMoveEvent** (QDragMoveEvent \*event) override
- void **dropEvent** (QDropEvent \*event) override
- void **leaveEvent** (QEvent \*event) override
- void **mouseMoveEvent** (QMouseEvent \*event) override
- void **mousePressEvent** (QMouseEvent \*event) override
- void **mouseReleaseEvent** (QMouseEvent \*event) override
- QModelIndex **moveCursor** (CursorAction cursorAction, Qt::KeyboardModifiers modifiers) override
- void **paintEvent** (QPaintEvent \*event) override
- void **resizeEvent** (QResizeEvent \*event) override
- void **setSelection** (const QRect &rect, QItemSelectionModel::SelectionFlags flags) override
- void **startDrag** (Qt::DropActions supportedActions) override

### 9.297.1 Detailed Description

[DCategorizedView](#) allows you to use it as it were a [QListView](#). Subclass [DCategorizedSortFilterProxyModel](#) to provide category information for items.

### 9.297.2 Member Function Documentation

#### 9.297.2.1 categorizedIndexesIn()

```
QModelIndexList Digikam::DCategorizedView::categorizedIndexesIn (
    const QRect & rect ) const [virtual]
```

## Parameters

<i>rect</i>	rectangle to test intersection with
-------------	-------------------------------------

## Note

Returns an empty list if the view is not categorized.

**9.297.2.2 categoryAt()**

```
QModelIndex Digikam::DCategorizedView::categoryAt (
    const QPoint & point ) const [virtual]
```

## Note

Returns QModelIndex() if the view is not categorized.

**9.297.2.3 categoryRange()**

```
QItemSelectionRange Digikam::DCategorizedView::categoryRange (
    const QModelIndex & index ) const [virtual]
```

## Note

Returns an empty range if the view is no categorized.

**9.297.2.4 categoryVisualRect()**

```
QRect Digikam::DCategorizedView::categoryVisualRect (
    const QModelIndex & index ) const [virtual]
```

## Note

Returns QRect() if the view is not categorized.

**9.297.2.5 setDrawDraggedItems()**

```
void Digikam::DCategorizedView::setDrawDraggedItems (
    bool drawDraggedItems )
```

Default: on. While dragging over the view, dragged items will be drawn transparently following the mouse cursor.

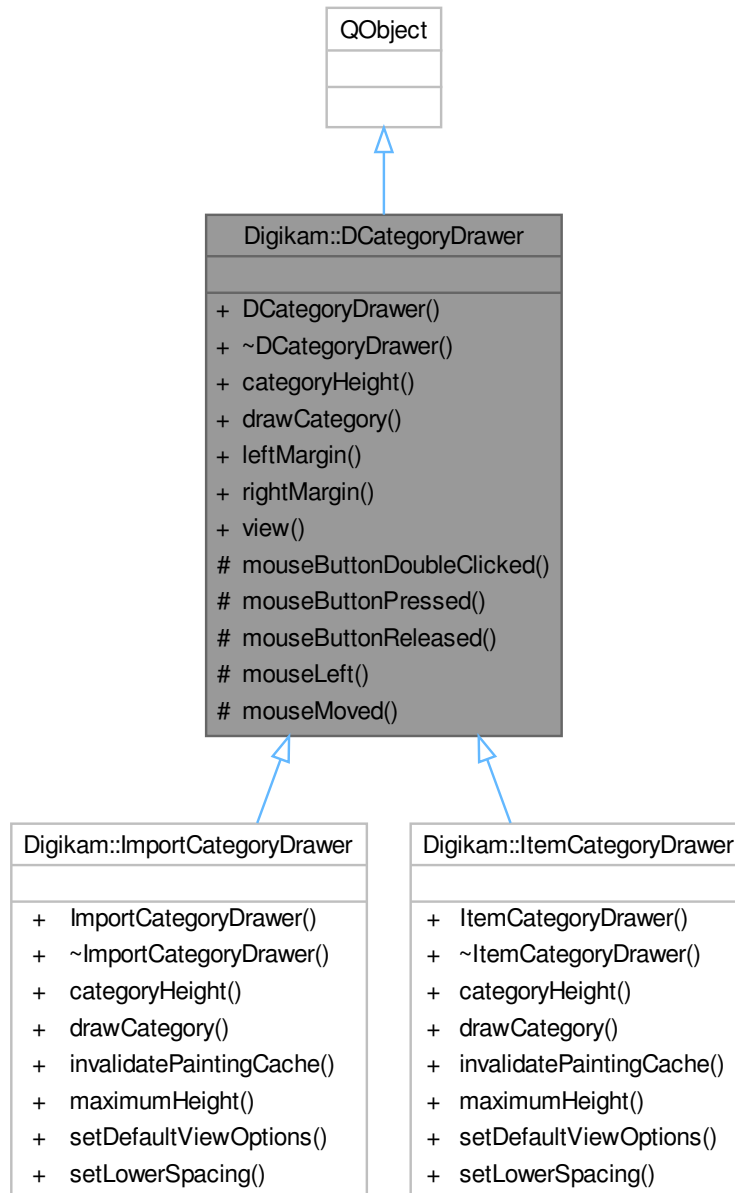
## Parameters

<i>drawDraggedItems</i>	if true, dragged items will be drawn
-------------------------	--------------------------------------

## 9.298 Digikam::DCategoryDrawer Class Reference

The category drawing is performed by this class.

Inheritance diagram for Digikam::DCategoryDrawer:



### Signals

- void `actionRequested` (int action, const `QModelIndex` &index)  
*Emit this signal on your subclass implementation to notify that something happened.*
- void `collapseOrExpandClicked` (const `QModelIndex` &index)  
*This signal becomes emitted when collapse or expand has been clicked.*

## Public Member Functions

- **DCategoryDrawer** ([DCategorizedView](#) \*const [view](#))  
*Construct a category drawer for a given view.*
- virtual int [categoryHeight](#) (const QModelIndex &index, const QStyleOption &option) const
- virtual void [drawCategory](#) (const QModelIndex &index, int sortRole, const QStyleOption &option, QPainter \*painter) const  
*This method purpose is to draw a category represented by the given.*
- virtual int [leftMargin](#) () const
- virtual int [rightMargin](#) () const
- [DCategorizedView](#) \* [view](#) () const

## Protected Member Functions

- virtual void [mouseButtonDoubleClicked](#) (const QModelIndex &index, const QRect &blockRect, QMouseEvent \*event)  
*Method called when the mouse button has been double clicked.*
- virtual void [mouseButtonPressed](#) (const QModelIndex &index, const QRect &blockRect, QMouseEvent \*event)  
*Method called when the mouse button has been pressed.*
- virtual void [mouseButtonReleased](#) (const QModelIndex &index, const QRect &blockRect, QMouseEvent \*event)  
*Method called when the mouse button has been released.*
- virtual void [mouseLeft](#) (const QModelIndex &index, const QRect &blockRect)  
*Method called when the mouse button has left this block.*
- virtual void [mouseMoved](#) (const QModelIndex &index, const QRect &blockRect, QMouseEvent \*event)  
*Method called when the mouse has been moved.*

## Friends

- class **DCategorizedView**

## 9.298.1 Detailed Description

It also gives information about the category height and margins.

## 9.298.2 Member Function Documentation

### 9.298.2.1 actionRequested

```
void Digikam::DCategoryDrawer::actionRequested (
    int action,
    const QModelIndex & index ) [signal]
```

Usually this will be triggered when you have received an event, and its position matched some "hot spot".

You give this action the integer you want, and having connected this signal to your code, the connected slot can perform the needed changes (view, model, selection model, delegate...)

### 9.298.2.2 categoryHeight()

```
int Digikam::DCategoryDrawer::categoryHeight (
    const QModelIndex & index,
    const QStyleOption & option ) const [virtual]
```

#### Returns

The category height for the category represented by index *index* with style options *option*.

Reimplemented in [Digikam::ItemCategoryDrawer](#), and [Digikam::ImportCategoryDrawer](#).

### 9.298.2.3 drawCategory()

```
void Digikam::DCategoryDrawer::drawCategory (
    const QModelIndex & index,
    int sortRole,
    const QStyleOption & option,
    QPainter * painter ) const [virtual]
```

#### Parameters

<i>index</i>	with the given
<i>sortRole</i>	sorting role
<i>option</i>	painter style options
<i>painter</i>	painter instance

#### Note

This method will be called one time per category, always with the first element in that category

Reimplemented in [Digikam::ItemCategoryDrawer](#), and [Digikam::ImportCategoryDrawer](#).

### 9.298.2.4 leftMargin()

```
int Digikam::DCategoryDrawer::leftMargin ( ) const [virtual]
```

#### Note

0 by default

### 9.298.2.5 mouseButtonDoubleClicked()

```
void Digikam::DCategoryDrawer::mouseButtonDoubleClicked (
    const QModelIndex & index,
    const QRect & blockRect,
    QMouseEvent * event ) [protected], [virtual]
```

**Parameters**

<i>index</i>	The representative index of the block of items.
<i>blockRect</i>	The rect occupied by the block of items.
<i>event</i>	The mouse event.

**Warning**

You explicitly have to determine whether the event has been accepted or not. You have to call `event->accept()` or `event->ignore()` at all possible case branches in your code.

**9.298.2.6 mouseButtonPressed()**

```
void Digikam::DCategoryDrawer::mouseButtonPressed (
    const QModelIndex & index,
    const QRect & blockRect,
    QMouseEvent * event ) [protected], [virtual]
```

**Parameters**

<i>index</i>	The representative index of the block of items.
<i>blockRect</i>	The rect occupied by the block of items.
<i>event</i>	The mouse event.

**Warning**

You explicitly have to determine whether the event has been accepted or not. You have to call `event->accept()` or `event->ignore()` at all possible case branches in your code.

**9.298.2.7 mouseButtonReleased()**

```
void Digikam::DCategoryDrawer::mouseButtonReleased (
    const QModelIndex & index,
    const QRect & blockRect,
    QMouseEvent * event ) [protected], [virtual]
```

**Parameters**

<i>index</i>	The representative index of the block of items.
<i>blockRect</i>	The rect occupied by the block of items.
<i>event</i>	The mouse event.

**Warning**

You explicitly have to determine whether the event has been accepted or not. You have to call `event->accept()` or `event->ignore()` at all possible case branches in your code.



**9.298.2.8 mouseLeft()**

```
void Digikam::DCategoryDrawer::mouseLeft (
    const QModelIndex & index,
    const QRect & blockRect ) [protected], [virtual]
```

**Parameters**

<i>index</i>	The representative index of the block of items.
<i>blockRect</i>	The rect occupied by the block of items.

**9.298.2.9 mouseMoved()**

```
void Digikam::DCategoryDrawer::mouseMoved (
    const QModelIndex & index,
    const QRect & blockRect,
    QMouseEvent * event ) [protected], [virtual]
```

**Parameters**

<i>index</i>	The representative index of the block of items.
<i>blockRect</i>	The rect occupied by the block of items.
<i>event</i>	The mouse event.

**9.298.2.10 rightMargin()**

```
int Digikam::DCategoryDrawer::rightMargin ( ) const [virtual]
```

**Note**

0 by default

**9.298.2.11 view()**

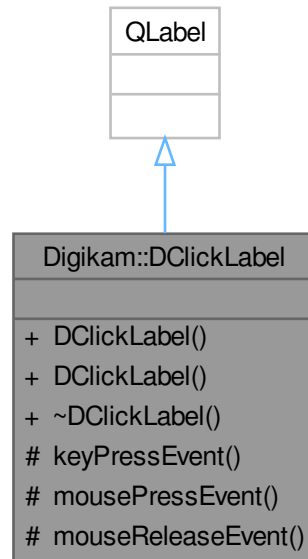
```
DategorizedView * Digikam::DCategoryDrawer::view ( ) const
```

**Returns**

The view this category drawer is associated with.

## 9.299 Digikam::DClickLabel Class Reference

Inheritance diagram for Digikam::DClickLabel:



### Signals

- void **activated** ()  
*Emitted when activated, by mouse or key press.*
- void **leftClicked** ()  
*Emitted when activated by left mouse click.*

### Public Member Functions

- **DClickLabel** (const QString &text, QWidget \*const parent=nullptr)
- **DClickLabel** (QWidget \*const parent=nullptr)

### Protected Member Functions

- void **keyPressEvent** (QKeyEvent \*event) override
- void **mousePressEvent** (QMouseEvent \*event) override
- void **mouseReleaseEvent** (QMouseEvent \*event) override

## 9.300 Digikam::DColor Class Reference

### Public Member Functions

- **DColor** ()=default  
*Initialize with default value, fully transparent eight bit black.*
- **DColor** (const QColor &color, bool sixteenBit=false)  
*Read values from QColor, convert to sixteenBit if sixteenBit is true.*
- **DColor** (int red, int green, int blue, int alpha, bool sixteenBit)  
*Initialize with given RGBA values.*
- **DColor** (uchar \*data, bool sixteenBit=false)  
*Read value from data.*
- int **alpha** () const
- void **blendAdd** (const DColor &src)
- void **blendAlpha16** (int alpha)
- void **blendAlpha8** (int alpha)
- void **blendClamp16** ()
- void **blendClamp8** ()
- void **blendInvAlpha16** (int alpha)
- void **blendInvAlpha8** (int alpha)
- void **blendZero** ()  
*Inline alpha blending helper functions.*
- int **blue** () const
- void **convertToEightBit** ()
- void **convertToSixteenBit** ()  
*Convert the color values of this color to and from sixteen bit and set the sixteenBit value accordingly.*
- void **demultiply** ()
- void **demultiply16** (int alpha)
- void **demultiply8** (int alpha)
- void **getHSL** (int \*const h, int \*const s, int \*const l) const  
*Return the current RGB color values of this color in the HSL color space.*
- QColor **getQColor** () const
- void **getYCbCr** (double \*const y, double \*const cb, double \*const cr) const  
*Return the current RGB color values of this color in the YCrCb color space.*
- int **green** () const
- bool **isPureGray** ()
- bool **isPureGrayValue** (int v)
- void **multiply** (float factor)
- void **premultiply** ()  
*Premultiply and demultiply this color.*
- void **premultiply16** (int alpha)
- void **premultiply8** (int alpha)
- int **red** () const
- void **setAlpha** (int alpha)
- void **setBlue** (int blue)
- void **setColor** (uchar \*const data, bool sixteenBit=false)  
*Read color values as RGBA from the given memory location.*
- void **setGreen** (int green)
- void **setHSL** (int h, int s, int l, bool sixteenBit)  
*Set the RGB color values of this color to the given HSL values converted to RGB.*
- void **setPixel** (uchar \*const data) const  
*Write the values of this color to the given memory location.*

- void **setRed** (int red)
- void **setSixteenBit** (bool sixteenBit)
- void **setYCbCr** (double y, double cb, double cr, bool sixteenBit)  
*Set the RGB color values of this color to the given YCrCb values converted to RGB.*
- bool **sixteenBit** () const

## 9.300.1 Constructor & Destructor Documentation

### 9.300.1.1 DColor()

```
Digikam::DColor::DColor (
    uchar * data,
    bool sixteenBit = false ) [inline], [explicit]
```

Equivalent to [setColor\(\)](#)

## 9.300.2 Member Function Documentation

### 9.300.2.1 blendZero()

```
void Digikam::DColor::blendZero ( ) [inline]
```

These functions are used by [DColorComposer](#). Look at that code to learn how to use them for composition if you want to use them in optimized code.

### 9.300.2.2 getHSL()

```
void Digikam::DColor::getHSL (
    int *const h,
    int *const s,
    int *const l ) const
```

Alpha is ignored for the conversion.

### 9.300.2.3 getYCbCr()

```
void Digikam::DColor::getYCbCr (
    double *const y,
    double *const cb,
    double *const cr ) const
```

Alpha is ignored for the conversion.

### 9.300.2.4 premultiply()

```
void Digikam::DColor::premultiply ( ) [inline]
```

[DImg](#) stores the color non-premultiplied. Inline methods.

### 9.300.2.5 setColor()

```
void Digikam::DColor::setColor (
    uchar *const data,
    bool sixteenBit = false ) [inline]
```

These methods are used in quite a few image effects, typically in loops iterating the data.

If sixteenBit is false, 4 bytes are read. If sixteenBit is true, 8 bytes are read. Inline method.

Providing them as inline methods allows the compiler to optimize better.

### 9.300.2.6 setHSL()

```
void Digikam::DColor::setHSL (
    int h,
    int s,
    int l,
    bool sixteenBit )
```

Alpha is set to be fully opaque. sixteenBit determines both how the HSL values are interpreted and the sixteenBit value of this color after this operation.

### 9.300.2.7 setPixel()

```
void Digikam::DColor::setPixel (
    uchar *const data ) const [inline]
```

If sixteenBit is false, 4 bytes are written. If sixteenBit is true, 8 bytes are written. Inline method.

### 9.300.2.8 setYCbCr()

```
void Digikam::DColor::setYCbCr (
    double y,
    double cb,
    double cr,
    bool sixteenBit )
```

Alpha is set to be fully opaque. sixteenBit determines both how the YCrCb values are interpreted and the sixteenBit value of this color after this operation.

## 9.301 Digikam::DColorComposer Class Reference

Inheritance diagram for Digikam::DColorComposer:



## Public Types

- enum [CompositingOperation](#) {  
**PorterDuffNone** , **PorterDuffClear** , **PorterDuffSrc** , **PorterDuffSrcOver** ,  
**PorterDuffDstOver** , **PorterDuffSrcIn** , **PorterDuffDstIn** , **PorterDuffSrcOut** ,  
**PorterDuffDstOut** , **PorterDuffSrcAtop** , **PorterDuffDstAtop** , **PorterDuffXor** }  
*The available rules to combine src and destination color.*
- enum **MultiplicationFlags** {  
**NoMultiplication** = 0x00 , **PremultiplySrc** = 0x01 , **PremultiplyDst** = 0x02 , **DemultiplyDst** = 0x04 ,  
**MultiplicationFlagsDImg** = PremultiplySrc | PremultiplyDst | DemultiplyDst , **MultiplicationFlags**↔  
**PremultipliedColorOnDImg** = PremultiplyDst | DemultiplyDst }

## Public Member Functions

- virtual void [compose](#) ([DColor](#) &dest, [DColor](#) &src)=0  
*Carry out the actual composition process.*
- virtual void [compose](#) ([DColor](#) &dest, [DColor](#) &src, MultiplicationFlags multiplicationFlags)  
*Compose the two colors by calling [compose\(dest, src\)](#).*

## Static Public Member Functions

- static [DColorComposer](#) \* [getComposer](#) ([CompositingOperation](#) rule)  
*Retrieve a [DColorComposer](#) object for one of the predefined rules.*

## 9.301.1 Member Enumeration Documentation

### 9.301.1.1 CompositingOperation

enum [Digikam::DColorComposer::CompositingOperation](#)

For the Porter-Duff rules, the formula is component = (source \* fs + destination \* fd) where fs, fd according to the following table with sa = source alpha, da = destination alpha:

None fs: sa fd: 1.0-sa Clear fs: 0.0 fd: 0.0 Src fs: 1.0 fd: 0.0 Src Over fs: 1.0 fd: 1.0-sa Dst Over fs: 1.0-da fd: 1.0  
 Src In fs: da fd: 0.0 Dst In fs: 0.0 fd: sa Src Out fs: 1.0-da fd: 0.0 Dst Out fs: 0.0 fd: 1.0-sa

Src Atop fs: da fd: 1.0-sa Dst Atop fs: 1.0-da fd: sa Xor fs: 1.0-da fd: 1.0-sa

None is the default, classical blending mode, a "Src over" simplification: Blend non-premultiplied RGBA data "src over" a fully opaque background. Src is the painter's algorithm. All other operations require premultiplied colors. The documentation of [java.awt.AlphaComposite](#) (Java 1.5) provides a good introduction and documentation on Porter Duff.

## 9.301.2 Member Function Documentation

### 9.301.2.1 compose() [1/2]

```
virtual void Digikam::DColorComposer::compose (
    DColor & dest,
    DColor & src ) [pure virtual]
```

Src and Dest are composed and the result is written to dest. No pre-/demultiplication is done by this method, use the other overloaded methods, which call this method, if you need pre- or demultiplication (you need it if any of the colors are read from or written to a [DImg](#)).

If you just pass the object to a [DImg](#) method, you do not need to call this. Call this function if you want to compose two colors. Implement this function if you create a custom [DColorComposer](#).

The bit depth of source and destination color must be identical.

**9.301.2.2 compose() [2/2]**

```
void Digikam::DColorComposer::compose (
    DColor & dest,
    DColor & src,
    DColorComposer::MultiplicationFlags multiplicationFlags ) [virtual]
```

Pre- and demultiplication operations are done as specified. For PorterDuff operations except PorterDuffNone, you need

- PremultiplySrc if src is not premultiplied (read from a [DImg](#))
- PremultiplyDst if dst is not premultiplied (read from a [DImg](#))
- DemultiplyDst if dst will be written to non-premultiplied data (a [DImg](#))

**9.301.2.3 getComposer()**

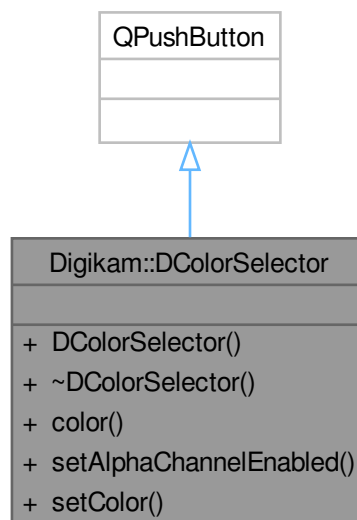
```
DColorComposer * Digikam::DColorComposer::getComposer (
    DColorComposer::CompositingOperation rule ) [static]
```

The object needs to be deleted by the caller.

**9.302 Digikam::DColorSelector Class Reference**

A widget to choose a color from a palette.

Inheritance diagram for Digikam::DColorSelector:



## Signals

- void **signalColorSelected** (const QColor &)

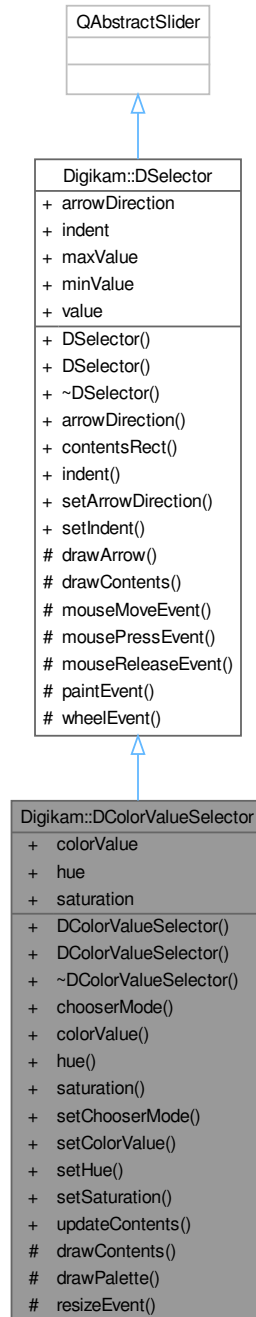
## Public Member Functions

- **DColorSelector** (QWidget \*const parent=nullptr)
- QColor **color** () const
- void **setAlphaChannelEnabled** (bool)
- void **setColor** (const QColor &color)



## 9.303 Digikam::DColorValueSelector Class Reference

Inheritance diagram for Digikam::DColorValueSelector:



### Public Member Functions

- **DColorValueSelector** (Qt::Orientation o, QWidget \*const parent=nullptr)
- **DColorValueSelector** (QWidget \*const parent=nullptr)

- DColorChooserMode [chooserMode](#) () const  
*Returns the current chooser mode.*
- int [colorValue](#) () const  
*Returns the current color value.*
- int [hue](#) () const  
*Returns the current hue value.*
- int [saturation](#) () const  
*Returns the current saturation value.*
- void [setChooserMode](#) (DColorChooserMode [chooserMode](#))  
*Sets the chooser mode.*
- void [setColorValue](#) (int colorValue)  
*Sets the color value.*
- void [setHue](#) (int hue)  
*Sets the hue value.*
- void [setSaturation](#) (int saturation)  
*Sets the saturation value.*
- void [updateContents](#) ()  
*Updates the widget's contents.*

## Public Member Functions inherited from [Digikam::DSelector](#)

- **DSelector** (Qt::Orientation o, QWidget \*const parent=nullptr)
- **DSelector** (QWidget \*const parent=nullptr)
- Qt::ArrowType [arrowDirection](#) () const
- QRect [contentsRect](#) () const
- bool [indent](#) () const
- void [setArrowDirection](#) (Qt::ArrowType direction)  
*Sets the arrow direction.*
- void [setIndent](#) (bool i)  
*Sets the indent option of the widget to i.*

## Protected Member Functions

- void [drawContents](#) (QPainter \*) override  
*Reimplemented from [DSelector](#).*
- virtual void [drawPalette](#) (QPixmap \*)  
*Draws the contents of the widget on a pixmap, which is used for buffering.*
- void [resizeEvent](#) (QResizeEvent \*) override

## Protected Member Functions inherited from [Digikam::DSelector](#)

- virtual void [drawArrow](#) (QPainter \*painter, const QPoint &pos)  
*Override this function to draw the cursor which indicates the current value.*
- void [mouseMoveEvent](#) (QMouseEvent \*e) override
- void [mousePressEvent](#) (QMouseEvent \*e) override
- void [mouseReleaseEvent](#) (QMouseEvent \*e) override
- void [paintEvent](#) (QPaintEvent \*) override
- void [wheelEvent](#) (QWheelEvent \*) override

## Properties

- int **colorValue**
- int **hue**
- int **saturation**

## Properties inherited from [Digikam::DSelector](#)

- Qt::ArrowType **arrowDirection**
- bool **indent**
- int **maxValue**
- int **minValue**
- int **value**

## Friends

- class **Private**

## 9.303.1 Member Function Documentation

### 9.303.1.1 chooserMode()

```
DColorChooserMode Digikam::DColorValueSelector::chooserMode ( ) const
```

#### Returns

The chooser mode (one of the DColorChooserMode constants)

### 9.303.1.2 colorValue()

```
int Digikam::DColorValueSelector::colorValue ( ) const
```

#### Returns

The color value (0-255)

### 9.303.1.3 drawContents()

```
void Digikam::DColorValueSelector::drawContents (
    QPainter * painter ) [override], [protected], [virtual]
```

The drawing is buffered in a pixmap here. As real drawing routine, [drawPalette\(\)](#) is used.

Reimplemented from [Digikam::DSelector](#).

#### 9.303.1.4 hue()

```
int Digikam::DColorValueSelector::hue ( ) const
```

##### Returns

The hue value (0-359)

#### 9.303.1.5 saturation()

```
int Digikam::DColorValueSelector::saturation ( ) const
```

##### Returns

The saturation value (0-255)

#### 9.303.1.6 setChooserMode()

```
void Digikam::DColorValueSelector::setChooserMode (
    DColorChooserMode chooserMode )
```

Doesn't automatically update the widget; you have to call `updateContents` manually.

##### Parameters

<i>chooserMode</i>	Sets the chooser mode (one of the <code>DColorChooserMode</code> constants)
--------------------	---

#### 9.303.1.7 setColorValue()

```
void Digikam::DColorValueSelector::setColorValue (
    int colorValue )
```

Doesn't automatically update the widget; you have to call `updateContents` manually.

##### Parameters

<i>colorValue</i>	Sets the color value (0-255)
-------------------	------------------------------

#### 9.303.1.8 setHue()

```
void Digikam::DColorValueSelector::setHue (
    int hue )
```

Doesn't automatically update the widget; you have to call `updateContents` manually.

## Parameters

<i>hue</i>	Sets the hue value (0-359)
------------	----------------------------

**9.303.1.9 setSaturation()**

```
void Digikam::DColorValueSelector::setSaturation (
    int saturation )
```

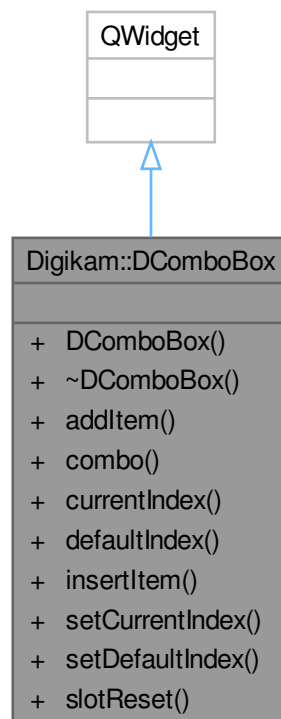
Doesn't automatically update the widget; you have to call `updateContents` manually.

## Parameters

<i>saturation</i>	Sets the saturation value (0-255)
-------------------	-----------------------------------

**9.304 Digikam::DComboBox Class Reference**

Inheritance diagram for Digikam::DComboBox:



### Public Slots

- void **slotReset** ()

### Signals

- void **activated** (int)
- void **currentIndexChanged** (int)
- void **reset** ()

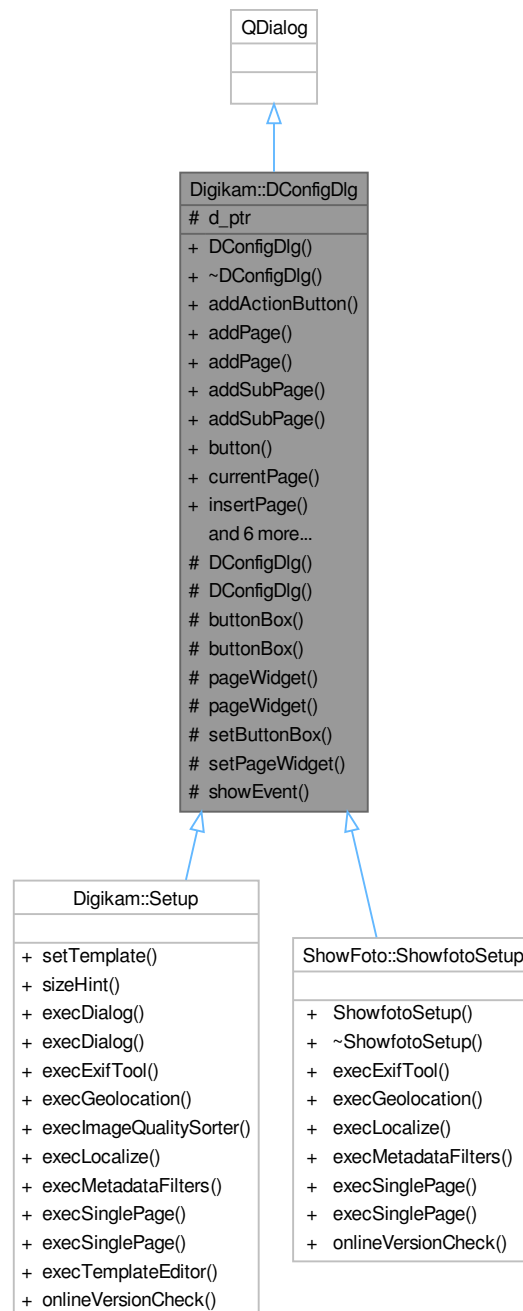
### Public Member Functions

- **DComboBox** (QWidget \*const parent=nullptr)
- void **addItem** (const QString &t, const QVariant &data=QVariant())
- QComboBox \* **combo** () const
- int **currentIndex** () const
- int **defaultIndex** () const
- void **insertItem** (int index, const QString &t, const QVariant &data=QVariant())
- void **setCurrentIndex** (int d)
- void **setDefaultIndex** (int d)

## 9.305 Digikam::DConfigDlg Class Reference

A dialog base class which can handle multiple pages.

Inheritance diagram for Digikam::DConfigDlg:



## Public Types

- enum `FaceType` {
  - `Auto` = `DConfigDlgView::Auto` , `Plain` = `DConfigDlgView::Plain` , `List` = `DConfigDlgView::List` , `Tree` = `DConfigDlgView::Tree` ,
  - `Tabbed` = `DConfigDlgView::Tabbed` }

## Signals

- void [currentPageChanged](#) ([DConfigDlgWdgItem](#) \*current, [DConfigDlgWdgItem](#) \*before)  
*This signal is emitted whenever the current page has changed.*
- void [pageRemoved](#) ([DConfigDlgWdgItem](#) \*page)  
*This signal is emitted whenever a page has been removed.*

## Public Member Functions

- **DConfigDlg** (QWidget \*const parent=nullptr, Qt::WindowFlags flags=Qt::WindowFlags())  
*Creates a new page dialog.*
- **~DConfigDlg** () override  
*Destroys the page dialog.*
- void **addActionButton** (QAbstractButton \*const button)  
*Set an action button.*
- void **addPage** ([DConfigDlgWdgItem](#) \*const item)  
*Adds a new top level page to the dialog.*
- [DConfigDlgWdgItem](#) \* **addPage** (QWidget \*const widget, const QString &name)  
*Adds a new top level page to the dialog.*
- void **addSubPage** ([DConfigDlgWdgItem](#) \*const parent, [DConfigDlgWdgItem](#) \*const item)  
*Inserts a new sub page in the dialog.*
- [DConfigDlgWdgItem](#) \* **addSubPage** ([DConfigDlgWdgItem](#) \*const parent, QWidget \*const widget, const QString &name)  
*Inserts a new sub page in the dialog.*
- QPushButton \* **button** (QDialogButtonBox::StandardButton which) const  
*Returns the QPushButton corresponding to the standard button which, or 0 if the standard button doesn't exist in this dialog.*
- [DConfigDlgWdgItem](#) \* **currentPage** () const  
*Returns the.*
- void **insertPage** ([DConfigDlgWdgItem](#) \*const before, [DConfigDlgWdgItem](#) \*const item)  
*Inserts a new page in the dialog.*
- [DConfigDlgWdgItem](#) \* **insertPage** ([DConfigDlgWdgItem](#) \*const before, QWidget \*const widget, const QString &name)  
*Inserts a new page in the dialog.*
- void **removePage** ([DConfigDlgWdgItem](#) \*const item)  
*Removes the page associated with the given.*
- void **setConfigGroup** (const QString &group)  
*Sets the config group name for restore or save dialog window size.*
- void **setCurrentPage** ([DConfigDlgWdgItem](#) \*const item)  
*Sets the page which is associated with the given.*
- void **setFaceType** ([FaceType](#) faceType)  
*Sets the face type of the dialog.*
- void **setStandardButtons** (QDialogButtonBox::StandardButtons buttons)  
*Sets the collection of standard buttons displayed by this dialog.*



## Protected Member Functions

- **DConfigDlg** (DConfigDlgPrivate &&, [DConfigDlgWdg](#) \*const widget, QWidget \*const parent, Qt::WindowFlags flags=Qt::WindowFlags())
- [DConfigDlg](#) ([DConfigDlgWdg](#) \*const widget, QWidget \*const parent, Qt::WindowFlags flags=Qt::WindowFlags())  
*This constructor can be used by subclasses to provide a custom page widget.*
- QDialogButtonBox \* **buttonBox** ()  
*Returns the button box of the dialog or 0 if no button box is set.*
- const QDialogButtonBox \* **buttonBox** () const  
*Returns the button box of the dialog or 0 if no button box is set.*
- [DConfigDlgWdg](#) \* **pageWidget** ()  
*Returns the page widget of the dialog or 0 if no page widget is set.*
- const [DConfigDlgWdg](#) \* **pageWidget** () const  
*Returns the page widget of the dialog or 0 if no page widget is set.*
- void [setButtonBox](#) (QDialogButtonBox \*const box)  
*Set the button box of the dialog.*
- void [setPageWidget](#) ([DConfigDlgWdg](#) \*const widget)  
*Set the page widget of the dialog.*
- void **showEvent** (QShowEvent \*) override

## Protected Attributes

- DConfigDlgPrivate \*const **d\_ptr** = nullptr

### 9.305.1 Detailed Description

This class provides a dialog base class which handles multiple pages and allows the user to switch between these pages in different ways.

Currently, `Auto`, `Plain`, `List`, `Tree` and `Tabbed` face types are available (

See also

[DConfigDlgView](#)).

### 9.305.2 Member Enumeration Documentation

#### 9.305.2.1 FaceType

enum `Digikam::DConfigDlg::FaceType`

- `Auto` - A dialog with a face based on the structure of the available pages. If only a single page is added, the dialog behaves like in `Plain` mode, with multiple pages without sub pages it behaves like in `List` mode and like in `Tree` mode otherwise.
- `Plain` - A normal dialog.
- `List` - A dialog with an icon list on the left side and a representation of the contents on the right side.
- `Tree` - A dialog with a tree on the left side and a representation of the contents on the right side.
- `Tabbed` - A dialog with a tab bar above the representation of the contents.

### 9.305.3 Constructor & Destructor Documentation

#### 9.305.3.1 DConfigDlg()

```
Digikam::DConfigDlg::DConfigDlg (
    DConfigDlgWdg *const widget,
    QWidget *const parent,
    Qt::WindowFlags flags = Qt::WindowFlags() ) [protected]
```

##### Parameters

<i>widget</i>	The <a href="#">DConfigDlgWdg</a> object will be reparented to this object, so you can create it without parent and you are not allowed to delete it.
<i>parent</i>	The widget parent instance
<i>flags</i>	The window flags

### 9.305.4 Member Function Documentation

#### 9.305.4.1 addPage() [1/2]

```
void Digikam::DConfigDlg::addPage (
    DConfigDlgWdgItem *const item )
```

##### Parameters

<i>item</i>	The
-------------	-----

##### See also

[DConfigDlgWdgItem](#) which describes the page.

#### 9.305.4.2 addPage() [2/2]

```
DConfigDlgWdgItem * Digikam::DConfigDlg::addPage (
    QWidget *const widget,
    const QString & name )
```

##### Parameters

<i>widget</i>	The widget of the page.
<i>name</i>	The name which is displayed in the navigation view.

##### Returns

The associated

See also

[DConfigDlgWdgItem](#).

#### 9.305.4.3 addSubPage() [1/2]

```
void Digikam::DConfigDlg::addSubPage (
    DConfigDlgWdgItem *const parent,
    DConfigDlgWdgItem *const item )
```

Parameters

<i>parent</i>	The new page will be insert as child of this
---------------	--

See also

[DConfigDlgWdgItem](#).

Parameters

<i>item</i>	The
-------------	-----

See also

[DConfigDlgWdgItem](#) which describes the page.

#### 9.305.4.4 addSubPage() [2/2]

```
DConfigDlgWdgItem * Digikam::DConfigDlg::addSubPage (
    DConfigDlgWdgItem *const parent,
    QWidget *const widget,
    const QString & name )
```

Parameters

<i>parent</i>	The new page will be insert as child of this
---------------	--

See also

[DConfigDlgWdgItem](#).

Parameters

<i>widget</i>	The widget of the page.
<i>name</i>	The name which is displayed in the navigation view.

**Returns**

The associated

**See also**

[DConfigDlgWdgItem](#).

**9.305.4.5 currentPage()**

```
DConfigDlgWdgItem * Digikam::DConfigDlg::currentPage ( ) const
```

**See also**

[DConfigDlgWdgItem](#) for the current page or 0 if there is no current page.

**9.305.4.6 currentPageChanged**

```
void Digikam::DConfigDlg::currentPageChanged (
    DConfigDlgWdgItem * current,
    DConfigDlgWdgItem * before ) [signal]
```

**Parameters**

<i>current</i>	The new current page or 0 if no current page is available.
----------------	--

**9.305.4.7 insertPage() [1/2]**

```
void Digikam::DConfigDlg::insertPage (
    DConfigDlgWdgItem *const before,
    DConfigDlgWdgItem *const item )
```

**Parameters**

<i>before</i>	The new page will be insert before this
---------------	---

**See also**

[DConfigDlgWdgItem](#) on the same level in hierarchy.

**Parameters**

<i>item</i>	The
-------------	-----

See also

[DConfigDlgWdgItem](#) which describes the page.

#### 9.305.4.8 insertPage() [2/2]

```
DConfigDlgWdgItem * Digikam::DConfigDlg::insertPage (
    DConfigDlgWdgItem *const before,
    QWidget *const widget,
    const QString & name )
```

Parameters

<i>before</i>	The new page will be insert before this
---------------	---

See also

[DConfigDlgWdgItem](#) on the same level in hierarchy.

Parameters

<i>widget</i>	The widget of the page.
<i>name</i>	The name which is displayed in the navigation view.

Returns

The associated

See also

[DConfigDlgWdgItem](#).

#### 9.305.4.9 pageRemoved

```
void Digikam::DConfigDlg::pageRemoved (
    DConfigDlgWdgItem * page ) [signal]
```

Parameters

<i>page</i>	The page which has been removed
-------------	---------------------------------

#### 9.305.4.10 removePage()

```
void Digikam::DConfigDlg::removePage (
    DConfigDlgWdgItem *const item )
```

See also

[DConfigDlgWdgItem](#).

#### 9.305.4.11 `setButtonBox()`

```
void Digikam::DConfigDlg::setButtonBox (
    QDialogButtonBox *const box ) [protected]
```

Note

the previous `buttonBox` will be deleted.

Parameters

<i>box</i>	The <code>QDialogButtonBox</code> object will be reparented to this object, so you can create it without parent and you are not allowed to delete it.
------------	---

#### 9.305.4.12 `setCurrentPage()`

```
void Digikam::DConfigDlg::setCurrentPage (
    DConfigDlgWdgItem *const item )
```

See also

[DConfigDlgWdgItem](#) to be the current page and emits the `currentPageChanged()` signal.

#### 9.305.4.13 `setPageWidget()`

```
void Digikam::DConfigDlg::setPageWidget (
    DConfigDlgWdg *const widget ) [protected]
```

Note

the previous `pageWidget` will be deleted.

Parameters

<i>widget</i>	The <a href="#">DConfigDlgWdg</a> object will be reparented to this object, so you can create it without parent and you are not allowed to delete it.
---------------	---

## 9.306 `Digikam::DConfigDlgMngr` Class Reference

The [DConfigDlgMngr](#) class provides a means of automatically retrieving, saving and resetting basic settings.

Inheritance diagram for Digikam::DConfigDlgMngr:



### Public Slots

- void [updateSettings](#) ()  
*Traverse the specified widgets, saving the settings of all known widgets in the settings object.*
- void [updateWidgets](#) ()  
*Traverse the specified widgets, sets the state of all known widgets according to the state in the settings object.*
- void [updateWidgetsDefault](#) ()  
*Traverse the specified widgets, sets the state of all known widgets according to the default state in the settings object.*

## Signals

- void [settingsChanged](#) ()  
*One or more of the settings have been saved (such as when the user clicks on the Apply button).*
- void [settingsChanged](#) (QWidget \*widget)  
*One or more of the settings have been changed.*
- void [widgetModified](#) ()  
*If retrieveSettings() was told to track changes then if any known setting was changed this signal will be emitted.*

## Public Member Functions

- [DConfigDlgMgr](#) (QWidget \*const parent, KConfigSkeleton \*const conf)  
*Constructor.*
- [~DConfigDlgMgr](#) () override  
*Destructor.*
- void [addWidget](#) (QWidget \*const widget)  
*Add additional widgets to manage.*
- bool [hasChanged](#) () const  
*Returns whether the current state of the known widgets are different from the state in the config object.*
- bool [isDefault](#) () const  
*Returns whether the current state of the known widgets are the same as the default state in the config object.*

## Static Public Member Functions

- static QHash< QString, QByteArray > \* [changedMap](#) ()  
*Retrieve the map between widgets class names and signals that are listened to detect changes in the configuration values.*
- static QHash< QString, QByteArray > \* [propertyMap](#) ()  
*Retrieve the map between widgets class names and the USER properties used for the configuration values.*

## Protected Member Functions

- QByteArray [getCustomProperty](#) (const QWidget \*widget) const  
*Find the property to use for a widget by querying the "kcfg\_property" property of the widget.*
- QByteArray [getCustomPropertyChangedSignal](#) (const QWidget \*widget) const  
*Find the changed signal of the property to use for a widget by querying the "kcfg\_propertyNotify" property of the widget.*
- QByteArray [getUserProperty](#) (const QWidget \*widget) const  
*Finds the USER property name using Qt's MetaProperty system, and caches it in the property map (the cache could be retrieved by [propertyMap\(\)](#)).*
- QByteArray [getUserPropertyChangedSignal](#) (const QWidget \*widget) const  
*Finds the changed signal of the USER property using Qt's MetaProperty system.*
- void [init](#) (bool trackChanges)
- bool [parseChildren](#) (const QWidget \*widget, bool trackChanges)  
*Recursive function that finds all known children.*
- QVariant [property](#) (QWidget \*w) const  
*Retrieve a property.*
- void [setProperty](#) (QWidget \*w, const QVariant &v)  
*Set a property.*
- void [setupWidget](#) (QWidget \*widget, KConfigSkeletonItem \*item)  
*Setup secondary widget properties.*



## Static Protected Member Functions

- static void **initMaps** ()  
*Initializes the property maps.*

### 9.306.1 Detailed Description

It also can emit signals when settings have been changed (settings were saved) or modified (the user changes a checkbox from on to off).

The object names of the widgets to be managed have to correspond to the names of the configuration entries in the KConfigSkeleton object plus an additional "kcfg\_" prefix. For example a widget with the object name "kcfg\_↔MyOption" would be associated to the configuration entry "MyOption".

The widget classes of Qt are supported out of the box.

Custom widget classes are supported if they have a Q\_PROPERTY defined for the property representing the value edited by the widget. By default the property is used for which "USER true" is set. For using another property, see below.

### 9.306.2 Constructor & Destructor Documentation

#### 9.306.2.1 DConfigDlgMngr()

```
Digikam::DConfigDlgMngr::DConfigDlgMngr (
    QWidget *const parent,
    KConfigSkeleton *const conf )
```

##### Parameters

<i>parent</i>	Dialog widget to manage
<i>conf</i>	Object that contains settings

### 9.306.3 Member Function Documentation

#### 9.306.3.1 addWidget()

```
void Digikam::DConfigDlgMngr::addWidget (
    QWidget *const widget )
```

##### Parameters

<i>widget</i>	Additional widget to manage, including all its children
---------------	---

#### 9.306.3.2 getCustomProperty()

```
QByteArray Digikam::DConfigDlgMngr::getCustomProperty (
```

```
const QWidget * widget ) const [protected]
```

Like a widget can use a property other than the USER property.

### 9.306.3.3 getCustomPropertyChangedSignal()

```
QByteArray Digikam::DConfigDlgMngr::getCustomPropertyChangedSignal (
    const QWidget * widget ) const [protected]
```

Like a widget can use a property change signal other than the one for USER property, if there even is one.

### 9.306.3.4 init()

```
void Digikam::DConfigDlgMngr::init (
    bool trackChanges ) [protected]
```

#### Parameters

<i>trackChanges</i>	- If any changes by the widgets should be tracked set true. This causes the emitting the modified() signal when something changes.
---------------------	--

### 9.306.3.5 parseChildren()

```
bool Digikam::DConfigDlgMngr::parseChildren (
    const QWidget * widget,
    bool trackChanges ) [protected]
```

Goes through the children of widget and if any are known and not being ignored, stores them in currentGroup. Also checks if the widget should be disabled because it is set immutable.

#### Parameters

<i>widget</i>	- Parent of the children to look at.
<i>trackChanges</i>	- If true then tracks any changes to the children of widget that are known.

#### Returns

bool - If a widget was set to something other than its default.

### 9.306.3.6 settingsChanged [1/2]

```
void Digikam::DConfigDlgMngr::settingsChanged ( ) [signal]
```

This is only emitted by [updateSettings\(\)](#) whenever one or more setting were changed and consequently saved.

### 9.306.3.7 settingsChanged [2/2]

```
void Digikam::DConfigDlgMngr::settingsChanged (
    QWidget * widget ) [signal]
```

## Parameters

<i>widget</i>	- The widget group (pass in via <a href="#">addWidget()</a> ) that contains the one or more modified setting.
---------------	---

## See also

[settingsChanged\(\)](#)**9.306.3.8 updateSettings**

```
void Digikam::DConfigDlgMngr::updateSettings ( ) [slot]
```

Example use: User clicks Ok or Apply button in a configure dialog.

**9.306.3.9 updateWidgets**

```
void Digikam::DConfigDlgMngr::updateWidgets ( ) [slot]
```

Example use: Initialisation of dialog. Example use: User clicks Reset button in a configure dialog.

**9.306.3.10 updateWidgetsDefault**

```
void Digikam::DConfigDlgMngr::updateWidgetsDefault ( ) [slot]
```

Example use: User clicks Defaults button in a configure dialog.

**9.306.3.11 widgetModified**

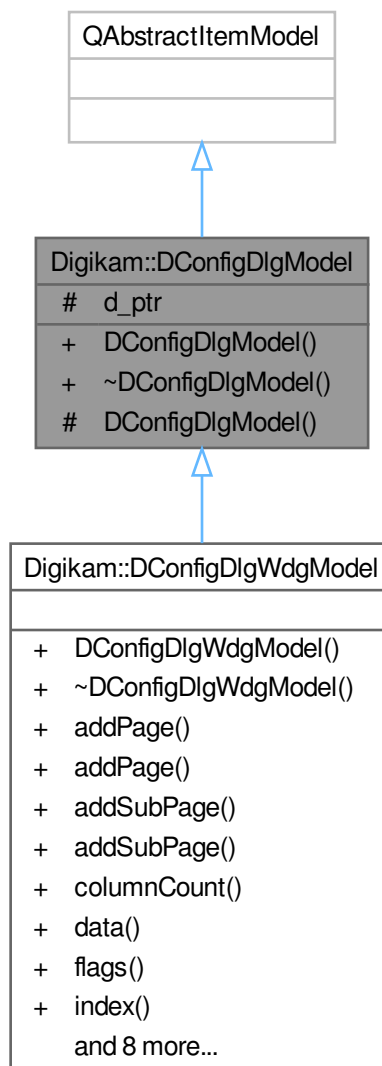
```
void Digikam::DConfigDlgMngr::widgetModified ( ) [signal]
```

Note that a settings can be modified several times and might go back to the original saved state. [hasChanged\(\)](#) will tell you if anything has actually changed from the saved values.

## 9.307 Digikam::DConfigDlgModel Class Reference

A base class for a model used by [DConfigDlgView](#).

Inheritance diagram for Digikam::DConfigDlgModel:



### Public Types

- enum `Role` { `HeaderRole` = `Qt::UserRole + 1` , `WidgetRole` }

*Additional roles that `DConfigDlgView` uses.*

## Public Member Functions

- **DConfigDlgModel** (QObject \*const parent=nullptr)  
*Constructs a page model with the given parent.*
- **~DConfigDlgModel** () override  
*Destroys the page model.*

## Protected Member Functions

- **DConfigDlgModel** (DConfigDlgModelPrivate &dd, QObject \*const parent)

## Protected Attributes

- DConfigDlgModelPrivate \*const **d\_ptr**

### 9.307.1 Detailed Description

This class is an abstract base class which must be used to implement custom models for [DConfigDlgView](#). Additional to the standard `Qt::ItemDataRoles` it provides the two roles

- HeaderRole
- WidgetRole

which are used to return a header string for a page and a `QWidget` pointer to the page itself.

### 9.307.2 Member Enumeration Documentation

#### 9.307.2.1 Role

```
enum Digikam::DConfigDlgModel::Role
```

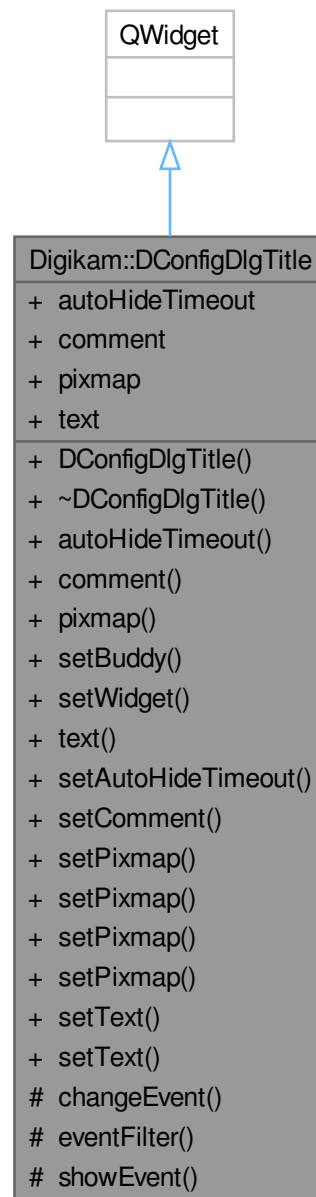
#### Enumerator

HeaderRole	A string to be rendered as page header.
WidgetRole	A pointer to the page widget. This is the widget that is shown when the item is selected.

## 9.308 Digikam::DConfigDlgTitle Class Reference

This class provides a widget often used for [DConfigDlg](#) titles.

Inheritance diagram for Digikam::DConfigDlgTitle:



## Public Types

- enum `ImageAlignment` { `ImageLeft` , `ImageRight` }  
*Possible title pixmap alignments.*
- enum `MessageType` { `PlainMessage` , `InfoMessage` , `WarningMessage` , `ErrorMessage` }  
*Comment message types.*

## Public Slots

- void [setAutoHideTimeout](#) (int msec)
  - Set the autohide timeout of the label Set value to 0 to disable autohide, which is the default.*
- void [setComment](#) (const QString &comment, [MessageType](#) type=[PlainText](#))
- void [setPixmap](#) (const QIcon &icon, [ImageAlignment](#) alignment=[ImageRight](#))
- void [setPixmap](#) (const QPixmap &pixmap, [ImageAlignment](#) alignment=[ImageRight](#))
- void [setPixmap](#) (const QString &icon, [ImageAlignment](#) alignment=[ImageRight](#))
- void [setPixmap](#) ([MessageType](#) type, [ImageAlignment](#) alignment=[ImageRight](#))
- void [setText](#) (const QString &text, [MessageType](#) type)
- void [setText](#) (const QString &text, Qt::Alignment alignment=[Qt::AlignLeft|Qt::AlignVCenter](#))

## Public Member Functions

- [DConfigDlgTitle](#) (QWidget \*const parent=nullptr)
  - Constructs a title widget with the given.*
- int [autoHideTimeout](#) () const
  - Get the current timeout value in milliseconds.*
- QString [comment](#) () const
- QPixmap [pixmap](#) () const
- void [setBuddy](#) (QWidget \*const buddy)
  - Sets this label's buddy to buddy.*
- void [setWidget](#) (QWidget \*const widget)
- QString [text](#) () const

## Protected Member Functions

- void [changeEvent](#) (QEvent \*) override
- bool [eventFilter](#) (QObject \*, QEvent \*) override
- void [showEvent](#) (QShowEvent \*) override

## Properties

- int [autoHideTimeout](#)
- QString [comment](#)
- QPixmap [pixmap](#)
- QString [text](#)

## 9.308.1 Detailed Description

[DConfigDlgTitle](#) uses the general application font at 1.4 times its size to style the text.

[DConfigDlgTitle](#) is very simple to use. You can either use its default text (and pixmap) properties or display your own widgets in the title widget.

## 9.308.2 Member Enumeration Documentation

### 9.308.2.1 ImageAlignment

enum [Digikam::DConfigDlgTitle::ImageAlignment](#)

- [ImageLeft](#): Display the pixmap left
- [ImageRight](#): Display the pixmap right (default)

## Enumerator

ImageLeft	Display the pixmap on the left.
ImageRight	Display the pixmap on the right.

**9.308.2.2 MessageType**

```
enum Digikam::DConfigDlgTitle::MessageType
```

## Enumerator

PlainTextMessage	Normal comment.
InfoMessage	Information the user should be alerted to.
WarningMessage	A warning the user should be alerted to.
ErrorMessage	An error message.

**9.308.3 Constructor & Destructor Documentation****9.308.3.1 DConfigDlgTitle()**

```
Digikam::DConfigDlgTitle::DConfigDlgTitle (
    QWidget *const parent = nullptr ) [explicit]
```

## Parameters

<i>parent</i>	.
---------------	---

**9.308.4 Member Function Documentation****9.308.4.1 autoHideTimeout()**

```
int Digikam::DConfigDlgTitle::autoHideTimeout ( ) const
```

## Returns

timeout value in msec

**9.308.4.2 comment()**

```
QString Digikam::DConfigDlgTitle::comment ( ) const
```

## Returns

the text displayed in the comment below the title, if any

## See also

[setComment\(\)](#)



### 9.308.4.3 pixmap()

```
QPixmap Digikam::DConfigDlgTitle::pixmap ( ) const
```

#### Returns

the pixmap displayed in the title

#### See also

[setPixmap\(\)](#)

### 9.308.4.4 setAutoHideTimeout

```
void Digikam::DConfigDlgTitle::setAutoHideTimeout (
    int msec ) [slot]
```

#### Parameters

<i>msec</i>	timeout value in milliseconds
-------------	-------------------------------

### 9.308.4.5 setBuddy()

```
void Digikam::DConfigDlgTitle::setBuddy (
    QWidget *const buddy )
```

When the user presses the shortcut key indicated by the label in this title widget, the keyboard focus is transferred to the label's buddy widget.

#### Parameters

<i>buddy</i>	the widget to activate when the shortcut key is activated
--------------	---

### 9.308.4.6 setComment

```
void Digikam::DConfigDlgTitle::setComment (
    const QString & comment,
    MessageType type = PlainMessage ) [slot]
```

#### Parameters

<i>comment</i>	Text displayed beneath the main title as a comment. It can either be plain text or rich text.
<i>type</i>	The sort of message it is.

See also

[MessageType](#)

`comment()`

#### 9.308.4.7 `setPixmap` [1/4]

```
void Digikam::DConfigDlgTitle::setPixmap (
    const QIcon & icon,
    ImageAlignment alignment = ImageRight ) [slot]
```

Parameters

<i>icon</i>	The pixmap to display in the header. The pixmap is by default right, but
<i>alignment</i>	can be used to display it also left.

See also

`pixmap()`

#### 9.308.4.8 `setPixmap` [2/4]

```
void Digikam::DConfigDlgTitle::setPixmap (
    const QPixmap & pixmap,
    ImageAlignment alignment = ImageRight ) [slot]
```

Parameters

<i>pixmap</i>	Pixmap displayed in the header. The pixmap is by default right, but
<i>alignment</i>	can be used to display it also left.

See also

`pixmap()`

#### 9.308.4.9 `setPixmap` [3/4]

```
void Digikam::DConfigDlgTitle::setPixmap (
    const QString & icon,
    ImageAlignment alignment = ImageRight ) [slot]
```

Parameters

<i>icon</i>	name of the icon to display in the header. The pixmap is by default right, but
<i>alignment</i>	can be used to display it also left.

See also

[pixmap\(\)](#)

#### 9.308.4.10 [setPixmap](#) [4/4]

```
void Digikam::DConfigDlgTitle::setPixmap (
    MessageType type,
    ImageAlignment alignment = ImageRight ) [slot]
```

Parameters

<i>type</i>	The message type to display as pixmap in the header. The message is by default right, but
<i>alignment</i>	can be used to display it also left.

See also

[pixmap\(\)](#)

#### 9.308.4.11 [setText](#) [1/2]

```
void Digikam::DConfigDlgTitle::setText (
    const QString & text,
    MessageType type ) [slot]
```

Parameters

<i>text</i>	Text displayed on the label. It can either be plain text or rich text. If it is plain text, the text is displayed as a bold title text.
<i>type</i>	The sort of message it is; will also set the icon accordingly

See also

[MessageType](#)

[text\(\)](#)

#### 9.308.4.12 [setText](#) [2/2]

```
void Digikam::DConfigDlgTitle::setText (
    const QString & text,
    Qt::Alignment alignment = Qt::AlignLeft | Qt::AlignVCenter ) [slot]
```

Parameters

<i>text</i>	Text displayed on the label. It can either be plain text or rich text. If it is plain text, the text is displayed as a bold title text.
<i>alignment</i>	Alignment of the text. Default is left and vertical centered.

**See also**

[text\(\)](#)

**9.308.4.13 setWidget()**

```
void Digikam::DConfigDlgTitle::setWidget (
    QWidget *const widget )
```

**Parameters**

<i>widget</i>	the widget displayed on the title widget.
---------------	---

**9.308.4.14 text()**

```
QString Digikam::DConfigDlgTitle::text ( ) const
```

**Returns**

the text displayed in the title

**See also**

[setText\(\)](#)

**9.309 Digikam::DConfigDlgView Class Reference**

A base class which can handle multiple pages.

Inheritance diagram for Digikam::DConfigDlgView:



## Public Types

- enum [FaceType](#) {  
**Auto** , **Plain** , **List** , **Tree** ,  
**Tabbed** }

*This enum is used to decide which type of navigation view shall be used in the page view.*

## Signals

- void [currentPageChanged](#) (const QModelIndex &current, const QModelIndex &previous)  
*This signal is emitted whenever the current page changes.*

## Public Member Functions

- **DConfigDlgView** (QWidget \*const parent=nullptr)  
*Creates a page view with given parent.*
- **~DConfigDlgView** () override  
*Destroys the page view.*
- QModelIndex **currentPage** () const  
*Returns the index for the current page or an invalid index if no current page exists.*
- [FaceType](#) **faceType** () const  
*Returns the face type of the page view.*
- QAbstractItemDelegate \* **itemDelegate** () const  
*Returns the item delegate of the page view.*
- QAbstractItemModel \* **model** () const  
*Returns the model of the page view.*
- void [setCurrentPage](#) (const QModelIndex &index)  
*Sets the page with.*
- void **setDefaultWidget** (QWidget \*widget)  
*Sets the `widget` which will be shown when a page is selected that has no own widget set.*
- void **setFaceType** ([FaceType](#) faceType)  
*Sets the face type of the page view.*
- void [setItemDelegate](#) (QAbstractItemDelegate \*delegate)  
*Sets the item.*
- void [setModel](#) (QAbstractItemModel \*model)  
*Sets the `model` of the page view.*

## Protected Member Functions

- **DConfigDlgView** (DConfigDlgViewPrivate &dd, QWidget \*const parent)
- virtual QAbstractItemView \* [createView](#) ()  
*Returns the navigation view, depending on the current face type.*
- virtual bool [showPageHeader](#) () const  
*Returns whether the page header should be visible.*
- virtual Qt::Alignment [viewPosition](#) () const  
*Returns the position where the navigation view should be located according to the page stack.*

## Protected Attributes

- DConfigDlgViewPrivate \*const **d\_ptr**

## Properties

- [FaceType](#) **faceType**

## 9.309.1 Detailed Description

This class provides a widget base class which handles multiple pages and allows the user to switch between these pages in different ways.

Currently, `Auto`, `Plain`, `List`, `Tree` and `Tabbed` face types are available.

See also

[DConfigDlgWdg](#)

## 9.309.2 Member Enumeration Documentation

### 9.309.2.1 FaceType

```
enum Digikam::DConfigDlgView::FaceType
```

- `Auto` - Depending on the number of pages in the model, the `Plain` (one page), the `List` (several pages) or the `Tree` face (nested pages) will be used. This is the default face type.
- `Plain` - No navigation view will be visible and only the first page of the model will be shown.
- `List` - An icon list is used as navigation view.
- `Tree` - A tree list is used as navigation view.
- `Tabbed` - A tab widget is used as navigation view.

## 9.309.3 Member Function Documentation

### 9.309.3.1 createView()

```
QAbstractItemView * Digikam::DConfigDlgView::createView ( ) [protected], [virtual]
```

This method can be reimplemented to provide custom navigation views.

### 9.309.3.2 currentPageChanged

```
void Digikam::DConfigDlgView::currentPageChanged (
    const QModelIndex & current,
    const QModelIndex & previous ) [signal]
```

The previous page index is replaced by the current index.

### 9.309.3.3 setCurrentPage()

```
void Digikam::DConfigDlgView::setCurrentPage (
    const QModelIndex & index )
```

**Parameters**

<i>index</i>	to be the current page and emits the
--------------	--------------------------------------

**See also**

[currentPageChanged](#) signal.

**9.309.3.4 setItemDelegate()**

```
void Digikam::DConfigDlgView::setItemDelegate (
    QAbstractItemDelegate * delegate )
```

**Parameters**

<i>delegate</i>	which can be used customize the page view.
-----------------	--

**9.309.3.5 setModel()**

```
void Digikam::DConfigDlgView::setModel (
    QAbstractItemModel * model )
```

The model has to provide data for the roles defined in [DConfigDlgModel::Role](#).

**9.309.3.6 showPageHeader()**

```
bool Digikam::DConfigDlgView::showPageHeader ( ) const [protected], [virtual]
```

This method can be reimplemented for adapting custom views.

**9.309.3.7 viewPosition()**

```
Qt::Alignment Digikam::DConfigDlgView::viewPosition ( ) const [protected], [virtual]
```

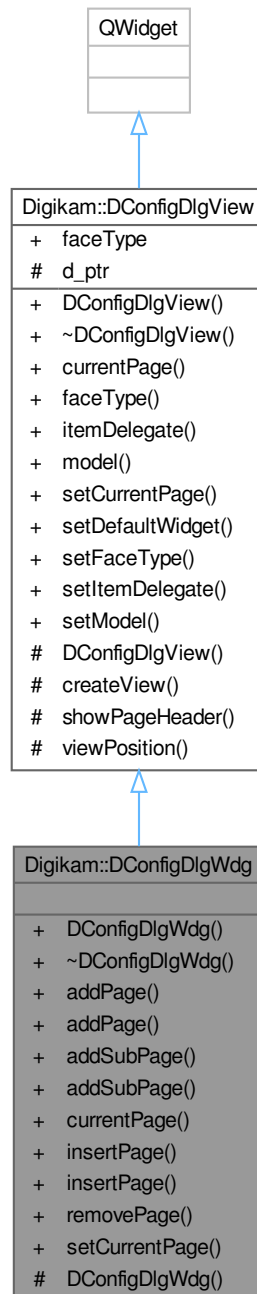
This method can be reimplemented for adapting custom views.



## 9.310 Digikam::DConfigDlgWdg Class Reference

Page widget with many layouts (faces).

Inheritance diagram for Digikam::DConfigDlgWdg:



### Signals

- void [currentPageChanged](#) (DConfigDlgWdgItem \*current, DConfigDlgWdgItem \*before)

- *This signal is emitted whenever the current page has changed.*
- void [pageRemoved](#) ([DConfigDlgWdgItem](#) \*page)  
*This signal is emitted when a page is removed.*
- void [pageToggled](#) ([DConfigDlgWdgItem](#) \*page, bool checked)  
*This signal is emitted whenever a checkable page changes its state.*

## Signals inherited from [Digikam::DConfigDlgView](#)

- void [currentPageChanged](#) (const QModelIndex &current, const QModelIndex &previous)  
*This signal is emitted whenever the current page changes.*

## Public Member Functions

- [DConfigDlgWdg](#) (QWidget \*const parent=nullptr)  
*Creates a new page widget.*
- [~DConfigDlgWdg](#) () override=default  
*Destroys the page widget.*
- void [addPage](#) ([DConfigDlgWdgItem](#) \*item)  
*Adds a new top level page to the widget.*
- [DConfigDlgWdgItem](#) \* [addPage](#) (QWidget \*widget, const QString &name)  
*Adds a new top level page to the widget.*
- void [addSubPage](#) ([DConfigDlgWdgItem](#) \*parent, [DConfigDlgWdgItem](#) \*item)  
*Inserts a new sub page in the widget.*
- [DConfigDlgWdgItem](#) \* [addSubPage](#) ([DConfigDlgWdgItem](#) \*parent, QWidget \*widget, const QString &name)  
*Inserts a new sub page in the widget.*
- [DConfigDlgWdgItem](#) \* [currentPage](#) () const  
*Returns the.*
- void [insertPage](#) ([DConfigDlgWdgItem](#) \*before, [DConfigDlgWdgItem](#) \*item)  
*Inserts a new page in the widget.*
- [DConfigDlgWdgItem](#) \* [insertPage](#) ([DConfigDlgWdgItem](#) \*before, QWidget \*widget, const QString &name)  
*Inserts a new page in the widget.*
- void [removePage](#) ([DConfigDlgWdgItem](#) \*item)  
*Removes the page associated with the given.*
- void [setCurrentPage](#) ([DConfigDlgWdgItem](#) \*item)  
*Sets the page which is associated with the given.*

## Public Member Functions inherited from [Digikam::DConfigDlgView](#)

- [DConfigDlgView](#) (QWidget \*const parent=nullptr)  
*Creates a page view with given parent.*
- [~DConfigDlgView](#) () override  
*Destroys the page view.*
- QModelIndex [currentPage](#) () const  
*Returns the index for the current page or an invalid index if no current page exists.*
- [FaceType](#) [faceType](#) () const  
*Returns the face type of the page view.*
- QAbstractItemDelegate \* [itemDelegate](#) () const  
*Returns the item delegate of the page view.*
- QAbstractItemModel \* [model](#) () const

- Returns the model of the page view.*
- void [setCurrentPage](#) (const QModelIndex &index)  
*Sets the page with.*
- void [setDefaultWidget](#) (QWidget \*widget)  
*Sets the widget which will be shown when a page is selected that has no own widget set.*
- void [setFaceType](#) ([FaceType](#) faceType)  
*Sets the face type of the page view.*
- void [setItemDelegate](#) (QAbstractItemDelegate \*delegate)  
*Sets the item.*
- void [setModel](#) (QAbstractItemModel \*model)  
*Sets the model of the page view.*

### Protected Member Functions

- [DConfigDlgWdg](#) (DConfigDlgWdgPrivate &dd, QWidget \*const parent)

### Protected Member Functions inherited from [Digikam::DConfigDlgView](#)

- [DConfigDlgView](#) (DConfigDlgViewPrivate &dd, QWidget \*const parent)
- virtual QAbstractItemView \* [createView](#) ()  
*Returns the navigation view, depending on the current face type.*
- virtual bool [showPageHeader](#) () const  
*Returns whether the page header should be visible.*
- virtual Qt::Alignment [viewPosition](#) () const  
*Returns the position where the navigation view should be located according to the page stack.*

### Additional Inherited Members

### Public Types inherited from [Digikam::DConfigDlgView](#)

- enum [FaceType](#) {  
  **Auto** , **Plain** , **List** , **Tree** ,  
  **Tabbed** }
- This enum is used to decide which type of navigation view shall be used in the page view.*

### Protected Attributes inherited from [Digikam::DConfigDlgView](#)

- DConfigDlgViewPrivate \*const **d\_ptr**

### Properties inherited from [Digikam::DConfigDlgView](#)

- [FaceType](#) faceType

## 9.310.1 Detailed Description

See also

[DConfigDlgView](#) with hierarchical page [model](#).

## 9.310.2 Constructor & Destructor Documentation

### 9.310.2.1 DConfigDlgWdg()

```
Digikam::DConfigDlgWdg::DConfigDlgWdg (
    QWidget *const parent = nullptr ) [explicit]
```

## Parameters

<i>parent</i>	The parent widget.
---------------	--------------------

### 9.310.3 Member Function Documentation

#### 9.310.3.1 `addPage()` [1/2]

```
void Digikam::DConfigDlgWdg::addPage (
    DConfigDlgWdgItem * item )
```

## Parameters

<i>item</i>	The
-------------	-----

## See also

[DConfigDlgWdgItem](#) which describes the page.

#### 9.310.3.2 `addPage()` [2/2]

```
DConfigDlgWdgItem * Digikam::DConfigDlgWdg::addPage (
    QWidget * widget,
    const QString & name )
```

## Parameters

<i>widget</i>	The widget of the page.
<i>name</i>	The name which is displayed in the navigation view.

## Returns

The associated

## See also

[DConfigDlgWdgItem](#).

#### 9.310.3.3 `addSubPage()` [1/2]

```
void Digikam::DConfigDlgWdg::addSubPage (
    DConfigDlgWdgItem * parent,
    DConfigDlgWdgItem * item )
```

## Parameters

<i>parent</i>	The new page will be insert as child of this
---------------	--

## See also

[DConfigDlgWdgItem](#).

## Parameters

<i>item</i>	The
-------------	-----

## See also

[DConfigDlgWdgItem](#) which describes the page.

**9.310.3.4 addSubPage() [2/2]**

```
DConfigDlgWdgItem * Digikam::DConfigDlgWdg::addSubPage (
    DConfigDlgWdgItem * parent,
    QWidget * widget,
    const QString & name )
```

## Parameters

<i>parent</i>	The new page will be insert as child of this
---------------	--

## See also

[DConfigDlgWdgItem](#).

## Parameters

<i>widget</i>	The widget of the page.
<i>name</i>	The name which is displayed in the navigation view.

## Returns

The associated

## See also

[DConfigDlgWdgItem](#).

**9.310.3.5 currentPage()**

```
DConfigDlgWdgItem * Digikam::DConfigDlgWdg::currentPage ( ) const
```

See also

[DConfigDlgWdgItem](#) for the current page or 0 if there is no current page.

### 9.310.3.6 `currentPageChanged`

```
void Digikam::DConfigDlgWdg::currentPageChanged (
    DConfigDlgWdgItem * current,
    DConfigDlgWdgItem * before ) [signal]
```

Parameters

<i>current</i>	The new current page or 0 if no current page is available.
----------------	--

### 9.310.3.7 `insertPage()` [1/2]

```
void Digikam::DConfigDlgWdg::insertPage (
    DConfigDlgWdgItem * before,
    DConfigDlgWdgItem * item )
```

Parameters

<i>before</i>	The new page will be insert before this
---------------	---

See also

[DConfigDlgWdgItem](#) on the same level in hierarchy.

Parameters

<i>item</i>	The
-------------	-----

See also

[DConfigDlgWdgItem](#) which describes the page.

### 9.310.3.8 `insertPage()` [2/2]

```
DConfigDlgWdgItem * Digikam::DConfigDlgWdg::insertPage (
    DConfigDlgWdgItem * before,
    QWidget * widget,
    const QString & name )
```

Parameters

<i>before</i>	The new page will be insert before this
---------------	---

## See also

[DConfigDlgWdgItem](#) on the same level in hierarchy.

## Parameters

<i>widget</i>	The widget of the page.
<i>name</i>	The name which is displayed in the navigation view.

## Returns

The associated

## See also

[DConfigDlgWdgItem](#).

**9.310.3.9 pageRemoved**

```
void Digikam::DConfigDlgWdg::pageRemoved (
    DConfigDlgWdgItem * page ) [signal]
```

## Parameters

<i>page</i>	The page which is removed
-------------	---------------------------

**9.310.3.10 pageToggled**

```
void Digikam::DConfigDlgWdg::pageToggled (
    DConfigDlgWdgItem * page,
    bool checked ) [signal]
```

## Parameters

<i>checked</i>	is true when the
<i>page</i>	is checked, or false if the
<i>page</i>	is unchecked.

**9.310.3.11 removePage()**

```
void Digikam::DConfigDlgWdg::removePage (
    DConfigDlgWdgItem * item )
```

## See also

[DConfigDlgWdgItem](#).

### 9.310.3.12 setCurrentPage()

```
void Digikam::DConfigDlgWdg::setCurrentPage (
    DConfigDlgWdgItem * item )
```

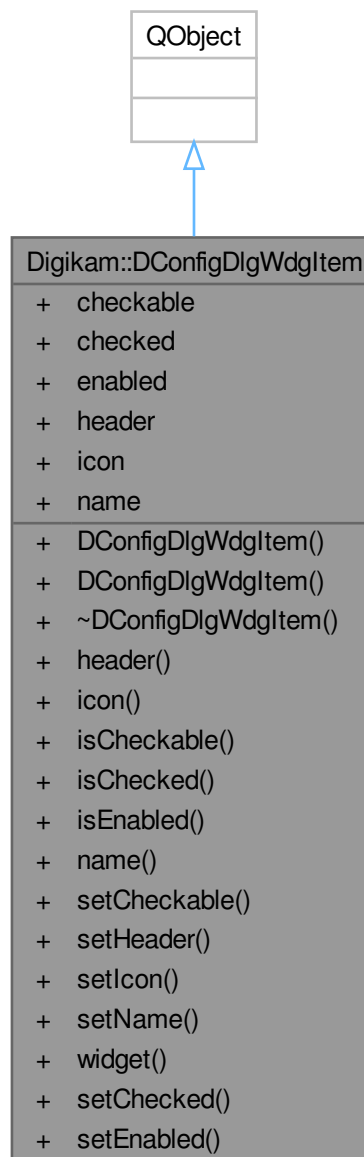
See also

[DConfigDlgWdgItem](#) to be the current page and emits the [currentPageChanged\(\)](#) signal.

## 9.311 Digikam::DConfigDlgWdgItem Class Reference

[DConfigDlgWdgItem](#) is used by [DConfigDlgWdg](#) and represents a page.

Inheritance diagram for Digikam::DConfigDlgWdgItem:





## Public Slots

- void **setChecked** (bool checked)  
*Sets whether the page widget item is checked.*
- void **setEnabled** (bool)  
*Sets whether the page widget item is enabled.*

## Signals

- void **changed** ()  
*This signal is emitted whenever the icon or header is changed.*
- void **toggled** (bool checked)  
*This signal is emitted whenever the user checks or unchecks the item of.*

## Public Member Functions

- [DConfigDlgWdgItem](#) (QWidget \*const widget)  
*Creates a new page widget item.*
- [DConfigDlgWdgItem](#) (QWidget \*const widget, const QString &name)  
*Creates a new page widget item.*
- [~DConfigDlgWdgItem](#) () override  
*Destroys the page widget item.*
- QString **header** () const  
*Returns the header of the page widget item.*
- QIcon **icon** () const  
*Returns the icon of the page widget item.*
- bool **isCheckable** () const  
*Returns whether the page widget item is checkable.*
- bool **isChecked** () const  
*Returns whether the page widget item is checked.*
- bool **isEnabled** () const  
*Returns whether the page widget item is enabled.*
- QString **name** () const  
*Returns the name of the page widget item.*
- void **setCheckable** (bool checkable)  
*Sets whether the page widget item is checkable in the view.*
- void **setHeader** (const QString &header)  
*Sets the header of the page widget item.*
- void **setIcon** (const QIcon &icon)  
*Sets the icon of the page widget item.*
- void **setName** (const QString &name)  
*Sets the name of the item as shown in the navigation view of the page widget.*
- QWidget \* **widget** () const  
*Returns the widget of the page widget item.*

## Properties

- bool **checkable**
- bool **checked**
- bool **enabled**

*This property holds whether the item is enabled.*

- QString **header**
- QIcon **icon**
- QString **name**

## 9.311.1 Constructor & Destructor Documentation

### 9.311.1.1 DConfigDlgWdgItem() [1/2]

```
Digikam::DConfigDlgWdgItem::DConfigDlgWdgItem (
    QWidget *const widget ) [explicit]
```

#### Parameters

<i>widget</i>	The widget that is shown as page in the <a href="#">DConfigDlgWdg</a> .
---------------	---

Hide the widget, otherwise when the widget has this [DConfigDlgView](#) as parent the widget is shown outside the QStackedWidget if the page was not selected ( and reparented ) yet.

### 9.311.1.2 DConfigDlgWdgItem() [2/2]

```
Digikam::DConfigDlgWdgItem::DConfigDlgWdgItem (
    QWidget *const widget,
    const QString & name )
```

#### Parameters

<i>widget</i>	The widget that is shown as page in the <a href="#">DConfigDlgWdg</a> .
<i>name</i>	The localized string that is show in the navigation view of the <a href="#">DConfigDlgWdg</a> .

Hide the widget, otherwise when the widget has this [DConfigDlgView](#) as parent the widget is shown outside the QStackedWidget if the page was not selected ( and reparented ) yet.

## 9.311.2 Member Function Documentation

### 9.311.2.1 setCheckable()

```
void Digikam::DConfigDlgWdgItem::setCheckable (
    bool checkable )
```

#### Parameters

<i>checkable</i>	True if the page widget is checkable, otherwise false.
------------------	--

### 9.311.2.2 setHeader()

```
void Digikam::DConfigDlgWdgItem::setHeader (
    const QString & header )
```

If `setHeader(QString())` is used, what is the default if the header does not got set explicit, then the defined `name()` will also be used for the header. If `setHeader("")` is used, the header will be hidden even if the [DConfigDlgView::FaceType](#) is something else then `Tabbed`.

#### Parameters

<i>header</i>	Header of the page widget item.
---------------	---------------------------------

### 9.311.2.3 setIcon()

```
void Digikam::DConfigDlgWdgItem::setIcon (
    const QIcon & icon )
```

#### Parameters

<i>icon</i>	Icon of the page widget item.
-------------	-------------------------------

### 9.311.2.4 toggled

```
void Digikam::DConfigDlgWdgItem::toggled (
    bool checked ) [signal]
```

#### See also

[setChecked\(\)](#) is called.

## 9.311.3 Property Documentation

### 9.311.3.1 enabled

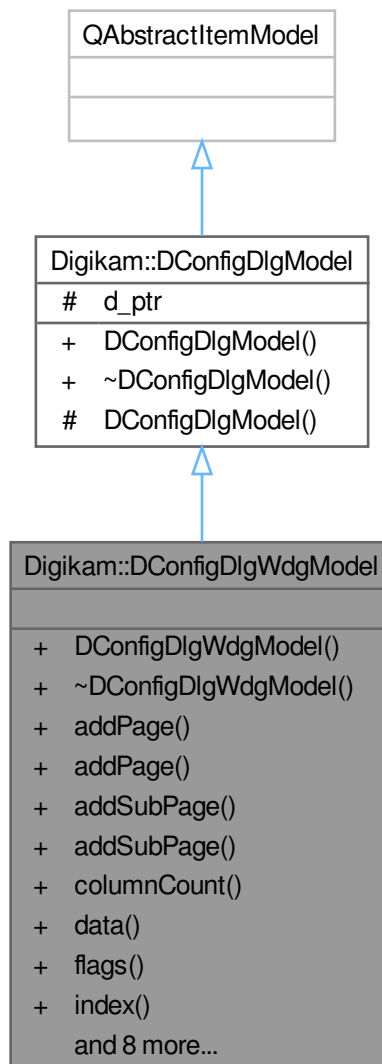
```
bool Digikam::DConfigDlgWdgItem::enabled [read], [write]
```

It dis-/enables both the widget and the item in the list-/treeview.

## 9.312 Digikam::DConfigDlgWdgModel Class Reference

This page model is used by.

Inheritance diagram for Digikam::DConfigDlgWdgModel:



## Signals

- void `toggled` (`DConfigDlgWdgItem *page`, bool checked)  
*This signal is emitted whenever a checkable page changes its state.*

## Public Member Functions

- `DConfigDlgWdgModel` (`QObject *const parent=nullptr`)  
*Creates a new page widget model.*
- `~DConfigDlgWdgModel` () `override=default`  
*Destroys the page widget model.*

- void `addPage` (`DConfigDlgWdgItem *item`)  
*Adds a new top level page to the model.*
- `DConfigDlgWdgItem * addPage` (`QWidget *widget`, `const QString &name`)  
*Adds a new top level page to the model.*
- void `addSubPage` (`DConfigDlgWdgItem *parent`, `DConfigDlgWdgItem *item`)  
*Inserts a new sub page in the model.*
- `DConfigDlgWdgItem * addSubPage` (`DConfigDlgWdgItem *parent`, `QWidget *widget`, `const QString &name`)  
*Inserts a new sub page in the model.*
- int **columnCount** (`const QModelIndex &parent=QModelIndex()`) `const` override  
*These methods are reimplemented from `QAbstractItemModel`.*
- `QVariant data` (`const QModelIndex &index`, `int role=Qt::DisplayRole`) `const` override
- `Qt::ItemFlags flags` (`const QModelIndex &index`) `const` override
- `QModelIndex index` (`const DConfigDlgWdgItem *item`) `const`  
*Returns the index for a given.*
- `QModelIndex index` (`int row`, `int column`, `const QModelIndex &parent=QModelIndex()`) `const` override
- void `insertPage` (`DConfigDlgWdgItem *before`, `DConfigDlgWdgItem *item`)  
*Inserts a new page in the model.*
- `DConfigDlgWdgItem * insertPage` (`DConfigDlgWdgItem *before`, `QWidget *widget`, `const QString &name`)  
*Inserts a new page in the model.*
- `DConfigDlgWdgItem * item` (`const QModelIndex &index`) `const`  
*Returns the.*
- `QModelIndex parent` (`const QModelIndex &index`) `const` override
- void `removePage` (`DConfigDlgWdgItem *item`)  
*Removes the page associated with the given.*
- int **rowCount** (`const QModelIndex &parent=QModelIndex()`) `const` override
- bool **setData** (`const QModelIndex &index`, `const QVariant &value`, `int role=Qt::EditRole`) `override`

### Public Member Functions inherited from [Digikam::DConfigDlgModel](#)

- **DConfigDlgModel** (`QObject *const parent=nullptr`)  
*Constructs a page model with the given parent.*
- `~DConfigDlgModel` () `override`  
*Destroys the page model.*

### Additional Inherited Members

### Public Types inherited from [Digikam::DConfigDlgModel](#)

- enum `Role` { `HeaderRole = Qt::UserRole + 1` , `WidgetRole` }  
*Additional roles that `DConfigDlgView` uses.*

### Protected Member Functions inherited from [Digikam::DConfigDlgModel](#)

- **DConfigDlgModel** (`DConfigDlgModelPrivate &dd`, `QObject *const parent`)

### Protected Attributes inherited from [Digikam::DConfigDlgModel](#)

- `DConfigDlgModelPrivate *const d_ptr`

### 9.312.1 Detailed Description

See also

[DConfigDlgWdg](#) to provide a hierarchical layout of pages.

### 9.312.2 Constructor & Destructor Documentation

#### 9.312.2.1 DConfigDlgWdgModel()

```
Digikam::DConfigDlgWdgModel::DConfigDlgWdgModel (
    QObject *const parent = nullptr ) [explicit]
```

Parameters

<i>parent</i>	The parent object.
---------------	--------------------

### 9.312.3 Member Function Documentation

#### 9.312.3.1 addPage() [1/2]

```
void Digikam::DConfigDlgWdgModel::addPage (
    DConfigDlgWdgItem * item )
```

Parameters

<i>item</i>	The
-------------	-----

See also

[DConfigDlgWdgItem](#) which describes the page.

#### 9.312.3.2 addPage() [2/2]

```
DConfigDlgWdgItem * Digikam::DConfigDlgWdgModel::addPage (
    QWidget * widget,
    const QString & name )
```

Parameters

<i>widget</i>	The widget of the page.
<i>name</i>	The name which is displayed in the navigation view.

Returns

The associated

See also

[DConfigDlgWdgItem](#).

### 9.312.3.3 addSubPage() [1/2]

```
void Digikam::DConfigDlgWdgModel::addSubPage (
    DConfigDlgWdgItem * parent,
    DConfigDlgWdgItem * item )
```

Parameters

<i>parent</i>	The new page will be insert as child of this
---------------	--

See also

[DConfigDlgWdgItem](#).

Parameters

<i>item</i>	The
-------------	-----

See also

[DConfigDlgWdgItem](#) which describes the page.

### 9.312.3.4 addSubPage() [2/2]

```
DConfigDlgWdgItem * Digikam::DConfigDlgWdgModel::addSubPage (
    DConfigDlgWdgItem * parent,
    QWidget * widget,
    const QString & name )
```

Parameters

<i>parent</i>	The new page will be insert as child of this
---------------	--

See also

[DConfigDlgWdgItem](#).

Parameters

<i>widget</i>	The widget of the page.
<i>name</i>	The name which is displayed in the navigation view.

**Returns**

The associated

**See also**

[DConfigDlgWdgItem](#).

**9.312.3.5 index()**

```
QModelIndex Digikam::DConfigDlgWdgModel::index (
    const DConfigDlgWdgItem * item ) const
```

**See also**

[DConfigDlgWdgItem](#). The index is invalid if the [item](#) can't be found in the model.

**9.312.3.6 insertPage() [1/2]**

```
void Digikam::DConfigDlgWdgModel::insertPage (
    DConfigDlgWdgItem * before,
    DConfigDlgWdgItem * item )
```

**Parameters**

<i>before</i>	The new page will be insert before this
---------------	---

**See also**

[DConfigDlgWdgItem](#) on the same level in hierarchy.

**Parameters**

<i>item</i>	The
-------------	-----

**See also**

[DConfigDlgWdgItem](#) which describes the page.

**9.312.3.7 insertPage() [2/2]**

```
DConfigDlgWdgItem * Digikam::DConfigDlgWdgModel::insertPage (
    DConfigDlgWdgItem * before,
    QWidget * widget,
    const QString & name )
```



## Parameters

<i>before</i>	The new page will be insert before this
---------------	---

## See also

[DConfigDlgWdgItem](#) on the same level in hierarchy.

## Parameters

<i>widget</i>	The widget of the page.
<i>name</i>	The name which is displayed in the navigation view.

## Returns

The associated

## See also

[DConfigDlgWdgItem](#).

**9.312.3.8 item()**

```
DConfigDlgWdgItem * Digikam::DConfigDlgWdgModel::item (
    const QModelIndex & index ) const
```

## See also

[DConfigDlgWdgItem](#) for a given index or 0 if the index is invalid.

**9.312.3.9 removePage()**

```
void Digikam::DConfigDlgWdgModel::removePage (
    DConfigDlgWdgItem * item )
```

## See also

[DConfigDlgWdgItem](#).

**9.312.3.10 toggled**

```
void Digikam::DConfigDlgWdgModel::toggled (
    DConfigDlgWdgItem * page,
    bool checked ) [signal]
```

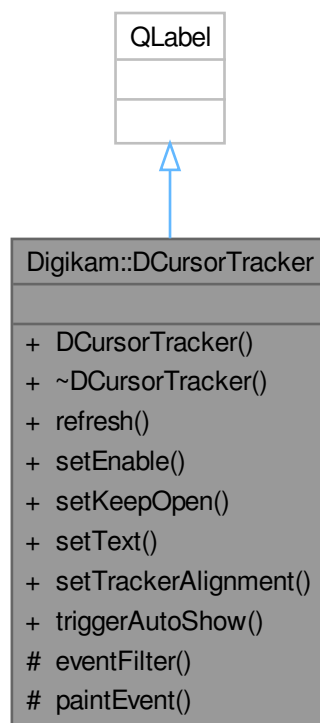
## Parameters

<i>checked</i>	is true when the
<i>page</i>	is checked, or false if the
<i>page</i>	is unchecked.

### 9.313 Digikam::DCursorTracker Class Reference

This class implements a window which looks like a tool tip.

Inheritance diagram for Digikam::DCursorTracker:



#### Public Member Functions

- **DCursorTracker** (const QString &txt, QWidget \*const parent, Qt::Alignment align=Qt::AlignCenter)
- void **refresh** ()
- void **setEnabled** (bool b)
- void **setKeepOpen** (bool b)
- void **setText** (const QString &txt)
  - Overload to make sure the widget size is correct.*
- void **setTrackerAlignment** (Qt::Alignment alignment)
- void **triggerAutoShow** (int timeout=2000)

### Protected Member Functions

- bool **eventFilter** (QObject \*, QEvent \*) override
- void **paintEvent** (QPaintEvent \*) override

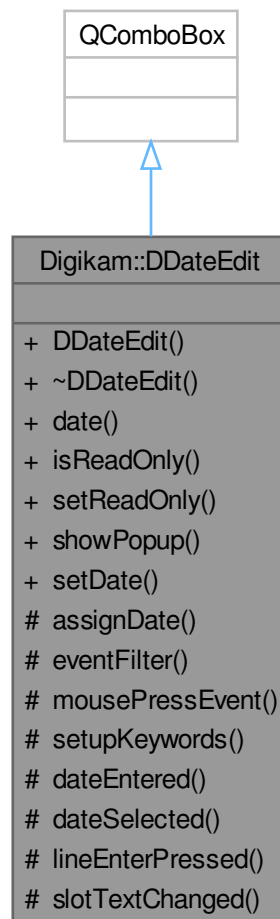
### 9.313.1 Detailed Description

It will follow the cursor when it's over a specified widget.

## 9.314 Digikam::DDateEdit Class Reference

A date editing widget that consists of an editable combo box.

Inheritance diagram for Digikam::DDateEdit:



### Public Slots

- void [setDate](#) (const QDate &date)  
*Sets the date.*

### Signals

- void [dateChanged](#) (const QDate &date)  
*This signal is emitted whenever the user modifies the date.*

### Public Member Functions

- **DDateEdit** (QWidget \*const parent=nullptr, const QString &name=QString())
- QDate [date](#) () const
- bool [isReadOnly](#) () const
- void [setReadOnly](#) (bool readOnly)  
*Sets whether the widget is read-only for the user.*
- void **showPopup** () override

### Protected Slots

- void **dateEntered** (const QDate &)
- void **dateSelected** (const QDate &)
- void **lineEnterPressed** ()
- void **slotTextChanged** (const QString &)

### Protected Member Functions

- virtual bool [assignDate](#) (const QDate &date)  
*Sets the date, without altering the display.*
- bool **eventFilter** (QObject \*, QEvent \*) override
- void **mousePressEvent** (QMouseEvent \*) override
- void [setupKeywords](#) ()  
*Fills the keyword map.*

## 9.314.1 Detailed Description

The combo box contains the date in text form, and clicking the combo box arrow will display a 'popup' style date picker.

This widget also supports advanced features like allowing the user to type in the day name to get the date. The following keywords are supported (in the native language): tomorrow, yesterday, today, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday.

## 9.314.2 Member Function Documentation

### 9.314.2.1 assignDate()

```
bool Digikam::DDateEdit::assignDate (
    const QDate & date ) [protected], [virtual]
```

This method is used internally to set the widget's date value. As a virtual method, it allows derived classes to perform additional validation on the date value before it is set. Derived classes should return true if `QDate::isValid(date)` returns false.

**Parameters**

<i>date</i>	The new date to set.
-------------	----------------------

**Returns**

True if the date was set, false if it was considered invalid and remains unchanged.

**9.314.2.2 date()**

```
QDate Digikam::DDateEdit::date ( ) const
```

**Returns**

The date entered. This date could be invalid, you have to check validity yourself.

**9.314.2.3 dateChanged**

```
void Digikam::DDateEdit::dateChanged (
    const QDate & date ) [signal]
```

The passed date can be invalid.

**9.314.2.4 isReadOnly()**

```
bool Digikam::DDateEdit::isReadOnly ( ) const
```

**Returns**

True if the widget is read-only, false if read-write.

**9.314.2.5 setDate**

```
void Digikam::DDateEdit::setDate (
    const QDate & date ) [slot]
```

**Parameters**

<i>date</i>	The new date to display. This date must be valid or it will not be set
-------------	--

**9.314.2.6 setReadOnly()**

```
void Digikam::DDateEdit::setReadOnly (
    bool readOnly )
```

If read-only, the date picker pop-up is inactive, and the displayed date cannot be edited.

#### Parameters

<i>readOnly</i>	True to set the widget read-only, false to set it read-write.
-----------------	---

#### 9.314.2.7 setupKeywords()

```
void Digikam::DDateEdit::setupKeywords ( ) [protected]
```

Re-implement it if you want additional keywords.

## 9.315 Digikam::DDatePicker Class Reference

Provides a widget for calendar date input.

Inheritance diagram for Digikam::DDatePicker:



## Signals

- void `dateChanged` (const `QDate &date`)  
*This signal is emitted each time the selected date is changed.*
- void `dateEntered` (const `QDate &date`)  
*This signal is emitted when enter is pressed and a VALID date has been entered before into the line edit.*
- void `dateSelected` (const `QDate &date`)

*This signal is emitted each time a day has been selected by clicking on the table (hitting a day in the current month).*

- void **tableClicked** ()

*This signal is emitted when the day has been selected by clicking on it in the table.*

## Public Member Functions

- [DDatePicker](#) (const QDate &dt, QWidget \*const parent=nullptr)

*The constructor.*

- [DDatePicker](#) (QWidget \*const parent=nullptr)

*The constructor.*

- [~DDatePicker](#) () override

*The destructor.*

- const QDate & **date** () const

- [DDateTable](#) \* **dateTable** () const

- int **fontSize** () const

*Returns the font size of the widget elements.*

- bool **hasCloseButton** () const

- void **setCloseButton** (bool enable)

*By calling this method with `enable = true`, [DDatePicker](#) will show a little close-button in the upper button-row.*

- bool **setDate** (const QDate &date)

*Sets the date.*

- void **setFontSize** (int)

*Sets the font size of the widgets elements.*

- QSize **sizeHint** () const override

*The size hint for date pickers.*

## Protected Slots

- void **dateChangedSlot** (const QDate &date)
- void **lineEnterPressed** ()
- void **monthBackwardClicked** ()
- void **monthForwardClicked** ()
- void **selectMonthClicked** ()
- void **selectYearClicked** ()
- void **tableClickedSlot** ()
- void **todayButtonClicked** ()
- void **uncheckYearSelector** ()
- void **weekSelected** (int)
- void **yearBackwardClicked** ()
- void **yearForwardClicked** ()

## Protected Member Functions

- void **changeEvent** (QEvent \*) override
- bool **eventFilter** (QObject \*, QEvent \*) override  
*to catch move keyEvents when QLineEdit has keyFocus*
- void **resizeEvent** (QResizeEvent \*) override  
*the resize event*



## Properties

- bool **closeButton**
- QDate **date**
- int **fontSize**

## Friends

- class **Private**

## 9.315.1 Constructor & Destructor Documentation

### 9.315.1.1 DDatePicker() [1/2]

```
Digikam::DDatePicker::DDatePicker (
    QWidget *const parent = nullptr ) [explicit]
```

The current date will be displayed initially.

### 9.315.1.2 DDatePicker() [2/2]

```
Digikam::DDatePicker::DDatePicker (
    const QDate & dt,
    QWidget *const parent = nullptr ) [explicit]
```

The given date will be displayed initially.

## 9.315.2 Member Function Documentation

### 9.315.2.1 date()

```
const QDate & Digikam::DDatePicker::date ( ) const
```

#### Returns

the selected date.

### 9.315.2.2 dateChanged

```
void Digikam::DDatePicker::dateChanged (
    const QDate & date ) [signal]
```

Usually, this does not mean that the date has been entered, since the date also changes, for example, when another month is selected.

#### See also

[dateSelected](#)

### 9.315.2.3 dateEntered

```
void Digikam::DDatePicker::dateEntered (
    const QDate & date ) [signal]
```

Connect to both [dateEntered\(\)](#) and [dateSelected\(\)](#) to receive all events where the user really enters a date.

### 9.315.2.4 dateSelected

```
void Digikam::DDatePicker::dateSelected (
    const QDate & date ) [signal]
```

It has the same meaning as [dateSelected\(\)](#) in older versions of [DDatePicker](#).

### 9.315.2.5 dateTable()

```
DDateTable * Digikam::DDatePicker::dateTable ( ) const
```

#### Returns

the [DDateTable](#) widget child of this [DDatePicker](#) widget.

### 9.315.2.6 hasCloseButton()

```
bool Digikam::DDatePicker::hasCloseButton ( ) const
```

#### Returns

true if a [DDatePicker](#) shows a close-button.

#### See also

[setCloseButton](#)

### 9.315.2.7 setCloseButton()

```
void Digikam::DDatePicker::setCloseButton (
    bool enable )
```

Clicking the close-button will cause the [DDatePicker](#)'s [topLevelWidget\(\)](#)'s [close\(\)](#) method being called. This is mostly useful for toplevel datepickers without a window manager decoration.

#### See also

[hasCloseButton](#)

### 9.315.2.8 setDate()

```
bool Digikam::DDatePicker::setDate (
    const QDate & date )
```

#### Returns

`false` and does not change anything if the date given is invalid.

### 9.315.2.9 sizeHint()

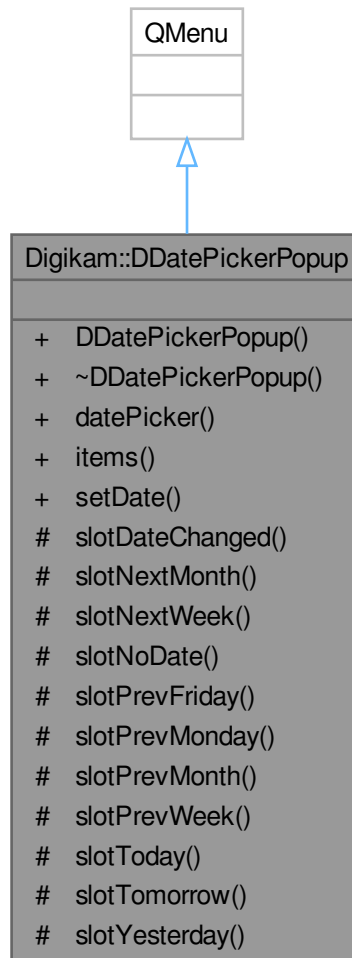
```
QSize Digikam::DDatePicker::sizeHint ( ) const [override]
```

The size hint recommends the minimum size of the widget so that all elements may be placed without clipping. This sometimes looks ugly, so when using the size hint, try adding 28 to each of the reported numbers of pixels.

## 9.316 Digikam::DDatePickerPopup Class Reference

This menu helps the user to select a date quickly.

Inheritance diagram for Digikam::DDatePickerPopup:



### Public Types

- enum **ItemFlag** { **NoDate** = 1 , **DatePicker** = 2 , **Words** = 4 }
- typedef QFlags< ItemFlag > **Items**

### Signals

- void **dateChanged** (const QDate &)  
*This signal emits the new date (selected with datepicker or other menu-items).*

### Public Member Functions

- **DDatePickerPopup** (Items **items**, const QDate &date=QDate::currentDate(), QWidget \*const parent=nullptr)  
*A constructor for the `DDatePickerPopup`.*
- **DDatePicker** \* **datePicker** () const
- int **items** () const
- void **setDate** (const QDate &date)

## Protected Slots

- void **slotDateChanged** (const QDate &)
- void **slotNextMonth** ()
- void **slotNextWeek** ()
- void **slotNoDate** ()
- void **slotPrevFriday** ()
- void **slotPrevMonday** ()
- void **slotPrevMonth** ()
- void **slotPrevWeek** ()
- void **slotToday** ()
- void **slotTomorrow** ()
- void **slotYesterday** ()

### 9.316.1 Detailed Description

This menu helps the user to select a date quickly. It offers various ways of selecting, e.g. with a [DDatePicker](#) or with words like "Tomorrow".

The available items are:

- NoDate: A menu-item with "No Date". If chosen, the datepicker will emit a null QDate.
- DatePicker: Show a DDatePicker-widget.
- Words: Show items like "Today", "Tomorrow" or "Next Week".

When supplying multiple items, separate each item with a bitwise OR.

### 9.316.2 Constructor & Destructor Documentation

#### 9.316.2.1 DDatePickerPopup()

```
Digikam::DDatePickerPopup::DDatePickerPopup (
    Items items,
    const QDate & date = QDate::currentDate(),
    QWidget *const parent = nullptr ) [explicit]
```

#### Parameters

<i>items</i>	List of all desirable items, separated with a bitwise OR.
<i>date</i>	Initial date of datepicker-widget.
<i>parent</i>	The object's parent.

### 9.316.3 Member Function Documentation

#### 9.316.3.1 datePicker()

```
DDatePicker * Digikam::DDatePickerPopup::datePicker ( ) const
```

**Returns**

A pointer to the private variable `mDatePicker`, an instance of [DDatePicker](#).

**9.316.3.2 items()**

```
int Digikam::DDatePickerPopup::items ( ) const
```

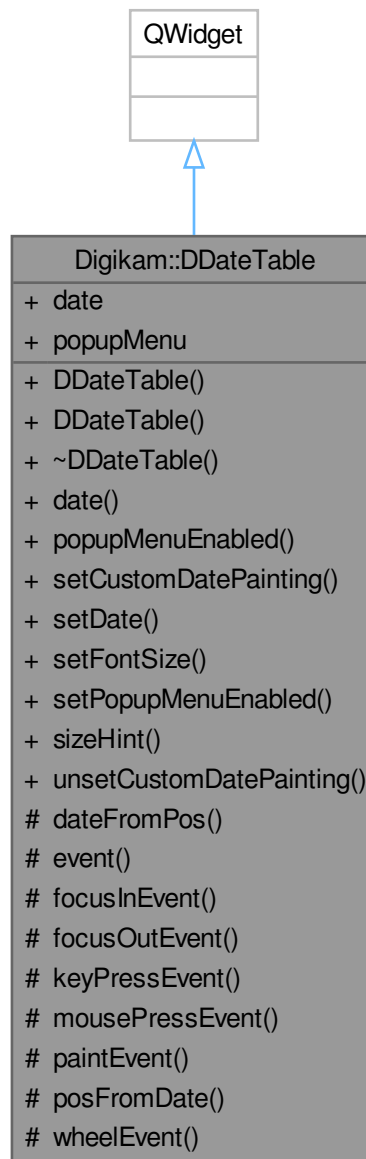
**Returns**

Returns the bitwise result of the active items in the popup.

**9.317 Digikam::DDateTable Class Reference**

This is a support class for the [DDatePicker](#) class.

Inheritance diagram for Digikam::DDateTable:



## Public Types

- enum **BackgroundMode** { **NoBgMode** = 0 , **RectangleMode** , **CircleMode** }

## Signals

- void [aboutToShowContextMenu](#) (QMenu \*menu, const QDate &dt)

*A popup menu for a given date is about to be shown (as when the user right clicks on that date and the popup menu is enabled).*

- void `dateChanged` (const `QDate` &cur, const `QDate` &old)  
*This function behaves essentially like the one above.*
- void `dateChanged` (const `QDate` &date)  
*The selected date changed.*
- void `tableClicked` ()  
*A date has been selected by clicking on the table.*

### Public Member Functions

- `DDateTable` (const `QDate` &dt, `QWidget` \*const parent=nullptr)
- `DDateTable` (`QWidget` \*const parent=nullptr)
- const `QDate` & `date` () const
- bool `popupMenuEnabled` () const  
*Returns if the popup menu is enabled or not.*
- void `setCustomDatePainting` (const `QDate` &date, const `QColor` &fgColor, `BackgroundMode` bgMode=`NoBgMode`, const `QColor` &bgColor=`QColor`())  
*Makes a given date be painted with a given foregroundColor, and background (a rectangle, or a circle/ellipse) in a given color.*
- bool `setDate` (const `QDate` &date)  
*Select and display this date.*
- void `setFontSize` (int size)  
*Set the font size of the date table.*
- void `setPopupMenuEnabled` (bool enable)  
*Enables a popup menu when right clicking on a date.*
- `QSize` `sizeHint` () const override  
*Returns a recommended size for the widget.*
- void `unsetCustomDatePainting` (const `QDate` &dt)  
*Unsets the custom painting of a date so that the date is painted as usual.*

### Protected Member Functions

- virtual `QDate` `dateFromPos` (int pos)  
*calculate the date that is displayed at a given cell in the matrix.*
- bool `event` (`QEvent` \*e) override  
*Cell highlight on mouse hovering.*
- void `focusInEvent` (`QFocusEvent` \*e) override
- void `focusOutEvent` (`QFocusEvent` \*e) override
- void `keyPressEvent` (`QKeyEvent` \*e) override
- void `mousePressEvent` (`QMouseEvent` \*e) override  
*React on mouse clicks that select a date.*
- void `paintEvent` (`QPaintEvent` \*e) override
- virtual int `posFromDate` (const `QDate` &dt)  
*calculate the position of the cell in the matrix for the given date.*
- void `wheelEvent` (`QWheelEvent` \*e) override

### Properties

- `QDate` `date`
- bool `popupMenu`



## Friends

- class **Private**

## 9.317.1 Detailed Description

It just draws the calendar table without titles, but could theoretically be used as a standalone.

When a date is selected by the user, it emits a signal: `dateSelected(QDate)`

## 9.317.2 Member Function Documentation

### 9.317.2.1 `aboutToShowContextMenu`

```
void Digikam::DDateTable::aboutToShowContextMenu (
    QMenu * menu,
    const QDate & dt ) [signal]
```

Connect the slot where you fill the menu to this signal.

### 9.317.2.2 `date()`

```
const QDate & Digikam::DDateTable::date ( ) const
```

#### Returns

the selected date.

### 9.317.2.3 `dateChanged`

```
void Digikam::DDateTable::dateChanged (
    const QDate & cur,
    const QDate & old ) [signal]
```

The selected date changed.

#### Parameters

<i>cur</i>	The current date
<i>old</i>	The date before the date was changed

### 9.317.2.4 `dateFromPos()`

```
QDate Digikam::DDateTable::dateFromPos (
    int pos ) [protected], [virtual]
```

`pos` is the 0-based index in the matrix. Inverse function to `posForDate()`.

### 9.317.2.5 posFromDate()

```
int Digikam::DDateTable::posFromDate (
    const QDate & dt ) [protected], [virtual]
```

The result is the 0-based index.

### 9.317.2.6 setPopupMenuEnabled()

```
void Digikam::DDateTable::setPopupMenuEnabled (
    bool enable )
```

When it's enabled, this object emits a `aboutToShowContextMenu` signal where you can fill in the menu items.

### 9.317.2.7 sizeHint()

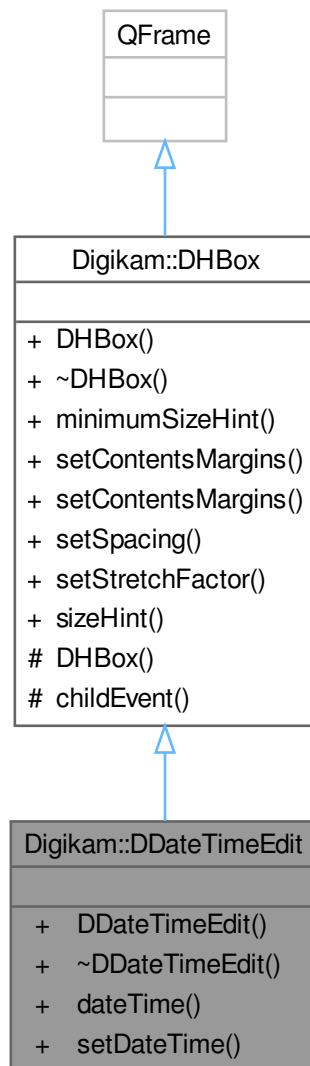
```
QSize Digikam::DDateTable::sizeHint ( ) const [override]
```

To save some time, the size of the largest used cell content is calculated in each `paintCell()` call, since all calculations have to be done there anyway. The size is stored in `maxCell`. The `sizeHint()` simply returns a multiple of `maxCell`.

## 9.318 Digikam::DDateTimeEdit Class Reference

This class is basically the same as the KDE Date Time widget with the exception that a `QTimeEdit` is placed directly besides it.

Inheritance diagram for Digikam::DDateTimeEdit:



## Signals

- void `dateTimeChanged` (const QDateTime &`dateTime`)  
*This signal is emitted whenever the user modifies the date or time.*

## Public Member Functions

- `DDateTimeEdit` (QWidget \*const parent, const QString &name)  
*constructor*
- `~DDateTimeEdit` () override  
*destructor*

- QDateTime [dateTime](#) () const  
*returns the date and time*
- void **setDateTime** (const QDateTime &[dateTime](#))  
*Sets the date and the time of this widget.*

## Public Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (QWidget \*const parent=nullptr)
- QSize **minimumSizeHint** () const override
- void **setContentsMargins** (const QMargins &margins)
- void **setContentsMargins** (int left, int top, int right, int bottom)
- void **setSpacing** (int space)
- void **setStretchFactor** (QWidget \*const widget, int stretch)
- QSize **sizeHint** () const override

## Additional Inherited Members

## Protected Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (bool vertical, QWidget \*const parent)
- void **childEvent** (QChildEvent \*e) override

## 9.318.1 Constructor & Destructor Documentation

### 9.318.1.1 DDateTimeEdit()

```
Digikam::DDateTimeEdit::DDateTimeEdit (
    QWidget *const parent,
    const QString & name ) [explicit]
```

#### Parameters

<i>parent</i>	the parent widget
<i>name</i>	the name of the widget

## 9.318.2 Member Function Documentation

### 9.318.2.1 dateTime()

```
QDateTime Digikam::DDateTimeEdit::dateTime ( ) const
```

#### Returns

a QDateTime with the currently chosen date and time

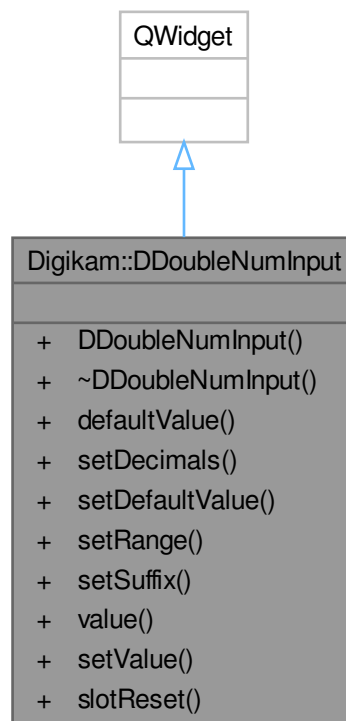
### 9.318.2.2 dateTimeChanged

```
void Digikam::DDateTimeEdit::dateTimeChanged (
    const QDateTime & dateTime ) [signal]
```

The passed date and time can be invalid.

## 9.319 Digikam::DDoubleNumInput Class Reference

Inheritance diagram for Digikam::DDoubleNumInput:



### Public Slots

- void **setValue** (double d)
- void **slotReset** ()

### Signals

- void **reset** ()
- void **valueChanged** (double)

**Public Member Functions**

- **DDoubleNumInput** (QWidget \*const parent=nullptr)
- double **defaultValue** () const
- void **setDecimals** (int p)
- void **setDefaultValue** (double d)
- void **setRange** (double min, double max, double step)
- void **setSuffix** (const QString &suffix)
- double **value** () const

## 9.320 Digikam::DDoubleSliderSpinBox Class Reference

Inheritance diagram for Digikam::DDoubleSliderSpinBox:



### Public Slots

- void **setValue** (double value)

## Signals

- void **valueChanged** (double value)

## Public Member Functions

- **DDoubleSliderSpinBox** (QWidget \*const parent=nullptr)
- double **fastSliderStep** () const
- double **maximum** () const
- double **minimum** () const
- void **setFastSliderStep** (double step)
- void **setMaximum** (double maximum)
- void **setMinimum** (double minimum)
- void **setRange** (double minimum, double maximum, int decimals=0)
- void **setSingleStep** (double value)
- double **value** ()

## Public Member Functions inherited from [Digikam::DAbstractSliderSpinBox](#)

- void **hideEdit** ()
- bool **isDragging** () const
- virtual QSize **minimumSize** () const
- QSize **minimumSizeHint** () const override
- void **setBlockUpdateSignalOnDrag** (bool block)
  - *If set to block, it informs inheriting classes that they shouldn't emit signals if the update comes from a mouse dragging the slider.*
- void **setExponentRatio** (double dbl)
- void **setPrefix** (const QString &prefix)
- void **setSuffix** (const QString &suffix)
- void **showEdit** ()
- QSize **sizeHint** () const override

## Protected Member Functions

- void **setInternalValue** (int value, bool blockUpdateSignal) override
  - *Sets the slider internal value.*
- QString **valueString** () const override

## Protected Member Functions inherited from [Digikam::DAbstractSliderSpinBox](#)

- **DAbstractSliderSpinBox** (QWidget \*const parent, DAbstractSliderSpinBoxPrivate \*const q)
- void **changeEvent** (QEvent \*e) override
- QRect **downButtonRect** (const QStyleOptionSpinBox &spinBoxOptions) const
- bool **eventFilter** (QObject \*recv, QEvent \*e) override
- void **focusInEvent** (QFocusEvent \*e) override
- void **keyPressEvent** (QKeyEvent \*e) override
- void **mouseMoveEvent** (QMouseEvent \*e) override
- void **mousePressEvent** (QMouseEvent \*e) override
- void **mouseReleaseEvent** (QMouseEvent \*e) override
- void **paint** (QPainter &painter)
- void **paintBreeze** (QPainter &painter)



- void **paintEvent** (QPaintEvent \*e) override
- void **paintFusion** (QPainter &painter)
- void **paintPlastique** (QPainter &painter)
- QStyleOptionProgressBar **progressBarOptions** () const
- QRect **progressRect** (const QStyleOptionSpinBox &spinBoxOptions) const
- QStyleOptionSpinBox **spinBoxOptions** () const
- QRect **upButtonRect** (const QStyleOptionSpinBox &spinBoxOptions) const
- int **valueForX** (int x, Qt::KeyboardModifiers modifiers=Qt::NoModifier) const
- void **wheelEvent** (QWheelEvent \*e) override

#### Additional Inherited Members

#### Protected Slots inherited from [Digikam::DAbstractSliderSpinBox](#)

- void **contextMenuEvent** (QContextMenuEvent \*event) override
- void **editLostFocus** ()

#### Protected Attributes inherited from [Digikam::DAbstractSliderSpinBox](#)

- DAbstractSliderSpinBoxPrivate \*const **d\_ptr**

### 9.320.1 Member Function Documentation

#### 9.320.1.1 setInternalValue()

```
void Digikam::DDoubleSliderSpinBox::setInternalValue (
    int value,
    bool blockUpdateSignal ) [override], [protected], [virtual]
```

Inheriting classes should respect blockUpdateSignal so that, in specific cases, we have a performance improvement. See setIgnoreMouseMoveEvents.

Implements [Digikam::DAbstractSliderSpinBox](#).

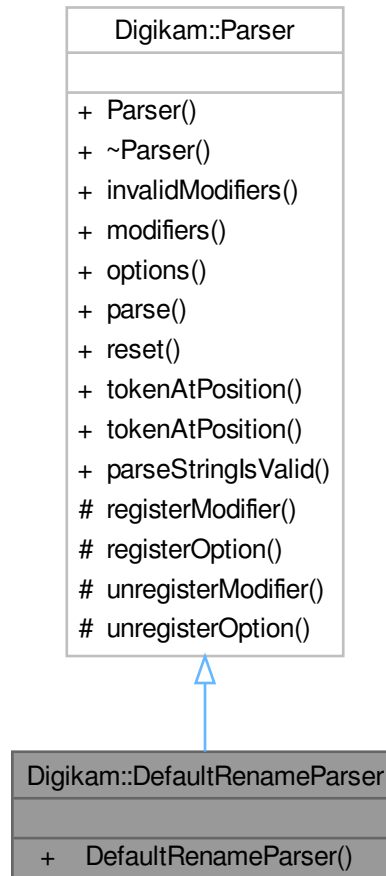
#### 9.320.1.2 valueString()

```
QString Digikam::DDoubleSliderSpinBox::valueString ( ) const [override], [protected], [virtual]
```

Implements [Digikam::DAbstractSliderSpinBox](#).

## 9.321 Digikam::DefaultRenameParser Class Reference

Inheritance diagram for Digikam::DefaultRenameParser:



### Additional Inherited Members

### Public Member Functions inherited from [Digikam::Parser](#)

- `ParseResults` `invalidModifiers` (`ParseSettings` &settings)
- `RulesList` `modifiers` () const
- `RulesList` `options` () const
- `QString` `parse` (`ParseSettings` &settings)
- `void` `reset` ()
- `bool` `tokenAtPosition` (`ParseSettings` &settings, int pos)
- `bool` `tokenAtPosition` (`ParseSettings` &settings, int pos, int &start, int &length)

### Static Public Member Functions inherited from [Digikam::Parser](#)

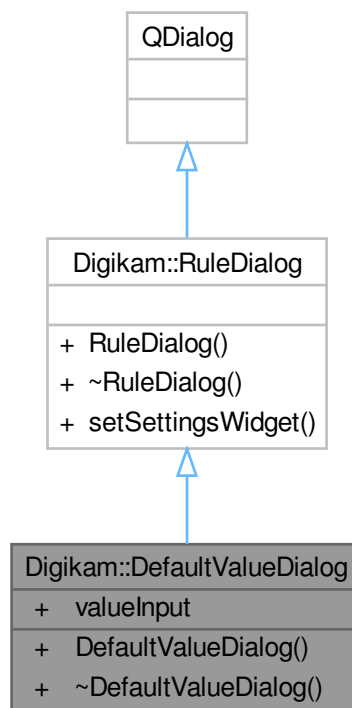
- static `bool` `parseStringsValid` (const `QString` &str)  
*check if the given parse string is valid*

### Protected Member Functions inherited from [Digikam::Parser](#)

- void **registerModifier** ([Rule](#) \*modifier)
- void **registerOption** ([Rule](#) \*option)
- void **unregisterModifier** (const [Rule](#) \*modifier)
- void **unregisterOption** (const [Rule](#) \*option)

## 9.322 Digikam::DefaultValueDialog Class Reference

Inheritance diagram for Digikam::DefaultValueDialog:



### Public Member Functions

- **DefaultValueDialog** ([Rule](#) \*parent)

### Public Member Functions inherited from [Digikam::RuleDialog](#)

- **RuleDialog** ([Rule](#) \*const parent)
- void **setSettingsWidget** ([QWidget](#) \*const settingsWidget)

## Public Attributes

- `QLineEdit * valueInput = nullptr`

## 9.323 Digikam::DefaultValueModifier Class Reference

Inheritance diagram for Digikam::DefaultValueModifier:



## Public Member Functions

- QString [parseOperation](#) ([ParseSettings](#) &settings, const QRegularExpressionMatch &match) override  
*TODO: describe me.*

## Public Member Functions inherited from [Digikam::Modifier](#)

- **Modifier** (const QString &name, const QString &description)
- **Modifier** (const QString &name, const QString &description, const QString &icon)

## Public Member Functions inherited from [Digikam::Rule](#)

- **Rule** (const QString &name)
- **Rule** (const QString &name, const QString &icon)
- QString **description** () const
- QPixmap **icon** (Rule::IconType type=Rule::Action) const
- bool **isValid** () const  
*Checks the validity of the parse object.*
- [ParseResults](#) **parse** ([ParseSettings](#) &settings)
- QRegularExpression & **regExp** () const  
*TODO: This is probably not needed anymore.*
- QPushButton \* **registerButton** (QWidget \*parent)  
*Register a button in the parent object.*
- QAction \* **registerMenu** (QMenu \*parent)  
*Register a menu action in the parent object.*
- virtual void **reset** ()  
*Resets the parser to its initial state.*
- TokenList & **tokens** () const
- bool **useTokenMenu** () const  
*Returns true if a token menu is used.*

## Additional Inherited Members

## Public Types inherited from [Digikam::Rule](#)

- enum **IconType** { **Action** = 0 , **Dialog** }

## Signals inherited from [Digikam::Rule](#)

- void **signalTokenTriggered** (const QString &)

## Static Public Member Functions inherited from [Digikam::Rule](#)

- static QString **escapeToken** (const QString &token)  
*Escape the token characters to make them work in regular expressions.*

## Protected Slots inherited from [Digikam::Rule](#)

- virtual void **slotTokenTriggered** (const QString &)

## Protected Member Functions inherited from [Digikam::Rule](#)

- bool **addToken** (const QString &id, const QString &description, const QString &actionName=QString())  
*add a token to the parser, every parser should at least assign one token object*
- void **setDescription** (const QString &desc)
- void **setIcon** (const QString &pixmap)
- void **setRegExp** (const QRegularExpression &regExp)
- void **setUseTokenMenu** (bool value)

*If multiple tokens have been assigned to a rule, a menu will be created.*

## 9.323.1 Member Function Documentation

### 9.323.1.1 parseOperation()

```
QString Digikam::DefaultValueModifier::parseOperation (
    ParseSettings & settings,
    const QRegularExpressionMatch & match ) [override], [virtual]
```

#### Parameters

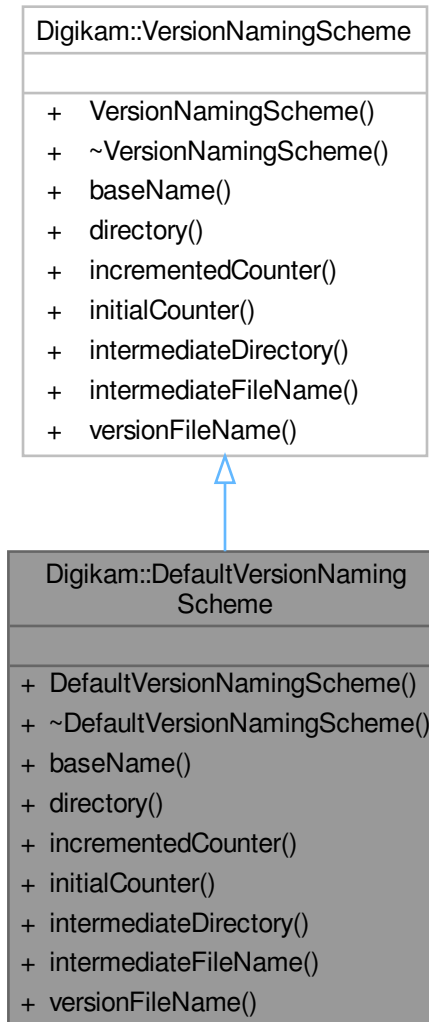
<i>settings</i>	contains settings
<i>match</i>	result of the regular expression match done in <code>Option::parse()</code>

#### Returns

Implements [Digikam::Modifier](#).

## 9.324 Digikam::DefaultVersionNamingScheme Class Reference

Inheritance diagram for Digikam::DefaultVersionNamingScheme:



### Public Member Functions

- virtual QString [baseName](#) (const QString &currentPath, const QString &filename, QVariant \*counter, QVariant \*intermediateCounter) override  
*Analyzes the given file name.*
- virtual QString [directory](#) (const QString &currentPath, const QString &filename) override  
*For a loaded file in directory path and with file name filename, returns the directory in which a new version (a new intermediate version, resp.) shall be stored.*
- virtual QVariant [incrementedCounter](#) (const QVariant &counter) override  
*Returns the given counter "incremented", that is, changed in a steady, repeatable fashion.*
- virtual QVariant [initialCounter](#) () override

Returns an initial counter value for version and intermediate number counters.

- virtual QString [intermediateDirectory](#) (const QString &currentPath, const QString &fileName) override
- virtual QString [intermediateFileName](#) (const QString &currentPath, const QString &filename, const QVariant &version, const QVariant &counter) override

Creates a version file name for an intermediate file in given directory, as previously returned by [directory\(\)](#), given baseName, as previously returned by [baseName](#), version and intermediate number counter.

- virtual QString [versionFileName](#) (const QString &currentPath, const QString &filename, const QVariant &counter) override

Creates a version file name for a file in given directory, as previously returned by [directory\(\)](#), given baseName, as previously returned by [baseName](#), and version counter.

## Public Member Functions inherited from [Digikam::VersionNamingScheme](#)

- [VersionNamingScheme](#) ()=default

Creates and analyzes file names of versioned files.

## 9.324.1 Member Function Documentation

### 9.324.1.1 [baseName\(\)](#)

```
QString Digikam::DefaultVersionNamingScheme::baseName (
    const QString & path,
    const QString & filename,
    QVariant * counter,
    QVariant * intermediateCounter ) [override], [virtual]
```

Returns the basename in the sense of stripping the file name of all added version information: A scheme that appends a number, like "MyFile-1.jpg", shall return "MyFile". Path is the directory, filename the file name, so path + filename is the file path. If counter is given, and the given file name has a version number, write it to counter. If intermediateCounter is given, and the given file name has an intermediate counter number, write it to counter. If not available, do not touch the given counters. See [initialCounter\(\)](#) for the valid counter formats.

Implements [Digikam::VersionNamingScheme](#).

### 9.324.1.2 [directory\(\)](#)

```
QString Digikam::DefaultVersionNamingScheme::directory (
    const QString & path,
    const QString & filename ) [override], [virtual]
```

Implements [Digikam::VersionNamingScheme](#).

### 9.324.1.3 [incrementedCounter\(\)](#)

```
QVariant Digikam::DefaultVersionNamingScheme::incrementedCounter (
    const QVariant & counter ) [override], [virtual]
```

You shall never return the given counter.

Implements [Digikam::VersionNamingScheme](#).



#### 9.324.1.4 initialCounter()

```
QVariant Digikam::DefaultVersionNamingScheme::initialCounter ( ) [override], [virtual]
```

There are two places where you shall generate counters You will receive the given QVariant in [incrementedCounter\(\)](#), [versionFileName\(\)](#) and [baseName\(\)](#), and you shall read a counter value from a generated file name in [baseName\(\)](#).

Implements [Digikam::VersionNamingScheme](#).

#### 9.324.1.5 intermediateDirectory()

```
QString Digikam::DefaultVersionNamingScheme::intermediateDirectory (
    const QString & currentPath,
    const QString & fileName ) [override], [virtual]
```

Implements [Digikam::VersionNamingScheme](#).

#### 9.324.1.6 intermediateFileName()

```
QString Digikam::DefaultVersionNamingScheme::intermediateFileName (
    const QString & path,
    const QString & filename,
    const QVariant & version,
    const QVariant & counter ) [override], [virtual]
```

Do not append a file suffix. You do not need to check if the file exists.

Implements [Digikam::VersionNamingScheme](#).

#### 9.324.1.7 versionFileName()

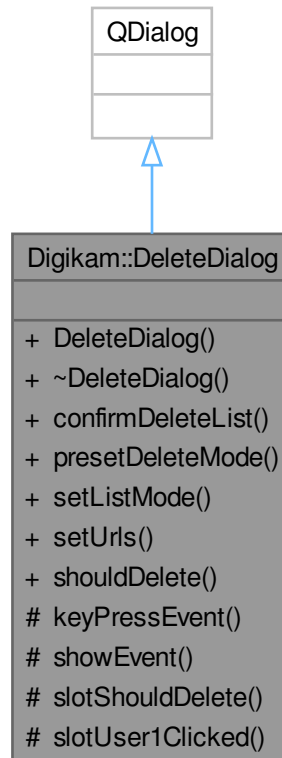
```
QString Digikam::DefaultVersionNamingScheme::versionFileName (
    const QString & path,
    const QString & baseName,
    const QVariant & counter ) [override], [virtual]
```

Do not append a file suffix. You do not need to check if the file exists.

Implements [Digikam::VersionNamingScheme](#).

## 9.325 Digikam::DeleteDialog Class Reference

Inheritance diagram for Digikam::DeleteDialog:



### Public Types

- enum **Mode** { **ModeFiles** , **ModeAlbums** , **ModeSubalbums** }

### Public Member Functions

- **DeleteDialog** (QWidget \*const parent)
- bool **confirmDeleteList** (const QList< QUrl > &condemnedURLs, DeleteDialogMode::ListMode listMode, DeleteDialogMode::DeleteMode deleteMode)
- void **presetDeleteMode** (DeleteDialogMode::DeleteMode mode)
- void **setListMode** (DeleteDialogMode::ListMode mode)
- void **setUrls** (const QList< QUrl > &urls)
- bool **shouldDelete** () const

### Protected Slots

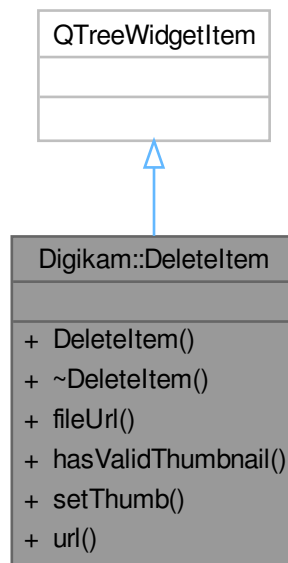
- void **slotShouldDelete** (bool)
- void **slotUser1Clicked** ()

### Protected Member Functions

- void **keyPressEvent** (QKeyEvent \*) override
- void **showEvent** (QShowEvent \*) override

## 9.326 Digikam::Deleteltem Class Reference

Inheritance diagram for Digikam::Deleteltem:

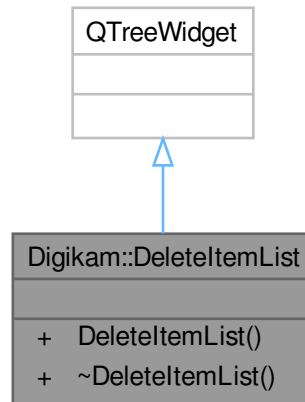


### Public Member Functions

- **Deleteltem** (QTreeWidgetItem \*const parent, const QUrl &url)
- QString **fileUrl** () const
- bool **hasValidThumbnail** () const
- void **setThumb** (const QPixmap &pix, bool hasThumb=true)
- QUrl **url** () const

## 9.327 Digikam::DeleteltemList Class Reference

Inheritance diagram for Digikam::DeleteltemList:

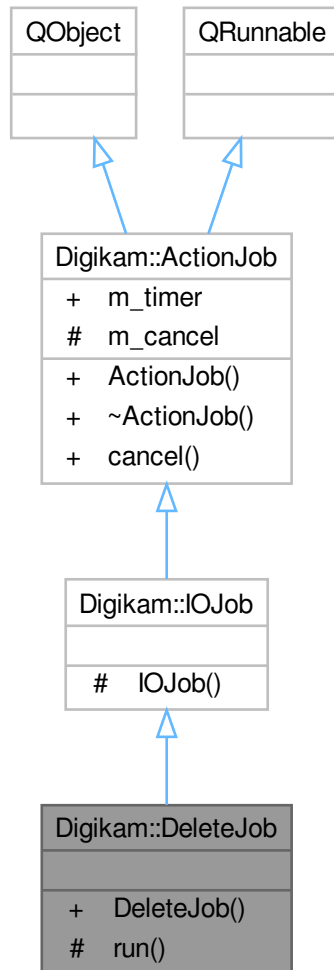


### Public Member Functions

- `DeleteltemList` (`QWidget *const parent=nullptr`)

## 9.328 Digikam::DeleteJob Class Reference

Inheritance diagram for Digikam::DeleteJob:



### Public Member Functions

- **DeleteJob** ([IOJobData](#) \*const data)

### Public Member Functions inherited from [Digikam::ActionJob](#)

- **ActionJob** ([QObject](#) \*const parent=nullptr)
  - Constructor which delegate deletion of [QRunnable](#) instance to [ActionThreadBase](#), not [QThreadPool](#).*
- **~ActionJob** () override
  - Re-implement destructor in you implementation.*

### Protected Member Functions

- void **run** () override

### Additional Inherited Members

### Public Slots inherited from [Digikam::ActionJob](#)

- void **cancel** ()  
*Call this method to cancel job.*

### Signals inherited from [Digikam::IOJob](#)

- void **signalError** (const QString &errMsg)
- void **signalOneProcessed** (const QUrl &url)

### Signals inherited from [Digikam::ActionJob](#)

- void **signalDone** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job is done.*
- void **signalProgress** (int)  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job progress.*
- void **signalStarted** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager that job is started.*

### Public Attributes inherited from [Digikam::ActionJob](#)

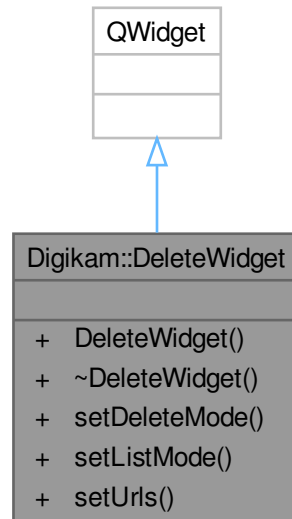
- QElapsedTimer **m\_timer**  
*Timer to determine the running time of the job.*

### Protected Attributes inherited from [Digikam::ActionJob](#)

- bool **m\_cancel** = false  
*You can use this boolean in your implementation to know if job must be canceled.*

## 9.329 Digikam::DeleteWidget Class Reference

Inheritance diagram for Digikam::DeleteWidget:



### Public Member Functions

- **DeleteWidget** (`QWidget *const parent=nullptr`)
- void **setDeleteMode** (`DeleteDialogMode::DeleteMode deleteMode`)
- void **setListMode** (`DeleteDialogMode::ListMode mode`)
- void **setUrls** (`const QList< QUrl > &urls`)

### Friends

- class **DeleteDialog**

## 9.330 Digikam::DeltaTime Class Reference

Container that hold the time difference for clock photo dialog.

### Public Member Functions

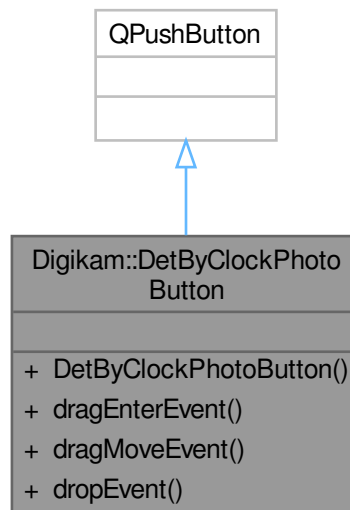
- bool **isNull** () const  
*Check if at least one option is selected.*

### Public Attributes

- int **deltaDays** = 0
- int **deltaHours** = 0
- int **deltaMinutes** = 0
- bool **deltaNegative** = false
- int **deltaSeconds** = 0

## 9.331 Digikam::DetByClockPhotoButton Class Reference

Inheritance diagram for Digikam::DetByClockPhotoButton:



### Signals

- void **signalClockPhotoDropped** (const QUrl &)

### Public Member Functions

- **DetByClockPhotoButton** (const QString &text)
- void **dragEnterEvent** (QDragEnterEvent \*event) override
- void **dragMoveEvent** (QDragMoveEvent \*event) override
- void **dropEvent** (QDropEvent \*event) override



## 9.332 Digikam::DetectionBenchmarker Class Reference

Inheritance diagram for Digikam::DetectionBenchmarker:



### Public Slots

- void **process** (const FacePipelineExtendedPackage::Ptr &package)

## Public Slots inherited from [Digikam::WorkerObject](#)

- void [deactivate](#) ([DeactivatingMode](#) mode=[FlushSignals](#))  
*Quits execution of this worker object.*
- void [schedule](#) ()  
*Starts execution of this worker object: The object is moved to a thread and an event loop started, so that queued signals will be received.*

## Signals

- void [processed](#) (const [FacePipelineExtendedPackage::Ptr](#) &package)

## Signals inherited from [Digikam::WorkerObject](#)

- void [finished](#) ()
- void [started](#) ()

## Public Member Functions

- [DetectionBenchmark](#) ([FacePipeline::Private](#) \*const d)
- [QString](#) [result](#) () const  
*NOTE: Bench performance code.*

## Public Member Functions inherited from [Digikam::WorkerObject](#)

- [WorkerObject](#) ()  
*Deriving from a worker object allows you to execute your slots in a thread.*
- bool [connectAndSchedule](#) (const [QObject](#) \*sender, const char \*signal, const char \*method, [Qt::](#)↔[ConnectionType](#) type=[Qt::AutoConnection](#)) const  
*You must normally call [schedule\(\)](#) to ensure that the object is active when you send a signal with work data.*
- [QThread::Priority](#) [priority](#) () const
- void [setPriority](#) ([QThread::Priority](#) priority)  
*Sets the priority for this dynamic thread.*
- State [state](#) () const
- void [wait](#) ()

## Protected Attributes

- [FacePipeline::Private](#) \*const **d** = nullptr
- double **facePixels** = 0.0
- int **faces** = 0
- int **falseNegativeFaces** = 0
- int **falsePositiveFaces** = 0
- int **falsePositiveImages** = 0
- int **totalImages** = 0
- double **totalPixels** = 0.0
- int **trueNegativeImages** = 0
- int **truePositiveFaces** = 0

## Additional Inherited Members

### Public Types inherited from [Digikam::WorkerObject](#)

- enum [DeactivatingMode](#) { [FlushSignals](#) , [KeepSignals](#) , [PhaseOut](#) }
- enum [State](#) { [Inactive](#) , [Scheduled](#) , [Running](#) , [Deactivating](#) }

### Static Public Member Functions inherited from [Digikam::WorkerObject](#)

- static bool **connectAndSchedule** (const QObject \*sender, const char \*signal, const [WorkerObject](#) \*receiver, const char \*method, Qt::ConnectionType type=Qt::AutoConnection)
- static bool **disconnectAndSchedule** (const QObject \*sender, const char \*signal, const [WorkerObject](#) \*receiver, const char \*method)

### Protected Member Functions inherited from [Digikam::WorkerObject](#)

- virtual void [aboutToDeactivate](#) ()  
*Called from [deactivate\(\)](#), typically from a different thread than the worker thread, possibly the UI thread.*
- virtual void [aboutToQuitLoop](#) ()  
*Called from within thread's event loop to quit processing.*
- void **addRunnable** (WorkerObjectRunnable \*loop)
- bool **event** (QEvent \*e) override
- void **removeRunnable** (WorkerObjectRunnable \*loop)
- void **run** ()
- void **setEventLoop** (QEventLoop \*loop)
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void **transitionToInactive** ()
- bool **transitionToRunning** ()

## 9.332.1 Member Function Documentation

### 9.332.1.1 result()

```
QString Digikam::DetectionBenchmarker::result ( ) const
```

No need i18n here

### 9.333 Digikam::DetectionWorker Class Reference

Inheritance diagram for Digikam::DetectionWorker:



#### Public Slots

- void **process** (const `FacePipelineExtendedPackage::Ptr` &package)
- void **setAccuracyAndModel** (int detectAccuracy, [FaceScanSettings::FaceDetectionModel](#) detectModel, [FaceScanSettings::FaceDetectionSize](#) detectSize, int recognizeAccuracy, [FaceScanSettings::FaceRecognitionModel](#) recognizeModel)

## Public Slots inherited from [Digikam::WorkerObject](#)

- void **deactivate** ([DeactivatingMode](#) mode=[FlushSignals](#))  
*Quits execution of this worker object.*
- void **schedule** ()  
*Starts execution of this worker object: The object is moved to a thread and an event loop started, so that queued signals will be received.*

## Signals

- void **processed** (const [FacePipelineExtendedPackage::Ptr](#) &package)

## Signals inherited from [Digikam::WorkerObject](#)

- void **finished** ()
- void **started** ()

## Public Member Functions

- **DetectionWorker** ([FacePipeline::Private](#) \*const dd)
- [QImage](#) **scaleForDetection** (const [DImg](#) &image) const

## Public Member Functions inherited from [Digikam::WorkerObject](#)

- [WorkerObject](#) ()  
*Deriving from a worker object allows you to execute your slots in a thread.*
- bool **connectAndSchedule** (const [QObject](#) \*sender, const char \*signal, const char \*method, [Qt::](#)↔[ConnectionType](#) type=[Qt::AutoConnection](#)) const  
*You must normally call [schedule\(\)](#) to ensure that the object is active when you send a signal with work data.*
- [QThread::Priority](#) **priority** () const
- void **setPriority** ([QThread::Priority](#) priority)  
*Sets the priority for this dynamic thread.*
- State **state** () const
- void **wait** ()

## Protected Attributes

- [FacePipeline::Private](#) \*const **d** = nullptr
- [FaceDetector](#) **detector**

## Additional Inherited Members

## Public Types inherited from [Digikam::WorkerObject](#)

- enum [DeactivatingMode](#) { [FlushSignals](#) , [KeepSignals](#) , [PhaseOut](#) }
- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

## Static Public Member Functions inherited from [Digikam::WorkerObject](#)

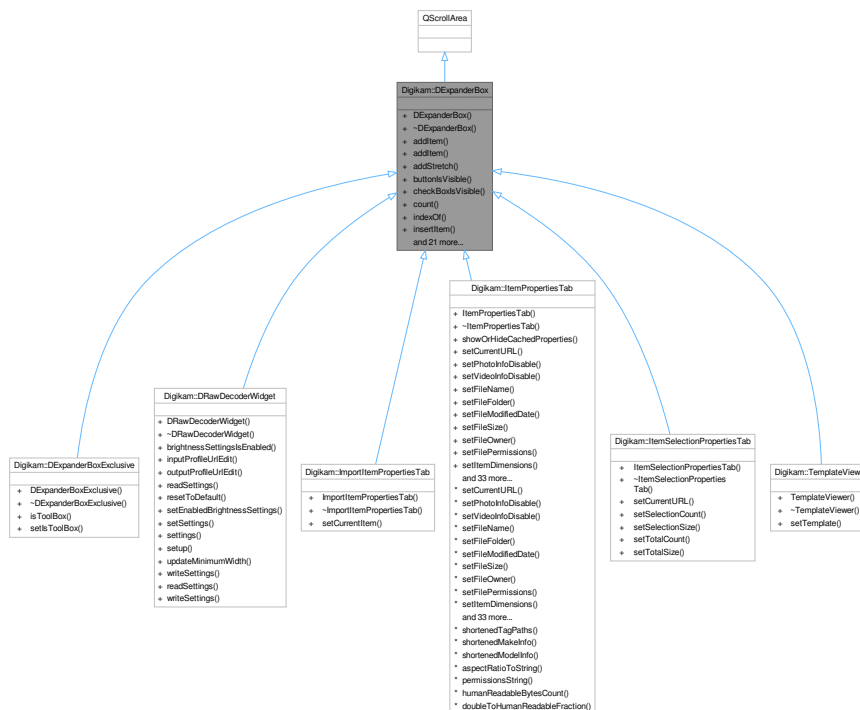
- static bool **connectAndSchedule** (const QObject \*sender, const char \*signal, const [WorkerObject](#) \*receiver, const char \*method, Qt::ConnectionType type=Qt::AutoConnection)
- static bool **disconnectAndSchedule** (const QObject \*sender, const char \*signal, const [WorkerObject](#) \*receiver, const char \*method)

## Protected Member Functions inherited from [Digikam::WorkerObject](#)

- virtual void **aboutToDeactivate** ()  
*Called from `deactivate()`, typically from a different thread than the worker thread, possibly the UI thread.*
- virtual void **aboutToQuitLoop** ()  
*Called from within thread's event loop to quit processing.*
- void **addRunnable** (WorkerObjectRunnable \*loop)
- bool **event** (QEvent \*e) override
- void **removeRunnable** (WorkerObjectRunnable \*loop)
- void **run** ()
- void **setEventLoop** (QEventLoop \*loop)
- void **shutDown** ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call `stop()` and `wait()`, knowing that nothing will call `start()` anymore after this 3) Be sure the thread will never be running at destruction.*
- void **transitionToInactive** ()
- bool **transitionToRunning** ()

## 9.334 Digikam::DExpanderBox Class Reference

Inheritance diagram for [Digikam::DExpanderBox](#):



## Signals

- void **signalItemButtonPressed** (int index)
- void **signalItemExpanded** (int index, bool b)
- void **signalItemToggled** (int index, bool b)

## Public Member Functions

- **DExpanderBox** (QWidget \*const parent=nullptr)
- void **addItem** (QWidget \*const w, const QIcon &icon, const QString &txt, const QString &objName, bool expandBydefault)
  - Add *DLabelExpander* item at end of box layout with these settings : 'w' : the widget hosted by *DLabelExpander*.
- void **addItem** (QWidget \*const w, const QString &txt, const QString &objName, bool expandBydefault)
- void **addStretch** ()
- bool **buttonIsVisible** (int index) const
- bool **checkboxIsVisible** (int index) const
- int **count** () const
- int **indexOf** (*DLabelExpander* \*const widget) const
- void **insertItem** (int index, QWidget \*const w, const QIcon &icon, const QString &txt, const QString &objName, bool expandBydefault)
  - Insert *DLabelExpander* item at box layout index with these settings : 'w' : the widget hosted by *DLabelExpander*.
- void **insertItem** (int index, QWidget \*const w, const QString &txt, const QString &objName, bool expandBydefault)
- void **insertStretch** (int index)
- bool **isChecked** (int index) const
- bool **isItemEnabled** (int index) const
- bool **isItemExpanded** (int index) const
- QIcon **itemIcon** (int index) const
- QString **itemText** (int index) const
- QString **itemToolTip** (int index) const
- virtual void **readSettings** (KConfigGroup &group)
- void **removeItem** (int index)
- void **setButtonIcon** (int index, const QIcon &icon)
- void **setButtonVisible** (int index, bool b)
- void **setCheckBoxVisible** (int index, bool b)
- void **setChecked** (int index, bool b)
- void **setItemEnabled** (int index, bool enabled)
- void **setItemExpanded** (int index, bool b)
- void **setItemIcon** (int index, const QIcon &icon)
- void **setItemText** (int index, const QString &txt)
- void **setItemToolTip** (int index, const QString &tip)
- *DLabelExpander* \* **widget** (int index) const
- virtual void **writeSettings** (KConfigGroup &group)

### 9.334.1 Member Function Documentation

#### 9.334.1.1 addItem()

```
void Digikam::DExpanderBox::addItem (
    QWidget *const w,
    const QIcon & icon,
    const QString & txt,
    const QString & objName,
    bool expandBydefault )
```

'pix' : pixmap used as icon to item title. 'txt' : text used as item title. 'objName' : item object name used to read/save expanded settings to rc file. 'expandBydefault' : item state by default (expanded or not).

### 9.334.1.2 insertItem()

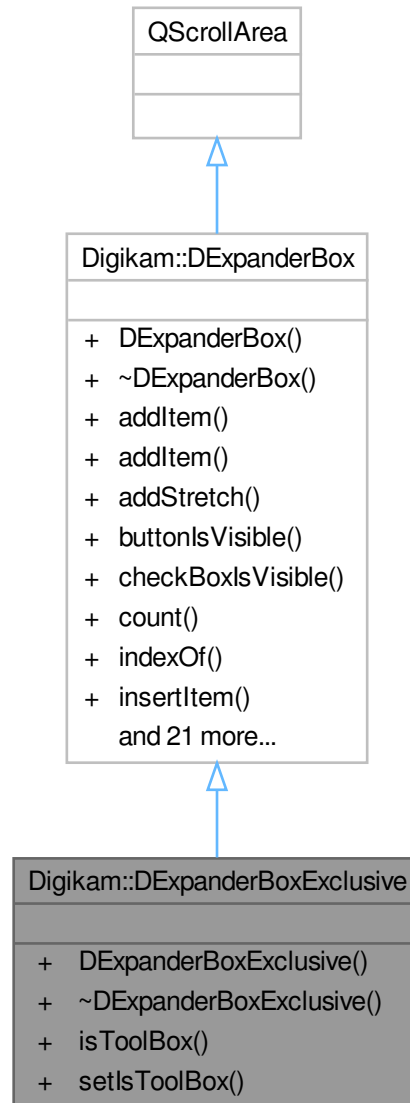
```
void Digikam::DExpanderBox::insertItem (
    int index,
    QWidget *const w,
    const QIcon & icon,
    const QString & txt,
    const QString & objName,
    bool expandBydefault )
```

'pix' : pixmap used as icon to item title. 'txt' : text used as item title. 'objName' : item object name used to read/save expanded settings to rc file. 'expandBydefault' : item state by default (expanded or not).



## 9.335 Digikam::DExpanderBoxExclusive Class Reference

Inheritance diagram for Digikam::DExpanderBoxExclusive:



### Public Member Functions

- **DExpanderBoxExclusive** (QWidget \*const parent=nullptr)
- bool **isToolBox** () const
- void **setIsToolBox** (bool b)

*Show one expander open at most.*

## Public Member Functions inherited from [Digikam::DExpanderBox](#)

- **DExpanderBox** (QWidget \*const parent=nullptr)
- void **addItem** (QWidget \*const w, const QIcon &icon, const QString &txt, const QString &objName, bool expandBydefault)
- Add [DLabelExpander](#) item at end of box layout with these settings : 'w' : the widget hosted by [DLabelExpander](#).*
- void **addItem** (QWidget \*const w, const QString &txt, const QString &objName, bool expandBydefault)
- void **addStretch** ()
- bool **buttonIsVisible** (int index) const
- bool **checkboxIsVisible** (int index) const
- int **count** () const
- int **indexOf** ([DLabelExpander](#) \*const widget) const
- void **insertItem** (int index, QWidget \*const w, const QIcon &icon, const QString &txt, const QString &objName, bool expandBydefault)
- Insert [DLabelExpander](#) item at box layout index with these settings : 'w' : the widget hosted by [DLabelExpander](#).*
- void **insertItem** (int index, QWidget \*const w, const QString &txt, const QString &objName, bool expandBydefault)
- void **insertStretch** (int index)
- bool **isChecked** (int index) const
- bool **isItemEnabled** (int index) const
- bool **isItemExpanded** (int index) const
- QIcon **itemIcon** (int index) const
- QString **itemText** (int index) const
- QString **itemToolTip** (int index) const
- virtual void **readSettings** (KConfigGroup &group)
- void **removeItem** (int index)
- void **setButtonIcon** (int index, const QIcon &icon)
- void **setButtonVisible** (int index, bool b)
- void **setCheckBoxVisible** (int index, bool b)
- void **setChecked** (int index, bool b)
- void **setItemEnabled** (int index, bool enabled)
- void **setItemExpanded** (int index, bool b)
- void **setItemIcon** (int index, const QIcon &icon)
- void **setItemText** (int index, const QString &txt)
- void **setItemToolTip** (int index, const QString &tip)
- [DLabelExpander](#) \* **widget** (int index) const
- virtual void **writeSettings** (KConfigGroup &group)

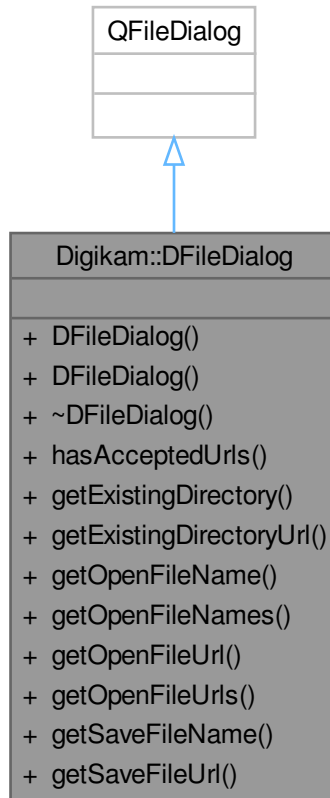
## Additional Inherited Members

## Signals inherited from [Digikam::DExpanderBox](#)

- void **signalItemButtonPressed** (int index)
- void **signalItemExpanded** (int index, bool b)
- void **signalItemToggled** (int index, bool b)

## 9.336 Digikam::DFileDialog Class Reference

Inheritance diagram for Digikam::DFileDialog:



### Public Member Functions

- **DFileDialog** (`QWidget *const parent`, `Qt::WindowFlags flags`)
- **DFileDialog** (`QWidget *const parent=nullptr`, `const QString &caption=QString()`, `const QString &directory=QString()`, `const QString &filter=QString()`)
- `bool hasAcceptedUrls () const`

### Static Public Member Functions

- `static QString getExistingDirectory (QWidget *const parent=nullptr, const QString &caption=QString(), const QString &dir=QString(), Options options=ShowDirsOnly)`
- `static QUrl getExistingDirectoryUrl (QWidget *const parent=nullptr, const QString &caption=QString(), const QUrl &dir=QUrl(), Options options=ShowDirsOnly, const QStringList &supportedSchemes=QStringList())`
- `static QString getOpenFileName (QWidget *const parent=nullptr, const QString &caption=QString(), const QString &dir=QString(), const QString &filter=QString(), QString *selectedFilter=nullptr, Options options=Options())`

- static QStringList **getOpenFileNames** (QWidget \*const parent=nullptr, const QString &caption=QString(), const QString &dir=QString(), const QString &filter=QString(), QString \*selectedFilter=nullptr, Options options=Options())
- static QUrl **getOpenFileUrl** (QWidget \*const parent=nullptr, const QString &caption=QString(), const QUrl &dir=QUrl(), const QString &filter=QString(), QString \*selectedFilter=nullptr, Options options=Options(), const QStringList &supportedSchemes=QStringList())
- static QList< QUrl > **getOpenFileUrls** (QWidget \*const parent=nullptr, const QString &caption=QString(), const QUrl &dir=QUrl(), const QString &filter=QString(), QString \*selectedFilter=nullptr, Options options=Options(), const QStringList &supportedSchemes=QStringList())
- static QString **getSaveFileName** (QWidget \*const parent=nullptr, const QString &caption=QString(), const QString &dir=QString(), const QString &filter=QString(), QString \*selectedFilter=nullptr, Options options=Options())
- static QUrl **getSaveFileUrl** (QWidget \*const parent=nullptr, const QString &caption=QString(), const QUrl &dir=QUrl(), const QString &filter=QString(), QString \*selectedFilter=nullptr, Options options=Options(), const QStringList &supportedSchemes=QStringList())

## 9.337 Digikam::DFileOperations Class Reference

### Public Types

- enum **SidecarAction** { **Rename** = 0 , **Copy** }

### Static Public Member Functions

- static bool **copyFile** (const QString &srcFile, const QString &dstFile, const bool \*const cancel=nullptr)  
*Copy file and keep the source file modification time.*
- static bool **copyFiles** (const QStringList &srcPaths, const QString &dstPath)  
*Copy a list of files to another place.*
- static bool **copyFolderRecursively** (const QString &srcPath, const QString &dstPath, const QString &item←Id=QString(), bool \*const cancel=nullptr, bool useDstPath=false)  
*Copy recursively a directory contents to another one.*
- static bool **copyModificationTime** (const QString &srcFile, const QString &dstFile)  
*Copy file modification time from source to destination file.*
- static QString **findExecutable** (const QString &name, const QStringList &hints=QStringList())  
*Returns the path to a program under Windows by searching in the Windows registry.*
- static QUrl **getUniqueFileUrl** (const QUrl &orgUrl, bool \*const newurl=nullptr)  
*Get unique file url if file exist by appending a counter suffix or return original url.*
- static QUrl **getUniqueFolderUrl** (const QUrl &orgUrl)  
*Get unique folder url if folder exist by appending a counter suffix or return original url.*
- static bool **localFileRename** (const QString &source, const QString &orgPath, const QString &destPath, bool ignoreSettings=false)  
*This method rename a local file 'orgPath' to 'destPath' with all ACL properties restoration taken from 'source' file.*
- static void **openFilesWithDefaultApplication** (const QList< QUrl > &urls)  
*Open file urls to default application relevant of file type-mimes desktop configuration.*
- static void **openInFileManager** (const QList< QUrl > &urls)  
*Open system file manager and select the item.*
- static bool **removeAndCopyFile** (const QString &srcFile, const QString &dstFile)  
*If the destination file already exists, it will be removed.*
- static bool **renameFile** (const QString &srcFile, const QString &dstFile)  
*Rename or move file and keep the source file modification time.*
- static bool **setModificationTime** (const QString &srcFile, const QDateTime &dateTime)  
*Set file modification time from QDateTime.*
- static bool **sidecarFiles** (const QString &srcFile, const QString &dstFile, SidecarAction action)  
*Rename/move or copy all possible sidecar files and keep the source file modification time.*

## 9.337.1 Member Function Documentation

### 9.337.1.1 findExecutable()

```
QString Digikam::DFileOperations::findExecutable (
    const QString & name,
    const QStringList & hints = QStringList() ) [static]
```

If the path is empty, QStandardPaths::findExecutable() is used as under Linux and macOS.

### 9.337.1.2 localFileRename()

```
bool Digikam::DFileOperations::localFileRename (
    const QString & source,
    const QString & orgPath,
    const QString & destPath,
    bool ignoreSettings = false ) [static]
```

Return true if operation is completed.

### 9.337.1.3 removeAndCopyFile()

```
bool Digikam::DFileOperations::removeAndCopyFile (
    const QString & srcFile,
    const QString & dstFile ) [static]
```

Copy file and keep the source file modification time.

### 9.337.1.4 setModificationTime()

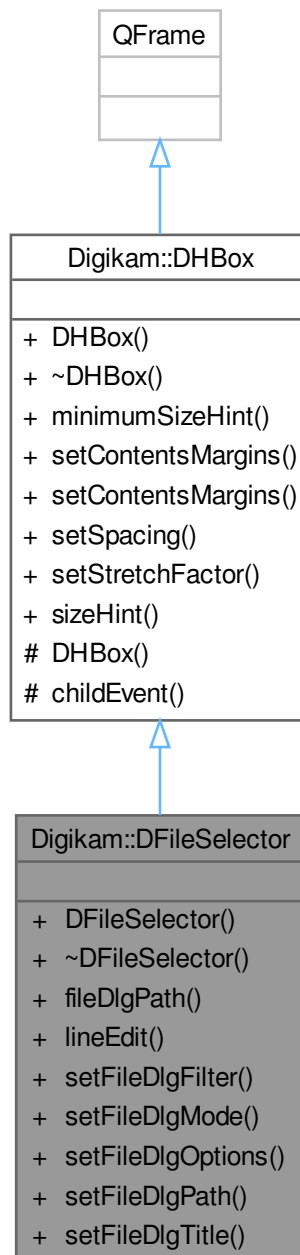
```
bool Digikam::DFileOperations::setModificationTime (
    const QString & srcFile,
    const QDateTime & dateTime ) [static]
```

Keep access time from source file.

## 9.338 Digikam::DFileSelector Class Reference

A widget to choose a single local file or path.

Inheritance diagram for Digikam::DFileSelector:



### Signals

- void **signalOpenFileDialog** ()
- void **signalUrlSelected** (const `QUrl` &)

### Public Member Functions

- **DFileSelector** (`QWidget *const parent=nullptr`)

- QString **fileDlgPath** () const
- QLineEdit \* **lineEdit** () const
- void **setFileDlgFilter** (const QString &filter)
- void **setFileDlgMode** (QFileDialog::FileMode mode)
- void **setFileDlgOptions** (QFileDialog::Options opts)
- void **setFileDlgPath** (const QString &path)
- void **setFileDlgTitle** (const QString &title)

### Public Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (QWidget \*const parent=nullptr)
- QSize **minimumSizeHint** () const override
- void **setContentsMargins** (const QMargins &margins)
- void **setContentsMargins** (int left, int top, int right, int bottom)
- void **setSpacing** (int space)
- void **setStretchFactor** (QWidget \*const widget, int stretch)
- QSize **sizeHint** () const override

### Additional Inherited Members

### Protected Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (bool vertical, QWidget \*const parent)
- void **childEvent** (QChildEvent \*e) override

## 9.338.1 Detailed Description

Use line edit and file dialog properties to customize operation modes.

## 9.339 Digikam::DFontProperties Class Reference

Inheritance diagram for Digikam::DFontProperties:



### Public Types

- enum [DisplayFlag](#) { **NoDisplayFlags** = 0 , **FixedFontsOnly** = 1 , **DisplayFrame** = 2 , **ShowDifferences** = 4 }
- typedef QFlags< [DisplayFlag](#) > **DisplayFlags**
- enum [FontColumn](#) { **FamilyList** = 0x01 , **StyleList** = 0x02 , **SizeList** = 0x04 }
- enum [FontDiff](#) { **NoFontDiffFlags** = 0 , **FontDiffFamily** = 1 , **FontDiffStyle** = 2 , **FontDiffSize** = 4 , **AllFontDiffs** = FontDiffFamily | FontDiffStyle | FontDiffSize }
- typedef QFlags< [FontDiff](#) > **FontDiffFlags**
- enum [FontListCriteria](#) { **FixedWidthFonts** = 0x01 , **ScalableFonts** = 0x02 , **SmoothScalableFonts** = 0x04 }

*The selection criteria for the font families shown in the dialog.*



## Signals

- void **fontSelected** (const QFont &font)  
*Emitted whenever the selected font changes.*

## Public Member Functions

- [DFontProperties](#) (QWidget \*const parent=nullptr, const DisplayFlags &flags=DisplayFrame, const QStringList &fontList=QStringList(), int visibleListSize=8, Qt::CheckState \*const sizelsRelativeState=nullptr)  
*Constructs a font picker widget.*
- [~DFontProperties](#) () override  
*Destructs the font chooser.*
- QColor [backgroundColor](#) () const
- QColor [color](#) () const
- void [enableColumn](#) (int column, bool state)  
*Enables or disable a font column in the chooser.*
- QFont [font](#) () const
- FontDiffFlags [fontDiffFlags](#) () const
- void [makeColumnVisible](#) (int column, bool state)  
*Makes a font column in the chooser visible or invisible.*
- QString [sampleText](#) () const
- void [setBackgroundColor](#) (const QColor &col)  
*Sets the background color to use in the preview.*
- void [setColor](#) (const QColor &col)  
*Sets the color to use in the preview.*
- void [setFont](#) (const QFont &font, bool onlyFixed=false)  
*Sets the currently selected font in the chooser.*
- void [setSampleBoxVisible](#) (bool visible)  
*Shows or hides the sample text box.*
- void [setSampleText](#) (const QString &text)  
*Sets the sample text.*
- void [setSizeRelative](#) (Qt::CheckState relative)  
*Sets the state of the checkbox indicating whether the font size is to be interpreted as relative size.*
- QSize [sizeHint](#) (void) const override  
*Reimplemented for internal reasons.*
- Qt::CheckState [sizelsRelative](#) () const

## Static Public Member Functions

- static void [getFontList](#) (QStringList &list, uint fontListCriteria)  
*Creates a list of font strings.*

## Properties

- QColor **backgroundColor**
- QColor **color**
- QFont **font**
- QString **sampleText**
- Qt::CheckState **sizelsRelative**

## 9.339.1 Member Enumeration Documentation

### 9.339.1.1 DisplayFlag

enum `Digikam::DFontProperties::DisplayFlag`

- `FixedFontsOnly` only show fixed fonts, excluding proportional fonts
- `DisplayFrame` show a visual frame around the chooser
- `ShowDifferences` display the font differences interfaces

### 9.339.1.2 FontColumn

enum `Digikam::DFontProperties::FontColumn`

- `FamilyList` - Identifies the family (leftmost) list.
- `StyleList` - Identifies the style (center) list.
- `SizeList` - Identifies the size (rightmost) list.

### 9.339.1.3 FontDiff

enum `Digikam::DFontProperties::FontDiff`

- `FontDiffFamily` - Identifies a requested change in the font family.
- `FontDiffStyle` - Identifies a requested change in the font style.
- `FontDiffSize` - Identifies a requested change in the font size.

### 9.339.1.4 FontListCriteria

enum `Digikam::DFontProperties::FontListCriteria`

- `FixedWidthFont` when included only fixed-width fonts are returned. The fonts where the width of every character is equal.
- `ScalableFont` when included only scalable fonts are returned; certain configurations allow bitmap fonts to remain unscaled and thus these fonts have limited number of sizes.
- `SmoothScalableFont` when included only return smooth scalable fonts. this will return only non-bitmap fonts which are scalable to any size requested. Setting this option to true will mean the "scalable" flag is irrelevant.

## 9.339.2 Constructor & Destructor Documentation

### 9.339.2.1 DFontProperties()

```
Digikam::DFontProperties::DFontProperties (
    QWidget *const parent = nullptr,
    const DisplayFlags & flags = DisplayFrame,
    const QStringList & fontList = QStringList(),
    int visibleListSize = 8,
    Qt::CheckState *const sizeIsRelativeState = nullptr ) [explicit]
```

It normally comes up with all font families present on the system; the `getFont` method below does allow some more fine-tuning of the selection of fonts that will be displayed in the dialog.

## Parameters

<i>parent</i>	The parent widget.
<i>flags</i>	Defines how the font chooser is displayed.

## See also

DisplayFlags

## Parameters

<i>fontList</i>	A list of fonts to display, in XLFD format.
<i>visibleListSize</i>	The minimum number of visible entries in the fontlists.
<i>sizelsRelativeState</i>	If not zero the widget will show a checkbox where the user may choose whether the font size is to be interpreted as relative size. Initial state of this checkbox will be set according to *sizelsRelativeState, user choice may be retrieved by calling sizelsRelative().

### 9.339.3 Member Function Documentation

#### 9.339.3.1 backgroundColor()

```
QColor Digikam::DFontProperties::backgroundColor ( ) const
```

## Returns

The background color currently used in the preview (default: the base color of the active colorgroup)

#### 9.339.3.2 color()

```
QColor Digikam::DFontProperties::color ( ) const
```

## Returns

The color currently used in the preview (default: the text color of the active color group)

#### 9.339.3.3 enableColumn()

```
void Digikam::DFontProperties::enableColumn (
    int column,
    bool state )
```

Use this function if your application does not need or supports all font properties.

## Parameters

<i>column</i>	Specify the columns. An or'ed combination of <code>FamilyList</code> , <code>StyleList</code> and <code>SizeList</code> is possible.
<i>state</i>	If <code>false</code> the columns are disabled.

**9.339.3.4 font()**

```
QFont Digikam::DFontProperties::font ( ) const
```

## Returns

The currently selected font in the chooser.

**9.339.3.5 fontDiffFlags()**

```
DFontProperties::FontDiffFlags Digikam::DFontProperties::fontDiffFlags ( ) const
```

## Returns

The bitmask corresponding to the attributes the user wishes to change.

**9.339.3.6 getFontList()**

```
void Digikam::DFontProperties::getFontList (
    QStringList & list,
    uint fontListCriteria ) [static]
```

## Parameters

<i>list</i>	The list is returned here.
<i>fontListCriteria</i>	should contain all the restrictions for font selection as OR-ed values

## See also

[DFontProperties::FontListCriteria](#) for the individual values

**9.339.3.7 makeColumnVisible()**

```
void Digikam::DFontProperties::makeColumnVisible (
    int column,
    bool state )
```

Use this function if your application does not need to show all font properties.

## Parameters

<i>column</i>	Specify the columns. An or'ed combination of <code>FamilyList</code> , <code>StyleList</code> and <code>SizeList</code> is possible.
<i>state</i>	If <code>false</code> the columns are made invisible.

**9.339.3.8 sampleText()**

```
QString Digikam::DFontProperties::sampleText ( ) const
```

## Returns

The current text in the sample text input area.

**9.339.3.9 setFont()**

```
void Digikam::DFontProperties::setFont (
    const QFont & font,
    bool onlyFixed = false )
```

## Parameters

<i>font</i>	The font to select.
<i>onlyFixed</i>	Readjust the font list to display only fixed width fonts if <code>true</code> , or vice-versa.

**9.339.3.10 setSampleBoxVisible()**

```
void Digikam::DFontProperties::setSampleBoxVisible (
    bool visible )
```

## Parameters

<i>visible</i>	Set it to true to show the box, to false to hide it.
----------------	--

**9.339.3.11 setSampleText()**

```
void Digikam::DFontProperties::setSampleText (
    const QString & text )
```

Normally you should not change this text, but it can be better to do this if the default text is too large for the edit area when using the default font of your application.

## Parameters

<i>text</i>	The new sample text. The current will be removed.
-------------	---

**9.339.3.12 setSizelsRelative()**

```
void Digikam::DFontProperties::setSizeIsRelative (
    Qt::CheckState relative )
```

**Note**

If parameter `sizelsRelative` was not set in the constructor of the widget this setting will be ignored.

**9.339.3.13 sizelsRelative()**

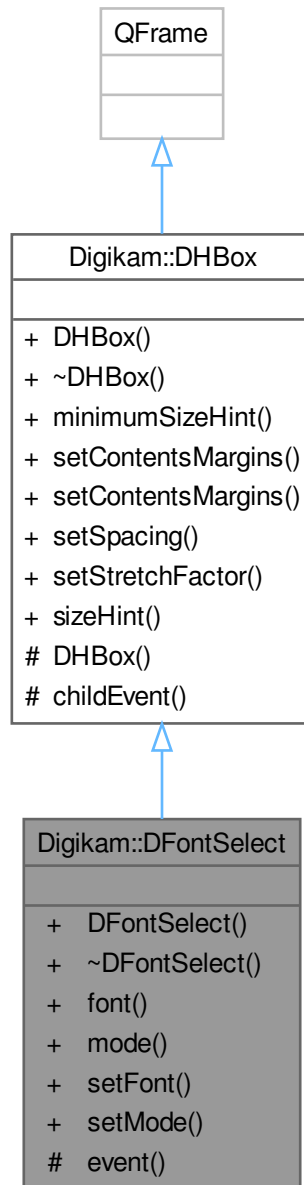
```
Qt::CheckState Digikam::DFontProperties::sizeIsRelative ( ) const
```

**Returns**

Whether the font size is to be interpreted as relative size (default: `QButton::Off`)

## 9.340 Digikam::DFontSelect Class Reference

Inheritance diagram for Digikam::DFontSelect:



### Public Types

- enum `FontMode` { `SystemFont = 0` , `CustomFont` }

### Signals

- void `signalFontChanged ()`

### Public Member Functions

- **DFontSelect** (const QString &text, QWidget \*const parent=nullptr)
- QFont **font** () const
- FontMode **mode** () const
- void **setFont** (const QFont &font)
- void **setMode** (FontMode mode)

### Public Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (QWidget \*const parent=nullptr)
- QSize **minimumSizeHint** () const override
- void **setContentsMargins** (const QMargins &margins)
- void **setContentsMargins** (int left, int top, int right, int bottom)
- void **setSpacing** (int space)
- void **setStretchFactor** (QWidget \*const widget, int stretch)
- QSize **sizeHint** () const override

### Protected Member Functions

- bool **event** (QEvent \*e) override

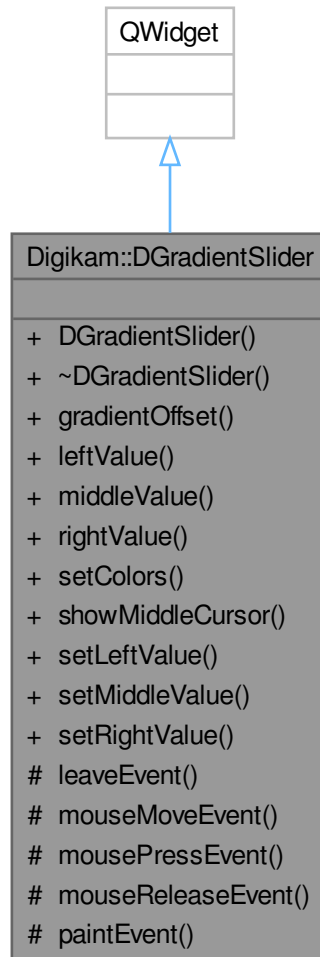
### Protected Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (bool vertical, QWidget \*const parent)
- void **childEvent** (QChildEvent \*e) override



## 9.341 Digikam::DGradientSlider Class Reference

Inheritance diagram for Digikam::DGradientSlider:



### Public Slots

- void **setLeftValue** (double)
- void **setMiddleValue** (double)
- void **setRightValue** (double)

### Signals

- void **leftValueChanged** (double)
- void **middleValueChanged** (double)
- void **rightValueChanged** (double)

## Public Member Functions

- **DGradientSlider** (QWidget \*const parent=nullptr)
- int **gradientOffset** () const
- double **leftValue** () const
- double **middleValue** () const
- double **rightValue** () const
- void **setColors** (const QColor &lcolor, const QColor &rcolor)
- void **showMiddleCursor** (bool b)

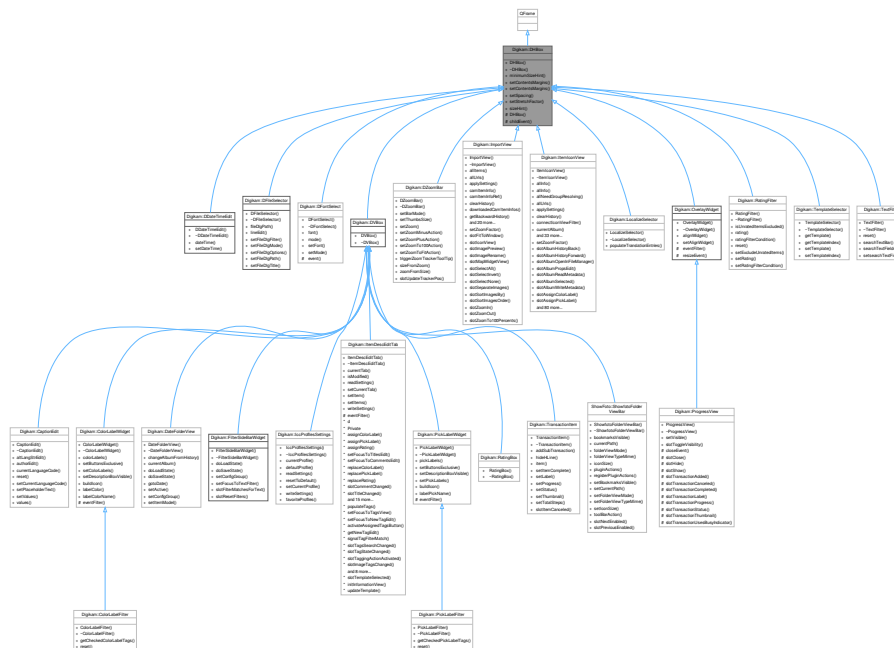
## Protected Member Functions

- void **leaveEvent** (QEvent \*) override
- void **mouseMoveEvent** (QMouseEvent \*) override
- void **mousePressEvent** (QMouseEvent \*) override
- void **mouseReleaseEvent** (QMouseEvent \*) override
- void **paintEvent** (QPaintEvent \*) override

## 9.342 Digikam::DHBox Class Reference

An Horizontal widget to host children widgets.

Inheritance diagram for Digikam::DHBox:



## Public Member Functions

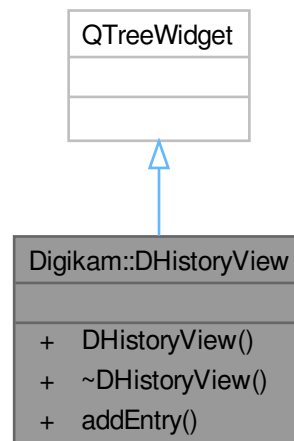
- **DHBox** (QWidget \*const parent=nullptr)
- QSize **minimumSizeHint** () const override
- void **setContentsMargins** (const QMargins &margins)
- void **setContentsMargins** (int left, int top, int right, int bottom)
- void **setSpacing** (int space)
- void **setStretchFactor** (QWidget \*const widget, int stretch)
- QSize **sizeHint** () const override

### Protected Member Functions

- **DHBox** (bool vertical, QWidget \*const parent)
- void **childEvent** (QChildEvent \*e) override

## 9.343 Digikam::DHistoryView Class Reference

Inheritance diagram for Digikam::DHistoryView:



### Public Types

- enum **EntryType** {  
    **StartingEntry** = 0 , **SuccessEntry** , **WarningEntry** , **ErrorEntry** ,  
    **ProgressEntry** , **CancelEntry** }

### Signals

- void **signalEntryClicked** (const QVariant &metadata)

### Public Member Functions

- **DHistoryView** (QWidget \*const parent)
- void **addEntry** (const QString &msg, EntryType type, const QVariant &metadata=QVariant())

## 9.344 Digikam::DHueSaturationSelector Class Reference

Inheritance diagram for Digikam::DHueSaturationSelector:



### Public Member Functions

- **DHueSaturationSelector** (QWidget \*const parent=nullptr)  
*Constructs a hue/saturation selection widget.*

- `~DHueSaturationSelector ()` override  
*Destructor.*
- `DColorChooserMode chooserMode ()` const  
*Returns the chooser mode.*
- `int colorValue ()` const  
*Returns the color value (also known as luminosity, 0-255)*
- `int hue ()` const  
*Returns the hue value.*
- `int saturation ()` const  
*Returns the saturation (0-255)*
- `void setChooserMode (DColorChooserMode chooserMode)`  
*Sets the chooser mode.*
- `void setColorValue (int color)`  
*Sets the color value (0-255)*
- `void setHue (int hue)`  
*Sets the hue value (0-360)*
- `void setSaturation (int saturation)`  
*Sets the saturation (0-255)*
- `void updateContents ()`  
*Updates the contents.*

## Public Member Functions inherited from Digikam::DPointSelect

- `DPointSelect (QWidget *const parent)`  
*Constructs a two-dimensional selector widget which has a value range of [0..100] in both directions.*
- `QRect contentsRect ()` const
- `QSize minimumSizeHint ()` const override  
*Reimplemented to give the widget a minimum size.*
- `void setMarkerColor (const QColor &col)`  
*Sets the color used to draw the marker.*
- `void setRange (int minX, int minY, int maxX, int maxY)`  
*Sets the range of possible values.*
- `void setValues (int xPos, int yPos)`  
*Sets the current values in horizontal and vertical direction.*
- `void setXValue (int xPos)`  
*Sets the current horizontal value.*
- `void setYValue (int yPos)`  
*Sets the current vertical value.*
- `int xValue ()` const
- `int yValue ()` const

## Protected Member Functions

- `void drawContents (QPainter *painter)` override  
*Reimplemented from [DPointSelect](#).*
- virtual `void drawPalette (QPixmap *pixmap)`  
*Draws the contents of the widget on a pixmap, which is used for buffering.*
- `void resizeEvent (QResizeEvent *)` override

## Protected Member Functions inherited from [Digikam::DPointSelect](#)

- virtual void **drawMarker** (QPainter \*p, int xp, int yp)  
*Override this function to draw the marker which indicates the currently selected value pair.*
- void **mouseMoveEvent** (QMouseEvent \*e) override
- void **mousePressEvent** (QMouseEvent \*e) override
- void **paintEvent** (QPaintEvent \*e) override
- void **valuesFromPosition** (int x, int y, int &xVal, int &yVal) const  
*Converts a pixel position to its corresponding values.*
- void **wheelEvent** (QWheelEvent \*) override

## Friends

- class **Private**

## Additional Inherited Members

## Signals inherited from [Digikam::DPointSelect](#)

- void **valueChanged** (int x, int y)  
*This signal is emitted whenever the user chooses a value, e.g.*

## Properties inherited from [Digikam::DPointSelect](#)

- int **xValue**
- int **yValue**

## 9.344.1 Member Function Documentation

### 9.344.1.1 chooserMode()

```
DColorChooserMode Digikam::DHueSaturationSelector::chooserMode ( ) const
```

#### Returns

The chooser mode (defined in DColorChooserMode)

### 9.344.1.2 colorValue()

```
int Digikam::DHueSaturationSelector::colorValue ( ) const
```

#### Returns

The color value (0-255)

### 9.344.1.3 drawContents()

```
void Digikam::DHueSaturationSelector::drawContents (
    QPainter * painter ) [override], [protected], [virtual]
```

This drawing is buffered in a pixmap here. As real drawing routine, [drawPalette\(\)](#) is used.

Reimplemented from [Digikam::DPointSelect](#).

### 9.344.1.4 hue()

```
int Digikam::DHueSaturationSelector::hue ( ) const
```

#### Returns

The hue value (0-360)

### 9.344.1.5 saturation()

```
int Digikam::DHueSaturationSelector::saturation ( ) const
```

#### Returns

The saturation (0-255)

### 9.344.1.6 setChooserMode()

```
void Digikam::DHueSaturationSelector::setChooserMode (
    DColorChooserMode chooserMode )
```

The allowed modes are defined in DColorChooserMode.

#### Parameters

<i>chooserMode</i>	The chooser mode as defined in DColorChooserMode
--------------------	--

### 9.344.1.7 setColorValue()

```
void Digikam::DHueSaturationSelector::setColorValue (
    int color )
```

#### Parameters

<i>color</i>	The color value (0-255)
--------------	-------------------------

### 9.344.1.8 setHue()

```
void Digikam::DHueSaturationSelector::setHue (
    int hue )
```

#### Parameters

<i>hue</i>	The hue value (0-360)
------------	-----------------------

### 9.344.1.9 setSaturation()

```
void Digikam::DHueSaturationSelector::setSaturation (
    int saturation )
```

#### Parameters

<i>saturation</i>	The saturation (0-255)
-------------------	------------------------



## 9.345 Digikam::DigikamApp Class Reference

Inheritance diagram for Digikam::DigikamApp:



### Signals

- void **queuedOpenCameraUiFromPath** (const QString &path)
- void **queuedOpenSolidDevice** (const QString &udi)

- void **signalCopyAlbumItemsSelection** ()
- void **signalCutAlbumItemsSelection** ()
- void **signalEscapePressed** ()
- void **signalFirstItem** ()
- void **signalLastItem** ()
- void **signalNextItem** ()
- void **signalNotificationError** (const QString &message, int type)
- void **signalPasteAlbumItemsSelection** ()
- void **signalPrevItem** ()
- void **signalWindowHasMoved** ()

### Public Member Functions

- void **autoDetect** ()
- void **downloadFrom** (const QString &cameraGuiPath)
- void **downloadFromUdi** (const QString &udi)
- void **enableAlbumBackwardHistory** (bool enable)
- void **enableAlbumForwardHistory** (bool enable)
- void **enableZoomMinusAction** (bool val)
- void **enableZoomPlusAction** (bool val)
- [DInfoInterface](#) \* **interface** ([DPluginAction](#) \*const ac) override  
*Return the interface instance to access to items information.*
- void **restoreSession** ()
- virtual void **show** ()
- [ItemIconView](#) \* **view** () const

### Public Member Functions inherited from [Digikam::DXmlGuiWindow](#)

- [DXmlGuiWindow](#) (QWidget \*const parent=nullptr, Qt::WindowFlags f=Qt::WindowFlags())
- QList< QAction \* > **allActions** () const  
*Return all actions from internal collection.*
- void **cleanupActions** ()  
*Cleanup unwanted actions from action collection.*
- QString **configGroupName** () const
- void [createFullScreenAction](#) (const QString &name)  
*Create Full-screen action to action collection instance from managed window set through [setManagedWindow\(\)](#).*
- void **createHelpActions** (const QString &handbookSection, bool coreOptions=true)  
*Create common actions from Help menu for all digiKam main windows.*
- void **createSettingsActions** ()  
*Create common actions to setup all digiKam main windows.*
- void **createSidebarActions** ()  
*Create common actions to handle side-bar through keyboard shortcuts.*
- bool **fullScreensActive** () const  
*Return true if managed window is currently in Full Screen Mode.*
- void **readFullScreenSettings** (const KConfigGroup &group)  
*Read full-screen settings from KDE config file.*
- virtual void **registerExtraPluginsActions** (QString &)
- void [registerPluginsActions](#) ()  
*Register all generic plugins action to this instance.*
- void **setConfigGroupName** (const QString &name)  
*Manage config group name used by window instance to get/set settings from config file.*
- void **setFullScreenOptions** (int options)  
*Set full-screen options to managed window.*
- void **unminimizeAndActivateWindow** ()

### Static Public Member Functions

- static [DigikamApp](#) \* **instance** ()

### Static Public Member Functions inherited from [Digikam::DXmlGuiWindow](#)

- static QAction \* **buildStdAction** (StdActionType type, const QObject \*const recvr, const char \*const slot, QObject \*const parent)
- static QString **configFullScreenHideSideBarsEntry** ()
- static QString **configFullScreenHideStatusBarEntry** ()
- static QString **configFullScreenHideThumbBarEntry** ()
- static QString **configFullScreenHideToolBarsEntry** ()

*Shared with [FullScreenSettings](#).*

- static void **restoreWindowSize** (QWindow \*const win, const KConfigGroup &group)
- static void **saveWindowSize** (QWindow \*const win, KConfigGroup &group)
- static void **setGoodDefaultWindowSize** (QWindow \*const win)
- static void **setupIconTheme** ()

*If we have some local breeze icon resource, prefer it.*

### Protected Member Functions

- void **closeEvent** (QCloseEvent \*e) override
- void **moveEvent** (QMoveEvent \*e) override
- bool **queryClose** () override

### Protected Member Functions inherited from [Digikam::DXmlGuiWindow](#)

- void **closeEvent** (QCloseEvent \*e) override
- void [editKeyboardShortcuts](#) (KActionCollection \*const extraac=nullptr, const QString &actitle=QString())  
*Call this method from your main window to show keyboard shortcut config dialog with an extra action collection to configure.*
- bool **eventFilter** (QObject \*obj, QEvent \*ev) override
- void **keyPressEvent** (QKeyEvent \*e) override
- QAction \* **showMenuBarAction** () const
- QAction \* **showStatusBarAction** () const

### Additional Inherited Members

### Protected Slots inherited from [Digikam::DXmlGuiWindow](#)

- bool **slotClose** ()

### Protected Attributes inherited from [Digikam::DXmlGuiWindow](#)

- [DLogoAction](#) \* **m\_animLogo** = nullptr

## 9.345.1 Member Function Documentation

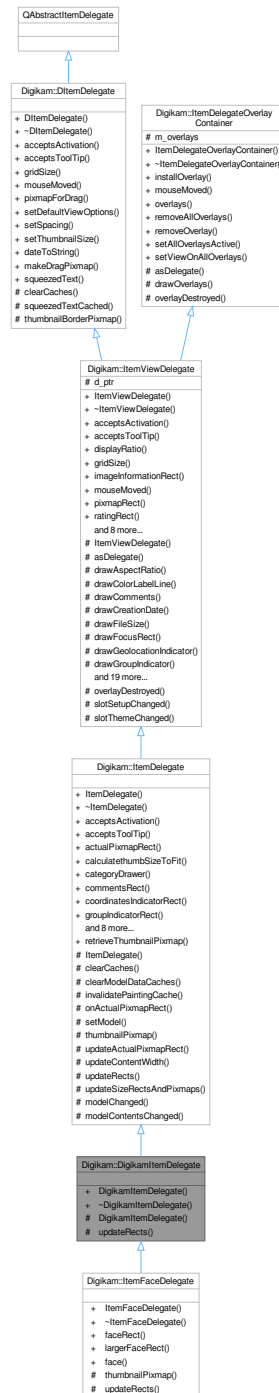
### 9.345.1.1 infoface()

```
DInfoInterface * Digikam::DigikamApp::infoIface (  
    DPluginAction *const ac ) [override], [virtual]
```

Implements [Digikam::DXmlGuiWindow](#).

## 9.346 Digikam::DigikamItemDelegate Class Reference

Inheritance diagram for Digikam::DigikamItemDelegate:



### Public Member Functions

- `DigikamItemDelegate` (`ItemCategorizedView` \*const parent)

## Public Member Functions inherited from [Digikam::ItemDelegate](#)

- **ItemDelegate** (QWidget \*const parent)
- bool [acceptsActivation](#) (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*activationRect=nullptr) const override
- bool [acceptsToolTip](#) (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*tooltipRect=nullptr) const override
 

*These methods take four parameters: The position on viewport, the rect on viewport, the index, and optionally a parameter into which, if the return value is true, a rectangle can be written for which the return value will be true as well.*
- QRect **actualPixmapRect** (const QModelIndex &index) const
- int **calculatethumbSizeToFit** (int ws)
- [ItemCategoryDrawer](#) \* **categoryDrawer** () const
- QRect **commentsRect** () const
- QRect **coordinatesIndicatorRect** () const
- QRect **groupIndicatorRect** () const
- QRect [imageInformationRect](#) () const override
 

*Returns the area where the image information is drawn, or null if empty / not supported.*
- void **paint** (QPainter \*painter, const QStyleOptionViewItem &option, const QModelIndex &index) const override
- QPixmap [pixmapForDrag](#) (const QStyleOptionViewItem &option, const QList< QModelIndex > &indexes) const override
- QRect [pixmapRect](#) () const override
 

*Returns the area where the pixmap is drawn, or null if not supported.*
- void [setDefaultViewOptions](#) (const QStyleOptionViewItem &option) override
 

*Style option with standard values to use for cached rendering.*
- void [setSpacing](#) (int spacing) override
- void **setView** ([ItemCategorizedView](#) \*view)
- QRect **tagsRect** () const

## Public Member Functions inherited from [Digikam::ItemViewDelegate](#)

- **ItemViewDelegate** (QWidget \*const parent)
- bool [acceptsActivation](#) (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*activationRect=nullptr) const override
- bool [acceptsToolTip](#) (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*tooltipRect=nullptr) const override
 

*These methods take four parameters: The position on viewport, the rect on viewport, the index, and optionally a parameter into which, if the return value is true, a rectangle can be written for which the return value will be true as well.*
- double **displayRatio** () const
- QSize [gridSize](#) () const override
 

*Returns the gridsize to be set by the view.*
- void [mouseMoved](#) (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index) override
- virtual QRect **ratingRect** () const
 

*Returns the rectangle where the rating is drawn, or a null rectangle if not supported.*
- QRect **rect** () const
- void [setDefaultViewOptions](#) (const QStyleOptionViewItem &option) override
 

*Style option with standard values to use for cached rendering.*
- void [setRatingEdited](#) (const QModelIndex &index)
 

*Can be used to temporarily disable drawing of the rating.*
- void [setSpacing](#) (int spacing) override
- void [setThumbnailSize](#) (const [ThumbnailSize](#) &thumbSize) override
 

*You must set these options from the view.*
- QSize **sizeHint** (const QStyleOptionViewItem &option, const QModelIndex &index) const override
- int **spacing** () const
- [ThumbnailSize](#) **thumbnailSize** () const

## Public Member Functions inherited from Digikam::DItemDelegate

- **DItemDelegate** (QObject \*const parent=nullptr)

## Public Member Functions inherited from Digikam::ItemDelegateOverlayContainer

- **ItemDelegateOverlayContainer** ()=default  
*This is a sample implementation for delegate management methods, to be inherited by a delegate.*
- void **installOverlay** (ItemDelegateOverlay \*overlay)
- void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)
- QList< ItemDelegateOverlay \* > **overlays** () const
- void **removeAllOverlays** ()
- void **removeOverlay** (ItemDelegateOverlay \*overlay)
- void **setAllOverlaysActive** (bool active)
- void **setViewOnAllOverlays** (QAbstractItemView \*view)

## Protected Member Functions

- **DigikamItemDelegate** (DigikamItemDelegatePrivate &dd, ItemCategorizedView \*parent)
- void **updateRects** () override

*In a subclass, you need to implement this method to set up the rects for drawing.*

## Protected Member Functions inherited from Digikam::ItemDelegate

- **ItemDelegate** (ItemDelegate::ItemDelegatePrivate &dd, QWidget \*const parent)
- void **clearCaches** () override
- virtual void **clearModelDataCaches** ()  
*Reimplement to clear caches based on model indexes (hash on row number etc.) Change signals are listened to this is called whenever such properties become invalid.*
- void **invalidatePaintingCache** () override
- bool **onActualPixmapRect** (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*actualRect) const
- void **setModel** (QAbstractItemModel \*model)
- virtual QPixmap **thumbnailPixmap** (const QModelIndex &index) const
- void **updateActualPixmapRect** (const QModelIndex &index, const QRect &rect)
- virtual void **updateContentWidth** ()  
*Reimplement this to set contentWidth.*
- void **updateSizeRectsAndPxmmaps** () override

## Protected Member Functions inherited from Digikam::ItemViewDelegate

- **ItemViewDelegate** (ItemViewDelegatePrivate &dd, QWidget \*const parent)
- QAbstractItemDelegate \* **asDelegate** () override  
*Returns the delegate, typically, the derived class.*
- void **drawAspectRatio** (QPainter \*p, const QRect &dimsRect, const QSize &dims) const
- void **drawColorLabelLine** (QPainter \*p, const QRect &pixRect, int colorId) const
- void **drawComments** (QPainter \*p, const QRect &commentsRect, const QString &comments) const
- void **drawCreationDate** (QPainter \*p, const QRect &dateRect, const QDateTime &date) const
- void **drawFileSize** (QPainter \*p, const QRect &r, qlonglong bytes) const
- void **drawFocusRect** (QPainter \*p, const QStyleOptionViewItem &option, bool isSelected) const

- void **drawGeolocationIndicator** (QPainter \*p, const QRect &r) const
  - void **drawGroupIndicator** (QPainter \*p, const QRect &r, int numberOfGroupedImages, bool open) const
  - void **drawImageFormat** (QPainter \*p, const QRect &r, const QString &f, bool drawTop) const
  - void **drawImageSize** (QPainter \*p, const QRect &dimsRect, const QSize &dims) const
  - void **drawModificationDate** (QPainter \*p, const QRect &dateRect, const QDateTime &date) const
  - void **drawMouseOverRect** (QPainter \*p, const QStyleOptionViewItem &option) const
  - void **drawName** (QPainter \*p, const QRect &nameRect, const QString &name) const
  - void **drawPanelSidelcon** (QPainter \*p, bool left, bool right) const
  - void **drawPickLabelIcon** (QPainter \*p, const QRect &r, int pickLabel) const
  - void **drawRating** (QPainter \*p, const QModelIndex &index, const QRect &ratingRect, int rating, bool isSelected) const
  - void **drawSpecialInfo** (QPainter \*p, const QRect &r, const QString &text) const
  - void **drawTags** (QPainter \*p, const QRect &r, const QString &tagsString, bool isSelected) const
  - QRect **drawThumbnail** (QPainter \*p, const QRect &thumbRect, const QPixmap &background, const QPixmap &thumbnail, bool isGrouped) const
- Use the tool methods for painting in subclasses.*
- void **drawTitle** (QPainter \*p, const QRect &titleRect, const QString &title) const
  - void **prepareBackground** ()
  - void **prepareFonts** ()
  - void **prepareMetrics** (int maxWidth)
  - void **prepareRatingPixmap** (bool composeOverBackground=true)
  - QPixmap **ratingPixmap** (int rating, bool selected) const

*Returns the relevant pixmap from the cached rating pixmaps.*

### Protected Member Functions inherited from [Digikam::DItemDelegate](#)

- QString **squeezedTextCached** (QPainter \*const p, int width, const QString &text) const
- QPixmap **thumbnailBorderPixmap** (const QSize &pixSize, bool isGrouped=false) const

### Protected Member Functions inherited from [Digikam::ItemDelegateOverlayContainer](#)

- virtual void **drawOverlays** (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index) const
- virtual void **overlayDestroyed** (QObject \*o)

*Declare as slot in the derived class calling this method.*

### Additional Inherited Members

### Signals inherited from [Digikam::ItemViewDelegate](#)

- void **hideNotification** ()
- void **requestNotification** (const QModelIndex &index, const QString &message)

### Signals inherited from [Digikam::DItemDelegate](#)

- void **gridSizeChanged** (const QSize &newSize)
- void **visualChange** ()



## Static Public Member Functions inherited from Digikam::ItemDelegate

- static QPixmap **retrieveThumbnailPixmap** (const QModelIndex &index, int thumbnailSize)  
*Retrieve the thumbnail pixmap in given size for the [ItemModel::ThumbnailRole](#) for the given index from the given index, which must adhere to [ItemThumbnailModel](#) semantics.*

## Static Public Member Functions inherited from Digikam::DItemDelegate

- static QString **dateToString** (const QDateTime &datetime)
- static QPixmap **makeDragPixmap** (const QStyleOptionViewItem &option, const QList< QModelIndex > &indexes, double displayRatio, const QPixmap &suggestedPixmap=QPixmap())
- static QString **squeezedText** (const QFontMetrics &fm, int width, const QString &text)

## Protected Slots inherited from Digikam::ItemDelegate

- void **modelChanged** ()
- void **modelContentsChanged** ()

## Protected Slots inherited from Digikam::ItemViewDelegate

- void **overlayDestroyed** (QObject \*o) override
- void **slotSetupChanged** ()
- void **slotThemeChanged** ()

## Protected Attributes inherited from Digikam::ItemViewDelegate

- ItemViewDelegatePrivate \*const **d\_ptr** = nullptr

## Protected Attributes inherited from Digikam::ItemDelegateOverlayContainer

- QList< [ItemDelegateOverlay](#) \* > **m\_overlays**

## 9.346.1 Member Function Documentation

### 9.346.1.1 updateRects()

```
void Digikam::DigikamItemDelegate::updateRects ( ) [override], [protected], [virtual]
```

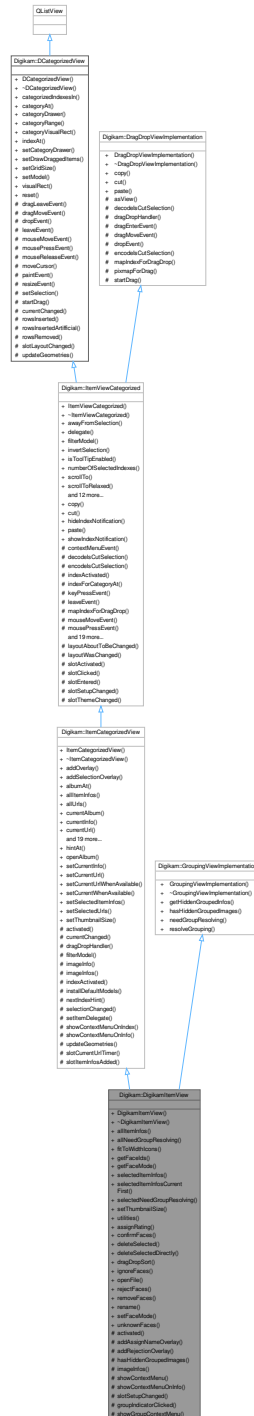
The paint() method operates depending on these rects.

Implements [Digikam::ItemDelegate](#).

Reimplemented in [Digikam::ItemFaceDelegate](#).

## 9.347 Digikam::DigikamItemView Class Reference

Inheritance diagram for Digikam::DigikamItemView:



### Public Slots

- void **assignRating** (const QList< QModelIndex > &index, int rating)
- void **confirmFaces** (const QList< QModelIndex > &indexes, int tagId)

*Confirm the face with a face tag (name) in the database.*

- void **deleteSelected** (const ItemViewUtilities::DeleteMode deleteMode=ItemViewUtilities::DeleteUseTrash)
- void **deleteSelectedDirectly** (const ItemViewUtilities::DeleteMode deleteMode=ItemViewUtilities::DeleteUseTrash)
- void **dragDropSort** (const ItemInfo &pick, const QList< ItemInfo > &infos)
- void **ignoreFaces** (const QList< QModelIndex > &indexes)

*Set Face to Ignore ID.*

- void **openFile** (const ItemInfo &info)
- void **rejectFaces** (const QList< QModelIndex > &indexes)

*This slot is connected to the reject signal of AssignNameOverlay, and handles two cases.*

- void **removeFaces** (const QList< QModelIndex > &indexes)

*Removes the face from the database.*

- void **rename** ()
- void **setFaceMode** (bool on)
- void **unknownFaces** (const QList< QModelIndex > &indexes)

*Ignored Face back to Unknown face.*

## Public Slots inherited from Digikam::ItemCategorizedView

- void **hintAt** (const ItemInfo &info)
 

*Does something to gain attention for info, but not changing current selection.*
- void **openAlbum** (const QList< Album \* > &album)
- void **setCurrentInfo** (const ItemInfo &info)
 

*Set as current item the item identified by the imageinfo.*
- void **setCurrentUrl** (const QUrl &url)
 

*Set as current item the item identified by its file url.*
- void **setCurrentUrlWhenAvailable** (const QUrl &url)
 

*Set as current item when it becomes available, the item identified by its file url.*
- void **setCurrentWhenAvailable** (qulonglong imageId)
 

*Scroll the view to the given item when it becomes available.*
- void **setSelectedItemInfos** (const QList< ItemInfo > &infos)
 

*Set selected items.*
- void **setSelectedUrls** (const QList< QUrl > &urlList)
 

*Set selected items identified by their file urls.*
- void **setThumbnailSize** (int size)

## Public Slots inherited from Digikam::ItemViewCategorized

- void **copy** () override
- void **cut** () override
- void **hideIndexNotification** ()
- void **paste** () override
- void **showIndexNotification** (const QModelIndex &index, const QString &message)

## Public Slots inherited from Digikam::DCategorizedView

- void **reset** () override

## Signals

- void **previewRequested** (const [ItemInfo](#) &info)
- void **signalSeparationModeChanged** (int category)
- void **signalShowContextMenu** (QContextMenuEvent \*event, const QList< QAction \* > &actions=QList< QAction \* >())
- void **signalShowContextMenuOnInfo** (QContextMenuEvent \*event, const [ItemInfo](#) &info, const QList< QAction \* > &actions, [ItemFilterModel](#) \*filterModel)
- void **signalShowGroupContextMenu** (QContextMenuEvent \*event, const QList< [ItemInfo](#) > &selectedInfos, [ItemFilterModel](#) \*filterModel)

## Signals inherited from [Digikam::ItemCategorizedView](#)

- void **currentChanged** (const [ItemInfo](#) &info)
- void **deselected** (const QList< [ItemInfo](#) > &nowDeselectedInfos)  
*Emitted when items are deselected. There may be other selected infos left. This signal is not emitted when the model is reset; then only selectionCleared is emitted.*
- void **imageActivated** (const [ItemInfo](#) &info)  
*Emitted when the given image is activated. Info is never null.*
- void **modelChanged** ()  
*Emitted when a new model is set.*
- void **selected** (const QList< [ItemInfo](#) > &newSelectedInfos)  
*Emitted when new items are selected. The parameter includes only the newly selected infos, there may be other already selected infos.*

## Signals inherited from [Digikam::ItemViewCategorized](#)

- void **clicked** (const QMouseEvent \*e, const QModelIndex &index)  
*For overlays: Like the respective parent class signals, but with additional info.*
- void **entered** (const QMouseEvent \*e, const QModelIndex &index)
- void **keyPressed** (QKeyEvent \*e)  
*Remember you may want to check if the event is accepted or ignored.*
- void **selectionChanged** ()  
*Emitted when any selection change occurs.*
- void **selectionCleared** ()  
*Emitted when the selection is completely cleared.*
- void **viewportClicked** (const QMouseEvent \*e)  
*While [clicked\(\)](#) is emitted with a valid index, this corresponds to clicking on empty space.*
- void **zoomInStep** ()
- void **zoomOutStep** ()

## Public Member Functions

- [DigikamItemView](#) (QWidget \*const parent=nullptr)
- [ItemInfoList](#) **allItemInfos** (bool grouping=false) const
- bool **allNeedGroupResolving** (const [OperationType](#) type) const
- int **fitToWidthIcons** ()
- QList< int > **getFaceIds** (const QList< QModelIndex > &indexes) const
- bool **getFaceMode** () const
- [ItemInfoList](#) **selectedItemInfos** (bool grouping=false) const
- [ItemInfoList](#) **selectedItemInfosCurrentFirst** (bool grouping=false) const
- bool **selectedNeedGroupResolving** (const [OperationType](#) type) const
- void **setThumbnailSize** (const [ThumbnailSize](#) &size) override
- [ItemViewUtilities](#) \* **utilities** () const

## Public Member Functions inherited from Digikam::ItemCategorizedView

- **ItemCategorizedView** (QWidget \*const parent=nullptr)
- void **addOverlay** (ItemDelegateOverlay \*overlay, ItemDelegate \*delegate=nullptr)
 

*Add and remove an overlay. It will as well be removed automatically when destroyed. Unless you pass a different delegate, the current delegate will be used.*
- void **addSelectionOverlay** (ItemDelegate \*delegate=nullptr)
- Album \* **albumAt** (const QPoint &pos) const
 

*If the model is categorized by an album, returns the album of the category that contains the position.*
- ItemInfoList **allItemInfos** () const
- QList< QUrl > **allUrls** () const
- Album \* **currentAlbum** () const
- ItemInfo **currentInfo** () const
- QUrl **currentUrl** () const
- ItemDelegate \* **delegate** () const
- QItemSelectionModel \* **getSelectionModel** () const
- ItemAlbumFilterModel \* **imageAlbumFilterModel** () const
- ItemAlbumModel \* **imageAlbumModel** () const
 

*Returns 0 if the ItemModel is not an ItemAlbumModel.*
- ItemFilterModel \* **imageFilterModel** () const
 

*Returns any ItemFilterMode in chain. May not be sourceModel()*
- ItemModel \* **imageModel** () const
- ImageSortFilterModel \* **imageSortFilterModel** () const
- ItemThumbnailModel \* **imageThumbnailModel** () const
 

*Returns 0 if the ItemModel is not an ItemThumbnailModel.*
- QModelIndex **indexForInfo** (const ItemInfo &info) const
- ItemInfo **nextInfo** (const ItemInfo &info)
- ItemInfo **nextInOrder** (const ItemInfo &startingPoint, int nth)
 

*Returns the n-th info after the given one.*
- ItemInfo **previousInfo** (const ItemInfo &info)
- void **removeOverlay** (ItemDelegateOverlay \*overlay)
- ItemInfoList **selectedItemInfos** () const
- ItemInfoList **selectedItemInfosCurrentFirst** () const
- void **setModels** (ItemModel \*model, ImageSortFilterModel \*filterModel)
- ThumbnailSize **thumbnailSize** () const
- void **toIndex** (const QUrl &url)
 

*Selects the index as current and scrolls to it.*

## Public Member Functions inherited from Digikam::ItemViewCategorized

- **ItemViewCategorized** (QWidget \*const parent=nullptr)
- void **awayFromSelection** ()
- DItemDelegate \* **delegate** () const
- void **invertSelection** ()
- bool **isToolTipEnabled** () const
- int **numberOfSelectedIndexes** () const
- void **scrollTo** (const QModelIndex &index, ScrollHint hint=EnsureVisible) override
- void **scrollToRelaxed** (const QModelIndex &index, ScrollHint hint=EnsureVisible)
 

*Like scrollTo, but only scrolls if the index is not visible, regardless of hint.*
- void **setInitialSelectedItem** (bool enabled)
 

*Ensure a initial selected item.*
- void **setScrollCurrentToCenter** (bool enabled)

- Scroll automatically the current index to center of the view.*
- void **setScrollStepGranularity** (int factor)
  - Determine a step size for scrolling: The larger this number, the smaller and more precise is the scrolling.*
- void **setSelectedIndexes** (const QList< QModelIndex > &indexes)
- void **setSpacing** (int spacing)
  - Sets the spacing.*
- void **setToolTipEnabled** (bool enabled)
- void **setUsePointingHandCursor** (bool useCursor)
  - Set if the PointingHand Cursor should be shown over the activation area.*
- void **toFirstIndex** ()
  - Selects the index as current and scrolls to it.*
- void **toIndex** (const QModelIndex &index)
- void **toLastIndex** ()
- void **toNextIndex** ()
- void **toPreviousIndex** ()

### Public Member Functions inherited from [Digikam::DCategorizedView](#)

- **DCategorizedView** (QWidget \*const parent=nullptr)
- virtual QModelIndexList **categorizedIndexesIn** (const QRect &rect) const
  - This method will return all indexes whose visual rect intersects *rect*.*
- virtual QModelIndex **categoryAt** (const QPoint &point) const
  - This method will return the first index of the category in the region of which *point* is found.*
- **DCategoryDrawer** \* **categoryDrawer** () const
- virtual QItemSelectionRange **categoryRange** (const QModelIndex &index) const
  - This method returns the range of indexes contained in the category in which *index* is sorted.*
- virtual QRect **categoryVisualRect** (const QModelIndex &index) const
  - This method will return the visual rect of the header of the category in which *index* is sorted.*
- QModelIndex **indexAt** (const QPoint &point) const override
- void **setCategoryDrawer** (**DCategoryDrawer** \*categoryDrawer)
- void **setDrawDraggedItems** (bool drawDraggedItems)
  - Switch on drawing of dragged items.*
- void **setGridSize** (const QSize &size)
- void **setModel** (QAbstractItemModel \*model) override
- QRect **visualRect** (const QModelIndex &index) const override

### Public Member Functions inherited from [Digikam::DragDropViewImplementation](#)

- virtual void **copy** ()
- virtual void **cut** ()
- virtual void **paste** ()

### Public Member Functions inherited from [Digikam::GroupingViewImplementation](#)

- **ItemInfoList** **getHiddenGroupedInfos** (const **ItemInfoList** &infos) const
- bool **needGroupResolving** (**OperationType** type, const **ItemInfoList** &infos) const
- **ItemInfoList** **resolveGrouping** (const **ItemInfoList** &infos) const

**Protected Slots**

- void **groupIndicatorClicked** (const QModelIndex &index)
- void **showGroupContextMenu** (const QModelIndex &index, QContextMenuEvent \*event)

**Protected Slots inherited from [Digikam::ItemCategorizedView](#)**

- void **slotCurrentUrlTimer** ()
- void **slotItemInfosAdded** ()

**Protected Slots inherited from [Digikam::ItemViewCategorized](#)**

- void **layoutAboutToBeChanged** ()
- void **layoutWasChanged** ()
- void **slotActivated** (const QModelIndex &index)
- void **slotClicked** (const QModelIndex &index)
- void **slotEntered** (const QModelIndex &index)
- virtual void **slotThemeChanged** ()

**Protected Slots inherited from [Digikam::DCategorizedView](#)**

- void **currentChanged** (const QModelIndex &current, const QModelIndex &previous) override
- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- virtual void **rowsInsertedArtificial** (const QModelIndex &parent, int start, int end)
- virtual void **slotLayoutChanged** ()
- void **updateGeometries** () override

**Protected Member Functions**

- void **activated** (const [ItemInfo](#) &info, Qt::KeyboardModifiers modifiers) override  
*Reimplement these in a subclass.*
- void **addAssignNameOverlay** ([ItemDelegate](#) \*delegate=nullptr)
- void **addRejectionOverlay** ([ItemDelegate](#) \*delegate=nullptr)
- bool **hasHiddenGroupedImages** (const [ItemInfo](#) &info) const override  
*must be implemented by parent view*
- [ItemInfoList](#) **imageInfos** (const QList< QModelIndex > &indexes, [OperationType](#) type) const
- void **showContextMenu** (QContextMenuEvent \*event) override
- void **showContextMenuOnInfo** (QContextMenuEvent \*event, const [ItemInfo](#) &info) override
- void **slotSetupChanged** () override

## Protected Member Functions inherited from [Digikam::ItemCategorizedView](#)

- void **currentChanged** (const QModelIndex &index, const QModelIndex &previous) override
- [AbstractItemDragDropHandler](#) \* **dragDropHandler** () const override
  - You need to implement these three methods Returns the drag drop handler.*
- QSortFilterProxyModel \* **filterModel** () const override
- [ItemInfo](#) **imageInfo** (const QModelIndex &index) const
- [ItemInfoList](#) **imageInfos** (const QList< QModelIndex > &indexes) const
- void **indexActivated** (const QModelIndex &index, Qt::KeyboardModifiers modifiers) override
- void **installDefaultModels** ()
  - install default [ItemAlbumModel](#) and filter model, ready for use*
- QModelIndex **nextIndexHint** (const QModelIndex &indexToAnchor, const QItemSelectionRange &removed) const override
  - Assuming the given indexes would be removed (hypothetically!), return the index to be selected instead, starting from anchor.*
- void **selectionChanged** (const QItemSelection &, const QItemSelection &) override
- void **setItemDelegate** ([ItemDelegate](#) \*delegate)
- void **showContextMenuOnIndex** (QContextMenuEvent \*event, const QModelIndex &index) override
  - Reimplement these in a subclass.*
- void **updateGeometries** () override

## Protected Member Functions inherited from [Digikam::ItemViewCategorized](#)

- void **contextMenuEvent** (QContextMenuEvent \*event) override
  - reimplemented from parent class*
- bool **decodelsCutSelection** (const QMimeData \*mimeType)
- void **encodelsCutSelection** (QMimeData \*mimeType, bool isCutSelection)
- QModelIndex **indexForCategoryAt** (const QPoint &pos) const
  - Returns an index that is representative for the category at position pos.*
- void **keyPressEvent** (QKeyEvent \*event) override
- void **leaveEvent** (QEvent \*event) override
- QModelIndex **mapIndexForDragDrop** (const QModelIndex &index) const override
  - Note: pure virtual [dragDropHandler\(\)](#) still open from [DragDropViewImplementation](#).*
- void **mouseMoveEvent** (QMouseEvent \*event) override
- void **mousePressEvent** (QMouseEvent \*event) override
- void **mouseReleaseEvent** (QMouseEvent \*event) override
- QModelIndex **moveCursor** (CursorAction cursorAction, Qt::KeyboardModifiers modifiers) override
- QPixmap **pixmapForDrag** (const QList< QModelIndex > &indexes) const override
  - Creates a pixmap for dragging the given indexes.*
- void **reset** () override
- void **resizeEvent** (QResizeEvent \*e) override
- void **rowsAboutToBeRemoved** (const QModelIndex &parent, int start, int end) override
- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- void **rowsRemoved** (const QModelIndex &parent, int start, int end) override
- void **selectionChanged** (const QItemSelection &, const QItemSelection &) override
- void **setItemDelegate** ([DItemDelegate](#) \*delegate)
- void **setToolTip** ([ItemViewToolTip](#) \*tip)
- virtual bool **showToolTip** (const QModelIndex &index, QStyleOptionViewItem &option, QHelpEvent \*e=nullptr)
  - Provides default behavior, can reimplement in a subclass.*
- void **updateDelegateSizes** ()
- void **userInteraction** ()
- bool **viewportEvent** (QEvent \*event) override
- void **wheelEvent** (QWheelEvent \*event) override



## Protected Member Functions inherited from [Digikam::DCategorizedView](#)

- void **dragLeaveEvent** (QDragLeaveEvent \*event) override
- void **dragMoveEvent** (QDragMoveEvent \*event) override
- void **dropEvent** (QDropEvent \*event) override
- void **leaveEvent** (QEvent \*event) override
- void **mouseMoveEvent** (QMouseEvent \*event) override
- void **mousePressEvent** (QMouseEvent \*event) override
- void **mouseReleaseEvent** (QMouseEvent \*event) override
- QModelIndex **moveCursor** (CursorAction cursorAction, Qt::KeyboardModifiers modifiers) override
- void **paintEvent** (QPaintEvent \*event) override
- void **resizeEvent** (QResizeEvent \*event) override
- void **setSelection** (const QRect &rect, QItemSelectionModel::SelectionFlags flags) override
- void **startDrag** (Qt::DropActions supportedActions) override

## Protected Member Functions inherited from [Digikam::DragDropViewImplementation](#)

- virtual QAbstractItemView \* **asView** ()=0  
*This one is implemented by DECLARE\_VIEW\_DRAG\_DROP\_METHODS.*
- bool **decodelsCutSelection** (const QMimeData \*mimeData)
- void **dragEnterEvent** (QDragEnterEvent \*event)  
*Implements the relevant QAbstractItemView methods for drag and drop.*
- void **dragMoveEvent** (QDragMoveEvent \*e)
- void **dropEvent** (QDropEvent \*e)
- void **encodelsCutSelection** (QMimeData \*mime, bool isCutSelection)
- void **startDrag** (Qt::DropActions supportedActions)

### 9.347.1 Member Function Documentation

#### 9.347.1.1 activated()

```
void Digikam::DigikamItemView::activated (
    const ItemInfo & info,
    Qt::KeyboardModifiers modifiers ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ItemCategorizedView](#).

#### 9.347.1.2 confirmFaces

```
void Digikam::DigikamItemView::confirmFaces (
    const QList< QModelIndex > & indexes,
    int tagId ) [slot]
```

You aren't allowed to "confirm" a person as Ignored. Marking as Ignored is treated as a changeTag() operation.

#### 9.347.1.3 hasHiddenGroupedImages()

```
bool Digikam::DigikamItemView::hasHiddenGroupedImages (
    const ItemInfo & ) const [override], [protected], [virtual]
```

Reimplemented from [Digikam::GroupingViewImplementation](#).

#### 9.347.1.4 rejectFaces

```
void Digikam::DigikamItemView::rejectFaces (
    const QList< QModelIndex > & indexes ) [slot]
```

If reject is done on an Unknown Face, it will mark the face as Ignored.

If reject is done on Unconfirmed suggestions, the suggestion is rejected and the face is marked as Unknown.

#### 9.347.1.5 removeFaces

```
void Digikam::DigikamItemView::removeFaces (
    const QList< QModelIndex > & indexes ) [slot]
```

You will have to run face detection again, to recover the face.

#### 9.347.1.6 setThumbnailSize()

```
void Digikam::DigikamItemView::setThumbnailSize (
    const ThumbnailSize & size ) [override], [virtual]
```

Reimplemented from [Digikam::ItemCategorizedView](#).

#### 9.347.1.7 showContextMenu()

```
void Digikam::DigikamItemView::showContextMenu (
    QContextMenuEvent * event ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ItemViewCategorized](#).

#### 9.347.1.8 showContextMenuOnInfo()

```
void Digikam::DigikamItemView::showContextMenuOnInfo (
    QContextMenuEvent * event,
    const ItemInfo & info ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ItemCategorizedView](#).

#### 9.347.1.9 slotSetupChanged()

```
void Digikam::DigikamItemView::slotSetupChanged ( ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ItemViewCategorized](#).

## 9.348 Digikam::DImageHistory Class Reference

### Classes

- class [Entry](#)

## Public Member Functions

- **DImageHistory** (const [DImageHistory](#) &other)
- const [FilterAction](#) & **action** (int i) const
- int **actionCount** () const  
*Returns the number of non-null actions.*
- void **adjustCurrentUuid** (const QString &uuid)  
*Changes the UUID of the current (last added current) referred image.*
- void **adjustReferredImages** ()  
*Adjusts the type of a Current [HistoryImageld](#): If it is the first entry, it becomes Original, if it is in an intermediate entry, it becomes Intermediate, if in the last entry, it stays current.*
- QList< [FilterAction](#) > **allActions** () const  
*Gets all actions which are not null.*
- QList< [HistoryImageld](#) > **allReferredImages** () const
- void **appendReferredImage** (const [HistoryImageld](#) &id)
- void **clearReferredImages** ()  
*Edit referred images.*
- [HistoryImageld](#) **currentReferredImage** () const
- QList< [DImageHistory::Entry](#) > & **entries** ()  
*Access entries.*
- const QList< [DImageHistory::Entry](#) > & **entries** () const
- bool **hasActions** () const  
*Access actions.*
- bool **hasCurrentReferredImage** () const
- bool **hasFilters** () const
- bool **hasOriginalReferredImage** () const
- bool **hasReferredImageOfType** ([HistoryImageld::Type](#) type) const
- bool **hasReferredImages** () const
- void **insertReferredImage** (int entryIndex, const [HistoryImageld](#) &id)
- bool **isEmpty** () const  
*A history is considered empty if there are no entries.*
- bool **isNull** () const  
*A history is null if it is constructed with the default constructor.*
- bool **isValid** () const  
*A history is a valid history (telling something about the past), if the history is not empty, and there is at least one referred image other than the "Current" entry, or there is a valid action.*
- void **moveCurrentReferredImage** (const QString &newPath, const QString &newFileName)  
*Change file path entries of the current referred image.*
- bool **operator!=** (const [DImageHistory](#) &other) const
- bool **operator<** (const [DImageHistory](#) &other) const
- [DImageHistory](#) & **operator<<** (const [FilterAction](#) &action)  
*Appends a new filter action to the history.*
- [DImageHistory](#) & **operator<<** (const [HistoryImageld](#) &imageld)  
*Appends a new referred image, representing the current state of the history.*
- [DImageHistory](#) & **operator=** (const [DImageHistory](#) &other)
- bool **operator==** (const [DImageHistory](#) &other) const
- bool **operator>** (const [DImageHistory](#) &other) const
- [Entry](#) & **operator[]** (int i)
- const [Entry](#) & **operator[]** (int i) const
- [HistoryImageld](#) **originalReferredImage** () const
- void **purgePathFromReferredImages** (const QString &path, const QString &fileName)  
*Remove file path entries pointing to the given absolute path from any referred images.*
- QList< [HistoryImageld](#) > & **referredImages** (int i)

*Access referred images.*

- const QList< [HistoryImageId](#) > & **referredImages** (int i) const
- QList< [HistoryImageId](#) > **referredImagesOfType** ([HistoryImageId::Type](#) type) const
- void **removeLast** ()

*Removes the last entry from the history.*

- int **size** () const

*Returns the number of entries.*

- QString **toXml** () const

*Serialize to and from XML.*

### Static Public Member Functions

- static [DImageHistory](#) **fromXml** (const QString &xml)

## 9.348.1 Member Function Documentation

### 9.348.1.1 clearReferredImages()

```
void Digikam::DImageHistory::clearReferredImages ( )
```

Remove all referredImages, leaving the entries list untouched

### 9.348.1.2 entries()

```
QList< DImageHistory::Entry > & Digikam::DImageHistory::entries ( )
```

There are [size\(\)](#) entries.

### 9.348.1.3 hasActions()

```
bool Digikam::DImageHistory::hasActions ( ) const
```

There is one action per entry, but the action may be null. Returns if there is any non-null action

### 9.348.1.4 operator<<()

```
DImageHistory & Digikam::DImageHistory::operator<< (
    const HistoryImageId & imageId )
```

If you add an id of type Current, [adjustReferredImages\(\)](#) will be called.

### 9.348.1.5 purgePathFromReferredImages()

```
void Digikam::DImageHistory::purgePathFromReferredImages (
    const QString & path,
    const QString & fileName )
```

This is useful when said file is about to be overwritten. All other [HistoryImageId](#) fields remain unchanged, no [HistoryImageId](#) is removed. path: directory path, without filename.

### 9.348.1.6 toXml()

```
QString Digikam::DImageHistory::toXml ( ) const
```

Note: The "Current" entry is skipped when writing to XML, so make sure the file into the metadata of which you write the XML, is the file marked as "Current" in this history.

## 9.349 Digikam::DImageHistory::Entry Class Reference

### Public Attributes

- [FilterAction](#) `action`  
A *DImageHistory* is a list of entries.
- `QList< HistoryImageId >` `referredImages`

### 9.349.1 Member Data Documentation

#### 9.349.1.1 action

```
FilterAction Digikam::DImageHistory::Entry::action
```

Each entry has one action. The action can be null, but it shall be null only if it is the action of the first entry, with the "Original" as referred image, representing the action of digitization.

There can be zero, one or any number of referred images per entry. A referred image is a file in the state after the action is applied.

## 9.350 Digikam::DImg Class Reference

### Public Types

- enum `ANGLE` { `ROT90` = 0 , `ROT180` , `ROT270` , `ROTNONE` }
  - enum `COLORMODEL` { `COLORMODELUNKNOWN` = 0 , `RGB` , `GRAYSCALE` , `MONOCHROME` , `INDEXED` , `YCBCR` , `CMYK` , `CIELAB` , `COLORMODELRAW` }
  - enum `FLIP` { `HORIZONTAL` = 0 , `VERTICAL` }
  - enum `FORMAT` { `NONE` = 0 , `JPEG` , `PNG` , `TIFF` , `JP2K` , `PGF` , `HEIF` , `RAW` , `QIMAGE` }
  - enum `PrepareMetadataFlag` { `RemoveOldMetadataPreviews` = 1 << 0 , `CreateNewMetadataPreview` = 1 << 1 , `ResetExifOrientationTag` = 1 << 2 , `CreateNewImageHistoryUUID` = 1 << 3 , `PrepareMetadataFlagsAll` }
- When saving, several changes to the image metadata are necessary before it can safely be written to the new file.*
- typedef `QFlags< PrepareMetadataFlag >` `PrepareMetadataFlags`

## Public Member Functions

- [DImg](#) ()  
*Create null image.*
- [DImg](#) (const [DImg](#) &image)  
*Copy image: Creates a shallow copy that refers to the same shared data.*
- [DImg](#) (const QByteArray &filePath, [DImgLoaderObserver](#) \*const observer=nullptr, const [DRawDecoding](#) &[rawDecodingSettings](#)=[DRawDecoding](#)())  
*Load image using QByteArray as file path.*
- [DImg](#) (const QImage &image)  
*Copy image: Creates a copy of a QImage object.*
- [DImg](#) (const QString &filePath, [DImgLoaderObserver](#) \*const observer=nullptr, const [DRawDecoding](#) &[rawDecodingSettings](#)=[DRawDecoding](#)())  
*Load image using QString as file path.*
- [DImg](#) (uint width, uint height, bool sixteenBit, bool alpha=false, uchar \*const data=nullptr, bool copy←Data=true)  
*Create image from data.*
- void **addAsReferredImage** (const [HistoryImageId](#) &id)
- [HistoryImageId](#) **addAsReferredImage** (const QString &filePath, [HistoryImageId::Type](#) type=[HistoryImageId::Intermediate](#))  
*If you have saved this DImg to filePath, and want to continue using this DImg object to add further changes to the image history, you can call this method to add to the image history a reference to the just saved image.*
- void **addCurrentUniquelImageId** (const QString &uuid)  
*In the history, adjusts the UUID of the ImageHistoryId of the current file.*
- void **addFilterAction** (const [FilterAction](#) &action)
- QVariant **attribute** (const QString &key) const
- void **bitBlendImage** ([DColorComposer](#) \*const composer, const [DImg](#) \*const src, int sx, int sy, int w, int h, int dx, int dy, [DColorComposer::MultiplicationFlags](#) multiplicationFlags=[DColorComposer::NoMultiplication](#))  
*Blend src image on this image (this is dest) with the specified composer and multiplication flags.*
- void **bitBlendImageOnColor** (const [DColor](#) &color)
- void **bitBlendImageOnColor** (const [DColor](#) &color, int x, int y, int w, int h)
- void **bitBlendImageOnColor** ([DColorComposer](#) \*const composer, const [DColor](#) &color, int x, int y, int w, int h, [DColorComposer::MultiplicationFlags](#) multiplicationFlags=[DColorComposer::NoMultiplication](#))  
*For the specified region, blend this image on the given color with the specified composer and multiplication flags.*
- void **bitBlitImage** (const [DImg](#) \*const src, int dx, int dy)  
*Copy a region of pixels from a source image to this image.*
- void **bitBlitImage** (const [DImg](#) \*const src, int sx, int sy, int dx, int dy)
- void **bitBlitImage** (const [DImg](#) \*const src, int sx, int sy, int w, int h, int dx, int dy)
- void **bitBlitImage** (const uchar \*const src, int sx, int sy, int w, int h, int dx, int dy, uint swidth, uint sheight, int sdepth)
- uchar \* **bits** () const
- int **bitsDepth** () const  
*Return the number of bits depth of one color component for one pixel : 8 (non sixteenBit) or 16 (sixteen)*
- int **bytesDepth** () const  
*Return the number of bytes depth of one pixel : 4 (non sixteenBit) or 8 (sixteen)*
- void **convertDepth** (int depth)  
*Convert depth of image.*
- void **convertToDepthOfImage** (const [DImg](#) \*const otherImage)
- void **convertToEightBit** ()
- QPixmap **convertToPixmap** () const
- QPixmap **convertToPixmap** ([IccTransform](#) &monitorICCTrans) const
- void **convertToSixteenBit** ()  
*Wrapper methods for convertDepth.*
- [DImg](#) **copy** () const

- Return a deep copy of full image.*

  - **DImg copy** (const QRect &rect) const

*Return a region of image.*

  - **DImg copy** (const QRectF &relativeRect) const
  - **DImg copy** (int x, int y, int w, int h) const
  - uchar \* **copyBits** () const
  - **DImg copyImageData** () const

*Return a deep copy of the image, but do not include metadata.*

  - **DImg copyMetaData** () const

*Return an image that contains a deep copy of this image's metadata and the information associated with the image data (width, height, hasAlpha, sixteenBit), but no image data, i.e.*

  - QImage **copyQImage** () const

*QImage wrapper methods.*

  - QImage **copyQImage** (const QRect &rect) const
  - QImage **copyQImage** (const QRectF &relativeRect) const
  - QImage **copyQImage** (int x, int y, int w, int h) const
  - QImage **copyQImage32** () const
  - **HistoryImageId createHistoryImageId** (const QString &filePath, [HistoryImageId::Type](#) type)

*Create a [HistoryImageId](#) for this image already saved at the given file path.*

  - QByteArray **createImageUniqueId** ()

*This method creates a new 256-bit UUID meant to be globally unique.*

  - void **crop** (const QRect &rect)

*Crop image to the specified region.*

  - void **crop** (int x, int y, int w, int h)
  - void **detach** ()

*Detaches from shared data and makes sure that this image is the only one referring to the data.*

  - **FORMAT detectedFormat** () const

*Returns the file format in form of the **FORMAT** enum that was detected in the `load()` method.*

  - QString **embeddedText** (const QString &key) const
  - int **exifOrientation** (const QString &filePath)

*Retrieves the Exif orientation, either from the [LoadSaveThread](#) info provider if available, or from the metadata.*

  - bool **exifRotate** (const QString &filePath)
  - QVariant **fileOriginData** () const

*When loaded from a file, some attributes like format and isReadOnly still depend on this originating file.*

  - void **fill** (const [DColor](#) &color)

*Fill whole image with specified color.*

  - void **flip** (**FLIP** direction)
  - QString **format** () const

*Returns the format string as written by the image loader this image was originally loaded from.*

  - **IccProfile getIccProfile** () const
  - **DImageHistory & getItemHistory** ()
  - const **DImageHistory & getItemHistory** () const
  - **MetaEngineData getMetadata** () const

*Metadata manipulation methods.*

  - **DImageHistory getOriginalImageHistory** () const
  - **DColor getPixelColor** (uint x, uint y) const

*Access a single pixel of the image.*

  - **DColor getSubPixelColor** (float x, float y) const
  - **DColor getSubPixelColorFast** (float x, float y) const
  - QByteArray **getUniqueHash** ()

*This methods return a 128-bit MD5 hex digest which is meant to uniquely identify the file.*

  - QByteArray **getUniqueHashVersion** (int version)

*Version 2: This methods return a 128-bit MD5 hex digest which is meant to uniquely identify the file.*

- bool **hasAlpha** () const
- bool **hasAttribute** (const QString &key) const
- bool **hasImageHistory** () const
- bool **hasTransparentPixels** () const
  - If the image has an alpha channel, check if there exist pixels which actually have non-opaque color, that is alpha < 1.0.*
- uint **height** () const
- void **imageSavedAs** (const QString &savePath)
  - It is common that images are not directly saved to the destination path.*
- void **insertAsReferredImage** (int afterHistoryStep, const HistoryImageId &otherImagesId)
- bool **isNull** () const
- bool **isReadOnly** () const
  - Return true if the original image file format cannot be saved.*
- QVariant **lastSavedFileOriginData** () const
- QString **lastSavedFilePath** () const
  - Returns the file path to which this DImg was saved.*
- bool **load** (const QString &filePath, bool loadMetadata, bool loadICCDData, bool loadUniqueHash, bool loadHistory, DImgLoaderObserver \*const observer=nullptr, const DRawDecoding &rawDecodingSettings=DRawDecoding())
- bool **load** (const QString &filePath, DImgLoaderObserver \*const observer=nullptr, const DRawDecoding &rawDecodingSettings=DRawDecoding())
- bool **load** (const QString &filePath, int loadFlags, DImgLoaderObserver \*const observer, const DRawDecoding &rawDecodingSettings=DRawDecoding())
- bool **loadItemInfo** (const QString &filePath, bool loadMetadata=true, bool loadICCDData=true, bool loadUniqueHash=true, bool loadImageHistory=true)
  - Loads most parts of the meta information, but never the image data.*
- quint64 **numBytes** () const
- quint64 **numPixels** () const
- DImg & **operator=** (const DImg &image)
  - Equivalent to the copy constructor.*
- bool **operator==** (const DImg &image) const
  - Returns whether two images are equal.*
- int **orientation** () const
  - Returns current DMetadata::Orientation from DImg.*
- int **originalBitDepth** () const
  - Returns the bit depth (in bits per channel, e.g.*
- COLORMODEL **originalColorModel** () const
  - Returns the color model in which the image was stored in the file.*
- QString **originalFilePath** () const
  - Returns the file path from which this DImg was originally loaded.*
- QSize **originalRatioSize** () const
  - Returns the size of the original file in the same aspect ratio as size().*
- QSize **originalSize** () const
  - Returns the size of the original file.*
- void **prepareMetadataToSave** (const QString &intendedDestPath, const QString &destMimeType, bool resetExifOrientationTag)
  - For convenience: Including all flags, except for ResetExifOrientationTag which can be selected.*
- void **prepareMetadataToSave** (const QString &intendedDestPath, const QString &destMimeType, const QString &originalFileName=QString(), PrepareMetadataFlags flags=PrepareMetadataFlagsAll)
- void **prepareSubPixelAccess** ()
- QImage **pureColorMask** (ExposureSettingsContainer \*const expoSettings) const
  - Return a mask image where pure white and pure black pixels are over-colored.*
- void **putImageData** (uchar \*const data, bool copyData=true)



*Overloaded function, provided for convenience, behaves essentially like the function above if data is not nullptr.*

- void **putImageData** (uint width, uint height, bool sixteenBit, bool alpha, uchar \*const data, bool copy↔Data=true)
 

*Replaces image data of this object.*
- **DRawDecoding rawDecodingSettings** () const
 

*Returns the [DRawDecoding](#) options that this [DImg](#) was loaded with.*
- void **removeAlphaChannel** ()
- void **removeAlphaChannel** (const [DColor](#) &destColor)
 

*If the image has an alpha channel and transparent pixels, it will be blended on the specified color and the alpha channel will be removed.*
- void **removeAttribute** (const QString &key)
- void **reset** ()
 

*Reset metadata and image data to null image.*
- void **resetMetaData** ()
 

*Reset metadata, but do not change image data.*
- void **resize** (int w, int h)
 

*Set width and height of this image, smoothScale it to the given size.*
- bool **reverseExifRotate** (const QString &filePath)
 

*Reverses the previous function.*
- bool **reverseRotateAndFlip** (int [orientation](#))
 

*Reverses the previous function.*
- void **rotate** (ANGLE angle)
- bool **rotateAndFlip** (int [orientation](#))
 

*Rotates and/or flip the [DImg](#) according to the given [DMetadata::Orientation](#), so that the current state is orientation and the resulting step is normal orientation.*
- bool **save** (const QString &filePath, const QString &format, [DImgLoaderObserver](#) \*const observer=nullptr)
- bool **save** (const QString &filePath, [FORMAT](#) frm, [DImgLoaderObserver](#) \*const observer=nullptr)
- QString **savedFormat** () const
 

*Returns the format string of the format that this image was last saved to.*
- uchar \* **scanLine** (uint i) const
- void **setAttribute** (const QString &key, const QVariant &value)
- void **setEmbeddedText** (const QString &key, const QString &text)
- void **setFileOriginData** (const QVariant &data)
- void **setHistoryBranch** (bool isBranch=true)
- void **setHistoryBranchAfter** (const [DImageHistory](#) &historyBeforeBranch, bool isBranch=true)
 

*Sets a step in the history to constitute the beginning of a branch.*
- void **setHistoryBranchForLastSteps** (int numberOfLastHistorySteps, bool isBranch=true)
- void **setIccProfile** (const [IccProfile](#) &profile)
- void **setItemHistory** (const [DImageHistory](#) &history)
- void **setMetadata** (const [MetaEngineData](#) &data)
- void **setPixelColor** (uint x, uint y, const [DColor](#) &color)
- bool **sixteenBit** () const
- QSize **size** () const
- [DImg](#) **smoothScale** (const QSize &destSize, Qt::AspectRatioMode aspectRatioMode=Qt::IgnoreAspect↔Ratio) const
- [DImg](#) **smoothScale** (int width, int height, Qt::AspectRatioMode aspectRatioMode=Qt::IgnoreAspectRatio) const
 

*Return a version of this image scaled to the specified size with the specified mode.*
- [DImg](#) **smoothScaleClipped** (const QSize &destSize, const QRect &clip, bool smooth=true) const
- [DImg](#) **smoothScaleClipped** (int width, int height, int clipx, int clipy, int clipwidth, int clipheight, bool smooth=true) const
 

*Executes the same scaling as [smoothScale\(width, height\)](#), but from the result of this call, returns only the section specified by clipx, clipy, clipwidth, clipheight.*

- **DImg smoothScaleSection** (const QRect &sourceRect, const QSize &destSize) const
- **DImg smoothScaleSection** (int sx, int sy, int sw, int sh, int dw, int dh) const  
*Take the region specified by the rectangle sx|sy, width and height sw \* sh, and scale it to an image with size dw \* dh.*
- uchar \* **stripImageData** ()  
*Returns the data of this image.*
- void **switchOriginToLastSaved** ()
- bool **transform** (int transformAction)  
*Rotates and/or flip the DImg according to the given transform action, which is a MetaEngineRotation::Transform←→Action.*
- bool **wasExifRotated** ()  
*Utility to make sure that an image is rotated according to Exif tag.*
- uint **width** () const

### Static Public Member Functions

- static QString **colorModelToString** (COLORMODEL colorModel)  
*Helper method to translate enum values to user presentable strings.*
- static **FORMAT fileFormat** (const QString &filePath)  
*Identify file format.*
- static QString **formatToMimeType** (**FORMAT** frm)
- static QByteArray **getUniqueHash** (const QString &filePath)
- static QByteArray **getUniqueHashVersion** (const QString &filePath, int version)
- static bool **isAnimatedImage** (const QString &filePath)  
*Return true if image file is an animation, as GIFa or NMG.*

### Friends

- class **DImgLoader**

## 9.350.1 Member Enumeration Documentation

### 9.350.1.1 FORMAT

enum `Digikam::DImg::FORMAT`

#### Enumerator

QIMAGE	QImage or ImageMagick.
--------	------------------------

### 9.350.1.2 PrepareMetadataFlag

enum `Digikam::DImg::PrepareMetadataFlag`

This method updates the stored meta engine object in preparation to a subsequent call to `save()` with the same target file. 'intendedDestPath' is the finally intended file name. Do not give the temporary file name if you are going to `save()` to a temp file. 'destMimeType' is destination type mime. In some cases, metadata is updated depending on this value. 'originalFileName' is the original file's name, for simplistic history tracking in metadata. This is completely

independent from the [DImageHistory](#) framework. For the 'flags' see below. Not all steps are optional and can be controlled with flags.

## Enumerator

RemoveOldMetadataPreviews	A small preview can be stored in the metadata. Remove old preview entries
CreateNewMetadataPreview	Create a new preview from current image data.
ResetExifOrientationTag	Set the exif orientation tag to "normal" Applicable if the image data was rotated according to the tag.
CreateNewImageHistoryUUID	Creates a new UUID for the image history. Applicable if the file was changed.

## 9.350.2 Constructor & Destructor Documentation

### 9.350.2.1 DImg() [1/4]

```
DigiKam::DImg::DImg ( )
```

[DImg](#) is a framework to support 16bits color depth image.

it doesn't aim to be a complete imaging library; it uses QImage/ImageMagick for load/save files which are not supported natively by it. some of the features:

- Native Image Loaders, for some imageformats which are of interest to us: JPEG (complete), TIFF (mostly complete), PNG (complete), JPEG2000 (complete), RAW (complete through libraw), PGF (complete). For the rest ImageMAGick codecs or qimageloader are used.
- Metadata preservation: when a file is loaded, its metadata like XMP, IPTC, EXIF, JFIF are read and held in memory. now when you save back the file to the original file or to a different file, the metadata is automatically written. All is delegate to Exiv2 library.
- Explicitly Shared Container format (see qt docs): this is necessary for performance reasons.
- 8 bits and 16 bits support: if the file format is 16 bits, it will load up the image in 16bits format (TIFF/PNG/↔ JPEG2000/RAW/PGF support) and all operations are done in 16 bits format, except when the rendering to screen is done, when its converted on the fly to a temporary 8 bits image and then rendered.
- Basic image manipulation: rotate, flip, color modifications, crop, scale. This has been ported from Imlib2 with 16 bits scaling support and support for scaling of only a section of the image.
- Rendering to Pixmap: using QImage/QPixmap. (see above for rendering of 16 bits images).
- Pixel format: the pixel format is different from QImage pixel format. In QImage the pixel data is stored as unsigned ints and to access the individual colors you need to use bit-shifting to ensure endian correctness. in [DImg](#), the pixel data is stored as unsigned char. the color layout is B,G,R,A (blue, green, red, alpha)

for 8 bits images: you can access individual color components like this:

```
uchar* const pixels = image.bits();
```

```
for (int i = 0 ; i < image.width() * image.height() ; ++i) { pixel[0] // blue pixel[1] // green pixel[2] // red pixel[3] // alpha
pixel += 4; // go to next pixel }
```

and for 16 bits images:

```
ushort* const pixels = (ushort*)image.bits();
```

```
for (int i = 0 ; i < image.width() * image.height() ; ++i) { pixel[0] // blue pixel[1] // green pixel[2] // red pixel[3] // alpha
pixel += 4; // go to next pixel }
```

The above is true for both big and little endian platforms. What this also means is that the pixel format is different from that of QImage for big endian machines. Functions are provided if you want to get a copy of the [DImg](#) as a QImage.

### 9.350.2.2 DImg() [2/4]

```
Digikam::DImg::DImg (
    const DImg & image )
```

The two images will be equal. Call [detach\(\)](#) or [copy\(\)](#) to create deep copies.

### 9.350.2.3 DImg() [3/4]

```
Digikam::DImg::DImg (
    const QImage & image ) [explicit]
```

If the QImage is null, a null [DImg](#) will be created.

### 9.350.2.4 DImg() [4/4]

```
Digikam::DImg::DImg (
    uint width,
    uint height,
    bool sixteenBit,
    bool alpha = false,
    uchar *const data = nullptr,
    bool copyData = true )
```

If data is 0, a new buffer will be allocated, otherwise the given data will be used: If copydata is true, the data will be copied to a newly allocated buffer. If copyData is false, this [DImg](#) object will take ownership of the data pointer. If there is an alpha channel, the data shall be in non-premultiplied form (unassociated alpha).

## 9.350.3 Member Function Documentation

### 9.350.3.1 addAsReferredImage()

```
HistoryImageId Digikam::DImg::addAsReferredImage (
    const QString & filePath,
    HistoryImageId::Type type = HistoryImageId::Intermediate )
```

First call [updateMetadata\(\)](#), then call [save\(\)](#), then call [addAsReferredImage\(\)](#). Do not call this directly after loading, before applying any changes: The history is correctly initialized when loading. If you need to insert the referred file to an entry which is not the last entry, which may happen if the added image was saved after this image's history was created, you can use [insertAsReferredImage](#). The added id is returned.

### 9.350.3.2 addCurrentUniqueImageId()

```
void Digikam::DImg::addCurrentUniqueImageId (
    const QString & uuid )
```

Call this if you have associated a UUID with this file which is not written to the metadata. If there is already a UUID present, read from metadata, it will not be replaced.

### 9.350.3.3 bitBlendImage()

```
void Digikam::DImg::bitBlendImage (
    DColorComposer *const composer,
    const DImg *const src,
    int sx,
    int sy,
    int w,
    int h,
    int dx,
    int dy,
    DColorComposer::MultiplicationFlags multiplicationFlags = DColorComposer::NoMultiplication
)
```

See documentation of [DColorComposer](#) for more info. For the other arguments, see documentation of [bitBlImage](#) above.

### 9.350.3.4 bitBlendImageOnColor()

```
void Digikam::DImg::bitBlendImageOnColor (
    DColorComposer *const composer,
    const DColor & color,
    int x,
    int y,
    int w,
    int h,
    DColorComposer::MultiplicationFlags multiplicationFlags = DColorComposer::NoMultiplication
)
```

See documentation of [DColorComposer](#) for more info. Note that the result pixel is again written to this image, which is, for the blending, source.

### 9.350.3.5 bitBlImage()

```
void Digikam::DImg::bitBlImage (
    const DImg *const src,
    int dx,
    int dy )
```

Parameters: *sx|sy* Coordinates in the source image of the rectangle to be copied *w h* Width and height of the rectangle (Default, or when both are -1: whole source image) *dx|dy* Coordinates in this image of the rectangle in which the region will be copied (Default: 0|0) The bit depth of source and destination must be identical.

### 9.350.3.6 convertDepth()

```
void Digikam::DImg::convertDepth (
    int depth )
```

Depth is bytesDepth \* bitsDepth. If depth is 32, converts to 8 bits, if depth is 64, converts to 16 bits.

### 9.350.3.7 copyMetaData()

```
DImg Digikam::DImg::copyMetaData ( ) const
```

isNull() is true.

### 9.350.3.8 createImageUniqueId()

```
QByteArray Digikam::DImg::createImageUniqueId ( )
```

The UUID will be returned as a 64-byte hexadecimal string. At least 128bits of the UUID will be created by the platform random number generator. The rest may be created from a content-based hash similar to the uniqueHash, see above. This method only generates a new UUID for this image without in any way changing this image object or saving the UUID anywhere.

### 9.350.3.9 detach()

```
void Digikam::DImg::detach ( )
```

If multiple images share common data, this image makes a copy of the data and detaches itself from the sharing mechanism. Nothing is done if there is just a single reference.

### 9.350.3.10 detectedFormat()

```
DImg::FORMAT Digikam::DImg::detectedFormat ( ) const
```

Other than the format attribute which is written by the [DImgLoader](#), this can include the QIMAGE or NONE values. Returns NONE for images that have not been loaded. For unknown image formats, a value of QIMAGE can be returned to indicate that the QImage-based loader will have been used. To find out if this has worked, check the return value you got from load().

### 9.350.3.11 fileOriginData()

```
QVariant Digikam::DImg::fileOriginData ( ) const
```

When saving in a different format to a different file, you may wish to switch these attributes to the new file.

#### See also

[fileOriginData](#) returns the current origin data, bundled in the returned QVariant.

[setFileOriginData](#) takes such a variant and adjusts the properties.

[lastSavedFileOriginData](#) returns the origin data as if the image was loaded from the last saved image.

[switchOriginToLastSaved](#) is equivalent to setting origin data returned from [lastSavedFileOriginData](#).

For example, an image loaded from a RAW and saved to PNG will be read-only and format RAW. After calling

**See also**

`switchOriginToLastSaved`, it will not be read-only, `format` will be PNG, and `rawDecodingSettings` will be null.

`detectedFormat` will not change. In the history, the last referred image that was added (as intermediate) is made the new Current image.

**Note**

Set the saved image path with

**See also**

`imageSavedAs` before!

**9.350.3.12 fill()**

```
void Digikam::DImg::fill (
    const DColor & color )
```

The bit depth of the color must be identical to the depth of this image.

**9.350.3.13 format()**

```
QString Digikam::DImg::format ( ) const
```

Format strings used include JPEG, PNG, TIFF, PGF, JP2K, RAW, PPM. For images loaded with the platform QImage loader, the file suffix is used. Returns null if this `DImg` was not loaded from a file, but created in memory.

**9.350.3.14 getPixelColor()**

```
DColor Digikam::DImg::getPixelColor (
    uint x,
    uint y ) const
```

These functions add some safety checks and then use the methods from `DColor`. In optimized code working directly on the data, better use the inline methods from `DColor`.

**9.350.3.15 getUniqueHash()**

```
QByteArray Digikam::DImg::getUniqueHash ( )
```

The hash is calculated on parts of the file and the file metadata. It cannot be used to find similar images. It is not calculated from the image data. The hash will be returned as a 32-byte hexadecimal string.

If you already have a `DImg` object of the file, use the member method. The object does not need to have the full image data loaded, but it shall at least have been loaded with `loadItemInfo` with `loadMetadata = true`, or have the metadata set later with `setComments`, `setExif`, `setIptc`, `setXmp`. If the object does not have the metadata loaded, a non-null, but invalid hash will be returned! In this case, use the static method. If the image has been loaded with `loadUniqueHash = true`, the hash can be retrieved with the member method.

You do not need a `DImg` object of the file to retrieve the unique hash; Use the static method and pass just the file path.



### 9.350.3.16 `getUniqueHashVersion()`

```
QByteArray Digikam::DImg::getUniqueHashVersion (
    int version )
```

The hash is calculated on parts of the file. It cannot be used to find similar images. It is not calculated from the image data. The hash will be returned as a 32-byte hexadecimal string.

Version 3: This methods return a 128-bit MD5 hex digest which is meant to uniquely identify the file. It cannot be used to find similar images. The hash is calculated from 6 blocks distributed across the file, the first block has a size of 100 kB (capture metadata), all other possible 5 blocks up to 25 kB. The hash will be returned as a 32-byte hexadecimal string.

If you already have a [DImg](#) object loaded from the file, use the member method. If the image has been loaded with `loadUniqueHash = true`, the hash will already be available.

You do not need a [DImg](#) object of the file to retrieve the unique hash; Use the static method and pass just the file path and version.

### 9.350.3.17 `hasTransparentPixels()`

```
bool Digikam::DImg::hasTransparentPixels ( ) const
```

Note that all pixels are scanned to reach a return value of "false". If `hasAlpha()` is false, always returns false.

### 9.350.3.18 `imageSavedAs()`

```
void Digikam::DImg::imageSavedAs (
    const QString & savePath )
```

For this reason, `save()` does not call [addAsReferredImage\(\)](#), and the stored save path may be wrong. Call this method after `save()` with the final destination path. This path will be stored in the image history as well.

### 9.350.3.19 `isReadOnly()`

```
bool Digikam::DImg::isReadOnly ( ) const
```

This is depending of `DImgLoader::save()` implementation. For example RAW file formats are supported by [DImg](#) using `dcrw` than cannot support writing operations.

### 9.350.3.20 `lastSavedFilePath()`

```
QString Digikam::DImg::lastSavedFilePath ( ) const
```

Returns the file path set with [imageSavedAs\(\)](#), if that was not called, `save()`, if that was not called, a null string.

### 9.350.3.21 loadItemInfo()

```
bool Digikam::DImg::loadItemInfo (
    const QString & filePath,
    bool loadMetadata = true,
    bool loadICCDData = true,
    bool loadUniqueHash = true,
    bool loadImageHistory = true )
```

If `loadMetadata` is true, the metadata will be available with `getComments`, `getExif`, `getIptc`, `getXmp` . If `loadICCDData` is true, the ICC profile will be available with `getICCProfile`.

### 9.350.3.22 operator==( )

```
bool Digikam::DImg::operator==(
    const DImg & image ) const
```

Two images are equal if and only if they refer to the same shared data. (Thus, `DImg() == DImg()` is not true, both instances refer to their own shared data. `image == DImg(image)` is true.) If two or more images refer to the same data, they have the same image data, `bits()` returns the same data, they have the same metadata, and a change to one image also affects the others. Call `detach()` to split one image from the group of equal images.

### 9.350.3.23 originalBitDepth()

```
int Digikam::DImg::originalBitDepth ( ) const
```

8 or 16) of the original file.

### 9.350.3.24 originalColorModel()

```
DImg::COLORMODEL Digikam::DImg::originalColorModel ( ) const
```

The color space of the loaded image data is always RGB.

### 9.350.3.25 originalFilePath()

```
QString Digikam::DImg::originalFilePath ( ) const
```

Returns a null string if the `DImg` was not loaded from a file.

### 9.350.3.26 prepareMetadataToSave()

```
void Digikam::DImg::prepareMetadataToSave (
    const QString & intendedDestPath,
    const QString & destMimeType,
    bool resetExifOrientationTag )
```

Uses `originalFilePath()` to fill the original file name.

### 9.350.3.27 pureColorMask()

```
QImage Digikam::DImg::pureColorMask (
    ExposureSettingsContainer *const expoSettings ) const
```

This way is used to identify over and under exposed pixels.

### 9.350.3.28 putImageData() [1/2]

```
void Digikam::DImg::putImageData (
    uchar *const data,
    bool copyData = true )
```

Uses current width, height, sixteenBit, and alpha values. If data is nullptr, the current data are deleted and the image is set to null (But metadata are unchanged).

### 9.350.3.29 putImageData() [2/2]

```
void Digikam::DImg::putImageData (
    uint width,
    uint height,
    bool sixteenBit,
    bool alpha,
    uchar *const data,
    bool copyData = true )
```

Metadata are unchanged. Parameters like constructor above.

### 9.350.3.30 rawDecodingSettings()

```
DRawDecoding Digikam::DImg::rawDecodingSettings ( ) const
```

If this is not a RAW image or no options were specified, returns DRawDecoding().

### 9.350.3.31 removeAlphaChannel()

```
void Digikam::DImg::removeAlphaChannel (
    const DColor & destColor )
```

This is a no-op if [hasTransparentPixels\(\)](#) is false, but this method can be expensive, therefore it is *not* checked inside [removeAlphaChannel\(\)](#). (the trivial [hasAlpha\(\)](#) is checked)

### 9.350.3.32 rotateAndFlip()

```
bool Digikam::DImg::rotateAndFlip (
    int orientation )
```

Returns true if the image was actually rotated or flipped (e.g. if ORIENTATION\_NORMAL is given, returns false, because no action is taken).

**9.350.3.33 savedFormat()**

```
QString Digikam::DImg::savedFormat ( ) const
```

An image can be loaded from a file - retrieve that format with [fileFormat\(\)](#) and [loadedFormat\(\)](#) - and can the multiple times be saved to different formats. Format strings used include JPG, PGF, PNG, TIFF and JP2K. If this file was not save, a null string is returned.

**9.350.3.34 setHistoryBranchAfter()**

```
void Digikam::DImg::setHistoryBranchAfter (
    const DImageHistory & historyBeforeBranch,
    bool isBranch = true )
```

Use [setHistoryBranch\(\)](#) to take [getOriginalImageHistory\(\)](#) and set the first added step as a branch. Use [setHistoryBranchForLastSteps\(n\)](#) to start the branch before the last n steps in the history. (Assume the history had 3 steps and you added 2, call [setHistoryBranchForLastSteps\(2\)](#)) Use [setHistoryBranchAfter\(\)](#) if have a copy of the history before branching, the first added step on top of that history will be made a branch.

**9.350.3.35 smoothScale()**

```
DImg Digikam::DImg::smoothScale (
    int width,
    int height,
    Qt::AspectRatioMode aspectRatioMode = Qt::IgnoreAspectRatio ) const
```

See QSize documentation for information on available modes

**9.350.3.36 smoothScaleClipped()**

```
DImg Digikam::DImg::smoothScaleClipped (
    int width,
    int height,
    int clipx,
    int clipy,
    int clipwidth,
    int clipheight,
    bool smooth = true ) const
```

This is thus equivalent to calling `Dimg scaled = smoothScale(width, height); scaled.crop(clipx, clipy, clipwidth, clipheight);` but potentially much faster. In [smoothScaleSection](#), you specify the source region, here, the result region. It will often not be possible to find *integer* source coordinates for a result region!

**9.350.3.37 stripImageData()**

```
uchar * Digikam::DImg::stripImageData ( )
```

Ownership of the buffer is passed to the caller, this image will be null afterwards.

### 9.350.3.38 transform()

```
bool Digikam::DImg::transform (
    int transformAction )
```

Returns true if the image was actually rotated or flipped.

### 9.350.3.39 wasExifRotated()

```
bool Digikam::DImg::wasExifRotated ( )
```

Detects if an image has previously already been rotated: You can call this method more than one time on the same image. Returns true if the image has actually been rotated or flipped. Returns false if a rotation was not needed.

## 9.351 Digikam::DImgBuiltinFilter Class Reference

### Public Types

- enum [Type](#) {  
**NoOperation** , **Rotate90** , **Rotate180** , **Rotate270** ,  
**FlipHorizontally** , **FlipVertically** , **Crop** , **Resize** ,  
**ConvertTo8Bit** , **ConvertTo16Bit** }

### Public Member Functions

- **DImgBuiltinFilter** ()=default  
*Create a filter performing no operation.*
- **DImgBuiltinFilter** (const [FilterAction](#) &action)  
*Create a filter for the given action.*
- **DImgBuiltinFilter** ([Type](#) type, const QVariant &arg=QVariant())  
*Create a filter of the given type.*
- void **apply** ([DImg](#) &image) const  
*Apply the described change to the given image reference.*
- [DImgThreadedFilter](#) \* **createThreadedFilter** ([DImg](#) \*const orgImage, QObject \*const parent=nullptr) const
- [DImgThreadedFilter](#) \* **createThreadedFilter** (QObject \*const parent=nullptr) const  
*Returns a [DImgThreadedFilter](#) which executes this builtin action.*
- QString **displayName** () const  
*Returns a displayName for this filter.*
- [FilterAction](#) **filterAction** () const  
*Returns the [FilterAction](#) describing this filter.*
- QString **filterIcon** () const
- QString **i18nDisplayName** () const
- bool **isReversible** () const
- bool **isValid** () const  
*Checks that the action is supported and valid arguments are set.*
- [DImgBuiltinFilter](#) **reverseFilter** () const  
*Returns the reverse action of this filter.*
- void **setAction** (const [FilterAction](#) &action)  
*same as constructor*
- void **setAction** ([Type](#) type, const QVariant &arg=QVariant())

## Static Public Member Functions

- static QString **filterIcon** (const QString &filterIdentifier)
- static QString **i18nDisplayName** (const QString &filterIdentifier)
- static bool **isSupported** (const QString &filterIdentifier)  
*Returns if the given filter and version are supported by [DImgBuiltinFilter](#).*
- static bool **isSupported** (const QString &filterIdentifier, int version)
- static QStringList **supportedFilters** ()
- static QList< int > **supportedVersions** (const QString &filterIdentifier)  
*Returns a list of supported versions of the given filter.*

## Protected Attributes

- QVariant **m\_arg**
- Type **m\_type** = NoOperation

## 9.351.1 Member Enumeration Documentation

### 9.351.1.1 Type

```
enum Digikam::DImgBuiltinFilter::Type
```

#### Enumerator

Crop	Argument: QRect.
Resize	Argument: QSize.

## 9.351.2 Constructor & Destructor Documentation

### 9.351.2.1 DImgBuiltinFilter() [1/2]

```
Digikam::DImgBuiltinFilter::DImgBuiltinFilter (
    const FilterAction & action ) [explicit]
```

If the action is not supported, the filter will perform no operation.

### 9.351.2.2 DImgBuiltinFilter() [2/2]

```
Digikam::DImgBuiltinFilter::DImgBuiltinFilter (
    Type type,
    const QVariant & arg = QVariant() ) [explicit]
```

See documentation of Type for required arguments.

## 9.351.3 Member Function Documentation

### 9.351.3.1 filterAction()

`FilterAction` Digikam::DImgBuiltinFilter::filterAction ( ) const

#### Note

The following methods are also accessed by the more general [DImgFilterManager](#) methods, so you usually do not need to call these directly.

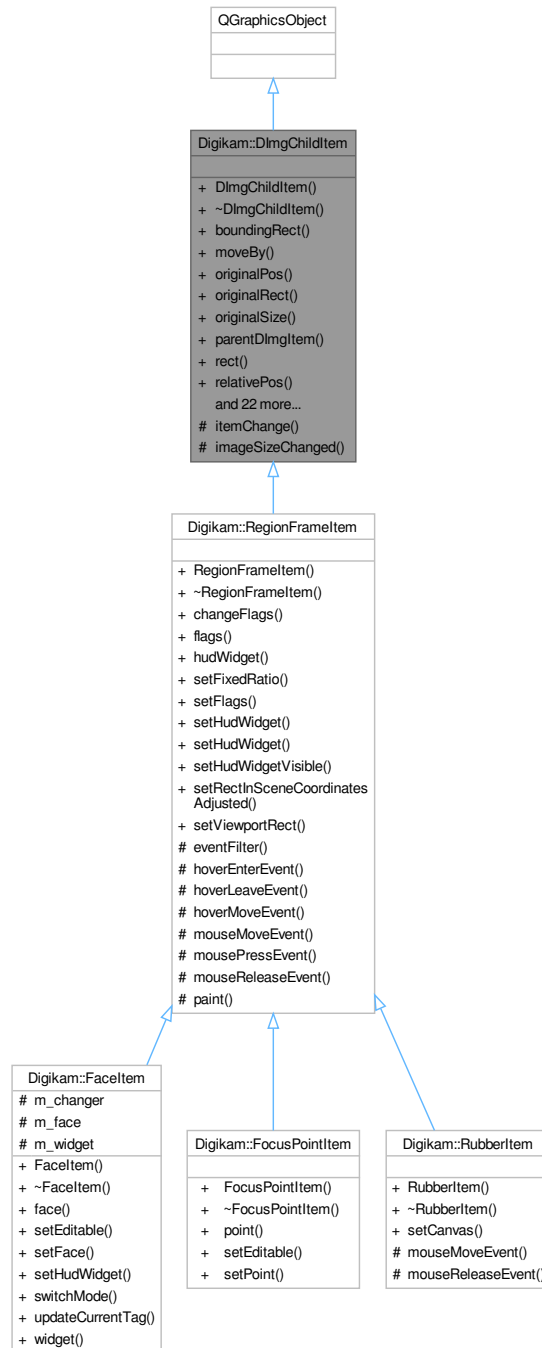
### 9.351.3.2 reverseFilter()

`DImgBuiltinFilter` Digikam::DImgBuiltinFilter::reverseFilter ( ) const

If the current action is not revertible, returns an invalid filter.

## 9.352 Digikam::DImgChildItem Class Reference

Inheritance diagram for Digikam::DImgChildItem:



### Signals

- void **geometryChanged** ()
- void **geometryOnImageChanged** ()



- void [positionChanged](#) ()  
*These signals are emitted in any case when the geometry changed: Either after changing the geometry relative to the original image, or when the size of the parent [GraphicsDImgItem](#) changed (zooming).*
- void [positionOnImageChanged](#) ()  
*These signals are emitted when the geometry, relative to the original image, of this item has changed.*
- void **sizeChanged** ()
- void **sizeOnImageChanged** ()

## Public Member Functions

- [DImgChildItem](#) (QGraphicsItem \*const parent=nullptr)  
*This is a base class for items that are positioned on top of a [GraphicsDImgItem](#), positioned in relative coordinates, i.e.*
- QRectF [boundingRect](#) () const override  
*Reimplemented.*
- void **moveBy** (qreal dx, qreal dy)
- QPoint **originalPos** () const
- QRect [originalRect](#) () const  
*Returns the position and size in coordinates of the original image.*
- QSize **originalSize** () const
- [GraphicsDImgItem](#) \* **parentDImgItem** () const  
*If the parent item is a [GraphicsDImgItem](#), return it, if the parent item is null or of a different class, returns 0.*
- QRectF [rect](#) () const  
*Returns position and size of this item, in coordinates of the parent [DImg](#) with the current zoom.*
- QPointF **relativePos** () const
- QRectF [relativeRect](#) () const  
*Returns the position and size relative to the [DImg](#) displayed in the parent item.*
- QSizeF **relativeSize** () const
- void [setOriginalPos](#) (const QPointF &posInOriginal)  
*Sets the position and size of this item, in coordinates of the original image.*
- void **setOriginalPos** (qreal x, qreal y)
- void **setOriginalRect** (const QRectF &rect)
- void **setOriginalRect** (qreal x, qreal y, qreal width, qreal height)
- void **setOriginalSize** (const QSizeF &sizeInOriginal)
- void **setOriginalSize** (qreal width, qreal height)
- void [setPos](#) (const QPointF &zoomedPos)  
*Sets the position and size of this item, in coordinates of the parent [DImg](#) item.*
- void **setPos** (qreal x, qreal y)
- void **setRect** (const QRectF &rect)
- void **setRect** (qreal x, qreal y, qreal width, qreal height)
- void **setRectInSceneCoordinates** (const QRectF &rect)  
*Equivalent to mapping the scene coordinates to the parent item, and calling [setRect](#)().*
- void [setRelativePos](#) (const QPointF &relativePosition)  
*Sets the position and size of this item, relative to the [DImg](#) displayed in the parent item.*
- void **setRelativePos** (qreal x, qreal y)
- void **setRelativeRect** (const QRectF &rect)
- void **setRelativeRect** (qreal x, qreal y, qreal width, qreal height)
- void **setRelativeSize** (const QSizeF &relativeSize)
- void **setRelativeSize** (qreal width, qreal height)
- void **setSize** (const QSizeF &zoomedSize)
- void **setSize** (qreal width, qreal height)
- QSizeF **size** () const

## Protected Slots

- void **imageSizeChanged** (const QSizeF &)

## Protected Member Functions

- QVariant **itemChange** (GraphicsItemChange change, const QVariant &value) override

## 9.352.1 Constructor & Destructor Documentation

### 9.352.1.1 DImgChildItem()

```
Digikam::DImgChildItem::DImgChildItem (
    QGraphicsItem *const parent = nullptr ) [explicit]
```

[0;1], on the image. From the set relative size, the [boundingRect\(\)](#) is calculated.

## 9.352.2 Member Function Documentation

### 9.352.2.1 boundingRect()

```
QRectF Digikam::DImgChildItem::boundingRect ( ) const [override]
```

Returns a rectangle starting at (0,0) (pos() in parent coordinates) and has a size determined by the relative size.

### 9.352.2.2 originalRect()

```
QRect Digikam::DImgChildItem::originalRect ( ) const
```

Note that the return value is integer based. At high zoom rates, different values of [relativeRect\(\)](#) or [zoomedRect\(\)](#) may result in the same [originalRect\(\)](#), when one pixel in the original is represented by more than one pixel on screen.

### 9.352.2.3 positionChanged

```
void Digikam::DImgChildItem::positionChanged ( ) [signal]
```

[positionChanged\(\)](#) is equivalent to listening to [xChanged\(\)](#) and [yChanged\(\)](#).

### 9.352.2.4 positionOnImageChanged

```
void Digikam::DImgChildItem::positionOnImageChanged ( ) [signal]
```

This happens by calling any of the methods above.

### 9.352.2.5 rect()

```
QRectF Digikam::DImgChildItem::rect ( ) const
```

This is the same result as `QRectF(pos(), boundingRect())`, `boundingRect` is virtual and may be overridden by base classes.

### 9.352.2.6 relativeRect()

```
QRectF Digikam::DImgChildItem::relativeRect ( ) const
```

All four values are in the interval [0;1].

### 9.352.2.7 setOriginalPos()

```
void Digikam::DImgChildItem::setOriginalPos (
    const QPointF & posInOriginal )
```

Requires a valid parent item.

### 9.352.2.8 setPos()

```
void Digikam::DImgChildItem::setPos (
    const QPointF & zoomedPos )
```

This is accepting unscaled parent coordinates, just like the "normal" `setPos()` does. Requires a valid parent item.

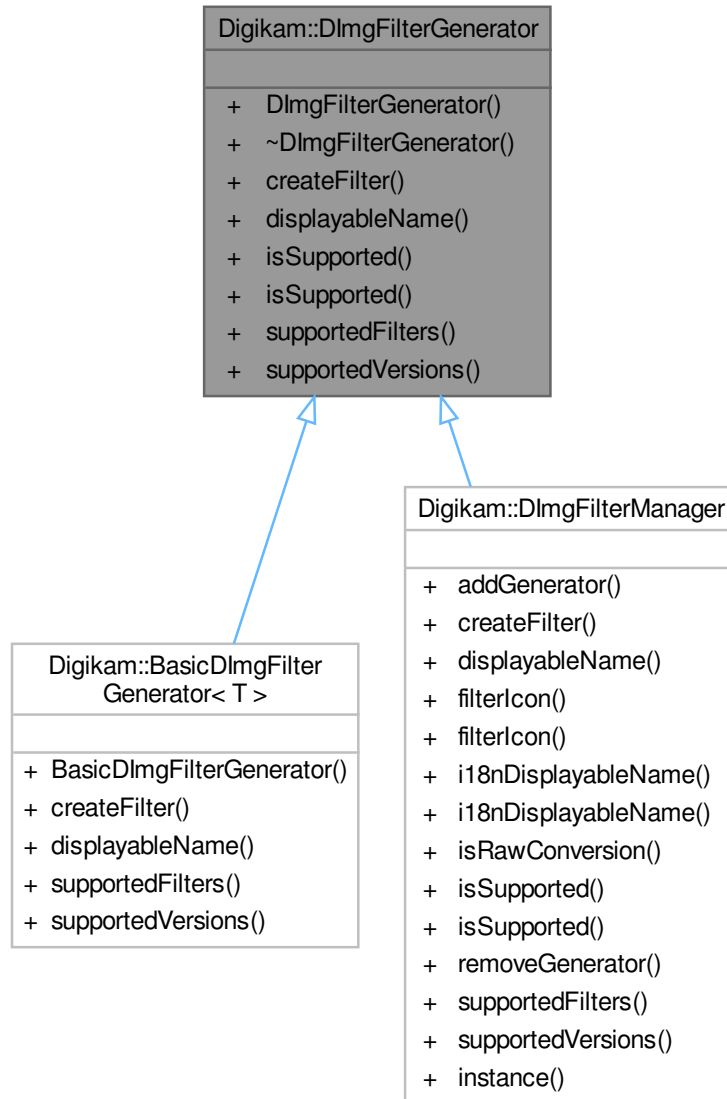
### 9.352.2.9 setRelativePos()

```
void Digikam::DImgChildItem::setRelativePos (
    const QPointF & relativePosition )
```

The values of `relativePosition` must be in the interval [0;1].

## 9.353 Digikam::DImgFilterGenerator Class Reference

Inheritance diagram for Digikam::DImgFilterGenerator:



### Public Member Functions

- virtual `DImgThreadedFilter * createFilter` (const QString &filterIdentifier, int version)=0  
*Create the filter for the given combination of identifier and version.*
- virtual QString `displayName` (const QString &filterIdentifier)=0  
*Returns a QString with filter name for displaying in views.*
- virtual bool `isSupported` (const QString &filterIdentifier)  
*Convenience methods.*
- virtual bool `isSupported` (const QString &filterIdentifier, int version)

- virtual QStringList [supportedFilters](#) ()=0  
*Returns a list with identifiers of supported filters.*
- virtual QList< int > [supportedVersions](#) (const QString &filterIdentifier)=0  
*Returns a list with the supported versions for the given identifier.*

## 9.353.1 Member Function Documentation

### 9.353.1.1 createFilter()

```
virtual DImgThreadedFilter * Digikam::DImgFilterGenerator::createFilter (
    const QString & filterIdentifier,
    int version ) [pure virtual]
```

Implemented in [Digikam::BasicDImgFilterGenerator< T >](#), and [Digikam::DImgFilterManager](#).

### 9.353.1.2 displayableName()

```
virtual QString Digikam::DImgFilterGenerator::displayableName (
    const QString & filterIdentifier ) [pure virtual]
```

Implemented in [Digikam::BasicDImgFilterGenerator< T >](#), and [Digikam::DImgFilterManager](#).

### 9.353.1.3 isSupported()

```
bool Digikam::DImgFilterGenerator::isSupported (
    const QString & filterIdentifier ) [virtual]
```

Reimplemented in [Digikam::DImgFilterManager](#).

### 9.353.1.4 supportedFilters()

```
virtual QStringList Digikam::DImgFilterGenerator::supportedFilters ( ) [pure virtual]
```

Implemented in [Digikam::BasicDImgFilterGenerator< T >](#), and [Digikam::DImgFilterManager](#).

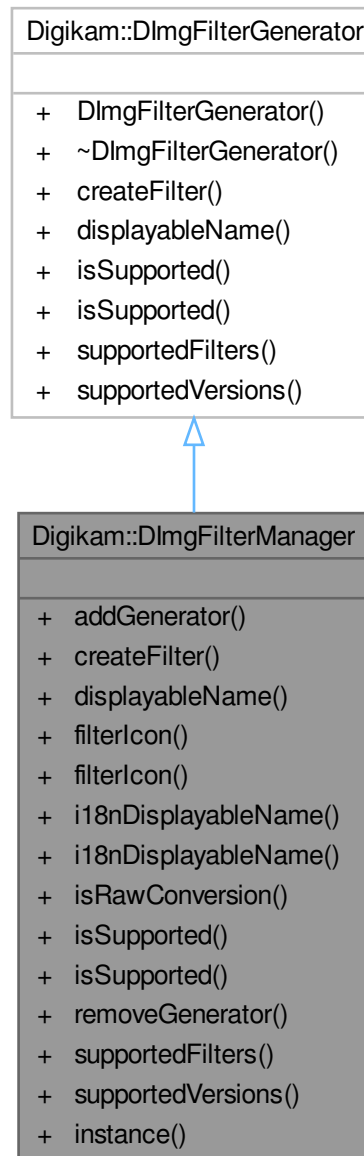
### 9.353.1.5 supportedVersions()

```
virtual QList< int > Digikam::DImgFilterGenerator::supportedVersions (
    const QString & filterIdentifier ) [pure virtual]
```

Implemented in [Digikam::BasicDImgFilterGenerator< T >](#), and [Digikam::DImgFilterManager](#).

## 9.354 Digikam::DImgFilterManager Class Reference

Inheritance diagram for Digikam::DImgFilterManager:



### Public Member Functions

- void **addGenerator** ([DImgFilterGenerator](#) \*const generator)  
*Registers all filter provided by this generator.*
- [DImgThreadedFilter](#) \* **createFilter** (const QString &filterIdentifier, int version) override  
*Create a filter from an installed manager.*
- QString **displayName** (const QString &filterIdentifier) override

Returns the (untranslated) displayable name for the given identifier.

- QString **filterIcon** (const [FilterAction](#) &action)
- QString **filterIcon** (const QString &filterIdentifier)

Returns an icon for the given filter.

- QString **i18nDisplayableName** (const [FilterAction](#) &action)
- QString **i18nDisplayableName** (const QString &filterIdentifier)

Returns the translated displayable name.

- bool **isRawConversion** (const QString &filterIdentifier)

Returns true if the given filter is to be considered as a step converting a RAW image to a normal image.

- bool **isSupported** (const QString &filterIdentifier) override

Returns true if the given filter, or, more specifically, the given filter in the given version is supported.

- bool **isSupported** (const QString &filterIdentifier, int version) override
- void **removeGenerator** ([DImgFilterGenerator](#) \*const generator)
- QStringList **supportedFilters** () override

Returns a list of the supported filter identifiers.

- QList< int > **supportedVersions** (const QString &filterIdentifier) override

Returns a list of supported versions of the given filter.

## Static Public Member Functions

- static [DImgFilterManager](#) \* **instance** ()

## Friends

- class [DImgFilterManagerCreator](#)

## 9.354.1 Member Function Documentation

### 9.354.1.1 createFilter()

```
DImgThreadedFilter * Digikam::DImgFilterManager::createFilter (
    const QString & filterIdentifier,
    int version ) [override], [virtual]
```

Returns 0 if no filter could be created. This is true if identifier/version is not supported, or the filter is builtin. Note: You probably want to use [FilterActionFilter](#).

Implements [Digikam::DImgFilterGenerator](#).

### 9.354.1.2 displayableName()

```
QString Digikam::DImgFilterManager::displayableName (
    const QString & filterIdentifier ) [override], [virtual]
```

This is only possible for supported filters. If you have a [FilterAction](#), it may already contain a displayable name.

Implements [Digikam::DImgFilterGenerator](#).

### 9.354.1.3 filterIcon()

```
QString Digikam::DImgFilterManager::filterIcon (
    const QString & filterIdentifier )
```

If no icon is known, returns a null string.

### 9.354.1.4 isSupported() [1/2]

```
bool Digikam::DImgFilterManager::isSupported (
    const QString & filterIdentifier ) [override], [virtual]
```

Reimplemented from [Digikam::DImgFilterGenerator](#).

### 9.354.1.5 isSupported() [2/2]

```
bool Digikam::DImgFilterManager::isSupported (
    const QString & filterIdentifier,
    int version ) [override], [virtual]
```

Reimplemented from [Digikam::DImgFilterGenerator](#).

### 9.354.1.6 supportedFilters()

```
QStringList Digikam::DImgFilterManager::supportedFilters ( ) [override], [virtual]
```

Implements [Digikam::DImgFilterGenerator](#).

### 9.354.1.7 supportedVersions()

```
QList< int > Digikam::DImgFilterManager::supportedVersions (
    const QString & filterIdentifier ) [override], [virtual]
```

Implements [Digikam::DImgFilterGenerator](#).

## 9.355 Digikam::DImgLoader Class Reference

### Public Types

- enum [LoadFlag](#) {
  - [LoadItemInfo](#) = 1 , [LoadMetadata](#) = 2 , [LoadICCDData](#) = 4 , [LoadImageData](#) = 8 ,
  - [LoadUniqueHash](#) = 16 , [LoadImageHistory](#) = 32 , [LoadPreview](#) = 64 , [LoadAll](#) = LoadItemInfo | LoadMetadata | LoadICCDData | LoadImageData | LoadUniqueHash | LoadImageHistory }

*This is the list of loading modes usable by [DImg](#) image plugins.*
- typedef QFlags< [LoadFlag](#) > **LoadFlags**



## Public Member Functions

- virtual bool **hasAlpha** () const =0
- virtual bool **hasLoadedData** () const
- virtual bool **isReadOnly** () const =0
- virtual bool **load** (const QString &filePath, [DImgLoaderObserver](#) \*const observer)=0
- template<typename Type >  
Q\_INLINE\_TEMPLATE Type \* **new\_failureTolerant** (quint64 w, quint64 h, uint typesPerPixel)  
*Allows safe multiplication of requested pixel number and bytes per pixel, avoiding particularly 32 bits overflow and exceeding the size\_t type.*
- template<typename Type >  
Q\_INLINE\_TEMPLATE Type \* **new\_failureTolerant** (size\_t size)
- virtual bool **save** (const QString &filePath, [DImgLoaderObserver](#) \*const observer)=0
- void **setLoadFlags** (LoadFlags flags)
- virtual bool **sixteenBit** () const =0

## Static Public Member Functions

- static qint64 **checkAllocation** (qint64 fullSize)  
*Value returned : -1 : unsupported platform 0 : parse failure from supported platform 1 : parse done with success from supported platform.*
- static int **convertCompressionForLibJpeg** (int value)
- static int **convertCompressionForLibPng** (int value)
- static unsigned char \* **new\_failureTolerant** (quint64 w, quint64 h, uint typesPerPixel)
- template<typename Type >  
static Type \* **new\_failureTolerant** (quint64 w, quint64 h, uint typesPerPixel)
- static unsigned char \* **new\_failureTolerant** (size\_t unsecureSize)
- template<typename Type >  
static Type \* **new\_failureTolerant** (size\_t unsecureSize)
- static unsigned short \* **new\_short\_failureTolerant** (quint64 w, quint64 h, uint typesPerPixel)
- static unsigned short \* **new\_short\_failureTolerant** (size\_t unsecureSize)

## Protected Member Functions

- **DImgLoader** ([DImg](#) \*const image)
- bool **checkExifWorkingColorSpace** () const
- virtual int **granularity** ([DImgLoaderObserver](#) \*const observer, int total, float progressSlice=1.0F)
- int **imageBitsDepth** () const
- int **imageBytesDepth** () const
- unsigned char \*& **imageData** ()
- QMap< QString, QString > & **imageEmbeddedText** () const
- QVariant **imageGetAttribute** (const QString &key) const
- QString **imageGetEmbeddedText** (const QString &key) const
- bool **imageHasAlpha** () const
- unsigned int & **imageHeight** ()
- quint64 **imageNumBytes** () const
- void **imageSetAttribute** (const QString &key, const QVariant &value)
- void **imageSetEmbeddedText** (const QString &key, const QString &text)
- void **imageSetIccProfile** (const [IccProfile](#) &profile)
- bool **imageSixteenBit** () const
- unsigned int & **imageWidth** ()
- void **loadingFailed** ()
- void **purgeExifWorkingColorSpace** ()
- virtual bool **readMetadata** (const QString &filePath)
- virtual bool **saveMetadata** (const QString &filePath)
- void **storeColorProfileInMetadata** ()

**Protected Attributes**

- `DImg * m_image = nullptr`
- `LoadFlags m_loadFlags = LoadAll`

**9.355.1 Member Enumeration Documentation****9.355.1.1 LoadFlag**

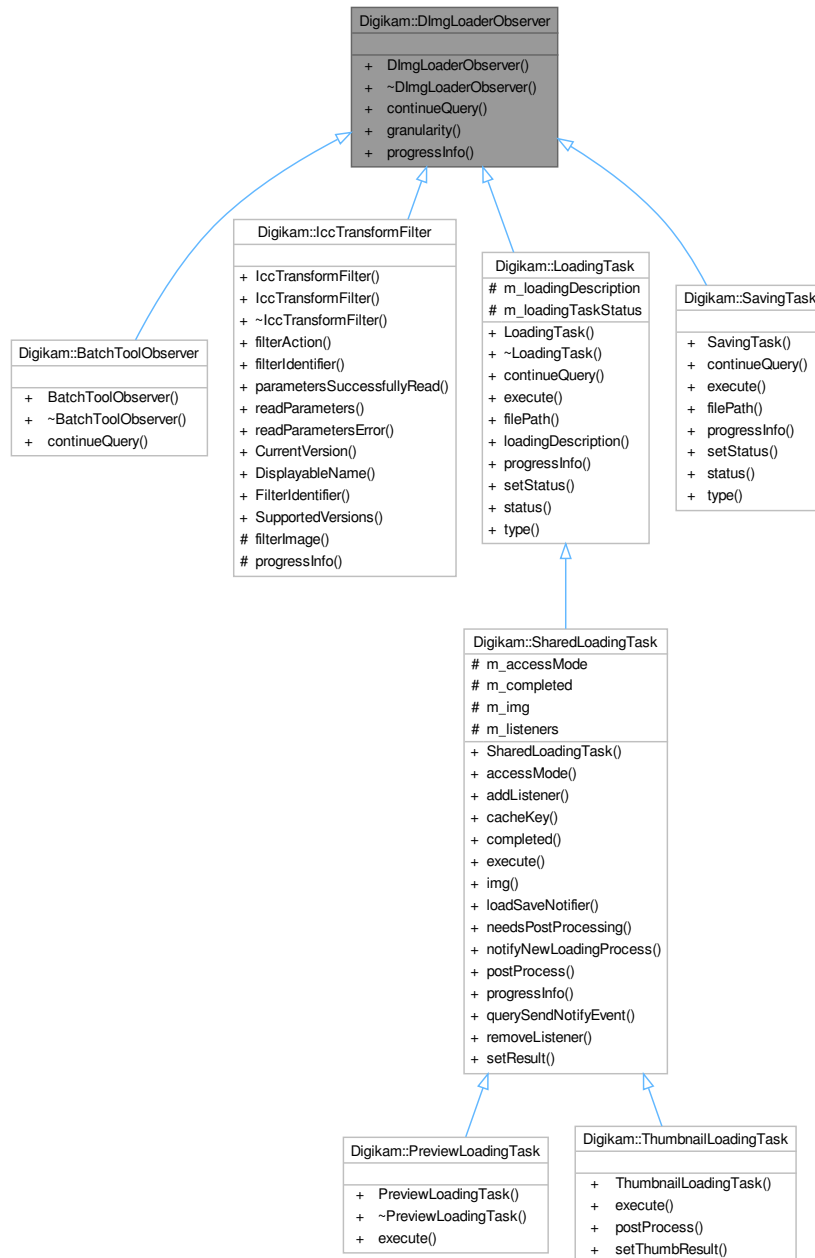
enum `Digikam::DImgLoader::LoadFlag`

**Enumerator**

<code>LoadItemInfo</code>	Load image information without image data. Image info as width and height
<code>LoadMetadata</code>	Image metadata.
<code>LoadICCData</code>	Image color profile.
<code>LoadImageData</code>	Full image data.
<code>LoadUniqueHash</code>	Image unique hash.
<code>LoadImageHistory</code>	Image version history.
<code>LoadPreview</code>	Special mode to load reduced image data. Load embedded preview image instead full size image
<code>LoadAll</code>	Helper to load all information, metadata and full image.

## 9.356 Digikam::DImgLoaderObserver Class Reference

Inheritance diagram for Digikam::DImgLoaderObserver:



### Public Member Functions

- virtual bool **continueQuery** ()  
*Queries whether the image IO operation shall be continued.*
- virtual float **granularity** ()  
*Return a relative value which determines the granularity, the frequency with which the `DImgLoaderObserver` is checked and progress is posted.*
- virtual void **progressInfo** (float progress)  
*Posts progress information about image IO.*

## 9.356.1 Member Function Documentation

### 9.356.1.1 granularity()

```
virtual float Digikam::DImgLoaderObserver::granularity ( ) [inline], [virtual]
```

Standard is 1.0. Values < 1 mean less granularity (fewer checks), values > 1 mean higher granularity (more checks).

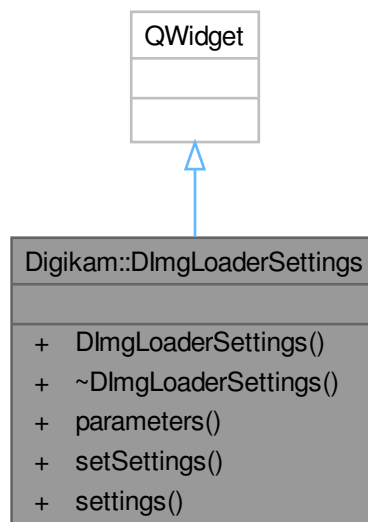
### 9.356.1.2 progressInfo()

```
virtual void Digikam::DImgLoaderObserver::progressInfo (
    float progress ) [inline], [virtual]
```

Reimplemented in [Digikam::lccTransformFilter](#).

## 9.357 Digikam::DImgLoaderSettings Class Reference

Inheritance diagram for Digikam::DImgLoaderSettings:



## Signals

- void **signalSettingsChanged** ()

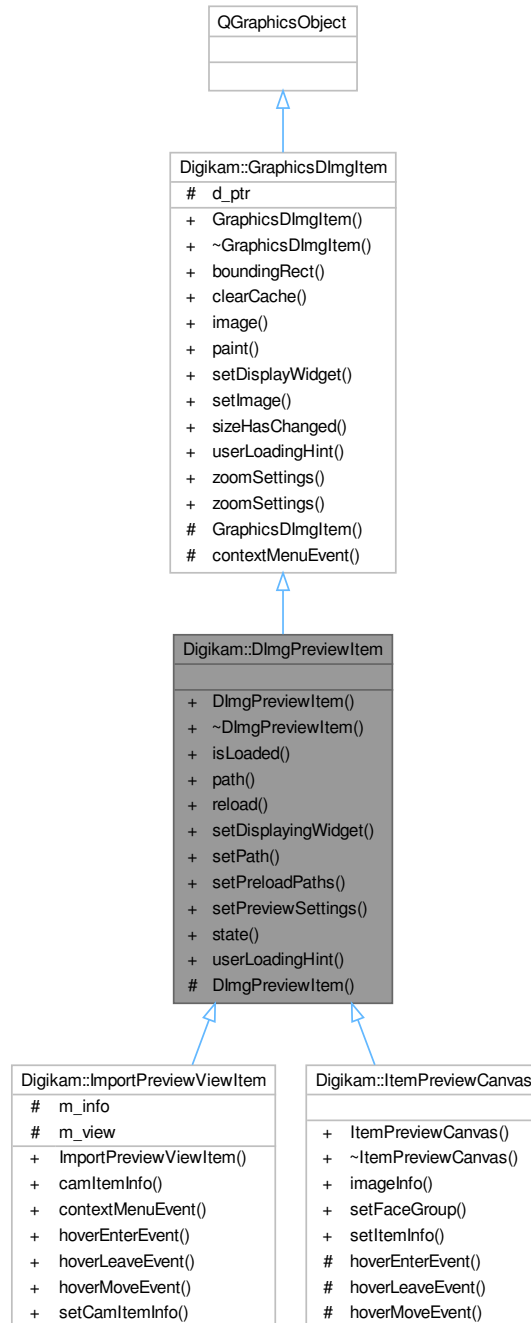
*Signal to emit when a settings is changed from the widget.*

### Public Member Functions

- **DImgLoaderSettings** (QWidget \*const parent=nullptr)
- QStringList **parameters** () const  
*Return the list of supported parameter names.*
- virtual void **setSettings** (const [DImgLoaderPrms](#) &set)=0  
*Set the parameters values in the widget from DImgLoaderPrms map container.*
- virtual [DImgLoaderPrms](#) **settings** () const =0  
*Return the DImgLoaderPrms map container of parameters/values from the Widget.*

## 9.358 Digikam::DImgPreviewItem Class Reference

Inheritance diagram for Digikam::DImgPreviewItem:



### Public Types

- enum **State** { **NoImage** , **Loading** , **ImageLoaded** , **ImageLoadingFailed** }

## Signals

- void **loaded** ()
- void **loadingFailed** ()
- void **stateChanged** (int state)

## Signals inherited from [Digikam::GraphicsDImgItem](#)

- void **imageChanged** ()
- void **imageSizeChanged** (const QSizeF &size)
- void **showContextMenu** (QGraphicsSceneContextMenuEvent \*e)

## Public Member Functions

- **DImgPreviewItem** (QGraphicsItem \*const parent=nullptr)
- bool **isLoading** () const
- QString **path** () const
- void **reload** ()
- void **setDisplayingWidget** (QWidget \*const widget)
- void **setPath** (const QString &path, bool rePreview=false)
- void **setPreloadPaths** (const QStringList &pathsToPreload)
- void **setPreviewSettings** (const [PreviewSettings](#) &settings)
- State **state** () const
- QString **userLoadingHint** () const override

## Public Member Functions inherited from [Digikam::GraphicsDImgItem](#)

- **GraphicsDImgItem** (QGraphicsItem \*const parent=nullptr)
- QRectF **boundingRect** () const override
- void **clearCache** ()
- [DImg](#) **image** () const
- void **paint** (QPainter \*painter, const QStyleOptionGraphicsItem \*option, QWidget \*widget) override
- void **setDisplayWidget** (QWidget \*const widget)
- void **setImage** (const [DImg](#) &img)
  - *Sets the [DImg](#) to be drawn by this item.*
- void **sizeHasChanged** ()
- [ImageZoomSettings](#) \* **zoomSettings** ()
- const [ImageZoomSettings](#) \* **zoomSettings** () const

## Protected Member Functions

- **DImgPreviewItem** (DImgPreviewItemPrivate &dd, QGraphicsItem \*const parent=nullptr)

## Protected Member Functions inherited from [Digikam::GraphicsDImgItem](#)

- **GraphicsDImgItem** (GraphicsDImgItemPrivate &dd, QGraphicsItem \*const parent)
- void **contextMenuEvent** (QGraphicsSceneContextMenuEvent \*e) override

### Additional Inherited Members

### Protected Attributes inherited from [Digikam::GraphicsDImgItem](#)

- GraphicsDImgItemPrivate \*const **d\_ptr**

## 9.358.1 Member Function Documentation

### 9.358.1.1 `userLoadingHint()`

```
QString Digikam::DImgPreviewItem::userLoadingHint ( ) const [override], [virtual]
```

Reimplemented from [Digikam::GraphicsDImgItem](#).



## 9.359 Digikam::DImgThreadedAnalyser Class Reference

Inheritance diagram for Digikam::DImgThreadedAnalyser:



### Public Member Functions

- [DImgThreadedAnalyser](#) (`DImg *const orgImage, QObject *const parent=nullptr, const QString &name=QString()`)  
Constructs an image analyser with all arguments (ready to use).
- [DImgThreadedAnalyser](#) (`QObject *const parent=nullptr, const QString &name=QString()`)  
Constructs a filter without argument.

## Public Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImg](#) \*const orgImage, QObject \*const parent, const QString &name=QString())  
*Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter](#) (QObject \*const parent=nullptr, const QString &name=QString())  
*Constructs a filter without argument.*
- virtual void [cancelFilter](#) ()  
*Cancel the threaded computation.*
- const QString & [filterName](#) ()
- int [filterVersion](#) () const
- [DImg](#) [getTargetImage](#) ()
- QList< int > [multithreadedSteps](#) (int stop, int start=0) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool [parametersSuccessfullyRead](#) () const  
*Optional: error handling for readParameters.*
- virtual QString [readParametersError](#) (const [FilterAction](#) &actionThatFailed) const
- void [setFilterName](#) (const QString &name)
- void [setFilterVersion](#) (int version)  
*Replaying a filter action: Set the filter version.*
- void [setOriginalImage](#) (const [DImg](#) &orgImage)
- void [setupAndStartDirectly](#) (const [DImg](#) &orgImage, [DImgThreadedFilter](#) \*const master, int progress↔ Begin=0, int progressEnd=100)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void [setupFilter](#) (const [DImg](#) &orgImage)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void [startFilter](#) ()  
*Start the threaded computation.*
- virtual void [startFilterDirectly](#) ()  
*Start computation of this filter, directly in this thread.*

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread](#) (QObject \*const parent=nullptr)  
*This class extends QRunnable, so you have to reimplement virtual void [run\(\)](#).*
- ~[DynamicThread](#) () override  
*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool [isFinished](#) () const
- bool [isRunning](#) () const
- QThread::Priority [priority](#) () const
- void [setEmitSignals](#) (bool emitThem)
- void [setPriority](#) (QThread::Priority priority)  
*Sets the priority for this dynamic thread.*
- State [state](#) () const

## Protected Member Functions

- virtual void [startAnalyse](#) ()=0  
*Main image analys method.*

## Protected Member Functions inherited from Digikam::DImgThreadedFilter

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from Digikam::DynamicThread

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Additional Inherited Members

## Public Types inherited from Digikam::DynamicThread

- enum [State](#) { [Inactive](#) , [Scheduled](#) , [Running](#) , [Deactivating](#) }

## Public Slots inherited from Digikam::DynamicThread

- void [start](#) ()
- void [stop](#) ()  
*Stop computation, sets the running flag to false.*
- void [wait](#) ()  
*Waits until the thread finishes.*

## Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

## Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Attributes inherited from [Digikam::DImgThreadedFilter](#)

- [DImg](#) **m\_destImage**  
*Output image data.*
- [DImgThreadedFilter](#) \* **m\_master** = nullptr  
*The master of this slave filter.*
- [QString](#) **m\_name**  
*Filter name.*
- [DImg](#) **m\_orgImage**  
*Copy of original Image data.*
- int **m\_progressBegin** = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int **m\_progressCurrent** = 0  
*To prevent signals bombarding with progress indicator value in [postProgress\(\)](#).*
- int **m\_progressSpan** = 0
- [DImgThreadedFilter](#) \* **m\_slave** = nullptr  
*The current slave.*
- int **m\_version** = 1
- bool **m\_wasCancelled** = false

## 9.359.1 Constructor & Destructor Documentation

### 9.359.1.1 [DImgThreadedAnalyser\(\)](#) [1/2]

```
Digikam::DImgThreadedAnalyser::DImgThreadedAnalyser (
    QObject *const parent = nullptr,
    const QString & name = QString() ) [explicit]
```

You need to call [setupFilter\(\)](#) and [startFilter\(\)](#) to start the threaded computation. To run filter without to use multi-threading, call [startFilterDirectly\(\)](#).

#### Warning

Versioning is not supported in this class

### 9.359.1.2 DImgThreadedAnalyser() [2/2]

```
Digikam::DImgThreadedAnalyser::DImgThreadedAnalyser (
    DImg *const orgImage,
    QObject *const parent = nullptr,
    const QString & name = QString() ) [explicit]
```

The given original image will be copied. You need to call [startFilter\(\)](#) to start the threaded computation. To run analyser without to use multithreading, call [startFilterDirectly\(\)](#).

## 9.359.2 Member Function Documentation

### 9.359.2.1 startAnalyse()

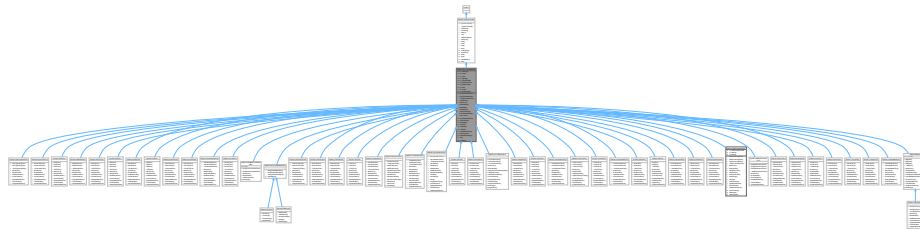
```
virtual void Digikam::DImgThreadedAnalyser::startAnalyse ( ) [protected], [pure virtual]
```

Override in subclass.

Implemented in [Digikam::NREstimate](#), and [Digikam::AutoCrop](#).

## 9.360 Digikam::DImgThreadedFilter Class Reference

Inheritance diagram for Digikam::DImgThreadedFilter:



### Classes

- class [DefaultFilterAction](#)  
*Convenience class to spare the few repeating lines of code.*

### Signals

- void [finished](#) (bool success)  
*Emitted when the computation has completed.*
- void [progress](#) (int progress)  
*Emitted when progress info from the calculation is available.*
- void [started](#) ()  
*This signal is emitted when image data is available and the computation has started.*

## Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
  - void **starting** ()
- Emitted if emitSignals is enabled.*

## Public Member Functions

- [DImgThreadedFilter](#) ([DImg](#) \*const orgImage, [QObject](#) \*const parent, const [QString](#) &name=[QString](#)())  
*Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter](#) ([QObject](#) \*const parent=nullptr, const [QString](#) &name=[QString](#)())  
*Constructs a filter without argument.*
- virtual void [cancelFilter](#) ()  
*Cancel the threaded computation.*
- virtual [FilterAction](#) [filterAction](#) ()=0  
*Returns the action description corresponding to currently set options.*
- virtual [QString](#) [filterIdentifier](#) () const =0  
*Return the identifier for this filter in the image history.*
- const [QString](#) & [filterName](#) ()
- int [filterVersion](#) () const
- [DImg](#) [getTargetImage](#) ()
- [QList](#)< int > [multithreadedSteps](#) (int stop, int start=0) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool [parametersSuccessfullyRead](#) () const  
*Optional: error handling for readParameters.*
- virtual void [readParameters](#) (const [FilterAction](#) &)=0
- virtual [QString](#) [readParametersError](#) (const [FilterAction](#) &actionThatFailed) const
- void [setFilterName](#) (const [QString](#) &name)
- void [setFilterVersion](#) (int version)  
*Replaying a filter action: Set the filter version.*
- void [setOriginalImage](#) (const [DImg](#) &orgImage)
- void [setupAndStartDirectly](#) (const [DImg](#) &orgImage, [DImgThreadedFilter](#) \*const master, int progress←Begin=0, int progressEnd=100)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void [setupFilter](#) (const [DImg](#) &orgImage)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void [startFilter](#) ()  
*Start the threaded computation.*
- virtual void [startFilterDirectly](#) ()  
*Start computation of this filter, directly in this thread.*
- virtual [QList](#)< int > [supportedVersions](#) () const

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread](#) ([QObject](#) \*const parent=nullptr)  
*This class extends [QRunnable](#), so you have to reimplement virtual void [run\(\)](#).*
- ~[DynamicThread](#) () override  
*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool [isFinished](#) () const
- bool [isRunning](#) () const
- [QThread::Priority](#) [priority](#) () const
- void [setEmitSignals](#) (bool emitThem)
- void [setPriority](#) ([QThread::Priority](#) priority)  
*Sets the priority for this dynamic thread.*
- State [state](#) () const

### Protected Member Functions

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [filterImage](#) ()=0  
*Main image filter method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

### Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

### Protected Attributes

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0

*The progress span that a slave filter uses in the parent filter's progress.*

- int **m\_progressCurrent** = 0  
*To prevent signals bombarding with progress indicator value in [postProgress\(\)](#).*
- int **m\_progressSpan** = 0
- [DImgThreadedFilter](#) \* **m\_slave** = nullptr  
*The current slave.*
- int **m\_version** = 1
- bool **m\_wasCancelled** = false

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

## 9.360.1 Constructor & Destructor Documentation

### 9.360.1.1 [DImgThreadedFilter\(\)](#) [1/3]

```
Digikam::DImgThreadedFilter::DImgThreadedFilter (
    QObject *const parent = nullptr,
    const QString & name = QString() ) [explicit]
```

You need to call [setupFilter\(\)](#) and [startFilter\(\)](#) to start the threaded computation. To run filter without to use multi-threading, call [startFilterDirectly\(\)](#).

### 9.360.1.2 [DImgThreadedFilter\(\)](#) [2/3]

```
Digikam::DImgThreadedFilter::DImgThreadedFilter (
    DImg *const orgImage,
    QObject *const parent,
    const QString & name = QString() )
```

The given original image will be copied. You need to call [startFilter\(\)](#) to start the threaded computation. To run filter without to use multithreading, call [startFilterDirectly\(\)](#).



### 9.360.1.3 DImgThreadedFilter() [3/3]

```
Digikam::DImgThreadedFilter::DImgThreadedFilter (
    DImgThreadedFilter *const master,
    const DImg & orgImage,
    const DImg & destImage,
    int progressBegin = 0,
    int progressEnd = 100,
    const QString & name = QString() ) [protected]
```

Do not call [startFilter\(\)](#) or [startFilterDirectly\(\)](#) on this. The computation will be started from [initFilter\(\)](#) which you must call from the derived class constructor.

Constructor for slave mode: Constructs a new slave filter with the specified master. The filter will be executed in the current thread. `orgImage` and `destImage` will not be copied. Note that the slave is still free to reallocate his `destImage`. `progressBegin` and `progressEnd` can indicate the progress span that the slave filter uses in the parent filter's progress. Any derived filter class that is publicly available to other filters should implement an additional constructor using this constructor.

## 9.360.2 Member Function Documentation

### 9.360.2.1 cancelFilter()

```
void Digikam::DImgThreadedFilter::cancelFilter ( ) [virtual]
```

Reimplemented in [Digikam::GreycstorationFilter](#).

### 9.360.2.2 cleanupFilter()

```
virtual void Digikam::DImgThreadedFilter::cleanupFilter ( ) [inline], [protected], [virtual]
```

Override in subclass.

### 9.360.2.3 filterAction()

```
virtual FilterAction Digikam::DImgThreadedFilter::filterAction ( ) [pure virtual]
```

Implemented in [Digikam::AutoExpoFilter](#), [Digikam::AutoLevelsFilter](#), [Digikam::EqualizeFilter](#), [Digikam::NormalizeFilter](#), [Digikam::StretchFilter](#), [Digikam::BCGFilter](#), [Digikam::BWSepiaFilter](#), [Digikam::InfraredFilter](#), [Digikam::MixerFilter](#), [Digikam::TonalityFilter](#), [Digikam::CBFilter](#), [Digikam::CurvesFilter](#), [Digikam::BorderFilter](#), [Digikam::TextureFilter](#), [Digikam::FilmFilter](#), [Digikam::FilterActionFilter](#), [Digikam::BlurFilter](#), [Digikam::BlurFXFilter](#), [Digikam::CharcoalFilter](#), [Digikam::ColorFXFilter](#), [Digikam::DistortionFXFilter](#), [Digikam::EmbossFilter](#), [Digikam::FilmGrainFilter](#), [Digikam::InvertFilter](#), [Digikam::OilPaintFilter](#), [Digikam::RainDropFilter](#), [Digikam::GreycstorationFilter](#), [Digikam::HotPixelFixer](#), [Digikam::HSLFilter](#), [Digikam::IccTransformFilter](#), [Digikam::LocalContrastFilter](#), [Digikam::AntiVignettingFilter](#), [Digikam::LensDistortionFilter](#), [Digikam::LensFunFilter](#), [Digikam::LevelsFilter](#), [Digikam::NRFilter](#), [Digikam::RawProcessingFilter](#), [Digikam::RedEyeCorrectionFilter](#), [Digikam::RefocusFilter](#), [Digikam::SharpenFilter](#), [Digikam::UnsharpMaskFilter](#), [Digikam::ContentAwareFilter](#), [Digikam::FreeRotationFilter](#), [Digikam::ShearFilter](#), and [Digikam::WBFilter](#).

### 9.360.2.4 filterIdentifier()

```
virtual QString Digikam::DImgThreadedFilter::filterIdentifier ( ) const [pure virtual]
```

Implemented in [Digikam::AutoExpoFilter](#), [Digikam::AutoLevelsFilter](#), [Digikam::EqualizeFilter](#), [Digikam::NormalizeFilter](#), [Digikam::StretchFilter](#), [Digikam::BCGFilter](#), [Digikam::BWSepiaFilter](#), [Digikam::InfraredFilter](#), [Digikam::MixerFilter](#), [Digikam::TonalityFilter](#), [Digikam::CBFilter](#), [Digikam::CurvesFilter](#), [Digikam::BorderFilter](#), [Digikam::TextureFilter](#), [Digikam::FilmFilter](#), [Digikam::FilterActionFilter](#), [Digikam::BlurFilter](#), [Digikam::BlurFXFilter](#), [Digikam::CharcoalFilter](#), [Digikam::ColorFXFilter](#), [Digikam::DistortionFXFilter](#), [Digikam::EmbossFilter](#), [Digikam::FilmGrainFilter](#), [Digikam::InvertFilter](#), [Digikam::OilPaintFilter](#), [Digikam::RainDropFilter](#), [Digikam::GreycstorationFilter](#), [Digikam::HotPixelFixer](#), [Digikam::HSLFilter](#), [Digikam::IccTransformFilter](#), [Digikam::LocalContrastFilter](#), [Digikam::AntiVignettingFilter](#), [Digikam::LensDistortionFilter](#), [Digikam::LensFunFilter](#), [Digikam::LevelsFilter](#), [Digikam::NRFilter](#), [Digikam::RawProcessingFilter](#), [Digikam::RedEyeCorrectionFilter](#), [Digikam::RefocusFilter](#), [Digikam::SharpenFilter](#), [Digikam::UnsharpMaskFilter](#), [Digikam::ContentAwareFilter](#), [Digikam::FreeRotationFilter](#), [Digikam::ShearFilter](#), and [Digikam::WBFilter](#).

### 9.360.2.5 filterImage()

```
virtual void Digikam::DImgThreadedFilter::filterImage ( ) [protected], [pure virtual]
```

Override in subclass.

Implemented in [Digikam::FilterActionFilter](#), [Digikam::IccTransformFilter](#), [Digikam::RawProcessingFilter](#), and [Digikam::WBFilter](#).

### 9.360.2.6 finished

```
void Digikam::DImgThreadedFilter::finished (
    bool success ) [signal]
```

#### Parameters

<i>success</i>	True if computation finished without interruption on valid data False if the thread was canceled, or no data is available.
----------------	--

### 9.360.2.7 initFilter()

```
void Digikam::DImgThreadedFilter::initFilter ( ) [protected], [virtual]
```

Must be called by your constructor.

### 9.360.2.8 initSlave()

```
void Digikam::DImgThreadedFilter::initSlave (
    DImgThreadedFilter *const master,
    int progressBegin = 0,
    int progressEnd = 100 ) [protected]
```

Note: Computation will be started from [setupFilter\(\)](#).

### 9.360.2.9 modulateProgress()

```
int Digikam::DImgThreadedFilter::modulateProgress (
    int progress ) [protected], [virtual]
```

Called by postProgress if master is not null.

### 9.360.2.10 multithreadedSteps()

```
QList< int > Digikam::DImgThreadedFilter::multithreadedSteps (
    int stop,
    int start = 0 ) const
```

Usually, start and stop are rows or columns from image to process. By default, whole image will be processed and start value is 0. In this case stop will be last row or column to process. Between range [start,stop], this method will divide by equal steps depending of number of CPU cores available. To be sure that all values will be processed, in case of CPU core division give rest, the last step compensate the difference. See Blur filter loop implementation for example to see how to use this method with QtConcurrents API.

### 9.360.2.11 parametersSuccessfullyRead()

```
bool Digikam::DImgThreadedFilter::parametersSuccessfullyRead ( ) const [virtual]
```

When readParameters() has been called, this method will return true if the call was successful, and false if not. If returning false, readParametersError() will give an error message. The default implementation always returns success. You only need to reimplement when a filter is likely to fail in a different environment, e.g. depending on availability of installed files. These methods have an undefined return value if readParameters() was not called previously.

Reimplemented in [Digikam::lccTransformFilter](#).

### 9.360.2.12 run()

```
void Digikam::DImgThreadedFilter::run ( ) [override], [protected], [virtual]
```

Implements [Digikam::DynamicThread](#).

### 9.360.2.13 setFilterVersion()

```
void Digikam::DImgThreadedFilter::setFilterVersion (
    int version )
```

A filter may implement different versions, to preserve image history when the algorithm is changed. Any value set here must be contained in supportedVersions, otherwise this call will be ignored. Default value is 1. (Note: If you intend to *record* a filter action, please look at [FilterAction](#)'s m\_version)

### 9.360.2.14 setSlave()

```
void Digikam::DImgThreadedFilter::setSlave (
    DImgThreadedFilter *const slave ) [protected]
```

At destruction of the slave, call with slave=0.

### 9.360.2.15 setupFilter()

```
void Digikam::DImgThreadedFilter::setupFilter (
    const DImg & orgImage )
```

The original image's data will not be copied.

## 9.360.3 Member Data Documentation

### 9.360.3.1 m\_master

```
DImgThreadedFilter* Digikam::DImgThreadedFilter::m_master = nullptr [protected]
```

Progress info will be routed to this one.

### 9.360.3.2 m\_slave

```
DImgThreadedFilter* Digikam::DImgThreadedFilter::m_slave = nullptr [protected]
```

Any filter might want to use another filter while processing.

## 9.361 Digikam::DImgThreadedFilter::DefaultFilterAction< Filter > Class Template Reference

Convenience class to spare the few repeating lines of code.

Inheritance diagram for Digikam::DImgThreadedFilter::DefaultFilterAction< Filter >:



### Public Member Functions

- **DefaultFilterAction** (bool isReproducible)
- **DefaultFilterAction** ([FilterAction::Category](#) category=[FilterAction::ReproducibleFilter](#))
- void **supportOlderVersionIf** (int [version](#), bool condition)

*Preserve backwards compatibility: If a given condition (some new feature is not used) is true, decrease the version so that older digikam versions can still replay the action.*

### Public Member Functions inherited from [Digikam::FilterAction](#)

- **FilterAction** (const QString &[identifier](#), int [version](#), [Category](#) category=[ReproducibleFilter](#))

- void **addFlag** (Flags flags)
- void **addParameter** (const QString &key, const QVariant &value)
  - Sets parameter, removing all other values for the same key.*
- [Category](#) **category** () const
- void **clearParameters** ()
  - Clear all parameters.*
- QString **description** () const
  - Returns a description / comment for this action.*
- QString **displayName** () const
- Flags **flags** () const
- bool **hasParameter** (const QString &key) const
- bool **hasParameters** () const
  - Access parameters.*
- QString **identifier** () const
  - Returns a technical identifier for the filter used to produce this action.*
- bool **isNull** () const
- bool **operator==** (const [FilterAction](#) &other) const
- QVariant & **parameter** (const QString &key)
- const QVariant **parameter** (const QString &key) const
- template<typename T >
  - T parameter** (const QString &key) const
    - Returns parameter converted from QVariant to given type.*
- template<typename T >
  - T parameter** (const QString &key, const T &defaultValue) const
    - Read parameter with a default value: If there is a parameter for the given key, return it converted from QVariant to the template type.*
- QHash< QString, QVariant > & **parameters** ()
- const QHash< QString, QVariant > & **parameters** () const
- void **removeFlag** (Flags flags)
- void **removeParameters** (const QString &key)
  - Removes all parameters for key.*
- void **setDescription** (const QString &description)
- void **setDisplayName** (const QString &displayName)
- void **setFlags** (Flags flags)
- void **setParameters** (const QHash< QString, QVariant > &params)
  - Replaces parameters.*
- int **version** () const
  - Returns the version (>= 1) of the filter used to produce this action.*

### Additional Inherited Members

### Public Types inherited from [Digikam::FilterAction](#)

- enum [Category](#) {
  - [ReproducibleFilter](#) = 0 , [ComplexFilter](#) = 1 , [DocumentedHistory](#) = 2 , **CategoryFirst** = [ReproducibleFilter](#) ,
  - CategoryLast** = [DocumentedHistory](#) }
- enum [Flag](#) { [ExplicitBranch](#) = 1 << 0 }
- typedef QFlags< [Flag](#) > **Flags**

### Protected Attributes inherited from [Digikam::FilterAction](#)

- Category `m_category` = [ReproducibleFilter](#)
- QString `m_description`
- QString `m_displayableName`
- Flags `m_flags`
- QString `m_identifier`
- QHash< QString, QVariant > `m_params`
- int `m_version` = 0

## 9.362 Digikam::DInfoInterface Class Reference

Inheritance diagram for Digikam::DInfoInterface:



### Public Types

- typedef `QList< int >` **DAAlbumIDs**  
List of *Album ids*.



- typedef QMap< QString, QVariant > **DInfoMap**  
*Map of properties name and value.*
- enum **SetupPage** { **ExifToolPage** = 0 , **ImageQualityPage** }

### Public Member Functions

- **DInfoInterface** (QObject \*const parent)
- virtual QList< [GPSItemContainer](#) \* > **currentGPSItems** () const
- virtual void [deleteImage](#) (const QUrl &url)  
*Manipulate with item.*
- virtual void [openSetupPage](#) (SetupPage page)  
*Open configuration dialog page.*
- virtual QMap< QString, QString > [passShortcutActionsToWidget](#) (QWidget \*const) const  
*Pass extra shortcut actions to widget and return prefixes of shortcuts.*
- Q\_SIGNAL void **signalAlbumItemsRecursiveCompleted** (const QList< QUrl > &imageList)
- Q\_SIGNAL void **signalSetupChanged** ()
- Q\_SIGNAL void **signalShortcutPressed** (const QString &shortcut, int val)
- virtual Q\_SLOT void [slotDateTimeForUrl](#) (const QUrl &url, const QDateTime &dt, bool updModDate)  
*Slot to call when date time stamp from item is changed.*
- virtual Q\_SLOT void [slotMetadataChangedForUrl](#) (const QUrl &url)  
*Slot to call when something in metadata from item is changed.*
- virtual QAbstractItemModel \* [tagFilterModel](#) ()  
*Return an instance of tag filter model if host application support this feature, else null pointer.*
  
- virtual QList< QUrl > [currentSelectedItems](#) () const  
*Low level items and albums methods.*
- virtual QList< QUrl > **currentAlbumItems** () const
- virtual QUrl **currentActiveItem** () const
- virtual void **parseAlbumItemsRecursive** ()
- virtual QList< QUrl > **albumItems** (int) const
- virtual QList< QUrl > **albumsItems** (const [DAlbumIDs](#) &) const
- virtual QList< QUrl > **allAlbumItems** () const
- virtual [DInfoMap](#) **albumInfo** (int) const
- virtual void **setAlbumInfo** (int, const [DInfoMap](#) &) const
- virtual [DInfoMap](#) **itemInfo** (const QUrl &) const
- virtual void **setItemInfo** (const QUrl &, const [DInfoMap](#) &)
- Q\_SIGNAL void **signalLastItemUrl** (const QUrl &)
  
- virtual QWidget \* [albumChooser](#) (QWidget \*const parent) const  
*Albums chooser view methods (to use items from albums before to process).*
- virtual [DAlbumIDs](#) **albumChooserItems** () const
- virtual bool **supportAlbums** () const
- Q\_SIGNAL void **signalAlbumChooserSelectionChanged** ()
  
- virtual QWidget \* [uploadWidget](#) (QWidget \*const parent) const  
*Album selector view methods (to upload items from an external place).*
- virtual QUrl **uploadUrl** () const
- Q\_SIGNAL void **signalUploadUrlChanged** ()
- virtual QUrl [defaultUploadUrl](#) () const  
*Url to upload new items without to use album selector.*
- Q\_SIGNAL void **signalImportedImage** (const QUrl &)

## Public Attributes

- bool **forceAlbumSelection** = false

## 9.362.1 Member Function Documentation

### 9.362.1.1 albumChooser()

```
QWidget * Digikam::DInfoInterface::albumChooser (
    QWidget *const parent ) const [virtual]
```

Reimplemented in [Digikam::DBInfofiface](#).

### 9.362.1.2 currentSelectedItems()

```
QList< QUrl > Digikam::DInfoInterface::currentSelectedItems ( ) const [virtual]
```

Reimplemented in [Digikam::DBInfofiface](#), and [Digikam::DMetaInfofiface](#).

### 9.362.1.3 defaultUploadUrl()

```
QUrl Digikam::DInfoInterface::defaultUploadUrl ( ) const [virtual]
```

Reimplemented in [Digikam::DBInfofiface](#), and [Digikam::DMetaInfofiface](#).

### 9.362.1.4 deleteImage()

```
void Digikam::DInfoInterface::deleteImage (
    const QUrl & url ) [virtual]
```

Reimplemented in [Digikam::DBInfofiface](#), and [Digikam::DMetaInfofiface](#).

### 9.362.1.5 openSetupPage()

```
void Digikam::DInfoInterface::openSetupPage (
    SetupPage page ) [virtual]
```

Reimplemented in [Digikam::DBInfofiface](#), and [ShowFoto::ShowfotoInfofiface](#).

### 9.362.1.6 passShortcutActionsToWidget()

```
QMap< QString, QString > Digikam::DInfoInterface::passShortcutActionsToWidget (
    QWidget * const ) const [virtual]
```

Reimplemented in [Digikam::DBInfofiface](#).

### 9.362.1.7 slotDateTimeForUrl()

```
void Digikam::DInfoInterface::slotDateTimeForUrl (
    const QUrl & url,
    const QDateTime & dt,
    bool updModDate ) [virtual]
```

Reimplemented in [Digikam::DMetalInfoface](#).

### 9.362.1.8 slotMetadataChangedForUrl()

```
void Digikam::DInfoInterface::slotMetadataChangedForUrl (
    const QUrl & url ) [virtual]
```

Reimplemented in [Digikam::DMetalInfoface](#).

### 9.362.1.9 tagFilterModel()

```
QAbstractItemModel * Digikam::DInfoInterface::tagFilterModel ( ) [virtual]
```

Reimplemented in [Digikam::DBInfoface](#).

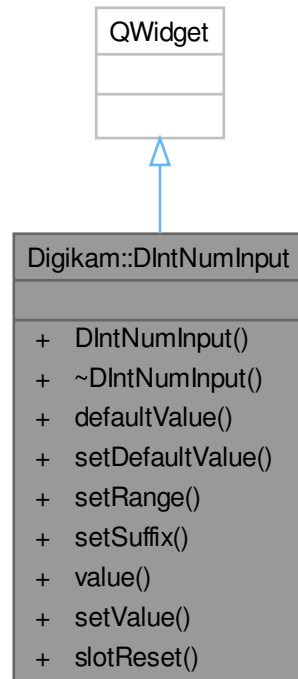
### 9.362.1.10 uploadWidget()

```
QWidget * Digikam::DInfoInterface::uploadWidget (
    QWidget *const parent ) const [virtual]
```

Reimplemented in [Digikam::DBInfoface](#), and [Digikam::DMetalInfoface](#).

## 9.363 Digikam::DIntNumInput Class Reference

Inheritance diagram for Digikam::DIntNumInput:



### Public Slots

- void **setValue** (int d)
- void **slotReset** ()

### Signals

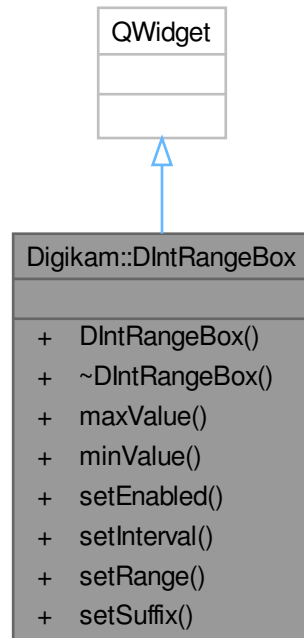
- void **reset** ()
- void **valueChanged** (int)

### Public Member Functions

- **DIntNumInput** (QWidget \*const parent=nullptr)
- int **defaultValue** () const
- void **setDefaultValue** (int d)
- void **setRange** (int min, int max, int step)
- void **setSuffix** (const QString &suffix)
- int **value** () const

## 9.364 Digikam::DIntRangeBox Class Reference

Inheritance diagram for Digikam::DIntRangeBox:



### Signals

- void **maxChanged** (int)
- void **minChanged** (int)

### Public Member Functions

- **DIntRangeBox** (QWidget \*const parent=nullptr)
- int **maxValue** ()  
*This method returns the maximum value of the interval.*
- int **minValue** ()  
*This method returns the minimum value of the interval.*
- void **setEnabled** (bool enabled)  
*This method enables or disables the embedded spinboxes.*
- void **setInterval** (int min, int max)  
*This method sets the minimum and maximum of the interval.*
- void **setRange** (int min, int max)  
*This method sets the lower and upper threshold of possible interval minimum and maximum values.*
- void **setSuffix** (const QString &suffix)  
*This method sets the suffix for the minimum and maximum value boxes.*

## 9.364.1 Member Function Documentation

### 9.364.1.1 `maxValue()`

```
int Digikam::DIntRangeBox::maxValue ( )
```

#### Returns

the maximum value.

### 9.364.1.2 `minValue()`

```
int Digikam::DIntRangeBox::minValue ( )
```

#### Returns

the minimum value.

### 9.364.1.3 `setEnabled()`

```
void Digikam::DIntRangeBox::setEnabled (
    bool enabled )
```

#### Parameters

<i>enabled</i>	If the interval boxes should be enabled.
----------------	--

### 9.364.1.4 `setInterval()`

```
void Digikam::DIntRangeBox::setInterval (
    int min,
    int max )
```

#### Parameters

<i>min</i>	The minimum value of the interval.
<i>max</i>	The maximum value of the interval.

### 9.364.1.5 `setRange()`

```
void Digikam::DIntRangeBox::setRange (
    int min,
    int max )
```

## Parameters

<i>min</i>	the lowest value to which the interval can be expanded.
<i>max</i>	the highest value to which the interval can be expanded.

## 9.364.1.6 setSuffix()

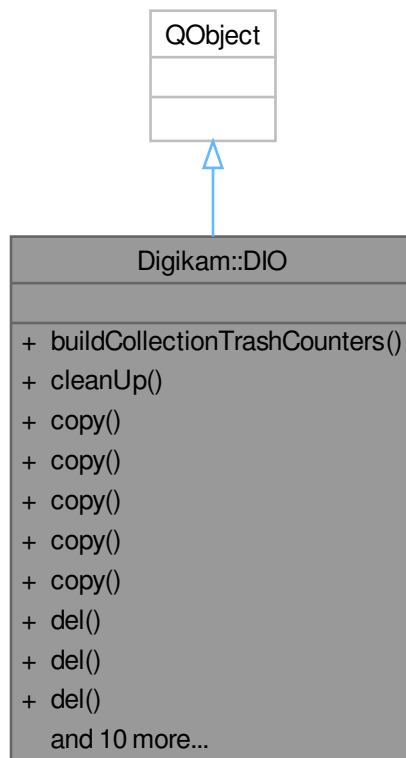
```
void Digikam::DIntRangeBox::setSuffix (
    const QString & suffix )
```

## Parameters

<i>suffix</i>	The suffix.
---------------	-------------

## 9.365 Digikam::DIO Class Reference

Inheritance diagram for Digikam::DIO:



## Signals

- void **signalRenameFailed** (const QUrl &url)
- void **signalRenameFinished** ()
- void **signalTrashCounters** ()
- void **signalTrashFinished** ()

## Static Public Member Functions

- static void **buildCollectionTrashCounters** ()
- static void **cleanUp** ()
- static void **copy** (const QList< [ItemInfo](#) > &infos, const QUrl &dest)  
*Copy items to external folder.*
- static void **copy** (const QList< [ItemInfo](#) > &infos, [PAlbum](#) \*const dest)  
*Copy items to another album.*
- static void **copy** (const QList< QUrl > &srcList, [PAlbum](#) \*const dest)  
*Copy external files to another album.*
- static void **copy** (const QUrl &src, [PAlbum](#) \*const dest)  
*Copy an external file to another album.*
- static void **copy** ([PAlbum](#) \*const src, [PAlbum](#) \*const dest)  
*All DIO methods will take care for sidecar files, if they exist.*
- static void **del** (const [ItemInfo](#) &info, bool useTrash)
- static void **del** (const QList< [ItemInfo](#) > &infos, bool useTrash)
- static void **del** ([PAlbum](#) \*const album, bool useTrash)
- static void **emptyTrash** (const DTrashItemInfoList &infos)
- static int **getTrashCounter** (const QString &albumRootPath)  
*Trash operations.*
- static [DIO](#) \* **instance** ()
- static bool **itemsUnderProcessing** ()
- static void **move** (const QList< [ItemInfo](#) > &infos, [PAlbum](#) \*const dest)  
*Move items to another album.*
- static void **move** (const QList< QUrl > &srcList, [PAlbum](#) \*const dest)  
*Move external files into another album.*
- static void **move** (const QUrl &src, [PAlbum](#) \*const dest)  
*Move external files another album.*
- static void **move** ([PAlbum](#) \*const src, [PAlbum](#) \*const dest)  
*Move an album into another album.*
- static void **rename** (const QUrl &src, const QString &newName, bool overwrite=false)  
*Rename item to new name.*
- static void **restoreTrash** (const DTrashItemInfoList &infos)

## Friends

- class [DIOCreator](#)

## 9.365.1 Member Function Documentation

### 9.365.1.1 copy()

```
void Digikam::DIO::copy (
    PAlbum *const src,
    PAlbum *const dest ) [static]
```

Copy an album to another album



## 9.366 Digikam::DirectoryNameOption Class Reference

Inheritance diagram for Digikam::DirectoryNameOption:



### Protected Member Functions

- QString [parseOperation](#) ([ParseSettings](#) &settings, const QRegularExpressionMatch &match) override  
*TODO: describe me.*

## Protected Member Functions inherited from [Digikam::Rule](#)

- bool [addToken](#) (const QString &id, const QString &description, const QString &actionName=QString())  
*add a token to the parser, every parser should at least assign one token object*
- void [setDescription](#) (const QString &desc)
- void [setIcon](#) (const QString &pixmap)
- void [setRegExp](#) (const QRegularExpression &regExp)
- void [setUseTokenMenu](#) (bool value)  
*If multiple tokens have been assigned to a rule, a menu will be created.*

## Additional Inherited Members

## Public Types inherited from [Digikam::Rule](#)

- enum [IconType](#) { [Action](#) = 0 , [Dialog](#) }

## Signals inherited from [Digikam::Rule](#)

- void [signalTokenTriggered](#) (const QString &)

## Public Member Functions inherited from [Digikam::Option](#)

- [Option](#) (const QString &name, const QString &description)
- [Option](#) (const QString &name, const QString &description, const QString &icon)

## Public Member Functions inherited from [Digikam::Rule](#)

- [Rule](#) (const QString &name)
- [Rule](#) (const QString &name, const QString &icon)
- QString [description](#) () const
- QPixmap [icon](#) (Rule::IconType type=Rule::Action) const
- bool [isValid](#) () const  
*Checks the validity of the parse object.*
- [ParseResults](#) [parse](#) ([ParseSettings](#) &settings)
- QRegularExpression & [regExp](#) () const  
*TODO: This is probably not needed anymore.*
- QPushButton \* [registerButton](#) (QWidget \*parent)  
*Register a button in the parent object.*
- QAction \* [registerMenu](#) (QMenu \*parent)  
*Register a menu action in the parent object.*
- virtual void [reset](#) ()  
*Resets the parser to its initial state.*
- TokenList & [tokens](#) () const
- bool [useTokenMenu](#) () const  
*Returns true if a token menu is used.*

## Static Public Member Functions inherited from [Digikam::Rule](#)

- static QString [escapeToken](#) (const QString &token)  
*Escape the token characters to make them work in regular expressions.*

## Protected Slots inherited from [Digikam::Rule](#)

- virtual void [slotTokenTriggered](#) (const QString &)

## 9.366.1 Member Function Documentation

### 9.366.1.1 [parseOperation\(\)](#)

```
QString Digikam::DirectoryNameOption::parseOperation (
    ParseSettings & settings,
    const QRegularExpressionMatch & match ) [override], [protected], [virtual]
```

#### Parameters

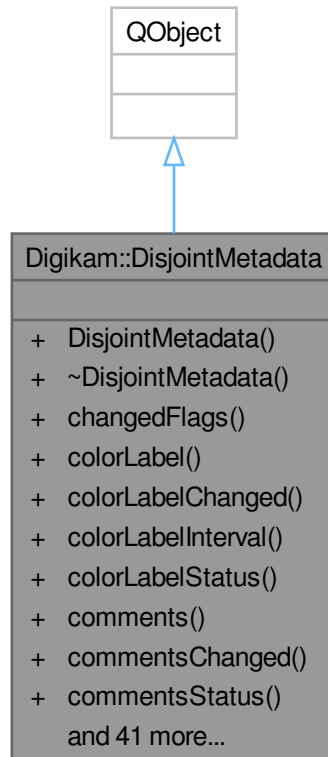
<i>settings</i>	contains settings
<i>match</i>	result of the regular expression match done in <a href="#">Option::parse()</a>

#### Returns

Implements [Digikam::Option](#).

## 9.367 Digikam::DisjointMetadata Class Reference

Inheritance diagram for Digikam::DisjointMetadata:



### Public Types

- enum `WriteMode` { `FullWrite` , `FullWriteIfChanged` , `PartialWrite` }

### Public Member Functions

- int `changedFlags` ()  
*changedFlags - used for selective metadata write.*
- int `colorLabel` () const  
*Returns the Color Label id (see ColorLabel values in globals.h).*
- bool `colorLabelChanged` () const
- void `colorLabelInterval` (int &lowest, int &highest) const  
*Returns the lowest and highest Color Label id (see ColorLabel values from globals.h).*
- `DisjointMetadataDataFields::Status` `colorLabelStatus` () const
- `CaptionsMap` `comments` () const  
*Returns a map all alternate language omments .*
- bool `commentsChanged` () const

- [DisjointMetadataDataFields::Status](#) **commentsStatus** () const
- [DisjointMetadataDataFields](#) **dataFields** () const
- QDateTime [dateTime](#) () const  
*Returns the dateTime.*
- bool **dateTimeChanged** () const  
*Returns if the metadata field has been changed with the corresponding setter method.*
- void [dateTimeInterval](#) (QDateTime &lowest, QDateTime &highest) const  
*Returns the earliest and latest date.*
- [DisjointMetadataDataFields::Status](#) **dateTimeStatus** () const  
*Returns the metadata field Status.*
- QStringList [keywords](#) () const  
*Returns a QStringList with all tags with status [DisjointMetadataDataFields::MetadataAvailable](#).*
- void **load** (const [ItemInfo](#) &info)
- [Template](#) [metadataTemplate](#) () const  
*Returns the metadata template.*
- int [pickLabel](#) () const  
*Returns the Pick Label id (see PickLabel values in globals.h).*
- bool **pickLabelChanged** () const
- void [pickLabelInterval](#) (int &lowest, int &highest) const  
*Returns the lowest and highest Pick Label id (see PickLabel values from globals.h).*
- [DisjointMetadataDataFields::Status](#) **pickLabelStatus** () const
- int [rating](#) () const  
*Returns the rating.*
- bool **ratingChanged** () const
- void [ratingInterval](#) (int &lowest, int &highest) const  
*Returns the lowest and highest rating.*
- [DisjointMetadataDataFields::Status](#) **ratingStatus** () const
- void [replaceColorLabel](#) (int colorId)  
*Special case if the metadata of color, pick or rating has already been changed outside.*
- void **replacePickLabel** (int pickId)
- void **replaceRating** (int [rating](#))
- void **reset** ()
- void **resetChanged** ()
- void **setColorLabel** (int colorId, [DisjointMetadataDataFields::Status](#) status=[DisjointMetadataDataFields::MetadataAvailable](#))
- void **setComments** (const [CaptionsMap](#) &comments, [DisjointMetadataDataFields::Status](#) status=[DisjointMetadataDataFields::MetadataAvailable](#))
- void **setDataFields** (const [DisjointMetadataDataFields](#) &data)
- void **setDateTime** (const QDateTime &dateTime, [DisjointMetadataDataFields::Status](#) status=[DisjointMetadataDataFields::MetadataAvailable](#))  
*Set metadata field to the given value, and the metadata field status to the corresponding [DisjointMetadataDataFields::MetadataAvailable](#).*
- void **setMetadataTemplate** (const [Template](#) &t, [DisjointMetadataDataFields::Status](#) status=[DisjointMetadataDataFields::MetadataAvailable](#))
- void **setPickLabel** (int pickId, [DisjointMetadataDataFields::Status](#) status=[DisjointMetadataDataFields::MetadataAvailable](#))
- void **setRating** (int [rating](#), [DisjointMetadataDataFields::Status](#) status=[DisjointMetadataDataFields::MetadataAvailable](#))
- void **setTag** (int albumId, [DisjointMetadataDataFields::Status](#) status=[DisjointMetadataDataFields::MetadataAvailable](#))
- void **setTitles** (const [CaptionsMap](#) &titles, [DisjointMetadataDataFields::Status](#) status=[DisjointMetadataDataFields::MetadataAvailable](#))
- QMap< int, [DisjointMetadataDataFields::Status](#) > [tags](#) () const  
*Returns a map with the status for each tag.*
- bool **tagsChanged** () const
- [DisjointMetadataDataFields::Status](#) **tagStatus** (const QString &tagPath) const
- [DisjointMetadataDataFields::Status](#) **tagStatus** (int albumId) const
- bool **templateChanged** () const
- [DisjointMetadataDataFields::Status](#) **templateStatus** () const
- [CaptionsMap](#) [titles](#) () const  
*Returns a map all alternate language titles.*

- bool **titlesChanged** () const
- [DisjointMetadataDataFields::Status](#) **titlesStatus** () const
- bool **willWriteMetadata** ([WriteMode](#) writeMode, const [MetaEngineSettingsContainer](#) &settings=[MetaEngineSettings::instance](#)()->settings()) const

*With the currently applied changes, the given writeMode and settings, returns if write(DMetadata), write(QString) or write(DImg) will actually apply any changes.*

- bool **write** ([ItemInfo](#) info, [WriteMode](#) writeMode=[FullWrite](#))

*Applies the set of metadata contained in this [MetadataHub](#) to the given [ItemInfo](#) object.*

## 9.367.1 Member Enumeration Documentation

### 9.367.1.1 WriteMode

```
enum Digikam::DisjointMetadata::WriteMode
```

Enumerator

FullWrite	Write all available information.
FullWriteIfChanged	Do a full write if and only if. <ul style="list-style-type: none"> <li>• metadata fields changed</li> <li>• the changed fields shall be written according to write settings "Changed" in this context means changed by one of the set... methods, the load() methods are ignored for this attribute. This mode allows to avoid write operations when e.g. the user does not want keywords to be written and only changes keywords.</li> </ul>
PartialWrite	Write only the changed parts. Metadata fields which cannot be changed from <a href="#">MetadataHub</a> (photographer ID etc.) will never be written

## 9.367.2 Member Function Documentation

### 9.367.2.1 changedFlags()

```
int Digikam::DisjointMetadata::changedFlags ( )
```

The result will be passed to metadatahub and it will

- write it to disk

**Returns**

- metadatahub flags encoded as int

### 9.367.2.2 colorLabel()

```
int Digikam::DisjointMetadata::colorLabel ( ) const
```

If status is [DisjointMetadataDataFields::MetadataDisjoint](#), the None Label is returned. (see [colorLabelInterval\(\)](#)) If status is [DisjointMetadataDataFields::MetadataInvalid](#), -1 is returned.

### 9.367.2.3 colorLabelInterval()

```
void Digikam::DisjointMetadata::colorLabelInterval (
    int & lowest,
    int & highest ) const
```

If status is [DisjointMetadataDataFields::MetadataAvailable](#), the values are the same. If status is [DisjointMetadataDataFields::MetadataInvalid](#), -1 is returned.

### 9.367.2.4 comments()

```
CaptionsMap Digikam::DisjointMetadata::comments ( ) const
```

If status is [DisjointMetadataDataFields::MetadataDisjoint](#), the first loaded map is returned. If status is [DisjointMetadataDataFields::MetadataInvalid](#), `CaptionMap()` is returned.

### 9.367.2.5 dateTime()

```
QDateTime Digikam::DisjointMetadata::dateTime ( ) const
```

If status is [DisjointMetadataDataFields::MetadataDisjoint](#), the earliest date is returned. (see [dateTimeInterval\(\)](#)) If status is [DisjointMetadataDataFields::MetadataInvalid](#), an invalid date is returned.

### 9.367.2.6 dateTimeInterval()

```
void Digikam::DisjointMetadata::dateTimeInterval (
    QDateTime & lowest,
    QDateTime & highest ) const
```

If status is [DisjointMetadataDataFields::MetadataAvailable](#), the values are the same. If status is [DisjointMetadataDataFields::MetadataInvalid](#), invalid dates are returned.

### 9.367.2.7 keywords()

```
QStringList Digikam::DisjointMetadata::keywords ( ) const
```

(i.e., the intersection of tags from all loaded metadata sets)

### 9.367.2.8 metadataTemplate()

```
Template Digikam::DisjointMetadata::metadataTemplate ( ) const
```

If status is [DisjointMetadataDataFields::MetadataDisjoint](#), the first loaded template is returned. If status is [DisjointMetadataDataFields::MetadataInvalid](#), 0 is returned.

**9.367.2.9 pickLabel()**

```
int Digikam::DisjointMetadata::pickLabel ( ) const
```

If status is [DisjointMetadataDataFields::MetadataDisjoint](#), the None Label is returned. (see [pickLabelInterval\(\)](#)) If status is [DisjointMetadataDataFields::MetadataInvalid](#), -1 is returned.

**9.367.2.10 pickLabelInterval()**

```
void Digikam::DisjointMetadata::pickLabelInterval (
    int & lowest,
    int & highest ) const
```

If status is [DisjointMetadataDataFields::MetadataAvailable](#), the values are the same. If status is [DisjointMetadataDataFields::MetadataInvalid](#), -1 is returned.

**9.367.2.11 rating()**

```
int Digikam::DisjointMetadata::rating ( ) const
```

If status is [DisjointMetadataDataFields::MetadataDisjoint](#), the lowest rating is returned. (see [ratingInterval\(\)](#)) If status is [DisjointMetadataDataFields::MetadataInvalid](#), -1 is returned.

**9.367.2.12 ratingInterval()**

```
void Digikam::DisjointMetadata::ratingInterval (
    int & lowest,
    int & highest ) const
```

If status is [DisjointMetadataDataFields::MetadataAvailable](#), the values are the same. If status is [DisjointMetadataDataFields::MetadataInvalid](#), -1 is returned.

**9.367.2.13 replaceColorLabel()**

```
void Digikam::DisjointMetadata::replaceColorLabel (
    int colorId )
```

Replace with current values as if there is no change.

**9.367.2.14 tags()**

```
QMap< int, DisjointMetadataDataFields::Status > Digikam::DisjointMetadata::tags ( ) const
```

Any tag that was set on one of the loaded images is contained in the map. (If a tag is not contained in the map, it was not set on any of the loaded images) If the tag was set on all loaded images, the status is [DisjointMetadataDataFields::MetadataAvailable](#). If the tag was set on at least one, but not all of the loaded images, the status is [DisjointMetadataDataFields::MetadataDisjoint](#).



### 9.367.2.15 titles()

```
CaptionsMap Digikam::DisjointMetadata::titles ( ) const
```

If status is [DisjointMetadataDataFields::MetadataDisjoint](#), the first loaded map is returned. If status is [DisjointMetadataDataFields::MetadataInvalid](#), `CaptionMap()` is returned.

### 9.367.2.16 write()

```
bool Digikam::DisjointMetadata::write (
    ItemInfo info,
    WriteMode writeMode = FullWrite )
```

#### Returns

Returns true if the info object has been changed

## 9.368 Digikam::DisjointMetadataDataFields Class Reference

This class was split from `DisjointMetadata::Private` to allow to use the automatic C++ copy constructor (`DisjointMetadata::Private` contains a `QMutex` and is thus non-copyable)

### Public Types

- enum [Status](#) { [MetadataInvalid](#) , [MetadataAvailable](#) , [MetadataDisjoint](#) }
- The status enum describes the result of joining several metadata sets.*

### Public Attributes

- int **colorLabel** = -1
- bool **colorLabelChanged** = false
- [Status](#) **colorLabelStatus** = [MetadataInvalid](#)
- [CaptionsMap](#) **comments**
- bool **commentsChanged** = false
- [Status](#) **commentsStatus** = [MetadataInvalid](#)
- int **count** = 0
- [QDateTime](#) **dateTime**
- bool **dateTimeChanged** = false
- [Status](#) **dateTimeStatus** = [MetadataInvalid](#)
- int **highestColorLabel** = -1
- int **highestPickLabel** = -1
- int **highestRating** = -1
- bool **invalid** = false
- [QDateTime](#) **lastDateTime**
- [Template](#) **metadataTemplate**
- int **pickLabel** = -1
- bool **pickLabelChanged** = false
- [Status](#) **pickLabelStatus** = [MetadataInvalid](#)
- int **rating** = -1

- bool **ratingChanged** = false
- **Status ratingStatus** = [MetadataInvalid](#)
- QList< int > **tagIds**
- QStringList **tagList**
- QMap< int, **Status** > **tags**
- bool **tagsChanged** = false
- bool **templateChanged** = false
- **Status templateStatus** = [MetadataInvalid](#)
- CaptionsMap **titles**
- bool **titlesChanged** = false
- **Status titlesStatus** = [MetadataInvalid](#)
- bool **withoutTags** = false

## 9.368.1 Member Enumeration Documentation

### 9.368.1.1 Status

```
enum Digikam::DisjointMetadataDataFields::Status
```

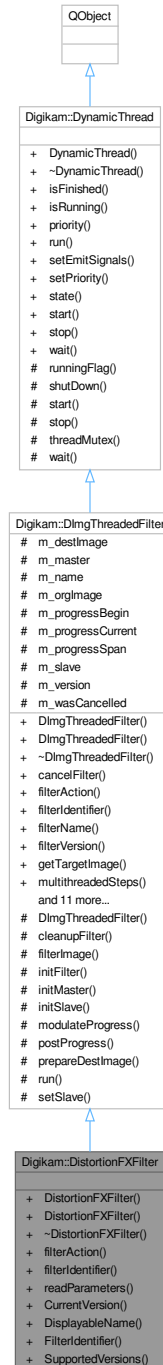
If only one set has been added, the status is always `MetadataAvailable`. If no set has been added, the status is always `MetadataInvalid`

#### Enumerator

<code>MetadataInvalid</code>	Not yet filled with any value.
<code>MetadataAvailable</code>	Only one data set has been added, or a common value is available.
<code>MetadataDisjoint</code>	No common value is available. For rating and dates, the interval is available.

## 9.369 Digikam::DistortionFXFilter Class Reference

Inheritance diagram for Digikam::DistortionFXFilter:



### Public Types

- enum **DistortionFXTypes** {  
**FishEye = 0** , **Twirl** , **CilindricalHor** , **CilindricalVert** ,

**CylindricalHV** , **Caricature** , **MultipleCorners** , **WavesHorizontal** , **WavesVertical** , **BlockWaves1** , **BlockWaves2** , **CircularWaves1** , **CircularWaves2** , **PolarCoordinates** , **UnpolarCoordinates** , **Tile** }

## Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

## Public Member Functions

- **DistortionFXFilter** ([DImg](#) \*const orgImage, [QObject](#) \*const parent=nullptr, int effectType=0, int level=0, int iteration=0, bool antialiasing=true)
- **DistortionFXFilter** ([QObject](#) \*const parent=nullptr)
- [FilterAction](#) filterAction () override
 

*Returns the action description corresponding to currently set options.*
- [QString](#) filterIdentifier () const override
 

*Return the identifier for this filter in the image history.*
- void [readParameters](#) (const [FilterAction](#) &action) override

## Public Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImg](#) \*const orgImage, [QObject](#) \*const parent, const [QString](#) &name=[QString](#)())
 

*Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter](#) ([QObject](#) \*const parent=nullptr, const [QString](#) &name=[QString](#)())
 

*Constructs a filter without argument.*
- virtual void [cancelFilter](#) ()
 

*Cancel the threaded computation.*
- const [QString](#) & [filterName](#) ()
- int [filterVersion](#) () const
- [DImg](#) [getTargetImage](#) ()
- [QList](#)< int > [multithreadedSteps](#) (int stop, int start=0) const
 

*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool [parametersSuccessfullyRead](#) () const
 

*Optional: error handling for readParameters.*
- virtual [QString](#) [readParametersError](#) (const [FilterAction](#) &actionThatFailed) const
- void [setFilterName](#) (const [QString](#) &name)
- void [setFilterVersion](#) (int version)
 

*Replaying a filter action: Set the filter version.*
- void [setOriginalImage](#) (const [DImg](#) &orgImage)
- void [setupAndStartDirectly](#) (const [DImg](#) &orgImage, [DImgThreadedFilter](#) \*const master, int progress←Begin=0, int progressEnd=100)
 

*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void [setupFilter](#) (const [DImg](#) &orgImage)
 

*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void [startFilter](#) ()
 

*Start the threaded computation.*
- virtual void [startFilterDirectly](#) ()
 

*Start computation of this filter, directly in this thread.*
- virtual [QList](#)< int > [supportedVersions](#) () const

## Public Member Functions inherited from Digikam::DynamicThread

- [DynamicThread](#) (QObject \*const parent=nullptr)
 

*This class extends QRunnable, so you have to reimplement virtual void [run\(\)](#).*
- [~DynamicThread](#) () override
 

*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool [isFinished](#) () const
- bool [isRunning](#) () const
- QThread::Priority [priority](#) () const
- void [setEmitSignals](#) (bool emitThem)
- void [setPriority](#) (QThread::Priority priority)
 

*Sets the priority for this dynamic thread.*
- State [state](#) () const

## Static Public Member Functions

- static int [CurrentVersion](#) ()
- static QString [DisplayableName](#) ()
- static QString [FilterIdentifier](#) ()
- static QList< int > [SupportedVersions](#) ()

## Additional Inherited Members

## Public Slots inherited from Digikam::DynamicThread

- void [start](#) ()
- void [stop](#) ()
 

*Stop computation, sets the running flag to false.*
- void [wait](#) ()
 

*Waits until the thread finishes.*

## Signals inherited from Digikam::DImgThreadedFilter

- void [finished](#) (bool success)
 

*Emitted when the computation has completed.*
- void [progress](#) (int progress)
 

*Emitted when progress info from the calculation is available.*
- void [started](#) ()
 

*This signal is emitted when image data is available and the computation has started.*

## Signals inherited from Digikam::DynamicThread

- void [finished](#) ()
- void [starting](#) ()
 

*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from [Digikam::DImgThreadedFilter](#)

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.369.1 Member Function Documentation

### 9.369.1.1 filterAction()

`FilterAction` Digikam::DistortionFXFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.369.1.2 filterIdentifier()

`QString` Digikam::DistortionFXFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

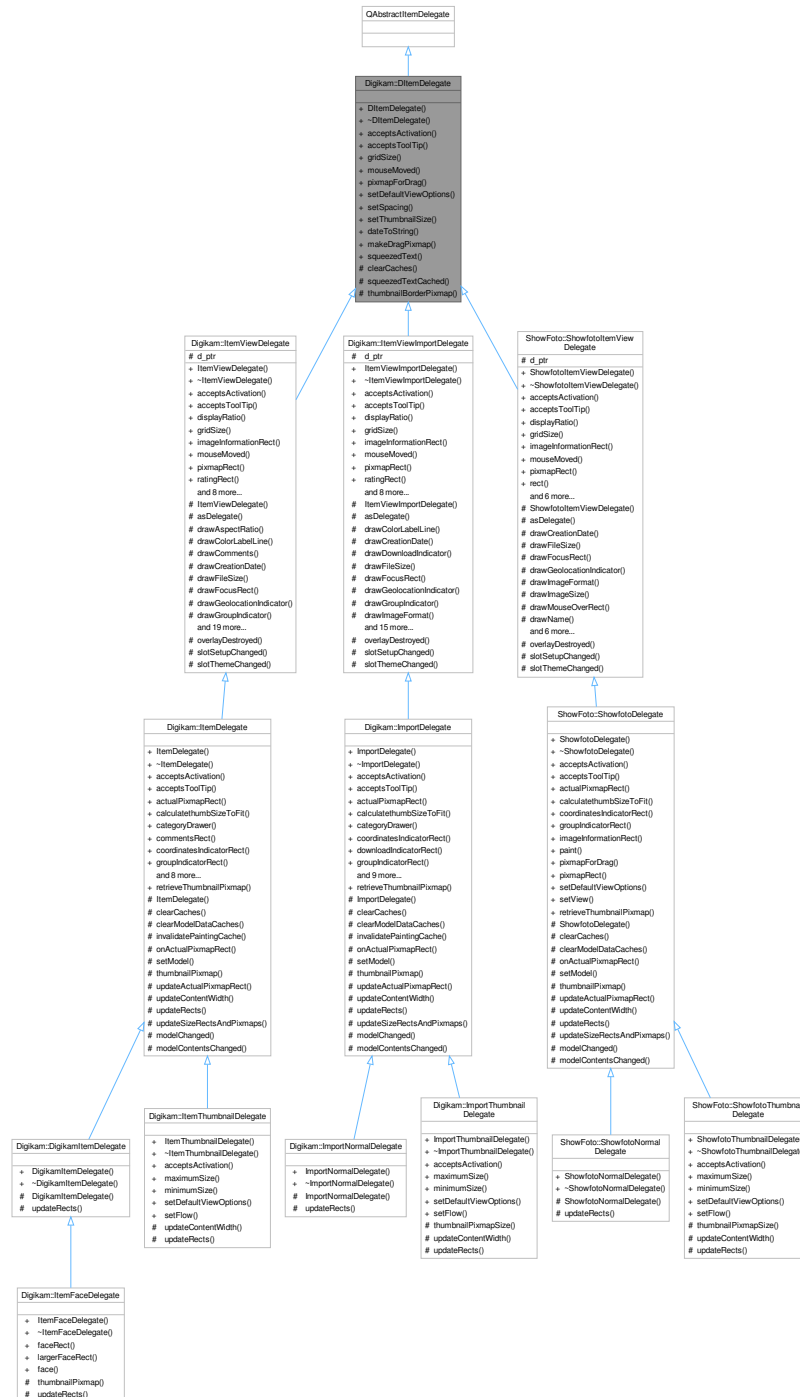
### 9.369.1.3 readParameters()

```
void Digikam::DistortionFXFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

## 9.370 Digikam::DItemDelegate Class Reference

Inheritance diagram for Digikam::DItemDelegate:



### Signals

- void **gridSizeChanged** (const QSize &newSize)
- void **visualChange** ()



## Public Member Functions

- **DItemDelegate** (QObject \*const parent=nullptr)
- virtual bool **acceptsActivation** (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*activationRect=nullptr) const =0
- virtual bool **acceptsToolTip** (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*tooltipRect=nullptr) const =0
 

*These methods take four parameters: The position on viewport, the rect on viewport, the index, and optionally a parameter into which, if the return value is true, a rectangle can be written for which the return value will be true as well.*
- virtual QSize **gridSize** () const =0
 

*Returns the gridsize to be set by the view.*
- virtual void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)=0
- virtual QPixmap **pixmapForDrag** (const QStyleOptionViewItem &option, const QList< QModelIndex > &indexes) const =0
- virtual void **setDefaultViewOptions** (const QStyleOptionViewItem &option)=0
 

*Style option with standard values to use for cached rendering.*
- virtual void **setSpacing** (int spacing)=0
- virtual void **setThumbnailSize** (const ThumbnailSize &thumbSize)=0
 

*You must set these options from the view.*

## Static Public Member Functions

- static QString **dateToString** (const QDateTime &datetime)
- static QPixmap **makeDragPixmap** (const QStyleOptionViewItem &option, const QList< QModelIndex > &indexes, double displayRatio, const QPixmap &suggestedPixmap=QPixmap())
- static QString **squeezedText** (const QFontMetrics &fm, int width, const QString &text)

## Protected Member Functions

- virtual void **clearCaches** ()
- QString **squeezedTextCached** (QPainter \*const p, int width, const QString &text) const
- QPixmap **thumbnailBorderPixmap** (const QSize &pixSize, bool isGrouped=false) const

## 9.370.1 Member Function Documentation

### 9.370.1.1 acceptsToolTip()

```
virtual bool Digikam::DItemDelegate::acceptsToolTip (
    const QPoint & pos,
    const QRect & visualRect,
    const QModelIndex & index,
    QRect * tooltipRect = nullptr ) const [pure virtual]
```

Implemented in [Digikam::ItemDelegate](#), [Digikam::ItemViewDelegate](#), [ShowFoto::ShowfotoDelegate](#), [ShowFoto::ShowfotoItemViewDelegate](#), [Digikam::ImportDelegate](#), and [Digikam::ItemViewImportDelegate](#).

### 9.370.1.2 gridSize()

```
virtual QSize Digikam::DItemDelegate::gridSize ( ) const [pure virtual]
```

It's sizeHint plus spacing.

Implemented in [Digikam::ItemViewDelegate](#), [ShowFoto::ShowfotoItemViewDelegate](#), and [Digikam::ItemViewImportDelegate](#).

### 9.370.1.3 mouseMoved()

```
virtual void Digikam::DItemDelegate::mouseMoved (
    QMouseEvent * e,
    const QRect & visualRect,
    const QModelIndex & index ) [pure virtual]
```

#### Note

to be called by [ItemViewCategorized](#) only

Implemented in [Digikam::ItemViewDelegate](#), [ShowFoto::ShowfotoItemViewDelegate](#), and [Digikam::ItemViewImportDelegate](#).

### 9.370.1.4 setDefaultViewOptions()

```
virtual void Digikam::DItemDelegate::setDefaultViewOptions (
    const QStyleOptionViewItem & option ) [pure virtual]
```

option.rect shall be the viewport rectangle. Call on resize, font change.

Implemented in [Digikam::ItemDelegate](#), [Digikam::ItemThumbnailDelegate](#), [Digikam::ItemViewDelegate](#), [ShowFoto::ShowfotoDelegate](#), [ShowFoto::ShowfotoThumbnailDelegate](#), [ShowFoto::ShowfotoItemViewDelegate](#), [Digikam::ImportDelegate](#), [Digikam::ImportThumbnailDelegate](#), and [Digikam::ItemViewImportDelegate](#).

### 9.370.1.5 setThumbnailSize()

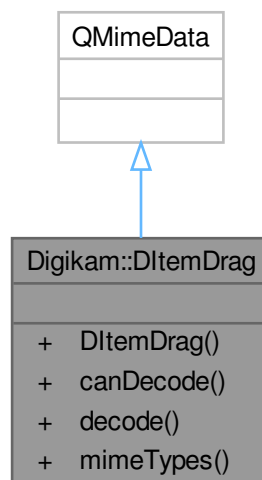
```
virtual void Digikam::DItemDelegate::setThumbnailSize (
    const ThumbnailSize & thumbSize ) [pure virtual]
```

Implemented in [Digikam::ItemViewDelegate](#), [ShowFoto::ShowfotoItemViewDelegate](#), and [Digikam::ItemViewImportDelegate](#).

## 9.371 Digikam::DItemDrag Class Reference

Provides a drag object with additional information for internal drag&drop.

Inheritance diagram for Digikam::DItemDrag:



## Public Member Functions

- **DItemDrag** (const QList< QUrl > &urls, const QList< int > &albumIDs, const QList< qlonglong > &imageIDs)

## Static Public Member Functions

- static bool **canDecode** (const QMimeData \*e)
- static bool **decode** (const QMimeData \*e, QList< QUrl > &urls, QList< int > &albumIDs, QList< qlonglong > &imageIDs)
- static QStringList **mimeTypes** ()

### 9.371.1 Detailed Description

Images can be moved through ItemDrag. It is possible to move them on another application which is supported through QT to e.g. copy the images. digiKam can use the IDs, if ItemDrag is dropped on digikam itself. The urls set via setUrls() are used for external drops (k3b, gimp, ...)

## 9.372 Digikam::DItemInfo Class Reference

[DItemInfo](#) is a class to get item information from host application (Showfoto or digiKam) The interface is re-implemented in host and depend how item information must be retrieved (from a database or by file metadata).

## Public Member Functions

- **DItemInfo** (const [DInfoInterface::DInfoMap](#) &)
- int **albumId** () const
- double **altitude** () const
- QString **aperture** () const
- [CaptionsMap](#) **captions** () const
- int **colorLabel** () const
- QString **comment** () const
- [MetaEngine::AltLangMap](#) **copyrightNotices** () const
- [MetaEngine::AltLangMap](#) **copyrights** () const
- QStringList **creators** () const
- QString **credit** () const
- QDateTime **dateTime** () const
- QSize **dimensions** () const
- QString **exposureTime** () const
- qlonglong **fileSize** () const
- QString **focalLength** () const
- QString **focalLength35mm** () const
- bool **hasGeolocationInfo** () const
- [DInfoInterface::DInfoMap](#) **infoMap** () const
- QStringList **keywords** () const
- double **latitude** () const
- QString **lens** () const
- double **longitude** () const
- QString **make** () const
- QString **model** () const

- QString **name** () const
- int **orientation** () const
- int **pickLabel** () const
- int **rating** () const
- QString **rights** () const
- QString **sensitivity** () const
- void **setCaptions** (const [CaptionsMap](#) &)
- void **setColorLabel** (int)
- void **setCopyrightNotices** (const [MetaEngine::AltLangMap](#) &map)
- void **setCopyrights** (const [MetaEngine::AltLangMap](#) &map)
- void **setOrientation** (int)
- void **setPickLabel** (int)
- void **setRating** (int)
- void **setTitles** (const [CaptionsMap](#) &)
- QString **source** () const
- QStringList **tagsPath** () const
- QString **title** () const
- [CaptionsMap](#) **titles** () const
- QString **videoCodec** () const

### 9.372.1 Detailed Description

The easy way to use this container is given below:

```
// READ INFO FROM HOST -----
```

```
QUrl itemUrl; // The item url that you want to retrieve information. DInfoInterface* hostface; // The host application interface instance.
```

```
DInfoInterface::DInfoMap info = hostface->itemInfo(itemUrl); // First stage is to get the information map from host application. DItemInfo item(info); // Second stage, is to create the DItemInfo instance for this item by url. QString title = item.name(); // Now you can retrieve the title, QString description = item.comment(); // The comment, QDateTime time = item.dateTime(); // The time stamp, etc.
```

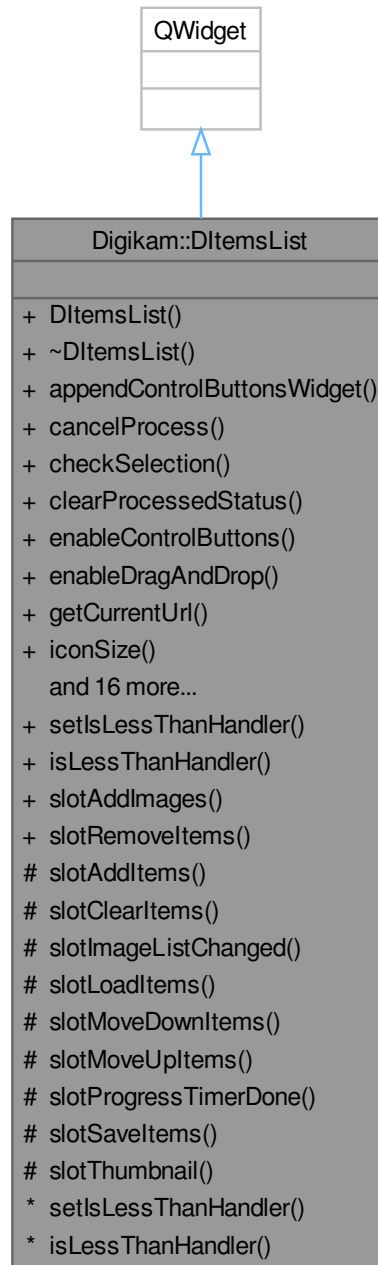
```
// WRITE INFO TO HOST -----
```

```
QUrl itemUrl; // The item url that you want to retrieve information. DInfoInterface* hostface; // The host application interface instance.
```

```
DItemInfo item; // Create the DItemInfo instance for this item with an empty internal info map. item.setRating(3); // Store rating to internal info map. item.setColorLabel(1); // Store color label to internal info map. hostface->setItemInfo(url, item.infoMap()); // Update item information to host using internal info map.
```

## 9.373 Digikam::DItemsList Class Reference

Inheritance diagram for Digikam::DItemsList:



### Public Types

- enum `ControlButton` {
  - Add** = 0x1 , **Remove** = 0x2 , **MoveUp** = 0x4 , **MoveDown** = 0x8 ,
  - Clear** = 0x10 , **Load** = 0x20 , **Save** = 0x40 }

- enum **ControlButtonPlacement** {  
**NoControlButtons** = 0 , **ControlButtonsLeft** , **ControlButtonsRight** , **ControlButtonsAbove** ,  
**ControlButtonsBelow** }
- typedef QFlags< ControlButton > **ControlButtons**

### Public Slots

- virtual void **slotAddImages** (const QList< QUrl > &list)
- virtual void **slotRemoveItems** ()

### Signals

- void **signalAddItems** (const QList< QUrl > &)
- void **signalContextMenuRequested** ()
- void **signalFoundRAWImages** (bool)
- void **signalImageListChanged** ()
- void **signalItemClicked** (QTreeWidgetItem \*)
- void **signalMoveDownItem** ()
- void **signalMoveUpItem** ()
- void **signalRemovedItems** (const QList< int > &)
- void **signalXMLCustomElements** (QXmlStreamReader &)
- void **signalXMLCustomElements** (QXmlStreamWriter &)
- void **signalXMLLoadImageElement** (QXmlStreamReader &)
- void **signalXMLSaveItem** (QXmlStreamWriter &, int)

### Public Member Functions

- **DItemsList** (QWidget \*const parent)
- void **appendControlButtonsWidget** (QWidget \*const widget)  
*Append a extra widget to the end of Control Button layout (as a progress bar for exemple).*
- void **cancelProcess** ()
- bool **checkSelection** ()  
*a function to check whether an image has been selected or not.*
- void **clearProcessedStatus** ()
- void **enableControlButtons** (bool enable=true)
- void **enableDragAndDrop** (const bool enable=true)
- QUrl **getCurrentUrl** () const
- int **iconSize** () const
- **DInfoInterface** \* **iface** () const
- virtual QList< QUrl > **imageUrls** (bool onlyUnprocessed=false) const
- **DItemsListView** \* **listView** () const
- void **loadImagesFromCurrentAlbum** ()  
*A function to load all the images from the album if no image has been selected by user.*
- void **loadImagesFromCurrentSelection** ()
- void **processed** (const QUrl &url, bool success)
- void **processing** (const QUrl &url)
- virtual void **removeItemByUrl** (const QUrl &url)
- void **setAllowDuplicate** (bool allow)
- void **setAllowRAW** (bool allow)
- void **setControlButtons** (ControlButtons buttonMask)
- QVBoxLayout \* **setControlButtonsPlacement** (ControlButtonPlacement placement)  
*Plug the control buttons near to the list, following 'placement' position.*

- void **setCurrentUrl** (const QUrl &url)
  - void **setIconSize** (int size)
  - void **setIface** (DInfoInterface \*const iface)
  - void **updateThumbnail** (const QUrl &url)
- 
- void **setIsLessThanHandler** (DItemsListIsLessThanHandler fncptr)  
*Methods to handle function pointer used to customize sort items in list.*
  - **DItemsListIsLessThanHandler isLessThanHandler** () const

### Protected Slots

- virtual void **slotAddItems** ()
- virtual void **slotClearItems** ()
- virtual void **slotImageListChanged** ()
- virtual void **slotLoadItems** ()
- virtual void **slotMoveDownItems** ()
- virtual void **slotMoveUpItems** ()
- void **slotProgressTimerDone** ()
- virtual void **slotSaveItems** ()
- virtual void **slotThumbnail** (const LoadingDescription &, const QPixmap &)

## 9.373.1 Member Function Documentation

### 9.373.1.1 appendControlButtonsWidget()

```
void Digikam::DItemsList::appendControlButtonsWidget (
    QWidget *const widget )
```

This method must be call before [setControlButtonsPlacement\(\)](#). Ownership of the widget is not transferred to the DItemList.

### 9.373.1.2 setControlButtonsPlacement()

```
QBoxLayout * Digikam::DItemsList::setControlButtonsPlacement (
    ControlButtonPlacement placement )
```

Return the instance of the layout supporting the control buttons, if any. This method must be calls after to use [appendControlButtonsWidget\(\)](#).

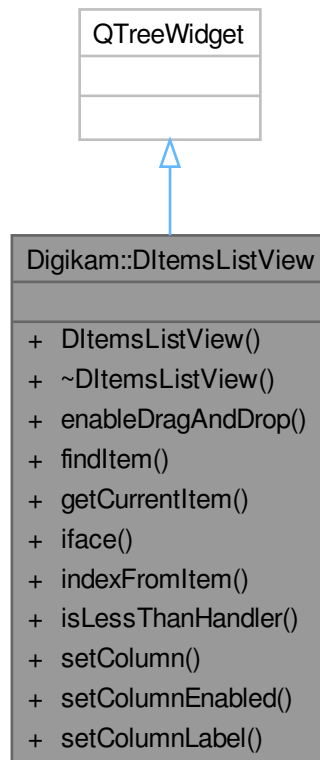
### 9.373.1.3 setIsLessThanHandler()

```
void Digikam::DItemsList::setIsLessThanHandler (
    DItemsListIsLessThanHandler fncptr )
```

See DItemsListIsLessThanHandler type for details.

## 9.374 Digikam::DItemsListView Class Reference

Inheritance diagram for Digikam::DItemsListView:



### Public Types

- enum `ColumnType` {  
`Thumbnail = 0`, `Filename`, `User1`, `User2`,  
`User3`, `User4`, `User5`, `User6` }

### Signals

- void `signalAddedDroppedItems` (const QList< QUrl > &)
- void `signalContextMenuRequested` ()
- void `signalItemClicked` (QTreeWidgetItem \*)

### Public Member Functions

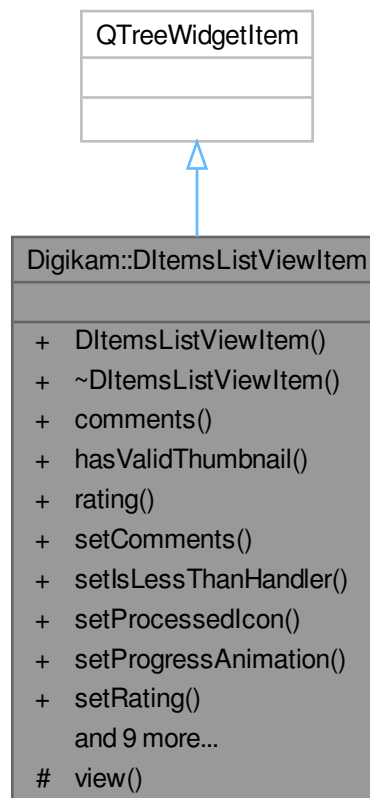
- `DItemsListView` (`DItemsList` \*const parent)
- void `enableDragAndDrop` (const bool enable=true)
- `DItemsListViewItem` \* `findItem` (const QUrl &url)



- [DItemsListViewItem](#) \* **getCurrentItem** () const
- [DInfoInterface](#) \* **iface** () const
- QModelIndex **indexFromItem** ([DItemsListViewItem](#) \*item, int column=0) const
- [DItemsListIsLessThanHandler](#) **isLessThanHandler** () const
- void **setColumn** (ColumnType column, const QString &label, bool enable)
- void **setColumnEnabled** (ColumnType column, bool enable)
- void **setColumnLabel** (ColumnType column, const QString &label)

## 9.375 Digikam::DItemsListViewItem Class Reference

Inheritance diagram for Digikam::DItemsListViewItem:



### Public Types

- enum **State** { **Waiting** , **Success** , **Failed** }

## Public Member Functions

- **DItemsListViewItem** ([DItemsListView](#) \*const view, const QUrl &url)
- QString **comments** () const
- bool **hasValidThumbnail** () const
- int **rating** () const
- void **setComments** (const QString &comments)
- void **setIsLessThanHandler** ([DItemsListIsLessThanHandler](#) fncptr)
- void **setProcessedIcon** (const QIcon &icon)
- void **setProgressAnimation** (const QPixmap &pix)
- void **setRating** (int rating)
- void **setState** (State state)
- void **setTags** (const QStringList &tags)
- void **setThumb** (const QPixmap &pix, bool hasThumb=true)
- void **setUrl** (const QUrl &url)
- State **state** () const
- QStringList **tags** () const
- void **updateInformation** ()
- virtual void [updateItemWidgets](#) ()
  - *Implement this, if you have special item widgets, e.g.*
- QUrl **url** () const

## Protected Member Functions

- [DItemsListView](#) \* **view** () const

## 9.375.1 Member Function Documentation

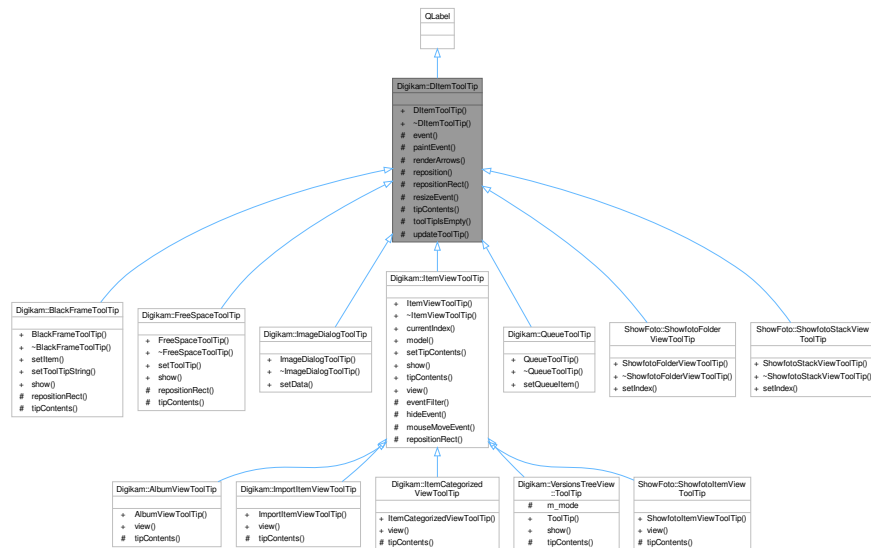
### 9.375.1.1 updateItemWidgets()

```
virtual void Digikam::DItemsListViewItem::updateItemWidgets ( ) [inline], [virtual]
```

an edit line they will be set automatically when adding items, changing order, etc.

## 9.376 Digikam::DItemToolTip Class Reference

Inheritance diagram for Digikam::DItemToolTip:



### Public Member Functions

- **DItemToolTip** (QWidget \*const parent=nullptr)

### Protected Member Functions

- bool **event** (QEvent \*) override
- void **paintEvent** (QPaintEvent \*) override
- void **renderArrows** ()
- void **reposition** ()
- virtual QRect **repositionRect** ()=0
- void **resizeEvent** (QResizeEvent \*) override
- virtual QString **tipContents** ()=0
- bool **toolTipsEmpty** () const
- void **updateToolTip** ()

### 9.376.1 Member Function Documentation

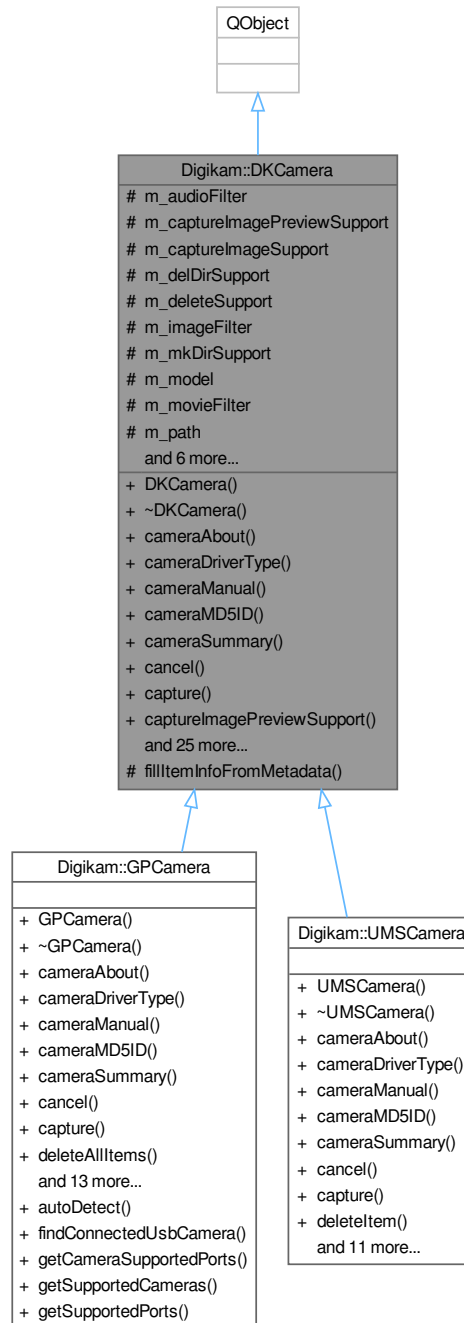
#### 9.376.1.1 tipContents()

```
virtual QString Digikam::DItemToolTip::tipContents ( ) [protected], [pure virtual]
```

Implemented in [Digikam::ItemViewToolTip](#).

## 9.377 Digikam::DKCamera Class Reference

Inheritance diagram for Digikam::DKCamera:



### Public Types

- enum **CameraDriverType** { **GPhotoDriver** = 0 , **UMSDriver** }

## Signals

- void **signalFolderList** (const QStringList &)

## Public Member Functions

- **DKCamera** (const QString &title, const QString &model, const QString &port, const QString &path)
- virtual bool **cameraAbout** (QString &about)=0
- virtual DKCamera::CameraDriverType **cameraDriverType** ()=0
- virtual bool **cameraManual** (QString &manual)=0
- virtual QByteArray **cameraMD5ID** ()=0
- virtual bool **cameraSummary** (QString &summary)=0
- virtual void **cancel** ()=0
- virtual bool **capture** (CamItemInfo &itemInfo)=0
- bool **captureImagePreviewSupport** () const
- bool **captureImageSupport** () const
- bool **delDirSupport** () const
- virtual bool **deleteltem** (const QString &folder, const QString &itemName)=0
- bool **deleteSupport** () const
- virtual bool **doConnect** ()=0
- virtual bool **downloadItem** (const QString &folder, const QString &itemName, const QString &saveFile)=0
- virtual bool **getFolders** (const QString &folder)=0
- virtual bool **getFreeSpace** (qint64 &bytesSize, qint64 &bytesAvail)=0
- virtual void **getItemInfo** (const QString &folder, const QString &itemName, CamItemInfo &info, bool use← Metadata)=0
- virtual bool **getItemsInfoList** (const QString &folder, bool useMetadata, CamItemInfoList &infoList)=0
  - If getImageDimensions is false, the camera shall set width and height to -1 if the values are not immediately available.*
- virtual bool **getMetadata** (const QString &folder, const QString &itemName, DMetadata &meta)=0
- virtual bool **getPreview** (QImage &preview)=0
- virtual bool **getThumbnail** (const QString &folder, const QString &itemName, QImage &thumbnail)=0
- QString **mimeType** (const QString &fileext) const
- bool **mkdirSupport** () const
- QString **model** () const
- QString **path** () const
- QString **port** () const
- void **printSupportedFeatures** ()
- virtual bool **setLockItem** (const QString &folder, const QString &itemName, bool lock)=0
- bool **thumbnailSupport** () const
- QString **title** () const
- virtual bool **uploadItem** (const QString &folder, const QString &itemName, const QString &localFile, CamItemInfo &itemInfo)=0
- bool **uploadSupport** () const
- QString **uuid** () const

## Protected Member Functions

- void **fillItemInfoFromMetadata** (CamItemInfo &item, const DMetadata &meta) const

## Protected Attributes

- QString `m_audioFilter`
- bool `m_captureImagePreviewSupport` = false
- bool `m_captureImageSupport` = false
- bool `m_delDirSupport` = false
- bool `m_deleteSupport` = false
- QString `m_imageFilter`
- bool `m_mkdirSupport` = false
- QString `m_model`
- QString `m_movieFilter`
- QString `m_path`
- QString `m_port`
- QString `m_rawFilter`
- bool `m_thumbnailSupport` = false
- QString `m_title`
- bool `m_uploadSupport` = false
- QString `m_uuid`

## 9.377.1 Member Function Documentation

### 9.377.1.1 capture()

```
virtual bool Digikam::DKCamera::capture (
    CamItemInfo & itemInfo ) [pure virtual]
```

Implemented in [Digikam::UMSCamera](#).

### 9.377.1.2 getFreeSpace()

```
virtual bool Digikam::DKCamera::getFreeSpace (
    qint64 & bytesSize,
    qint64 & bytesAvail ) [pure virtual]
```

Implemented in [Digikam::UMSCamera](#).

### 9.377.1.3 getItemsInfoList()

```
virtual bool Digikam::DKCamera::getItemsInfoList (
    const QString & folder,
    bool useMetadata,
    CamItemInfoList & infoList ) [pure virtual]
```

Implemented in [Digikam::UMSCamera](#), and [Digikam::GPCamera](#).

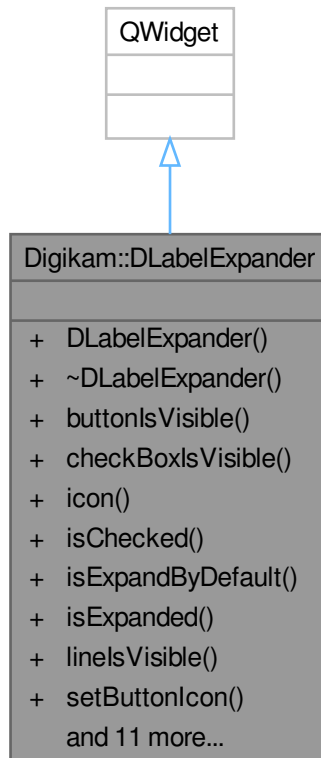
### 9.377.1.4 getPreview()

```
virtual bool Digikam::DKCamera::getPreview (
    QImage & preview ) [pure virtual]
```

Implemented in [Digikam::UMSCamera](#).

## 9.378 Digikam::DLabelExpander Class Reference

Inheritance diagram for Digikam::DLabelExpander:



### Signals

- void **signalButtonPressed** ()
- void **signalExpanded** (bool)
- void **signalToggled** (bool)

### Public Member Functions

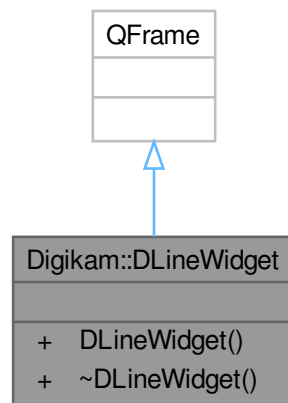
- **DLabelExpander** (QWidget \*const parent=nullptr)
- bool **buttonIsVisible** () const
- bool **checkboxIsVisible** () const
- QIcon **icon** () const
- bool **isChecked** () const
- bool **isExpandByDefault** () const
- bool **isExpanded** () const
- bool **lineIsVisible** () const
- void **setButtonIcon** (const QIcon &icon)
- void **setButtonVisible** (bool b)

- void **setCheckBoxVisible** (bool b)
- void **setChecked** (bool b)
- void **setExpandByDefault** (bool b)
- void **setExpanded** (bool b)
- void **setIcon** (const QIcon &icon)
- void **setLineVisible** (bool b)
- void **setText** (const QString &txt)
- void **setWidget** (QWidget \*const widget)
- QString **text** () const
- QWidget \* **widget** () const

## 9.379 Digikam::DLineWidget Class Reference

A widget to show an horizontal or vertical line separator.

Inheritance diagram for Digikam::DLineWidget:



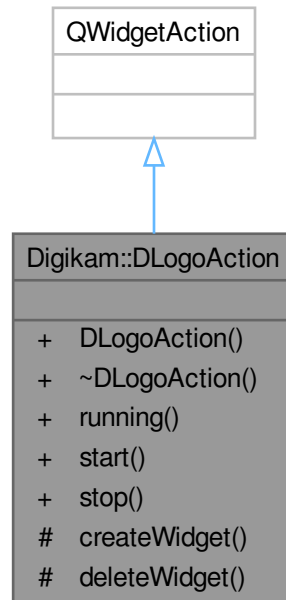
### Public Member Functions

- **DLineWidget** (Qt::Orientation orientation, QWidget \*const parent=nullptr)



## 9.380 Digikam::DLogoAction Class Reference

Inheritance diagram for Digikam::DLogoAction:



### Public Member Functions

- **DLogoAction** (QObject \*const parent, bool alignOnright=true)
- bool **running** () const
- void **start** ()
- void **stop** ()

### Protected Member Functions

- QWidget \* **createWidget** (QWidget \*parent) override
- void **deleteWidget** (QWidget \*widget) override

## 9.381 Digikam::DMessageBox Class Reference

### Static Public Member Functions

- static bool [readMsgBoxShouldBeShown](#) (const QString &dontShowAgainName)
- static void [saveMsgBoxShouldBeShown](#) (const QString &dontShowAgainName, bool value)

*Save the fact that the message box should not be shown again.*

- static int [showContinueCancel](#) (QMessageBox::Icon icon, QWidget \*const parent, const QString &caption, const QString &text, const QString &dontAskAgainName=QString())  
*Show a message box with Continue and Cancel buttons, and wait user feedback.*
- static int [showContinueCancelList](#) (QMessageBox::Icon icon, QWidget \*const parent, const QString &caption, const QString &text, const QStringList &items, const QString &dontAskAgainName=QString())  
*Show List of items to process into a message box with Continue and Cancel buttons, and wait user feedback.*
- static int [showContinueCancelWidget](#) (QMessageBox::Icon icon, QWidget \*const parent, const QString &caption, const QString &text, QWidget \*const listWidget, const QString &dontAskAgainName)  
*Show widget into a message box with Continue and Cancel buttons, and wait user feedback.*
- static void [showInformationList](#) (QMessageBox::Icon icon, QWidget \*const parent, const QString &caption, const QString &text, const QStringList &items, const QString &dontShowAgainName=QString())  
*Show List of items into an informative message box.*
- static void [showInformationWidget](#) (QMessageBox::Icon icon, QWidget \*const parent, const QString &caption, const QString &text, QWidget \*const listWidget, const QString &dontShowAgainName)  
*Show widget into an informative message box.*
- static int [showYesNo](#) (QMessageBox::Icon icon, QWidget \*const parent, const QString &caption, const QString &text, const QString &dontAskAgainName=QString())  
*Show a message box with Yes and No buttons, and wait user feedback.*
- static int [showYesNoList](#) (QMessageBox::Icon icon, QWidget \*const parent, const QString &caption, const QString &text, const QStringList &items, const QString &dontAskAgainName=QString())  
*Show List of items to process into a message box with Yes and No buttons, and wait user feedback.*
- static int [showYesNoWidget](#) (QMessageBox::Icon icon, QWidget \*const parent, const QString &caption, const QString &text, QWidget \*const listWidget, const QString &dontAskAgainName=QString())  
*Show widget into a message box with Yes and No buttons, and wait user feedback.*

## 9.381.1 Member Function Documentation

### 9.381.1.1 readMsgBoxShouldBeShown()

```
bool Digikam::DMessageBox::readMsgBoxShouldBeShown (
    const QString & dontShowAgainName ) [static]
```

#### Returns

true if the corresponding message box should be shown.

#### Parameters

<i>dontShowAgainName</i>	the name that identify the message box. If empty, this method return false.
--------------------------	---

### 9.381.1.2 saveMsgBoxShouldBeShown()

```
void Digikam::DMessageBox::saveMsgBoxShouldBeShown (
    const QString & dontShowAgainName,
    bool value ) [static]
```

#### Parameters

<i>dontShowAgainName</i>	the name that identify the message box. If empty, this method does nothing.
<i>value</i>	the value chosen in the message box to show it again next time.

### 9.381.1.3 showContinueCancel()

```
int Digikam::DMessageBox::showContinueCancel (
    QMessageBox::Icon icon,
    QWidget *const parent,
    const QString & caption,
    const QString & text,
    const QString & dontAskAgainName = QString() ) [static]
```

Return QMessageBox::Yes or QMessageBox::Cancel.

### 9.381.1.4 showContinueCancelList()

```
int Digikam::DMessageBox::showContinueCancelList (
    QMessageBox::Icon icon,
    QWidget *const parent,
    const QString & caption,
    const QString & text,
    const QStringList & items,
    const QString & dontAskAgainName = QString() ) [static]
```

Return QMessageBox::Yes or QMessageBox::Cancel.

### 9.381.1.5 showContinueCancelWidget()

```
int Digikam::DMessageBox::showContinueCancelWidget (
    QMessageBox::Icon icon,
    QWidget *const parent,
    const QString & caption,
    const QString & text,
    QWidget *const listWidget,
    const QString & dontAskAgainName ) [static]
```

Return QMessageBox::Yes or QMessageBox::Cancel.

### 9.381.1.6 showYesNo()

```
int Digikam::DMessageBox::showYesNo (
    QMessageBox::Icon icon,
    QWidget *const parent,
    const QString & caption,
    const QString & text,
    const QString & dontAskAgainName = QString() ) [static]
```

Return QMessageBox::Yes or QMessageBox::No.

### 9.381.1.7 showYesNoList()

```
int Digikam::DMessageBox::showYesNoList (
    QMessageBox::Icon icon,
    QWidget *const parent,
    const QString & caption,
    const QString & text,
    const QStringList & items,
    const QString & dontAskAgainName = QString() ) [static]
```

Return QMessageBox::Yes or QMessageBox::No.

### 9.381.1.8 showYesNoWidget()

```
int Digikam::DMessageBox::showYesNoWidget (
    QMessageBox::Icon icon,
    QWidget *const parent,
    const QString & caption,
    const QString & text,
    QWidget *const listWidget,
    const QString & dontAskAgainName = QString() ) [static]
```

Return QMessageBox::Yes or QMessageBox::No.



- enum [VIDEOCOLORMODEL](#) {  
**VIDEOCOLORMODEL\_UNKNOWN** = 1000 , **VIDEOCOLORMODEL\_OTHER** , **VIDEOCOLORMODEL\_SRGB** , **VIDEOCOLORMODEL\_BT709** ,  
**VIDEOCOLORMODEL\_BT601** }

*Video color model reported by FFmpeg following XMP DM Spec from Adobe.*

## Public Types inherited from [Digikam::MetaEngine](#)

- typedef QMap< QString, QString > [AltLangMap](#)  
*A map used to store a list of Alternative Language values.*
- enum [Backend](#) {  
[Exiv2Backend](#) = 0 , [LibRawBackend](#) , [LibHeifBackend](#) , [ImageMagickBackend](#) ,  
[FFmpegBackend](#) , [ExifToolBackend](#) , [VideoMergeBackend](#) , [NoBackend](#) }  
*Metadata Backend used to populate information.*
- enum [ImageColorWorkSpace](#) { **WORKSPACE\_UNSPECIFIED** = 0 , **WORKSPACE\_SRGB** = 1 ,  
**WORKSPACE\_ADOBERGB** = 2 , **WORKSPACE\_UNCALIBRATED** = 65535 }  
*The item color workspace values given by Exif metadata.*
- enum [ImageOrientation](#) {  
**ORIENTATION\_UNSPECIFIED** = 0 , **ORIENTATION\_NORMAL** = 1 , **ORIENTATION\_HFLIP** = 2 ,  
**ORIENTATION\_ROT\_180** = 3 ,  
**ORIENTATION\_VFLIP** = 4 , **ORIENTATION\_ROT\_90\_HFLIP** = 5 , **ORIENTATION\_ROT\_90** = 6 ,  
**ORIENTATION\_ROT\_90\_VFLIP** = 7 ,  
**ORIENTATION\_ROT\_270** = 8 }  
*The item orientation values given by Exif metadata.*
- typedef QMap< QString, QString > **MetaDataMap**  
*A map used to store Tags Key and Tags Value.*
- enum [MetadataWritingMode](#) { [WRITE\\_TO\\_FILE\\_ONLY](#) = 0 , [WRITE\\_TO\\_SIDECAR\\_ONLY](#) = 1 ,  
[WRITE\\_TO\\_SIDECAR\\_AND\\_FILE](#) = 2 , [WRITE\\_TO\\_SIDECAR\\_ONLY\\_FOR\\_READ\\_ONLY\\_FILES](#) = 3 }  
*The item metadata writing mode, between item file metadata and XMP sidecar file, depending on the context.*
- typedef QMap< QString, QStringList > [TagsMap](#)  
*A map used to store Tags Key and a list of Tags properties :*
- enum [XmpTagType](#) {  
**NormalTag** = 0 , **ArrayBagTag** = 1 , **StructureTag** = 2 , **ArrayLangTag** = 3 ,  
**ArraySeqTag** = 4 }  
*Xmp tag types, used by setXmpTag, only first three types are used.*

## Public Member Functions

- **DMetadata** (const [MetaEngineData](#) &data)
- **DMetadata** (const QString &filePath)
- bool [addToXmpTagStringBag](#) (const char \*const xmpTagName, const QStringList &entriesToAdd) const  
*Set an Xmp tag content using a list of strings defined by the 'entriesToAdd' parameter.*
- bool **applyChanges** (bool setVersion=false) const
- bool **getACDSeeTagsPath** (QStringList &tagsPath) const
- QString **getCameraSerialNumber** () const  
*Return a string with Camera serial number.*
- bool [getCopyrightInformation](#) ([Template](#) &t) const  
*Fills only the copyright values in the template.*
- [IptcCoreContactInfo](#) **getCreatorContactInfo** () const
- [IccProfile](#) [getIccProfile](#) () const  
*Reads an [IccProfile](#) that is described or embedded in the metadata.*

- [IptcCoreLocationInfo](#) **getIptcCoreLocation** () const
- [QStringList](#) **getIptcCoreSubjects** () const
- int **getItemColorLabel** (const [DMetadataSettingsContainer](#) &settings=[DMetadataSettings::instance\(\)](#) ->settings()) const
- [CaptionsMap](#) **getItemComments** (const [DMetadataSettingsContainer](#) &settings=[DMetadataSettings::instance\(\)](#) ->settings()) const
- bool **getItemFacesMap** ([QMultiMap](#)< [QString](#), [QVariant](#) > &facesPath) const  
*Get Images Face Map based on tags stored in Picassa/Metadatagroup format.*
- [QString](#) **getItemHistory** () const
- int **getItemPickLabel** (const [DMetadataSettingsContainer](#) &settings=[DMetadataSettings::instance\(\)](#) ->settings()) const
- int **getItemRating** (const [DMetadataSettingsContainer](#) &settings=[DMetadataSettings::instance\(\)](#) ->settings()) const
- bool **getItemTagsPath** ([QStringList](#) &tagsPath, const [DMetadataSettingsContainer](#) &settings=[DMetadataSettings::instance\(\)](#) ->settings()) const
- [CaptionsMap](#) **getItemTitles** (const [DMetadataSettingsContainer](#) &settings=[DMetadataSettings::instance\(\)](#) ->settings()) const
- [QString](#) **getItemUniqueld** () const
- [QString](#) **getLensDescription** () const  
*Return a string with Lens mounted on the front of camera.*
- [QVariant](#) **getMetadataField** ([MetadataInfo::Field](#) field) const  
*Returns the requested metadata field as a QVariant.*
- [QVariantList](#) **getMetadataFields** (const [MetadataFields](#) &fields) const
- [Template](#) **getMetadataTemplate** () const
- int **getMSecsInfo** () const  
*Returns millisecond time-stamp from Exif tags or 0 if not found.*
- [PhotoInfoContainer](#) **getPhotographInformation** () const
- [VideoInfoContainer](#) **getVideoInformation** () const  
*Returns video metadata from Xmp tags.*
- [QStringList](#) **getXmpKeywords** () const  
*Return a strings list of Xmp keywords from image.*
- [QStringList](#) **getXmpSubCategories** () const  
*Return a strings list of Xmp sub-categories from image.*
- [QStringList](#) **getXmpSubjects** () const  
*Return a strings list of Xmp subjects from image.*
- bool **hasItemHistoryTag** () const
- bool **load** (const [QString](#) &filePath, bool videoAll=false, [Backend](#) \*backend=nullptr)  
*Re-implemented from [MetaEngine](#) to use libraw identify, libheif, ffmpeg probe, and ImageMAGick identify methods if Exiv2 failed.*
- bool **loadUsingFFmpeg** (const [QString](#) &filePath)  
*Try to extract metadata using FFMpeg probe method (libav).*
- bool **loadUsingRawEngine** (const [QString](#) &filePath)  
*Try to extract metadata using Raw Engine identify method (libraw).*
- bool **mSecTimeStamp** (const char \*const exifTagName, int &ms) const  
*Extract milliseconds time-stamp of photo from an Exif tag and store it to 'ms'.*
- void **registerMetadataSettings** ()
- bool **removeExifColorSpace** () const  
*Remove the Exif color space identification from the image.*
- bool **removeExifTags** (const [QStringList](#) &tagFilters)
- bool **removeFromXmpTagStringBag** (const char \*const xmpTagName, const [QStringList](#) &entriesToRemove) const  
*Remove those Xmp tag entries that are listed in entriesToRemove from the entries in metadata.*
- bool **removelptcTags** (const [QStringList](#) &tagFilters)

- bool **removeItemFacesMap** () const  
*Remove Images Face Map tags from Picassa/Metadatagroup format.*
- bool **removeMetadataTemplate** () const
- bool **removeXmpKeywords** (const QStringList &keywordsToRemove)  
*Remove those Xmp keywords that are listed in keywordsToRemove from the keywords in metadata.*
- bool **removeXmpSubCategories** (const QStringList &categoriesToRemove)  
*Remove those Xmp sub-categories that are listed in categoriesToRemove from the sub-categories in metadata.*
- bool **removeXmpSubjects** (const QStringList &subjectsToRemove)  
*Remove those Xmp subjects that are listed in subjectsToRemove from the subjects in metadata.*
- bool **removeXmpTags** (const QStringList &tagFilters)
- bool **save** (const QString &filePath, bool setVersion=false) const
- bool **setACDSeeTagsPath** (const QStringList &tagsPath) const
- bool **setCreatorContactInfo** (const [IptcCoreContactInfo](#) &info) const
- bool **setIccProfile** (const [IccProfile](#) &profile)  
*Sets the IccProfile embedded in the Exif metadata.*
- bool **setIptcCoreLocation** (const [IptcCoreLocationInfo](#) &location) const
- bool **setItemColorLabel** (int colorId, const [DMetadadataSettingsContainer](#) &settings=[DMetadadataSettings::instance\(\)](#) ->settings()) const
- bool **setItemComments** (const [CaptionsMap](#) &comments, const [DMetadadataSettingsContainer](#) &settings=[DMetadadataSettings::instance\(\)](#) ->settings()) const
- bool **setItemFacesMap** (const QMap< QString, QVariant > &facesPath, bool write, const QSize &size=QSize()) const  
*Set Images Face Map tags in Picassa/Metadatagroup format.*
- bool **setItemHistory** (const QString &imageHistoryXml) const
- bool **setItemPickLabel** (int pickId, const [DMetadadataSettingsContainer](#) &settings=[DMetadadataSettings::instance\(\)](#) ->settings()) const
- bool **setItemRating** (int rating, const [DMetadadataSettingsContainer](#) &settings=[DMetadadataSettings::instance\(\)](#) ->settings()) const
- bool **setItemTagsPath** (const QStringList &tagsPath, const [DMetadadataSettingsContainer](#) &settings=[DMetadadataSettings::instance\(\)](#) ->settings()) const
- bool **setItemTitles** (const [CaptionsMap](#) &title, const [DMetadadataSettingsContainer](#) &settings=[DMetadadataSettings::instance\(\)](#) ->settings()) const
- bool **setItemUniqueld** (const QString &uuid) const
- bool **setMetadataTemplate** (const [Template](#) &t) const
- void **setSettings** (const [MetaEngineSettingsContainer](#) &settings)
- bool **setXmpKeywords** (const QStringList &newKeywords) const  
*Set Xmp keywords using a list of strings defined by 'newKeywords' parameter.*
- bool **setXmpSubCategories** (const QStringList &newSubCategories) const  
*Set Xmp sub-categories using a list of strings defined by 'newSubCategories' parameter.*
- bool **setXmpSubjects** (const QStringList &newSubjects) const  
*Set Xmp subjects using a list of strings defined by 'newSubjects' parameter.*

## Public Member Functions inherited from [Digikam::MetaEngine](#)

- **MetaEngine** ()  
*Standard constructor.*
- **MetaEngine** (const [MetaEngineData](#) &data)  
*Constructor to load from parsed data.*
- **MetaEngine** (const QString &filePath)  
*Constructor to Load Metadata from item file.*
- virtual **~MetaEngine** ()  
*Standard destructor.*



- [MetaEngineData](#) **data** () const
- void **setData** (const [MetaEngineData](#) &data)
- bool **loadFromData** (const QByteArray &imgData)  
*Load all metadata (Exif, Iptc, Xmp, and JFIF Comments) from a byte array.*
- bool **loadFromDataAndMerge** (const QByteArray &imgData, const QStringList &exclude=QStringList())  
*Load and merge metadata (Exif, Iptc and Xmp) from a byte array.*
- bool **isEmpty** () const  
*Return 'true' if metadata container in memory as no Comments, Exif, Iptc, and Xmp.*
- QSize **getPixelSize** () const  
*Returns the pixel size of the current item.*
- QString **getMimeType** () const  
*Returns the mime type of this item.*
- void **setReadWithExifTool** (const bool on)  
*Enable or disable reading metadata operations with ExifTool.*
- bool **readWithExifTool** () const  
*Return true if reading metadata operations with ExifTool is enabled.*
- void **setWriteWithExifTool** (const bool on)  
*Enable or disable writing metadata operations with ExifTool.*
- bool **writeWithExifTool** () const  
*Return true if writing metadata operations with ExifTool is enabled.*
- void **setWriteRawFiles** (const bool on)  
*Enable or disable writing metadata operations to RAW files.*
- bool **writeRawFiles** () const  
*Return true if writing metadata operations on RAW files is enabled.*
- void **setWriteDngFiles** (const bool on)  
*Enable or disable writing metadata operations to DNG files.*
- bool **writeDngFiles** () const  
*Return true if writing metadata operations on DNG files is enabled.*
- void **setUseXMPSidecar4Reading** (const bool on)  
*Enable or disable using XMP sidecar for reading metadata.*
- bool **useXMPSidecar4Reading** () const  
*Return true if using XMP sidecar for reading metadata is enabled.*
- void **setUseCompatibleFileName** (const bool on)  
*Enable or disable using compatible file name for sidecar files.*
- bool **useCompatibleFileName** () const  
*Return true if using compatible file name for sidecar files.*
- void **setMetadataWritingMode** (const int mode)  
*Set metadata writing mode.*
- int **metadataWritingMode** () const  
*Return the metadata writing mode.*
- void **setUpdateFileTimeStamp** (bool on)  
*Enable or disable file timestamp updating when metadata are saved.*
- bool **updateFileTimeStamp** () const  
*Return true if file timestamp is updated when metadata are saved.*
  
- bool **setItemProgramId** (const QString &program, const QString &version) const  
*Set Program name and program version in Exif and Iptc Metadata.*
- QSize **getItemDimensions** () const  
*Return the size of item in pixels using Exif tags.*
- bool **setItemDimensions** (const QSize &size) const

- Set the size of item in pixels in Exif tags.*

  - [MetaEngine::ImageOrientation](#) [getItemOrientation](#) () const
    - Return the item orientation set in Exif metadata.*
  - bool [setItemOrientation](#) ([ImageOrientation](#) orientation) const
    - Set the Exif orientation tag of item.*
  - [MetaEngine::ImageColorWorkSpace](#) [getItemColorWorkSpace](#) () const
    - Return the item color-space set in Exif metadata.*
  - bool [setItemColorWorkSpace](#) ([ImageColorWorkSpace](#) workspace) const
    - Set the Exif color-space tag of item.*
  - QDateTime [getItemDateTime](#) () const
    - Return the time stamp of item.*
  - bool [setImageDateTime](#) (const QDateTime &dateTime, bool setDateDigitized=false) const
    - Set the Exif and Iptc time stamp.*
  - QDateTime [getDigitizationDateTime](#) (bool fallbackToCreationTime=false) const
    - Return the digitization time stamp of the item.*
  - bool [getItemPreview](#) (QImage &preview) const
    - Return a QImage copy of Iptc preview image.*
  - bool [setItemPreview](#) (const QImage &preview) const
    - Set the Iptc preview image.*
  - QByteArray [getItemIccProfile](#) () const
    - Get image ICC profile.*
  - bool [setItemIccProfile](#) (const QByteArray &iccData) const
    - Set image ICC profile.*
  
- bool [initializeGPSInfo](#) ()
  - Make sure all static required GPS EXIF and XMP tags exist.*
  
- bool [getGPSInfo](#) (double &altitude, double &latitude, double &longitude) const
  - Get all GPS location information set in item.*
  
- QString [getGPSLatitudeString](#) () const
  - Get GPS location information set in the item, in the GPSCoordinate format as described in the XMP specification.*
  
- QString [getGPSLongitudeString](#) () const
  
- bool [getGPSLatitudeNumber](#) (double \*const latitude) const
  - Get GPS location information set in the item, as a double floating point number as in degrees where the sign determines the direction ref (North + / South - ; East + / West -).*
  
- bool [getGPSLongitudeNumber](#) (double \*const longitude) const
  
- bool [getGPSAltitude](#) (double \*const altitude) const
  - Get GPS altitude information, in meters, relative to sea level (positive sign above sea level)*
  
- bool [setGPSInfo](#) (const double altitude, const double latitude, const double longitude)
  - Set all GPS location information into item.*

- bool [setGPSInfo](#) (const double \*const altitude, const double latitude, const double longitude)  
*Set all GPS location information into item.*
- bool [setGPSInfo](#) (const double altitude, const QString &latitude, const QString &longitude)  
*Set all GPS location information into item.*
- bool [removeGPSInfo](#) ()  
*Remove all Exif tags relevant of GPS location information.*
- void [setFilePath](#) (const QString &path)  
*Set the file path of current item.*
- QString [getFilePath](#) () const  
*Return the file path of current item.*
- bool [load](#) (const QString &filePath, [Backend](#) \*backend=nullptr)  
*Load all metadata (Exif, Iptc, Xmp, and JFIF Comments) from a picture (JPEG, RAW, TIFF, PNG, DNG, etc...).*
- bool [loadFromSidecarAndMerge](#) (const QString &filePath)  
*Load metadata from a sidecar file and merge.*
- bool [save](#) (const QString &filePath, bool setVersion=false) const  
*Save all metadata to a file.*
- bool [applyChanges](#) (bool setVersion=false) const  
*The same than [save\(\)](#) method, but it apply on current item.*
- bool [exportChanges](#) (const QString &exvTmpFile) const  
*Export metadata to a temporary EXV file container.*
- bool [hasComments](#) () const  
*Return 'true' if metadata container in memory as Comments.*
- bool [clearComments](#) () const  
*Clear the Comments metadata container in memory.*
- QByteArray [getComments](#) () const  
*Return a Qt byte array copy of Comments container get from current item.*
- QString [getCommentsDecoded](#) () const  
*Return a Qt string object of Comments from current item decoded using the 'detectEncodingAndDecode()' method.*
- bool [setComments](#) (const QByteArray &data) const  
*Set the Comments data using a Qt byte array.*

- [TagsMap](#) **getStdExifTagsList** () const  
*Return a map of all standard Exif tags supported by Exiv2.*
- [TagsMap](#) **getMakernoteTagsList** () const  
*Return a map of all non-standard Exif tags (makernotes) supported by Exiv2.*
- bool **hasExif** () const  
*Return 'true' if metadata container in memory as Exif.*
- bool **clearExif** () const  
*Clear the Exif metadata container in memory.*
- QByteArray [getExifEncoded](#) (bool addExifHeader=false) const  
*Returns the exif data encoded to a QByteArray in a form suitable for storage in a JPEG image.*
- bool [setExif](#) (const QByteArray &data) const  
*Set the Exif data using a Qt byte array.*
- QImage [getExifThumbnail](#) (bool fixOrientation) const  
*Return a QImage copy of Exif thumbnail image.*
- bool [rotateExifQImage](#) (QImage &image, [ImageOrientation](#) orientation) const  
*Fix orientation of a QImage image accordingly with Exif orientation tag.*
- bool [setExifThumbnail](#) (const QImage &thumb) const  
*Set the Exif Thumbnail image.*
- bool **removeExifThumbnail** () const  
*Remove the Exif Thumbnail from the item.*
- bool [setTiffThumbnail](#) (const QImage &thumb) const  
*Adds a JPEG thumbnail to a TIFF images.*
- QString [getExifComment](#) (bool readDescription=true) const  
*Return a QString copy of Exif user comments.*
- QString [getExifTagComment](#) (const char \*exifTagName) const  
*Return a Exif tag comment like a string.*
- bool [setExifComment](#) (const QString &comment, bool writeDescription=true) const  
*Set the Exif user comments from item.*
- QString [getExifTagString](#) (const char \*exifTagName, bool escapeCR=true) const  
*Get an Exif tags content like a string.*

- bool [setExifTagString](#) (const char \*exifTagName, const QString &value) const  
*Set an Exif tag content using a string.*
- bool [getExifTagLong](#) (const char \*exifTagName, long &val) const  
*Get an Exif tag content like a long value.*
- bool [getExifTagLong](#) (const char \*exifTagName, long &val, int component) const  
*Get an Exif tag content like a long value.*
- bool [setExifTagLong](#) (const char \*exifTagName, long val) const  
*Set an Exif tag content using a long value.*
- bool [setExifTagUShort](#) (const char \*exifTagName, unsigned int val) const  
*Set an Exif tag content using a unsigned short value.*
- bool [getExifTagRational](#) (const char \*exifTagName, long int &num, long int &den, int component=0) const  
*Get the 'component' index of an Exif tags content like a rational value.*
- bool [setExifTagRational](#) (const char \*exifTagName, long int num, long int den) const  
*Set an Exif tag content using a rational value.*
- bool [setExifTagURational](#) (const char \*exifTagName, unsigned long int num, unsigned long int den) const  
*Set an Exif tag content using a unsigned rational value.*
- QByteArray [getExifTagData](#) (const char \*exifTagName) const  
*Get an Exif tag content like a bytes array.*
- bool [setExifTagData](#) (const char \*exifTagName, const QByteArray &data) const  
*Set an Exif tag content using a bytes array.*
- QVariant [getExifTagVariant](#) (const char \*exifTagName, bool rationalAsListOfInts=true, bool escapeCR=true, int component=0) const  
*Get an Exif tags content as a QVariant.*
- bool [setExifTagVariant](#) (const char \*exifTagName, const QVariant &data, bool rationalWantSmallDenominator=true) const  
*Set an Exif tag content using a QVariant.*
- bool [removeExifTag](#) (const char \*exifTagName) const  
*Remove the Exif tag 'exifTagName' from Exif metadata.*
- QString [getExifTagTitle](#) (const char \*exifTagName)  
*Return the Exif Tag title or a null string.*
- QString [getExifTagDescription](#) (const char \*exifTagName)

*Return the Exif Tag description or a null string.*

- `QString createExifUserStringFromValue` (const char \*exifTagName, const QVariant &val, bool escapeCR=true)
 

*Takes a QVariant value as it could have been retrieved by `getExifTagVariant` with the given `exifTagName`, and returns its value properly converted to a string (including translations from Exiv2).*
- `MetaEngine::MetaDataMap getExifTagsDataList` (const QStringList &exifKeysFilter=QStringList(), bool invertSelection=false, bool extractBinary=true) const
 

*Return a map of Exif tags name/value found in metadata sorted by Exif keys given by 'exifKeysFilter'.*
- `MetaEngine::TagsMap getIptcTagsList` () const
 

*Return a map of all standard Iptc tags supported by Exiv2.*
- `bool hasIptc` () const
 

*Return 'true' if metadata container in memory as Iptc.*
- `bool clearIptc` () const
 

*Clear the Iptc metadata container in memory.*
- `QByteArray getIptc` (bool addIrbHeader=false) const
 

*Return a Qt byte array copy of Iptc container get from current item.*
- `bool setIptc` (const QByteArray &data) const
 

*Set the Iptc data using a Qt byte array.*
- `QString getIptcTagString` (const char \*iptcTagName, bool escapeCR=true) const
 

*Get an Iptc tag content like a string.*
- `bool setIptcTagString` (const char \*iptcTagName, const QString &value) const
 

*Set an Iptc tag content using a string.*
- `QStringList getIptcTagsStringList` (const char \*iptcTagName, bool escapeCR=true) const
 

*Returns a strings list with of multiple Iptc tags from the item.*
- `bool setIptcTagsStringList` (const char \*iptcTagName, int maxSize, const QStringList &oldValues, const QStringList &newValues) const
 

*Set multiple Iptc tags contents using a strings list.*
- `QByteArray getIptcTagData` (const char \*iptcTagName) const
 

*Get an Iptc tag content as a bytes array.*
- `bool setIptcTagData` (const char \*iptcTagName, const QByteArray &data) const
 

*Set an Iptc tag content using a bytes array.*

- bool [removeIptcTag](#) (const char \*iptcTagName) const  
*Remove the all instance of Iptc tags 'iptcTagName' from Iptc metadata.*
- QString [getIptcTagTitle](#) (const char \*iptcTagName)  
*Return the Iptc Tag title or a null string.*
- QString [getIptcTagDescription](#) (const char \*iptcTagName)  
*Return the Iptc Tag description or a null string.*
- [MetaEngine::MetaDataMap](#) [getIptcTagsDataList](#) (const QStringList &iptcKeysFilter=QStringList(), bool invertSelection=false) const  
*Return a map of Iptc tags name/value found in metadata sorted by Iptc keys given by 'iptcKeysFilter'.*
- QStringList [getIptcKeywords](#) () const  
*Return a strings list of Iptc keywords from item.*
- bool [setIptcKeywords](#) (const QStringList &oldKeywords, const QStringList &newKeywords) const  
*Set Iptc keywords using a list of strings defined by 'newKeywords' parameter.*
- QStringList [getIptcSubjects](#) () const  
*Return a strings list of Iptc subjects from item.*
- bool [setIptcSubjects](#) (const QStringList &oldSubjects, const QStringList &newSubjects) const  
*Set Iptc subjects using a list of strings defined by 'newSubjects' parameter.*
- QStringList [getIptcSubCategories](#) () const  
*Return a strings list of Iptc sub-categories from item.*
- bool [setIptcSubCategories](#) (const QStringList &oldSubCategories, const QStringList &newSubCategories) const  
*Set Iptc sub-categories using a list of strings defined by 'newSubCategories' parameter.*
- [MetaEngine::TagsMap](#) [getXmpTagsList](#) () const  
*Return a map of all standard Xmp tags supported by Exiv2.*
- bool [hasXmp](#) () const  
*Return 'true' if metadata container in memory as Xmp.*
- bool [clearXmp](#) () const  
*Clear the Xmp metadata container in memory.*
- QByteArray [getXmp](#) () const  
*Return a Qt byte array copy of XMP container get from current item.*
- bool [setXmp](#) (const QByteArray &data) const

*Set the Xmp data using a Qt byte array.*

- **QString** [getXmpTagString](#) (const char \*xmpTagName, bool escapeCR=true) const  
*Get a Xmp tag content like a string.*
- **bool** [setXmpTagString](#) (const char \*xmpTagName, const QString &value) const  
*Set a Xmp tag content using a string.*
- **bool** [setXmpTagString](#) (const char \*xmpTagName, const QString &value, [XmpTagType](#) type) const  
*Set a Xmp tag with a specific type.*
- **QString** [getXmpTagTitle](#) (const char \*xmpTagName)  
*Return the Xmp Tag title or a null string.*
- **QString** [getXmpTagDescription](#) (const char \*xmpTagName)  
*Return the Xmp Tag description or a null string.*
- **MetaEngine::MetaDataMap** [getXmpTagsDataList](#) (const QStringList &xmpKeysFilter=QStringList(), bool invertSelection=false) const  
*Return a map of Xmp tags name/value found in metadata sorted by Xmp keys given by 'xmpKeysFilter'.*
- **MetaEngine::AltLangMap** [getXmpTagStringListLangAlt](#) (const char \*xmpTagName, bool escapeCR=true) const  
*Get all redondant Alternative Language Xmp tags content like a map.*
- **bool** [setXmpTagStringListLangAlt](#) (const char \*xmpTagName, const [MetaEngine::AltLangMap](#) &values) const  
*Set an Alternative Language Xmp tag content using a map.*
- **QString** [getXmpTagStringLangAlt](#) (const char \*xmpTagName, const QString &langAlt, bool escapeCR) const  
*Get a Xmp tag content like a string set with an alternative language header 'langAlt' (like "fr-FR" for French - RFC3066 notation) If 'escapeCR' parameter is true, the CR characters will be removed.*
- **bool** [setXmpTagStringLangAlt](#) (const char \*xmpTagName, const QString &value, const QString &langAlt) const  
*Set a Xmp tag content using a string with an alternative language header.*
- **QStringList** [getXmpTagStringSeq](#) (const char \*xmpTagName, bool escapeCR=true) const  
*Get a Xmp tag content like a sequence of strings.*
- **bool** [setXmpTagStringSeq](#) (const char \*xmpTagName, const QStringList &seq) const  
*Set a Xmp tag content using the sequence of strings 'seq'.*
- **QStringList** [getXmpTagStringBag](#) (const char \*xmpTagName, bool escapeCR) const  
*Get a Xmp tag content like a bag of strings.*



- bool [setXmpTagStringBag](#) (const char \*xmpTagName, const QStringList &bag) const  
*Set a Xmp tag content using the bag of strings 'bag'.*
- bool [addToXmpTagStringBag](#) (const char \*xmpTagName, const QStringList &entriesToAdd) const  
*Set an Xmp tag content using a list of strings defined by the 'entriesToAdd' parameter.*
- bool [removeFromXmpTagStringBag](#) (const char \*xmpTagName, const QStringList &entriesToRemove) const  
*Remove those Xmp tag entries that are listed in entriesToRemove from the entries in metadata.*
- QVariant [getXmpTagVariant](#) (const char \*xmpTagName, bool rationalAsListOfInts=true, bool stringEscape↵  
CR=true) const  
*Get an Xmp tag content as a QVariant.*
- QStringList [getXmpKeywords](#) () const  
*Return a strings list of Xmp keywords from item.*
- bool [setXmpKeywords](#) (const QStringList &newKeywords) const  
*Set Xmp keywords using a list of strings defined by 'newKeywords' parameter.*
- bool [removeXmpKeywords](#) (const QStringList &keywordsToRemove)  
*Remove those Xmp keywords that are listed in keywordsToRemove from the keywords in metadata.*
- QStringList [getXmpSubjects](#) () const  
*Return a strings list of Xmp subjects from item.*
- bool [setXmpSubjects](#) (const QStringList &newSubjects) const  
*Set Xmp subjects using a list of strings defined by 'newSubjects' parameter.*
- bool [removeXmpSubjects](#) (const QStringList &subjectsToRemove)  
*Remove those Xmp subjects that are listed in subjectsToRemove from the subjects in metadata.*
- QStringList [getXmpSubCategories](#) () const  
*Return a strings list of Xmp sub-categories from item.*
- bool [setXmpSubCategories](#) (const QStringList &newSubCategories) const  
*Set Xmp sub-categories using a list of strings defined by 'newSubCategories' parameter.*
- bool [removeXmpSubCategories](#) (const QStringList &categoriesToRemove)  
*Remove those Xmp sub-categories that are listed in categoriesToRemove from the sub-categories in metadata.*
- bool [removeXmpTag](#) (const char \*xmpTagName, bool family=false) const  
*Remove the Xmp tag 'xmpTagName' from Xmp metadata.*

### Static Public Member Functions

- static double **apexApertureToFNumber** (double aperture)
- static double **apexShutterSpeedToExposureTime** (double shutterSpeed)
- static CountryCodeMap **countryCodeMap** ()  
*Return a map of ISO-639-1 2 letters country codes with country names.*
- static CountryCodeMap **countryCodeMap2** ()  
*Return a map of ISO-639-2 3 letters country codes with country names.*
- static QMap< int, QString > **possibleValuesForEnumField** (MetadataInfo::Field field)  
*Returns a map of possible enum values and their user-presentable, i18n'ed representation.*
- static **MetaEngine::AltLangMap toAltLangMap** (const QVariant &var)
- static QStringList **valuesToString** (const QVariantList &list, const MetadataFields &fields)
- static QString **valueToString** (const QVariant &value, MetadataInfo::Field field)  
*Convert a QVariant value of the specified field to a user-presentable, i18n'ed string.*
- static QString **videoColorModelToString** (**VIDEOCOLORMODEL** videoColorModel)  
*Helper method to translate enum values to user presentable strings.*

### Static Public Member Functions inherited from **Digikam::MetaEngine**

- static bool **initializeExiv2** ()  
*Return true if Exiv2 library initialization is done properly.*
- static bool **supportXmp** ()  
*Return true if Exiv2 library is compiled with Xmp metadata support.*
- static bool **supportJpegXL** ()  
*Return true if Exiv2 library is compiled with JpegXL metadata support.*
- static bool **supportBmff** ()  
*Return true if library support Base Media File Format (aka CR3, HEIF, HEIC, and AVIF).*
- static bool **supportMetadataWriting** (const QString &typeMime)  
*Return true if library can write metadata to typeMime file format.*
- static QString **Exiv2Version** ()  
*Return a string version of Exiv2 release in format "major.minor.patch".*
- static void **convertToRational** (const double number, long int \*const numerator, long int \*const denominator, const int rounding)  
*This method converts 'number' to a rational value, returned in the 'numerator' and 'denominator' parameters.*
- static void **convertToRationalSmallDenominator** (const double number, long int \*const numerator, long int \*const denominator)  
*This method convert a 'number' to a rational value, returned in 'numerator' and 'denominator' parameters.*
- static double **convertDegreeAngleToDouble** (double degrees, double minutes, double seconds)  
*Converts degrees values as a double representation.*
- static QString **convertToGPSCoordinateString** (const long int numeratorDegrees, const long int denominatorDegrees, const long int numeratorMinutes, const long int denominatorMinutes, const long int numeratorSeconds, const long int denominatorSeconds, const char directionReference)  
*Converts a GPS position stored as rationals in Exif to the form described as GPSCoordinate in the XMP specification, either in the form "256,45,34N" or "256,45.566667N".*

- static QString **convertToGPSCoordinateString** (const bool isLatitude, double coordinate)  
*Converts a GPS position stored as double floating point number in degrees to the form described as GPSCoordinate in the XMP specification.*
- static bool **convertFromGPSCoordinateString** (const QString &coordinate, long int \*const numeratorDegrees, long int \*const denominatorDegrees, long int \*const numeratorMinutes, long int \*const denominatorMinutes, long int \*const numeratorSeconds, long int \*const denominatorSeconds, char \*const directionReference)  
*Converts a GPSCoordinate string as defined by XMP to three rationals and the direction reference.*
- static bool **convertFromGPSCoordinateString** (const QString &gpsString, double \*const coordinate)  
*Convert a GPSCoordinate string as defined by XMP to a double floating point number in degrees where the sign determines the direction ref (North + / South - ; East + / West -).*
- static bool **convertToUserPresentableNumbers** (const QString &coordinate, int \*const degrees, int \*const minutes, double \*const seconds, char \*const directionReference)  
*Converts a GPSCoordinate string to user presentable numbers, integer degrees and minutes and double floating point seconds, and a direction reference ('N' or 'S', 'E' or 'W')*
- static void **convertToUserPresentableNumbers** (const bool isLatitude, double coordinate, int \*const degrees, int \*const minutes, double \*const seconds, char \*const directionReference)  
*Converts a double floating point number to user presentable numbers, integer degrees and minutes and double floating point seconds, and a direction reference ('N' or 'S', 'E' or 'W').*
- static QString **sidecarFilePathForFile** (const QString &path)  
*Return the XMP Sidecar file path for a item file path.*
- static QString **sidecarPath** (const QString &path)  
*Like [sidecarFilePathForFile\(\)](#), but works for local file path.*
- static QUrl **sidecarUrl** (const QUrl &url)  
*Like [sidecarFilePathForFile\(\)](#), but works for remote URLs.*
- static QUrl **sidecarUrl** (const QString &path)  
*Gives a file url for a local path.*
- static bool **hasSidecar** (const QString &path)  
*Performs a QFileInfo based check if the given local file has a sidecar.*
- static QString **backendName** (Backend t)  
*Return a string of backend name used to parse metadata from file.*
- static bool **canWriteComment** (const QString &filePath)  
*Return 'true' if Comments can be written in file.*
- static QString **detectLanguageAlt** (const QString &value, QString &lang)  
*Language Alternative autodetection.*

- static bool **canWriteExif** (const QString &filePath)  
*Return 'true' if Exif can be written in file.*
- static bool **canWriteIptc** (const QString &filePath)  
*Return 'true' if Iptc can be written in file.*
- static bool **canWriteXmp** (const QString &filePath)  
*Return 'true' if Xmp can be written in file.*
- static bool **registerXmpNameSpace** (const QString &uri, const QString &prefix)  
*Register a namespace which Exiv2 doesn't know yet.*
- static bool **unregisterXmpNameSpace** (const QString &uri)  
*Unregister a previously registered custom namespace.*

### Additional Inherited Members

### Protected Member Functions inherited from [Digikam::MetaEngine](#)

- bool **setProgramId** () const  
*Set the Program Name and Program Version information in Exif and Iptc metadata.*

## 9.382.1 Member Enumeration Documentation

### 9.382.1.1 VIDEOCOLORMODEL

```
enum Digikam::DMetadata::VIDEOCOLORMODEL
```

These values are stored in DB as Image color model properties (extension of DImg::ColorModel)

## 9.382.2 Member Function Documentation

### 9.382.2.1 addToXmpTagStringBag()

```
bool Digikam::DMetadata::addToXmpTagStringBag (
    const char *const xmpTagName,
    const QStringList & entriesToAdd ) const
```

The existing entries are preserved. The method will compare all new with all already existing entries to prevent duplicates in the image. Return true if the entries have been added to metadata.

### 9.382.2.2 getCopyrightInformation()

```
bool Digikam::DMetadata::getCopyrightInformation (
    Template & t ) const
```

Use getMetadataTemplate() usually. Returns true if valid fields were read.

**9.382.2.3 getIccProfile()**

```
IccProfile Digikam::DMetadata::getIccProfile ( ) const
```

This method does not retrieve profiles embedded in the image but from the Exif metadata, e.g. embedded profiles in JPEG images. Returns a null profile if no profile is found.

**9.382.2.4 getItemFacesMap()**

```
bool Digikam::DMetadata::getItemFacesMap (
    QMap< QString, QVariant > & facesPath ) const
```

Read face tags only if Exiv2 can write them, otherwise garbage tags will be generated on image transformation

**9.382.2.5 getLensDescription()**

```
QString Digikam::DMetadata::getLensDescription ( ) const
```

There no standard Exif tag for Lens information. Camera makernotes and Xmp tags are parsed. Take a care : lens information are not standardized and string content is not homogeneous between camera model/maker. < Canon Cameras Makernote.

- < Canon Cameras Makernote.
- < Alternative Canon Cameras Makernote.
- < Nikon Cameras Makernote.
- < Nikon Cameras Makernote.
- < Nikon Cameras Makernote.
- < Minolta Cameras Makernote.
- < Sony Cameras Makernote.
- < Sony Cameras Makernote.
- < Sony Cameras Makernote.
- < Pentax Cameras Makernote.
- < Pentax Cameras Makernote.
- < Panasonic Cameras Makernote.
- < Panasonic Cameras Makernote.
- < Sigma Cameras Makernote.
- < Samsung Cameras Makernote.
- < Non-standard Exif tag set by Camera Raw.
- < Olympus Cameras Makernote.
- < Olympus Cameras Makernote.

### 9.382.2.6 getMetadataField()

```
QVariant Digikam::DMetadata::getMetadataField (
    MetadataInfo::Field field ) const
```

See `metadainfo.h` for a specification of the format of the QVariant.

### 9.382.2.7 getXmpKeywords()

```
QStringList Digikam::DMetadata::getXmpKeywords ( ) const
```

Return an empty list if no keyword are set.

### 9.382.2.8 getXmpSubCategories()

```
QStringList Digikam::DMetadata::getXmpSubCategories ( ) const
```

Return an empty list if no sub-category are set.

### 9.382.2.9 getXmpSubjects()

```
QStringList Digikam::DMetadata::getXmpSubjects ( ) const
```

Return an empty list if no subject are set.

### 9.382.2.10 load()

```
bool Digikam::DMetadata::load (
    const QString & filePath,
    bool videoAll = false,
    Backend * backend = nullptr )
```

If backend is non null, return the backend used to populate metadata (Exiv2). See [MetaEngine::Backend](#) enum for details.

### 9.382.2.11 mSecTimeStamp()

```
bool Digikam::DMetadata::mSecTimeStamp (
    const char *const exifTagName,
    int & ms ) const
```

Returns true if data are extracted.

### 9.382.2.12 possibleValuesForEnumField()

```
QMap< int, QString > Digikam::DMetadata::possibleValuesForEnumField (
    MetadataInfo::Field field ) [static]
```

Valid fields are those which are described as "enum from" or "bit mask from" in metadatainfo.h. Int, enum from libMetaEngine

Int, enum from Exif

Int, enum from Exif

Int, enum from Exif

Int, enum from Exif

int, enum from Exif

Int, bit mask from Exif

### 9.382.2.13 removeFromXmpTagStringBag()

```
bool Digikam::DMetadata::removeFromXmpTagStringBag (
    const char *const xmpTagName,
    const QStringList & entriesToRemove ) const
```

Return true if tag entries are no longer contained in metadata. All other entries are preserved.

### 9.382.2.14 removeXmpKeywords()

```
bool Digikam::DMetadata::removeXmpKeywords (
    const QStringList & keywordsToRemove )
```

Return true if keywords are no longer contained in metadata.

### 9.382.2.15 removeXmpSubCategories()

```
bool Digikam::DMetadata::removeXmpSubCategories (
    const QStringList & categoriesToRemove )
```

Return true if subjects are no longer contained in metadata.

### 9.382.2.16 removeXmpSubjects()

```
bool Digikam::DMetadata::removeXmpSubjects (
    const QStringList & subjectsToRemove )
```

Return true if subjects are no longer contained in metadata.

### 9.382.2.17 setItemFacesMap()

```
bool Digikam::DMetadata::setItemFacesMap (
    const QMap< QString, QVariant > & facesPath,
    bool write,
    const QSize & size = QSize() ) const
```

## Parameters

<i>facesPath</i>	The face map to register in metadata based on tags stored in Picassa/Metadatagroup
<i>write</i>	If true all faces will be written, else update mode search if at least a face tag exist and write if true.
<i>size</i>	The size of the area grouping all faces in image.

**9.382.2.18 setXmpKeywords()**

```
bool Digikam::DMetadata::setXmpKeywords (
    const QStringList & newKeywords ) const
```

The existing keywords from image are preserved. The method will compare all new keywords with all already existing keywords to prevent duplicate entries in image. Return true if keywords have been changed in metadata.

**9.382.2.19 setXmpSubCategories()**

```
bool Digikam::DMetadata::setXmpSubCategories (
    const QStringList & newSubCategories ) const
```

The existing sub-categories from image are preserved. The method will compare all new sub-categories with all already existing sub-categories to prevent duplicate entries in image. Return true if sub-categories have been changed in metadata.

**9.382.2.20 setXmpSubjects()**

```
bool Digikam::DMetadata::setXmpSubjects (
    const QStringList & newSubjects ) const
```

The existing subjects from image are preserved. The method will compare all new subject with all already existing subject to prevent duplicate entries in image. Return true if subjects have been changed in metadata.

**9.382.2.21 valueToString()**

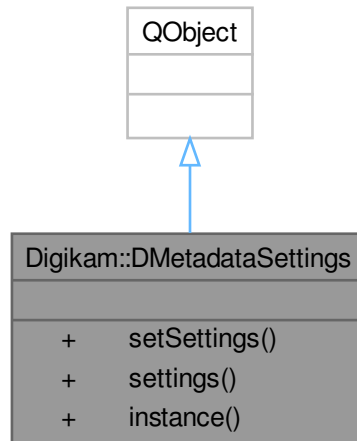
```
QString Digikam::DMetadata::valueToString (
    const QVariant & value,
    MetadataInfo::Field field ) [static]
```

The QVariant must be of the type as specified in metadatainfo.h and as obtained by getMetadataField.



## 9.383 Digikam::DMetadadataSettings Class Reference

Inheritance diagram for Digikam::DMetadadataSettings:



### Signals

- void **signalDMetadadataSettingsChanged** (const [DMetadadataSettingsContainer](#) &current, const [DMetadadataSettingsContainer](#) &previous)
- void **signalSettingsChanged** ()

### Public Member Functions

- void **setSettings** (const [DMetadadataSettingsContainer](#) &settings)  
*Sets the current Metadata settings and writes them to config.*
- [DMetadadataSettingsContainer](#) **settings** () const  
*Returns the current Metadata settings.*

### Static Public Member Functions

- static [DMetadadataSettings](#) \* **instance** ()  
*Global container for Metadata settings.*

### Friends

- class **DMetadadataSettingsCreator**

## 9.383.1 Member Function Documentation

### 9.383.1.1 instance()

```
DMetadataSettings * Digikam::DMetadataSettings::instance ( ) [static]
```

All accessor methods are thread-safe.

## 9.384 Digikam::DMetadataSettingsContainer Class Reference

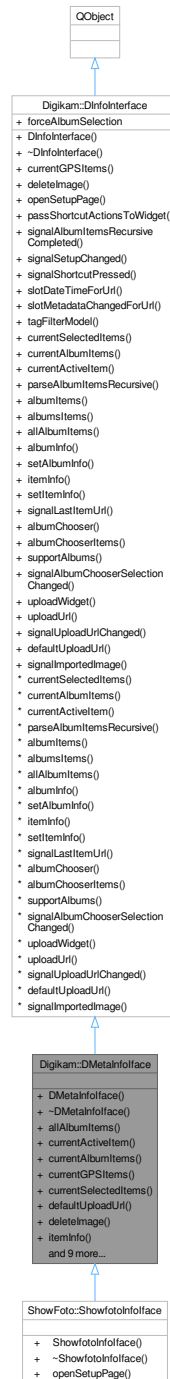
The class [DMetadataSettingsContainer](#) is designed to dynamically add namespaces.

### Public Member Functions

- **DMetadataSettingsContainer** (const [DMetadataSettingsContainer](#) &other)
- void **addMapping** (const QString &key)
- void **defaultValues** ()
  - defaultValues - default namespaces used by digiKam*
- QList< [NamespaceEntry](#) > & **getReadMapping** (const QString &key) const
- QList< [NamespaceEntry](#) > & **getWriteMapping** (const QString &key) const
- QList< QString > **mappingKeys** () const
- [DMetadataSettingsContainer](#) & **operator=** (const [DMetadataSettingsContainer](#) &other)
- void **readFromConfig** (KConfigGroup &group)
- bool **readingAllTags** () const
- void **setReadingAllTags** (bool b)
- void **setUnifyReadWrite** (bool b)
- QString **translateMappingKey** (const QString &key) const
- bool **unifyReadWrite** () const
- void **writeToConfig** (KConfigGroup &group) const

## 9.385 Digikam::DMetaInfoface Class Reference

Inheritance diagram for Digikam::DMetaInfoface:



### Public Member Functions

- **DMetaInfoface** (QObject \*const parent, const QList< QUrl > &lst, const QUrl &currentActive)
- QList< QUrl > [allAlbumItems](#) () const override

- [QUrl currentActiveItem](#) () const override
- [QList< QUrl > currentAlbumItems](#) () const override
- [QList< \[GPSItemContainer\]\(#\) \\* > currentGPSItems](#) () const override
- [QList< QUrl > currentSelectedItems](#) () const override
- *Low level items and albums methods.*
- [QUrl defaultUploadUrl](#) () const override
- *Url to upload new items without to use album selector.*
- void [deleteImage](#) (const [QUrl](#) &url) override
- *Manipulate with item.*
- [DInfoMap itemInfo](#) (const [QUrl](#) &) const override
- void [parseAlbumItemsRecursive](#) () override
- void [setItemInfo](#) (const [QUrl](#) &, const [DInfoMap](#) &) override
- Q\_SIGNAL void [signalItemChanged](#) (const [QUrl](#) &url)
- Q\_SIGNAL void [signalRemoveImageFromAlbum](#) (const [QUrl](#) &)
- Q\_SLOT void [slotDateTimeForUrl](#) (const [QUrl](#) &url, const [QDateTime](#) &dt, bool updModDate) override
- *Slot to call when date time stamp from item is changed.*
- Q\_SLOT void [slotMetadataChangedForUrl](#) (const [QUrl](#) &url) override
- *Slot to call when something in metadata from item is changed.*
- bool [supportAlbums](#) () const override
- [QUrl uploadUrl](#) () const override
- [QWidget](#) \* [uploadWidget](#) ([QWidget](#) \*const parent) const override
- *Album selector view methods (to upload items from an external place).*

## Public Member Functions inherited from [Digikam::DInfoInterface](#)

- [DInfoInterface](#) ([QObject](#) \*const parent)
- virtual void [openSetupPage](#) ([SetupPage](#) page)
- *Open configuration dialog page.*
- virtual [QMap< QString, QString >](#) [passShortcutActionsToWidget](#) ([QWidget](#) \*const) const
- *Pass extra shortcut actions to widget and return prefixes of shortcuts.*
- Q\_SIGNAL void [signalAlbumItemsRecursiveCompleted](#) (const [QList< QUrl >](#) &imageList)
- Q\_SIGNAL void [signalSetupChanged](#) ()
- Q\_SIGNAL void [signalShortcutPressed](#) (const [QString](#) &shortcut, int val)
- virtual [QAbstractItemModel](#) \* [tagFilterModel](#) ()
- *Return an instance of tag filter model if host application support this feature, else null pointer.*
- virtual [QList< QUrl >](#) [albumItems](#) (int) const
- virtual [QList< QUrl >](#) [albumItems](#) (const [DAlbumIDs](#) &) const
- virtual [DInfoMap](#) [albumInfo](#) (int) const
- virtual void [setAlbumInfo](#) (int, const [DInfoMap](#) &) const
- Q\_SIGNAL void [signalLastItemUrl](#) (const [QUrl](#) &)
- virtual [QWidget](#) \* [albumChooser](#) ([QWidget](#) \*const parent) const
- *Albums chooser view methods (to use items from albums before to process).*
- virtual [DAlbumIDs](#) [albumChooserItems](#) () const
- Q\_SIGNAL void [signalAlbumChooserSelectionChanged](#) ()
- Q\_SIGNAL void [signalUploadUrlChanged](#) ()
- Q\_SIGNAL void [signalImportedImage](#) (const [QUrl](#) &)

## Additional Inherited Members

### Public Types inherited from [Digikam::DInfoInterface](#)

- typedef QList< int > **DAlbumIDs**  
*List of [Album](#) ids.*
- typedef QMap< QString, QVariant > **DInfoMap**  
*Map of properties name and value.*
- enum **SetupPage** { **ExifToolPage** = 0 , **ImageQualityPage** }

### Public Attributes inherited from [Digikam::DInfoInterface](#)

- bool **forceAlbumSelection** = false

## 9.385.1 Member Function Documentation

### 9.385.1.1 allAlbumItems()

```
QList< QUrl > Digikam::DMetaInfoIface::allAlbumItems ( ) const [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

### 9.385.1.2 currentActiveItem()

```
QUrl Digikam::DMetaInfoIface::currentActiveItem ( ) const [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

### 9.385.1.3 currentAlbumItems()

```
QList< QUrl > Digikam::DMetaInfoIface::currentAlbumItems ( ) const [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

### 9.385.1.4 currentGPSItems()

```
QList< GPSItemContainer * > Digikam::DMetaInfoIface::currentGPSItems ( ) const [override],  
[virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

### 9.385.1.5 currentSelectedItems()

```
QList< QUrl > Digikam::DMetaInfoIface::currentSelectedItems ( ) const [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

### 9.385.1.6 defaultUploadUrl()

```
QUrl Digikam::DMetaInfoIface::defaultUploadUrl ( ) const [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

### 9.385.1.7 deleteImage()

```
void Digikam::DMetaInfoIface::deleteImage (
    const QUrl & url ) [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

### 9.385.1.8 itemInfo()

```
DMetaInfoIface::DInfoMap Digikam::DMetaInfoIface::itemInfo (
    const QUrl & url ) const [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

### 9.385.1.9 parseAlbumItemsRecursive()

```
void Digikam::DMetaInfoIface::parseAlbumItemsRecursive ( ) [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

### 9.385.1.10 setItemInfo()

```
void Digikam::DMetaInfoIface::setItemInfo (
    const QUrl & url,
    const DInfoMap & map ) [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

### 9.385.1.11 slotDateTimeForUrl()

```
void Digikam::DMetaInfoIface::slotDateTimeForUrl (
    const QUrl & url,
    const QDateTime & dt,
    bool updModDate ) [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

### 9.385.1.12 slotMetadataChangedForUrl()

```
void Digikam::DMetaInfoIface::slotMetadataChangedForUrl (
    const QUrl & url ) [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

**9.385.1.13 supportAlbums()**

```
bool Digikam::DMetaInfoIface::supportAlbums ( ) const [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

**9.385.1.14 uploadUrl()**

```
QUrl Digikam::DMetaInfoIface::uploadUrl ( ) const [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

**9.385.1.15 uploadWidget()**

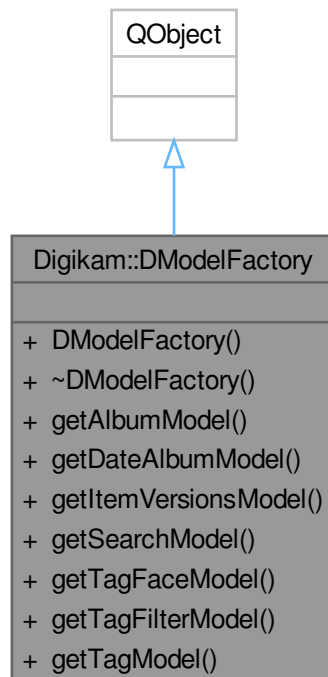
```
QWidget * Digikam::DMetaInfoIface::uploadWidget (
    QWidget *const parent ) const [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

**9.386 Digikam::DModelFactory Class Reference**

This class is simply a factory of all models that build the core of the digikam application.

Inheritance diagram for Digikam::DModelFactory:



### Public Member Functions

- [AlbumModel](#) \* **getAlbumModel** () const
- [DateAlbumModel](#) \* **getDateAlbumModel** () const
- [ItemVersionsModel](#) \* **getItemVersionsModel** () const
- [SearchModel](#) \* **getSearchModel** () const
- [TagModel](#) \* **getTagFaceModel** () const
- [TagModel](#) \* **getTagFilterModel** () const
- [TagModel](#) \* **getTagModel** () const

### 9.386.1 Detailed Description

#### Author

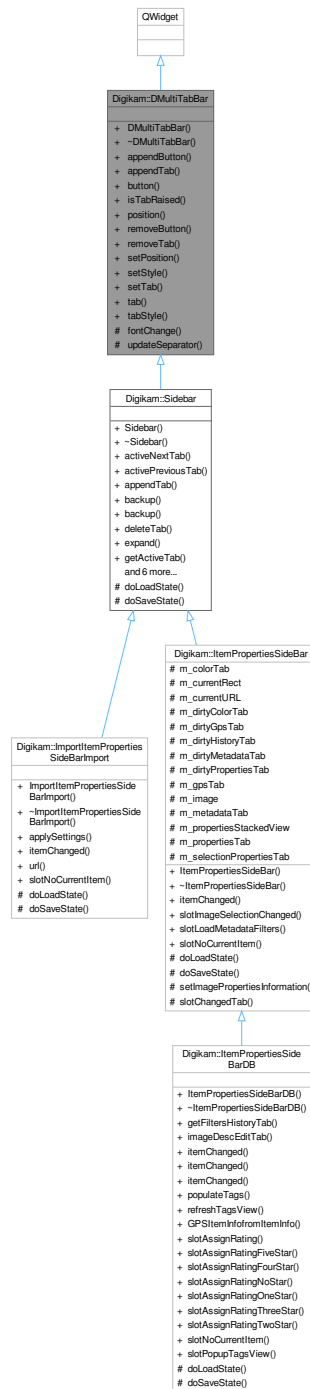
jwienke

## 9.387 Digikam::DMultiTabBar Class Reference

A Widget for horizontal and vertical tabs.



Inheritance diagram for Digikam::DMultiTabBar:



## Public Types

- enum `TextStyle` { `ActiveIconText = 0` , `AllIconsText = 2` }  
*The list of available styles for `DMultiTabBar`.*

## Public Member Functions

- DMultiTabBar** (Qt::Edge pos, QWidget \*const parent=nullptr)

- void [appendButton](#) (const QIcon &pic, int id=-1, QMenu \*const popup=nullptr, const QString &not\_used\_↔ yet=QString())  
*append a new button to the button area.*
- void [appendTab](#) (const QIcon &pic, int id=-1, const QString &text=QString())  
*append a new tab to the tab area.*
- [DMultiTabBarButton](#) \* **button** (int id) const  
*get a pointer to a button within the button area identified by its ID*
- bool **isTabRaised** (int id) const  
*return the state of a tab, identified by its ID*
- Qt::Edge [position](#) () const  
*get the tabbar position.*
- void **removeButton** (int id)  
*remove a button with the given ID*
- void **removeTab** (int id)  
*remove a tab with a given ID*
- void [setPosition](#) (Qt::Edge pos)  
*set the real position of the widget.*
- void **setStyle** ([TextStyle](#) style)  
*set the display style of the tabs*
- void [setTab](#) (int id, bool state)  
*set a tab to "raised"*
- [DMultiTabBarTab](#) \* **tab** (int id) const  
*get a pointer to a tab within the tab area, identified by its ID*
- [TextStyle](#) [tabStyle](#) () const  
*get the display style of the tabs*

### Protected Member Functions

- virtual void **fontChange** (const QFont &)
- void **updateSeparator** ()

### Friends

- class **DMultiTabBarButton**

## 9.387.1 Member Enumeration Documentation

### 9.387.1.1 TextStyle

enum [Digikam::DMultiTabBar::TextStyle](#)

#### Enumerator

ActiveIconText	Always shows icon, only show the text of active tabs.
AllIconsText	Always shows the text and icons.

## 9.387.2 Member Function Documentation

### 9.387.2.1 appendButton()

```
void Digikam::DMultiTabBar::appendButton (
    const QIcon & pic,
    int id = -1,
    QMenu *const popup = nullptr,
    const QString & not_used_yet = QString() )
```

The button can later on be accessed with button(ID) eg for connecting signals to it

#### Parameters

<i>pic</i>	a icon for the button
<i>id</i>	an arbitrary ID value. It will be emitted in the clicked signal for identifying the button if more than one button is connected to a signals.
<i>popup</i>	A popup menu which should be displayed if the button is clicked
<i>not_used_yet</i>	will be used for a popup text in the future

### 9.387.2.2 appendTab()

```
void Digikam::DMultiTabBar::appendTab (
    const QIcon & pic,
    int id = -1,
    const QString & text = QString() )
```

It can be accessed later on with tabb(id);

#### Parameters

<i>pic</i>	a icon for the tab
<i>id</i>	an arbitrary ID which can be used later on to identify the tab
<i>text</i>	if a mode with text is used it will be the tab text, otherwise a mouse over hint

### 9.387.2.3 position()

```
Qt::Edge Digikam::DMultiTabBar::position ( ) const
```

#### Returns

position

### 9.387.2.4 setPosition()

```
void Digikam::DMultiTabBar::setPosition (
    Qt::Edge pos )
```

**Parameters**

<i>pos</i>	if the mode is horizontal, only use top, bottom, if it is vertical use left or right
------------	--

**9.387.2.5 setTab()**

```
void Digikam::DMultiTabBar::setTab (
    int id,
    bool state )
```

**Parameters**

<i>id</i>	The ID of the tab to manipulate
<i>state</i>	true == activated/raised, false == not active

**9.387.2.6 tabStyle()**

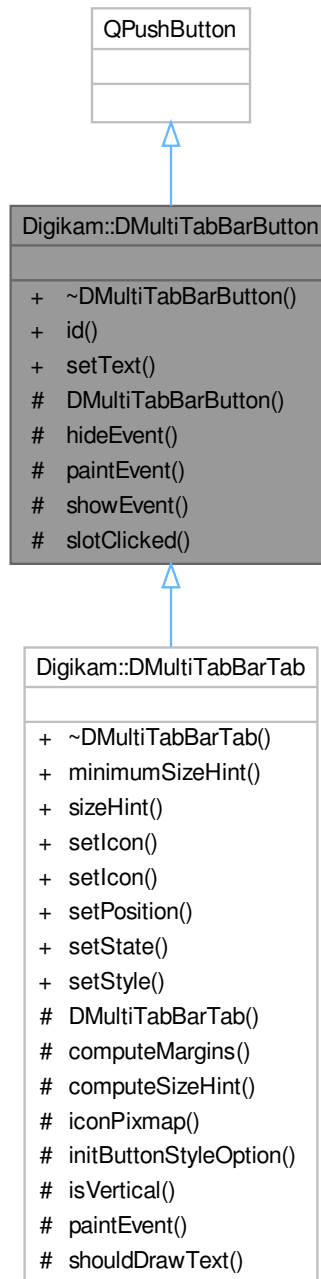
```
DMultiTabBar::TextStyle Digikam::DMultiTabBar::tabStyle ( ) const
```

**Returns**

display style

## 9.388 Digikam::DMultiTabBarButton Class Reference

Inheritance diagram for Digikam::DMultiTabBarButton:



### Public Slots

- void **setText** (const QString &text)

## Signals

- void `signalClicked` (int id)  
*this is emitted if the button is clicked*

## Public Member Functions

- int `id` () const

## Protected Slots

- virtual void `slotClicked` ()

## Protected Member Functions

- `DMultiTabBarButton` (const QIcon &pic, const QString &, int id, QWidget \*const parent)
- void `hideEvent` (QHideEvent \*) override
- void `paintEvent` (QPaintEvent \*) override
- void `showEvent` (QShowEvent \*) override

## Friends

- class `DMultiTabBar`

## 9.388.1 Member Function Documentation

### 9.388.1.1 `signalClicked`

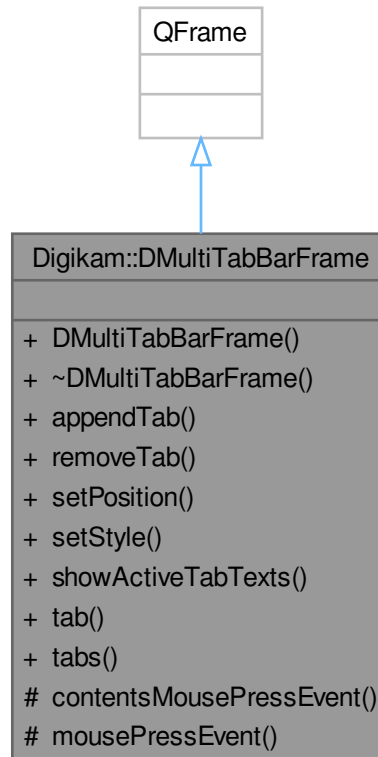
```
void Digikam::DMultiTabBarButton::signalClicked (  
    int id ) [signal]
```

#### Parameters

<i>id</i>	the ID identifying the button
-----------	-------------------------------

## 9.389 Digikam::DMultiTabBarFrame Class Reference

Inheritance diagram for Digikam::DMultiTabBarFrame:



### Public Member Functions

- **DMultiTabBarFrame** (QWidget \*const parent, Qt::Edge pos)
- void **appendTab** (const QIcon &, int=-1, const QString &=QString())
- void **removeTab** (int)
- void **setPosition** (Qt::Edge pos)
- void **setStyle** (DMultiTabBar::TextStyle style)
- void **showActiveTabTexts** (bool show)
- [DMultiTabBarTab](#) \* **tab** (int) const
- QList< [DMultiTabBarTab](#) \* > \* **tabs** ()

### Protected Member Functions

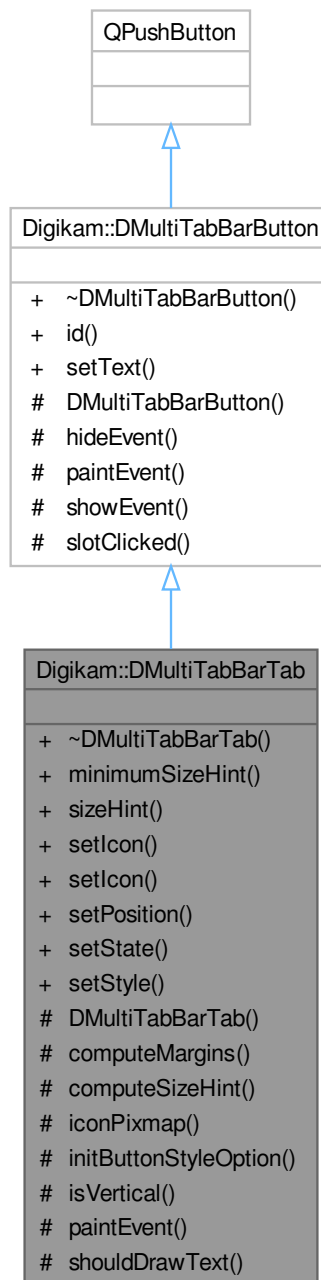
- virtual void **contentsMouseEvent** (QMouseEvent \*)  
*Reimplemented from QScrollView in order to ignore all mouseEvents on the viewport, so that the parent can handle them.*
- void **mousePressEvent** (QMouseEvent \*) override

**Friends**

- class **DMultiTabBar**

**9.390 Digikam::DMultiTabBarTab Class Reference**

Inheritance diagram for Digikam::DMultiTabBarTab:





## Public Slots

- void **setIcon** (const QIcon &)
- void **setIcon** (const QString &)
- void **setPosition** (Qt::Edge)
 

*this is used internally, but can be used by the user.*
- void **setState** (bool state)
 

*set the active state of the tab*
- void **setStyle** (DMultiTabBar::TextStyle)
 

*this is used internally, but can be used by the user.*

## Public Slots inherited from [Digikam::DMultiTabBarButton](#)

- void **setText** (const QString &text)

## Public Member Functions

- QSize **minimumSizeHint** () const override
- QSize **sizeHint** () const override

## Public Member Functions inherited from [Digikam::DMultiTabBarButton](#)

- int **id** () const

## Protected Member Functions

- **DMultiTabBarTab** (const QIcon &pic, const QString &, int id, QWidget \*const parent, Qt::Edge pos, [DMultiTabBar::TextStyle](#) style)
 

*This class should never be created except with the `appendTab` call of [DMultiTabBar](#).*
- void **computeMargins** (int \*hMargin, int \*vMargin) const
- QSize **computeSizeHint** (bool withText) const
- QPixmap **iconPixmap** () const
- void **initButtonStyleOption** (QStyleOptionToolButton \*opt) const
- bool **isVertical** () const
- void **paintEvent** (QPaintEvent \*) override
- bool **shouldDrawText** () const

## Protected Member Functions inherited from [Digikam::DMultiTabBarButton](#)

- **DMultiTabBarButton** (const QIcon &pic, const QString &, int id, QWidget \*const parent)
- void **hideEvent** (QHideEvent \*) override
- void **paintEvent** (QPaintEvent \*) override
- void **showEvent** (QShowEvent \*) override

## Friends

- class **DMultiTabBarFrame**

## Additional Inherited Members

### Signals inherited from [Digikam::DMultiTabBarButton](#)

- void [signalClicked](#) (int id)  
*this is emitted if the button is clicked*

### Protected Slots inherited from [Digikam::DMultiTabBarButton](#)

- virtual void [slotClicked](#) ()

## 9.390.1 Member Function Documentation

### 9.390.1.1 setPosition

```
void Digikam::DMultiTabBarTab::setPosition (
    Qt::Edge pos ) [slot]
```

It the according call of [DMultiTabBar](#) is invoked though this modifications will be overwritten

### 9.390.1.2 setState

```
void Digikam::DMultiTabBarTab::setState (
    bool state ) [slot]
```

#### Parameters

<i>state</i>	true==active false==not active
--------------	--------------------------------

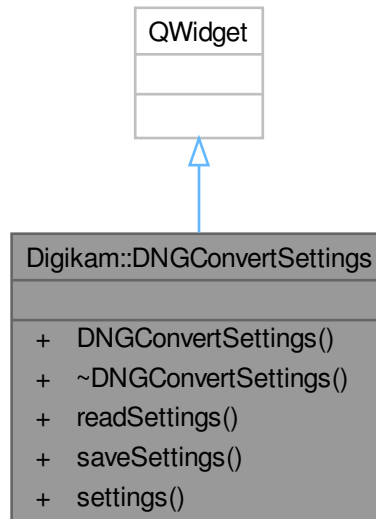
### 9.390.1.3 setStyle

```
void Digikam::DMultiTabBarTab::setStyle (
    DMultiTabBar::TextStyle style ) [slot]
```

It the according call of [DMultiTabBar](#) is invoked though this modifications will be overwritten

## 9.391 Digikam::DNGConvertSettings Class Reference

Inheritance diagram for Digikam::DNGConvertSettings:



### Signals

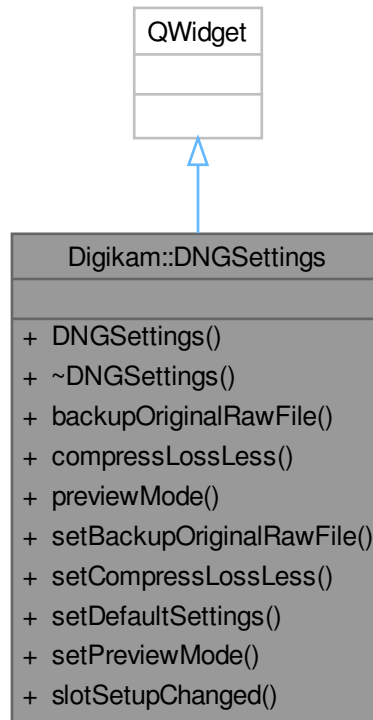
- void **signalDownloadNameChanged** ()

### Public Member Functions

- **DNGConvertSettings** (`QWidget *const parent=nullptr`)
- void **readSettings** (`const KConfigGroup &group`)
- void **saveSettings** (`KConfigGroup &group`)
- void **settings** ([DownloadSettings](#) `*const settings`)

## 9.392 Digikam::DNGSettings Class Reference

Inheritance diagram for Digikam::DNGSettings:



### Public Slots

- void **slotSetupChanged** ()  
*To handle changes from host application [Setup](#) dialog.*

### Signals

- void **signalSettingsChanged** ()
- void **signalSetupExifTool** ()

### Public Member Functions

- **DNGSettings** (QWidget \*const parent=nullptr)
- bool **backupOriginalRawFile** () const
- bool **compressLossLess** () const
- int **previewMode** () const
- void **setBackupOriginalRawFile** (bool b)
- void **setCompressLossLess** (bool b)
- void **setDefaultSettings** ()
- void **setPreviewMode** (int mode)

## 9.393 Digikam::DNGWriter Class Reference

### Public Types

- enum `ConvertError` {  
`PROCESS_CONTINUE = 1` , `PROCESS_COMPLETE = 0` , `PROCESS_FAILED = -1` , `PROCESS_CANCELED = -2` ,  
`FILE_NOT_SUPPORTED = -3` , `DNG_SDK_INTERNAL_ERROR = -4` }
- enum `JPEGPreview` { `NONE = 0` , `MEDIUM` , `FULL_SIZE` }

### Public Member Functions

- bool `backupOriginalRawFile` () const
- void `cancel` ()
- bool `compressLossLess` () const
- int `convert` ()
- QString `inputFile` () const
- QString `outputFile` () const
- int `previewMode` () const
- void `reset` ()
- void `setBackupOriginalRawFile` (bool b)
- void `setCompressLossLess` (bool b)
- void `setInputFile` (const QString &filePath)
- void `setOutputFile` (const QString &filePath)
- void `setPreviewMode` (int mode)
- void `setUpdateFileDate` (bool b)
- bool `updateFileDate` () const

### Static Public Member Functions

- static QString `dngSdkVersion` ()
- static QString `xmpSdkVersion` ()

## 9.393.1 Member Enumeration Documentation

### 9.393.1.1 ConvertError

enum `Digikam::DNGWriter::ConvertError`

#### Enumerator

<code>PROCESS_CONTINUE</code>	Current stages is done.
<code>PROCESS_COMPLETE</code>	All stages done.
<code>PROCESS_FAILED</code>	A failure happen while processing.
<code>PROCESS_CANCELED</code>	User has canceled processing.
<code>FILE_NOT_SUPPORTED</code>	Raw file format is not supported by converter.
<code>DNG_SDK_INTERNAL_ERROR</code>	Adobe DNG SDK has generated an error while processing.

### 9.393.1.2 JPEGPreview

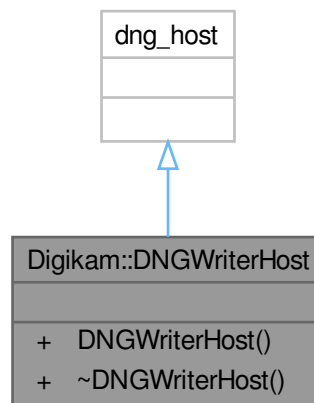
enum `Digikam::DNGWriter::JPEGPreview`

#### Enumerator

NONE	No preview will be generated.
MEDIUM	A medium size preview will be generated.
FULL_SIZE	A full size preview will be generated.

## 9.394 Digikam::DNGWriterHost Class Reference

Inheritance diagram for `Digikam::DNGWriterHost`:

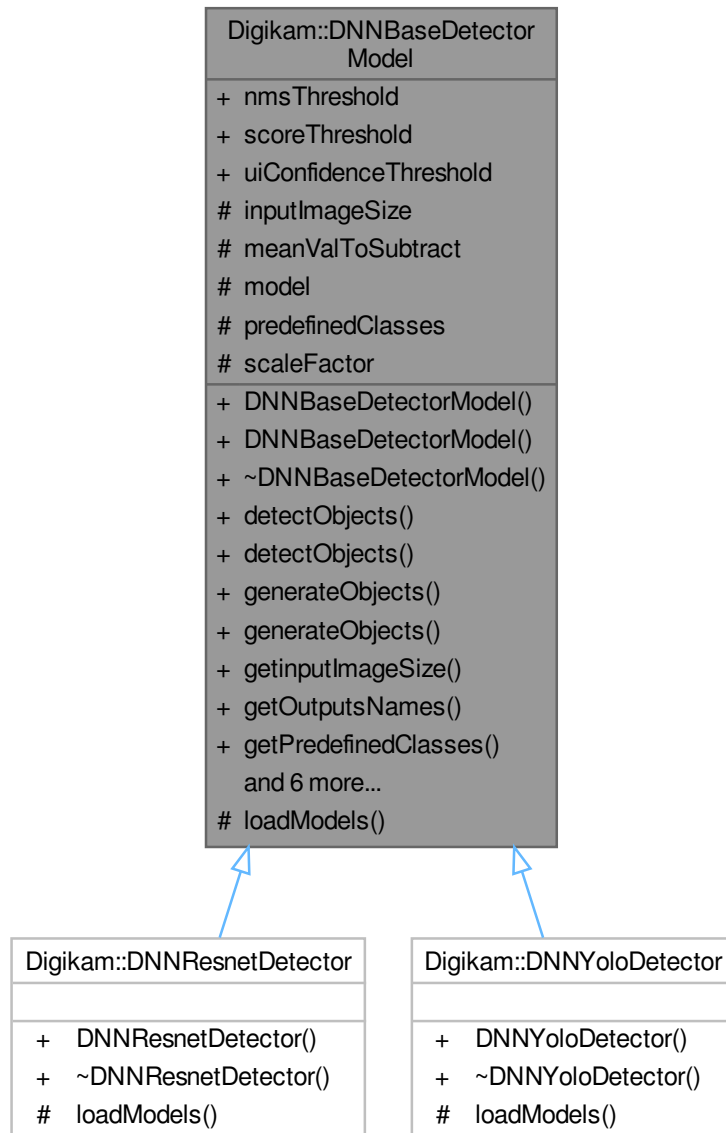


#### Public Member Functions

- **DNGWriterHost** (`DNGWriter::Private *const priv, dng_memory_allocator *const allocator=nullptr`)

## 9.395 Digikam::DNNBaseDetectorModel Class Reference

Inheritance diagram for Digikam::DNNBaseDetectorModel:



### Public Member Functions

- **DNNBaseDetectorModel** (float scale, const cv::Scalar &val, const cv::Size &inputImgSize)
- virtual QHash< QString, QVector< QRect > > **detectObjects** (const cv::Mat &inputImage)  
*detectObjects* return the predicted objects and localization as well (if we use deeplearning for object detection like YOLO, etc) otherwise the map whose the key is the objects name and their values are empty.
- virtual QList< QHash< QString, QVector< QRect > > > **detectObjects** (const std::vector< cv::Mat > &inputBatchImages)

- detectObjects in batch images (fixed batch size).*
- QList< QString > **generateObjects** (const cv::Mat &inputImage)
  - generateObjects in one image return just the predicted objects without locations of objects using for the assignment tagging names.*
- QList< QList< QString > > **generateObjects** (const std::vector< cv::Mat > &inputImage)
  - generateObjects in batch images return just the predicted objects without locations of objects using for the assignment tagging names.*
- cv::Size **getInputImageSize** () const
  - Return the input Image Size from Deep NN model.*
- std::vector< cv::String > **getOutputsNames** () const
- virtual QList< QString > **getPredefinedClasses** () const
  - Get predefined objects according to selected model.*
- QList< QString > **loadDetectionClasses** ()
- virtual QMap< QString, QVector< QRect > > **postprocess** (const cv::Mat &inputImage, const cv::Mat &out) const =0
- QList< QMap< QString, QVector< QRect > > > **postprocess** (const std::vector< cv::Mat > &inputBatchImages, const std::vector< cv::Mat > &outs) const
- std::vector< cv::Mat > **preprocess** (const cv::Mat &inputImage)
- std::vector< cv::Mat > **preprocess** (const std::vector< cv::Mat > &inputBatchImages)
- double **showInferenceTime** ()

### Static Public Attributes

- static float **nmsThreshold** = 0.4F
  - Threshold for nms suppression.*
- static float **scoreThreshold** = 0.45F
  - Threshold for class detection score.*
- static int **uiConfidenceThreshold** = DNN\_MODEL\_THRESHOLD\_NOT\_SET
  - Threshold for bbox detection. It can be init and changed in the GUI.*

### Protected Member Functions

- virtual bool **loadModels** ()=0

### Protected Attributes

- cv::Size **inputImageSize**
- cv::Scalar **meanValToSubtract**
- [DNNModelBase](#) \* **model** = nullptr
- QList< QString > **predefinedClasses**
- float **scaleFactor** = 1.0F

## 9.395.1 Member Data Documentation

### 9.395.1.1 uiConfidenceThreshold

```
int Digikam::DNNBaseDetectorModel::uiConfidenceThreshold = DNN_MODEL_THRESHOLD_NOT_SET [static]
```

setting 1000 will use the value from dnnmodels.conf if passed in



## 9.396 Digikam::DNNFaceDetectorBase Class Reference

Inheritance diagram for Digikam::DNNFaceDetectorBase:



### Public Member Functions

- **DNNFaceDetectorBase** (float scale, const cv::Scalar &val, const cv::Size &inputImgSize)
- virtual void **detectFaces** (const cv::Mat &inputImage, const cv::Size &paddedSize, std::vector< cv::Rect > &detectedBboxes)=0
- cv::Size **nnInputSizeRequired** () const
- virtual void **setFaceDetectionSize** ([FaceScanSettings::FaceDetectionSize](#) faceSize)

### Static Public Attributes

- static float **nmsThreshold** = 0.4F  
*Threshold for nms suppression.*
- static int **uiConfidenceThreshold** = DNN\_MODEL\_THRESHOLD\_NOT\_SET  
*Threshold for bbox detection. It can be init and changed in the GUI.*

### Protected Member Functions

- void **correctBbox** (cv::Rect &bbox, const cv::Size &paddedSize) const
- void **selectBbox** (const cv::Size &paddedSize, float confidence, int left, int right, int top, int bottom, std::vector< float > &goodConfidences, std::vector< cv::Rect > &goodBoxes, std::vector< float > &doubtConfidences, std::vector< cv::Rect > &doubtBoxes) const

## Protected Attributes

- `cv::Size inputImageSize = cv::Size(300, 300)`
- `cv::Scalar meanValToSubtract = cv::Scalar(0.0, 0.0, 0.0)`
- `DNNModelBase * model = nullptr`
- `float scaleFactor = 1.0F`

## 9.396.1 Member Function Documentation

### 9.396.1.1 selectBbox()

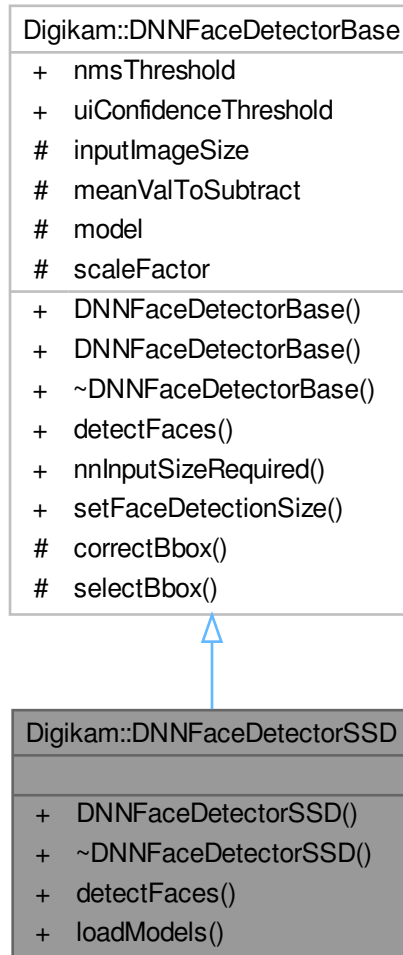
```
void Digikam::DNNFaceDetectorBase::selectBbox (
    const cv::Size & paddedSize,
    float confidence,
    int left,
    int right,
    int top,
    int bottom,
    std::vector< float > & goodConfidences,
    std::vector< cv::Rect > & goodBoxes,
    std::vector< float > & doubtConfidences,
    std::vector< cv::Rect > & doubtBoxes ) const [protected]
```

Classify bounding boxes detected. Good bounding boxes are defined as boxes that reside within the non-padded zone or those that are out only for min of (10% of padded range, 10% of bbox dim).

Bad bounding boxes are defined as boxes that have at maximum 25% of each dimension out of non-padded zone.

## 9.397 Digikam::DNNFaceDetectorSSD Class Reference

Inheritance diagram for Digikam::DNNFaceDetectorSSD:



### Public Member Functions

- void `detectFaces` (const cv::Mat &inputImage, const cv::Size &paddedSize, std::vector< cv::Rect > &detectedBboxes) override
- bool `loadModels` ()

### Public Member Functions inherited from [Digikam::DNNFaceDetectorBase](#)

- `DNNFaceDetectorBase` (float scale, const cv::Scalar &val, const cv::Size &inputImgSize)
- cv::Size `nnInputSizeRequired` () const
- virtual void `setFaceDetectionSize` ([FaceScanSettings::FaceDetectionSize](#) faceSize)

## Additional Inherited Members

### Static Public Attributes inherited from [Digikam::DNNFaceDetectorBase](#)

- static float **nmsThreshold** = 0.4F  
*Threshold for nms suppression.*
- static int **uiConfidenceThreshold** = DNN\_MODEL\_THRESHOLD\_NOT\_SET  
*Threshold for bbox detection. It can be init and changed in the GUI.*

### Protected Member Functions inherited from [Digikam::DNNFaceDetectorBase](#)

- void **correctBbox** (cv::Rect &bbox, const cv::Size &paddedSize) const
- void **selectBbox** (const cv::Size &paddedSize, float confidence, int left, int right, int top, int bottom, std::vector< float > &goodConfidences, std::vector< cv::Rect > &goodBoxes, std::vector< float > &doubtConfidences, std::vector< cv::Rect > &doubtBoxes) const

### Protected Attributes inherited from [Digikam::DNNFaceDetectorBase](#)

- cv::Size **inputImageSize** = cv::Size(300, 300)
- cv::Scalar **meanValToSubtract** = cv::Scalar(0.0, 0.0, 0.0)
- [DNNModelBase](#) \* **model** = nullptr
- float **scaleFactor** = 1.0F

## 9.397.1 Member Function Documentation

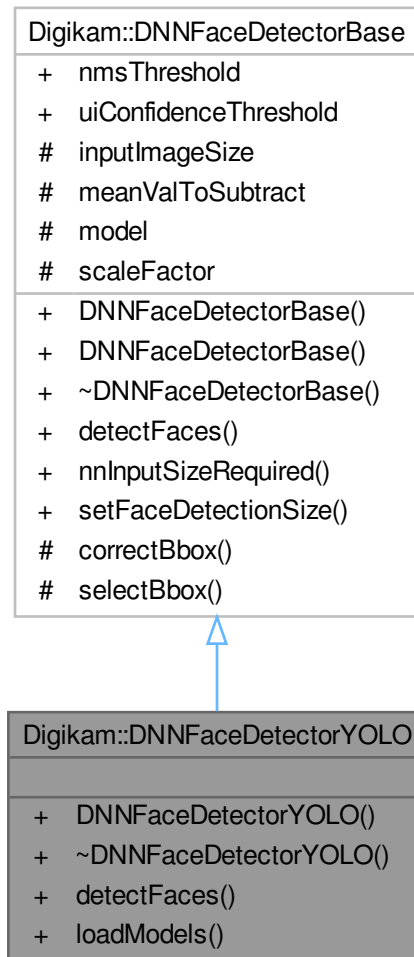
### 9.397.1.1 detectFaces()

```
void Digikam::DNNFaceDetectorSSD::detectFaces (
    const cv::Mat & inputImage,
    const cv::Size & paddedSize,
    std::vector< cv::Rect > & detectedBboxes ) [override], [virtual]
```

Implements [Digikam::DNNFaceDetectorBase](#).

## 9.398 Digikam::DNNFaceDetectorYOLO Class Reference

Inheritance diagram for Digikam::DNNFaceDetectorYOLO:



### Public Member Functions

- void `detectFaces` (const cv::Mat &inputImage, const cv::Size &paddedSize, std::vector< cv::Rect > &detectedBboxes) override
- bool `loadModels` ()

### Public Member Functions inherited from [Digikam::DNNFaceDetectorBase](#)

- `DNNFaceDetectorBase` (float scale, const cv::Scalar &val, const cv::Size &inputImgSize)
- cv::Size `nnInputSizeRequired` () const
- virtual void `setFaceDetectionSize` ([FaceScanSettings::FaceDetectionSize](#) faceSize)

## Additional Inherited Members

### Static Public Attributes inherited from [Digikam::DNNFaceDetectorBase](#)

- static float **nmsThreshold** = 0.4F  
*Threshold for nms suppression.*
- static int **uiConfidenceThreshold** = DNN\_MODEL\_THRESHOLD\_NOT\_SET  
*Threshold for bbox detection. It can be init and changed in the GUI.*

### Protected Member Functions inherited from [Digikam::DNNFaceDetectorBase](#)

- void **correctBbox** (cv::Rect &bbox, const cv::Size &paddedSize) const
- void **selectBbox** (const cv::Size &paddedSize, float confidence, int left, int right, int top, int bottom, std::vector< float > &goodConfidences, std::vector< cv::Rect > &goodBoxes, std::vector< float > &doubtConfidences, std::vector< cv::Rect > &doubtBoxes) const

### Protected Attributes inherited from [Digikam::DNNFaceDetectorBase](#)

- cv::Size **inputImageSize** = cv::Size(300, 300)
- cv::Scalar **meanValToSubtract** = cv::Scalar(0.0, 0.0, 0.0)
- [DNNModelBase](#) \* **model** = nullptr
- float **scaleFactor** = 1.0F

## 9.398.1 Member Function Documentation

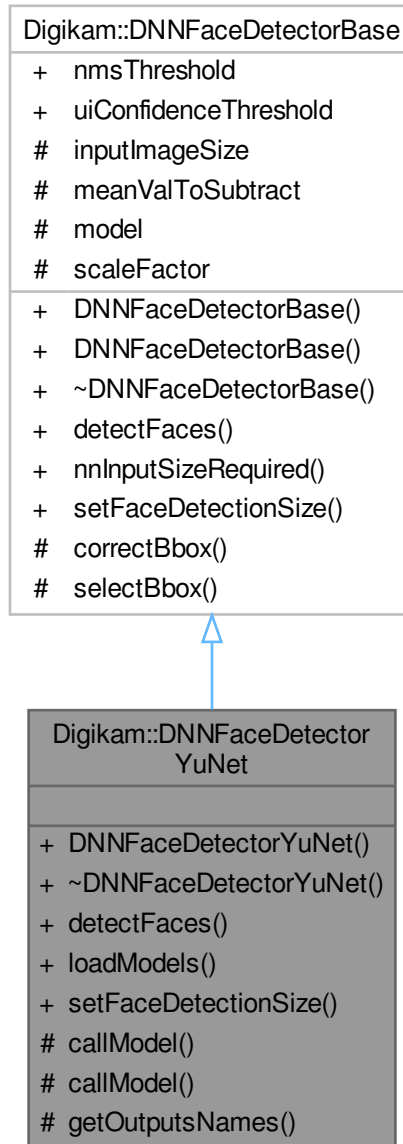
### 9.398.1.1 detectFaces()

```
void Digikam::DNNFaceDetectorYOLO::detectFaces (
    const cv::Mat & inputImage,
    const cv::Size & paddedSize,
    std::vector< cv::Rect > & detectedBboxes ) [override], [virtual]
```

Implements [Digikam::DNNFaceDetectorBase](#).

## 9.399 Digikam::DNNFaceDetectorYuNet Class Reference

Inheritance diagram for Digikam::DNNFaceDetectorYuNet:



### Public Member Functions

- void `detectFaces` (const cv::Mat &inputImage, const cv::Size &paddedSize, std::vector< cv::Rect > &detectedBboxes) override
- bool `loadModels` ()
- virtual void `setFaceDetectionSize` (`FaceScanSettings::FaceDetectionSize` faceSize) override

## Public Member Functions inherited from [Digikam::DNNFaceDetectorBase](#)

- **DNNFaceDetectorBase** (float scale, const cv::Scalar &val, const cv::Size &inputImgSize)
- cv::Size **nnInputSizeRequired** () const

## Protected Member Functions

- cv::Mat **callModel** (const cv::Mat &inputImage)
- cv::UMat **callModel** (const cv::UMat &inputImage)
- std::vector< cv::String > **getOutputsNames** () const

## Protected Member Functions inherited from [Digikam::DNNFaceDetectorBase](#)

- void **correctBbox** (cv::Rect &bbox, const cv::Size &paddedSize) const
- void **selectBbox** (const cv::Size &paddedSize, float confidence, int left, int right, int top, int bottom, std::vector< float > &goodConfidences, std::vector< cv::Rect > &goodBoxes, std::vector< float > &doubtConfidences, std::vector< cv::Rect > &doubtBoxes) const

## Friends

- class **FacePipelineDetectRecognize**

## Additional Inherited Members

## Static Public Attributes inherited from [Digikam::DNNFaceDetectorBase](#)

- static float **nmsThreshold** = 0.4F  
*Threshold for nms suppression.*
- static int **uiConfidenceThreshold** = DNN\_MODEL\_THRESHOLD\_NOT\_SET  
*Threshold for bbox detection. It can be init and changed in the GUI.*

## Protected Attributes inherited from [Digikam::DNNFaceDetectorBase](#)

- cv::Size **inputImageSize** = cv::Size(300, 300)
- cv::Scalar **meanValToSubtract** = cv::Scalar(0.0, 0.0, 0.0)
- [DNNModelBase](#) \* **model** = nullptr
- float **scaleFactor** = 1.0F

## 9.399.1 Member Function Documentation

### 9.399.1.1 detectFaces()

```
void Digikam::DNNFaceDetectorYuNet::detectFaces (
    const cv::Mat & inputImage,
    const cv::Size & paddedSize,
    std::vector< cv::Rect > & detectedBboxes ) [override], [virtual]
```

Implements [Digikam::DNNFaceDetectorBase](#).



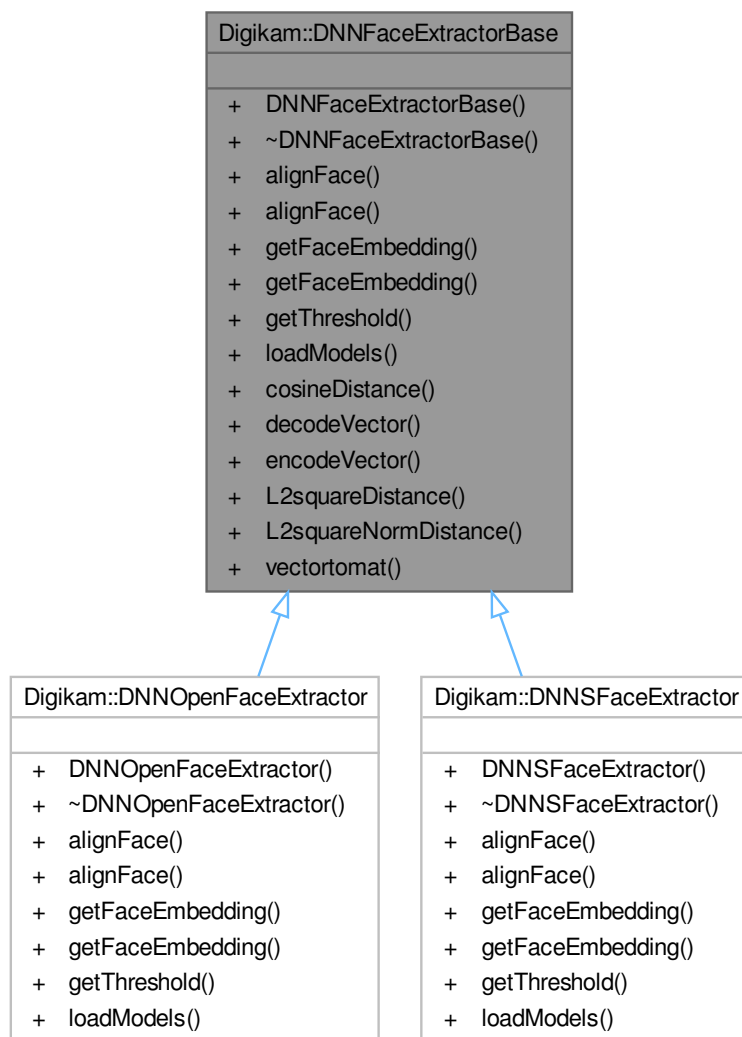
## 9.399.1.2 setFaceDetectionSize()

```
void Digikam::DNNFaceDetectorYuNet::setFaceDetectionSize (
    FaceScanSettings::FaceDetectionSize faceSize ) [override], [virtual]
```

Reimplemented from [Digikam::DNNFaceDetectorBase](#).

## 9.400 Digikam::DNNFaceExtractorBase Class Reference

Inheritance diagram for Digikam::DNNFaceExtractorBase:



## Public Member Functions

- virtual cv::Mat **alignFace** (const cv::Mat &inputImage) const =0
- virtual cv::UMat **alignFace** (const cv::UMat &inputImage) const =0
- virtual cv::Mat **getFaceEmbedding** (const cv::Mat &faceImage)=0
- virtual cv::UMat **getFaceEmbedding** (const cv::UMat &faceImage)=0
- virtual float **getThreshold** (int uiThreshold=DNN\_MODEL\_THRESHOLD\_NOT\_SET) const =0  
*cover the UI threshold to a float using the conversion factor built into the model*
- virtual bool **loadModels** ()=0  
*Read pretrained neural network for face recognition.*

## Static Public Member Functions

- static double **cosineDistance** (const std::vector< float > &v1, const std::vector< float > &v2)  
*Calculate different between 2 vectors.*
- static std::vector< float > **decodeVector** (const QJsonArray &json)
- static QJsonArray **encodeVector** (const std::vector< float > &vector)
- static double **L2squareDistance** (const std::vector< float > &v1, const std::vector< float > &v2)
- static double **L2squareNormDistance** (const std::vector< float > &v1, const std::vector< float > &v2)
- static cv::Mat **vectortomat** (const std::vector< float > &vector)  
*Convert face embedding between different formats.*

## 9.400.1 Member Function Documentation

### 9.400.1.1 getThreshold()

```
virtual float Digikam::DNNFaceExtractorBase::getThreshold (
    int uiThreshold = DNN_MODEL_THRESHOLD_NOT_SET ) const [pure virtual]
```

Implemented in [Digikam::DNNOpenFaceExtractor](#), and [Digikam::DNNSFaceExtractor](#).

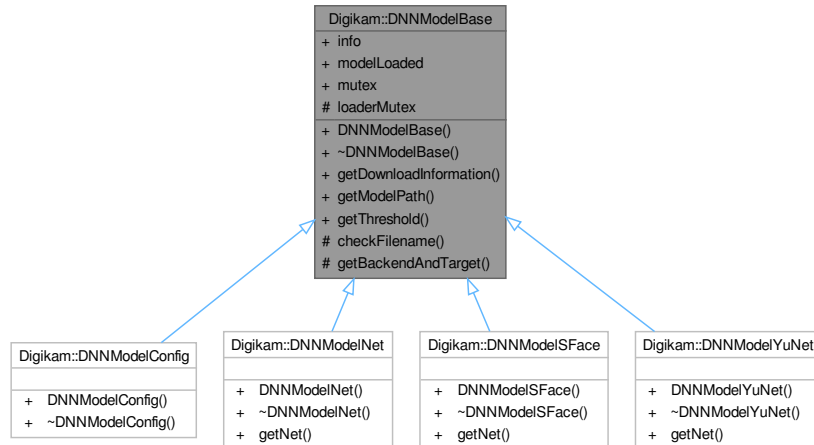
### 9.400.1.2 loadModels()

```
virtual bool Digikam::DNNFaceExtractorBase::loadModels ( ) [pure virtual]
```

Implemented in [Digikam::DNNOpenFaceExtractor](#), and [Digikam::DNNSFaceExtractor](#).

## 9.401 Digikam::DNNModelBase Class Reference

Inheritance diagram for Digikam::DNNModelBase:



### Public Member Functions

- **DNNModelBase** (const [DNNModelInfoContainer](#) &\_info)
- **DownloadInfo** **getDownloadInformation** () const
- const QString **getModelPath** () const  
*Return path to the model, or null string if path cannot be found.*
- float **getThreshold** (int uiThreshold=DNN\_MODEL\_THRESHOLD\_NOT\_SET) const  
*input: uiThreshold is the slider value from the UI.*

### Public Attributes

- const [DNNModelInfoContainer](#) **info**  
*information about the model.*
- bool **modelLoaded** = false  
*check if the model has been loaded.*
- QMutex **mutex**  
*mutex to sigle-thread model during critical processing functions.*

### Protected Member Functions

- bool **checkFilename** () const
- const QPair< int, int > **getBackendAndTarget** () const

### Protected Attributes

- QMutex **loaderMutex**

## 9.401.1 Member Function Documentation

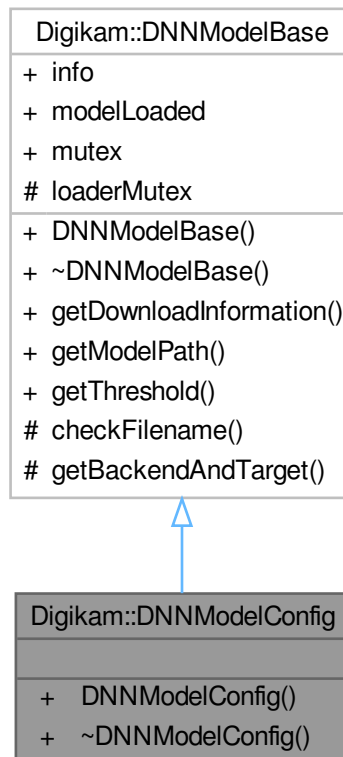
### 9.401.1.1 getThreshold()

```
float Digikam::DNNModelBase::getThreshold (
    int uiThreshold = DNN_MODEL_THRESHOLD_NOT_SET ) const
```

return: float threshold to be used by processing ([FaceDetector](#), FaceRecognizer, etc...).

## 9.402 Digikam::DNNModelConfig Class Reference

Inheritance diagram for Digikam::DNNModelConfig:



### Public Member Functions

- `DNNModelConfig` (const [DNNModelInfoContainer](#) &\_info)

## Public Member Functions inherited from Digikam::DNNModelBase

- **DNNModelBase** (const [DNNModelInfoContainer](#) &\_info)
- [DownloadInfo](#) **getDownloadInformation** () const
- const QString **getModelPath** () const  
*Return path to the model, or null string if path cannot be found.*
- float **getThreshold** (int uiThreshold=DNN\_MODEL\_THRESHOLD\_NOT\_SET) const  
*input: uiThreshold is the slider value from the UI.*

## Additional Inherited Members

## Public Attributes inherited from Digikam::DNNModelBase

- const [DNNModelInfoContainer](#) **info**  
*information about the model.*
- bool **modelLoaded** = false  
*check if the model has been loaded.*
- QMutex **mutex**  
*mutex to sigle-thread model during critical processing functions.*

## Protected Member Functions inherited from Digikam::DNNModelBase

- bool **checkFilename** () const
- const QPair< int, int > **getBackendAndTarget** () const

## Protected Attributes inherited from Digikam::DNNModelBase

- QMutex **loaderMutex**

## 9.403 Digikam::DNNModelInfoContainer Class Reference

### Public Member Functions

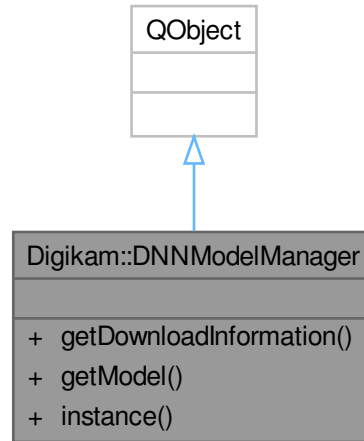
- **DNNModelInfoContainer** (const [DNNModelInfoContainer](#) &)
- **DNNModelInfoContainer** (const QString &\_displayName, const QString &\_fileName, const DNNModelUsageList &\_usage, const QVersionNumber &\_minVersion, const QString &\_downloadPath, const QString &\_sha256, const qint64 &\_fileSize, int \_defaultThreshold, int \_minUsableThreshold, int \_maxUsableThreshold, DNNLoaderType \_loaderType, const QString &\_classList, const QString &\_configName, const cv::Scalar &\_meanValToSubtract, int \_imageSize)
- [DNNModelInfoContainer](#) & **operator=** (const [DNNModelInfoContainer](#) &)
- [DNNModelInfoContainer](#) & **operator=** ([DNNModelInfoContainer](#) &&)
- bool **operator==** (const [DNNModelInfoContainer](#) &t) const

## Public Attributes

- QString **classList**  
*Name of model containing list of class names for classification.*
- QString **configName**
- int **defaultThreshold** = 0  
*Threshold used for models that aren't configured by the UI.*
- QString **displayName**  
*Name used for display in UI (QComboBox).*
- QString **downloadPath**  
*Used by the downloader for the download path.*
- QString **fileName**  
*Used by the downloader and model loader.*
- qint64 **fileSize** = 0  
*Used by the downloader to verify size.*
- int **imageSize** = 0  
*Max dimension of a side of an image.*
- DNNLoaderType **loaderType** = DNNLoaderNet  
*Model loader type custom (YuNet/SFace), Caffe, Darknet, Torch, Tensorflow.*
- int **maxUsableThreshold** = 0  
*Used to convert UI 1-10 slider to float for processing.*
- cv::Scalar **meanValToSubtract** = cv::Scalar(0.0, 0.0, 0.0)
- int **minUsableThreshold** = 0  
*Used to convert UI 1-10 slider to float for processing.*
- QVersionNumber **minVersion**  
*Minimum version of digiKam needed to use this model.*
- QString **sha256**  
*SHA265 hash of the file for download.*
- DNNModelUsageList **usage**  
*How the model can be used. | for more than one use. face\_detection, face\_recognition, weight, object\_detection, etc...*

## 9.404 Digikam::DNNModelManager Class Reference

Inheritance diagram for Digikam::DNNModelManager:



### Public Member Functions

- const [QList< DownloadInfo >](#) & **getDownloadInformation** (DNNModelUsage usage)  
*Used by the filesdownload to get a stream containing the files and information to download.*
- [DNNModelBase \\*](#) **getModel** (const [QString](#) &modelName, DNNModelUsage usage) const  
*Retrieve a [DNNModelBase](#) pointer by name.*

### Static Public Member Functions

- static [DNNModelManager \\*](#) **instance** ()  
*Global instance of internal model manager.*

### Friends

- class [DNNModelManagerCreator](#)

## 9.404.1 Member Function Documentation

### 9.404.1.1 getModel()

```

DNNModelBase * Digikam::DNNModelManager::getModel (
    const QString & modelName,
    DNNModelUsage usage ) const
  
```

This will load and create the model on first use. It will also find the best OpenCV Target and Backend for the model based on computer capabilities. Returns nullptr if 'modelName' cannot be found.

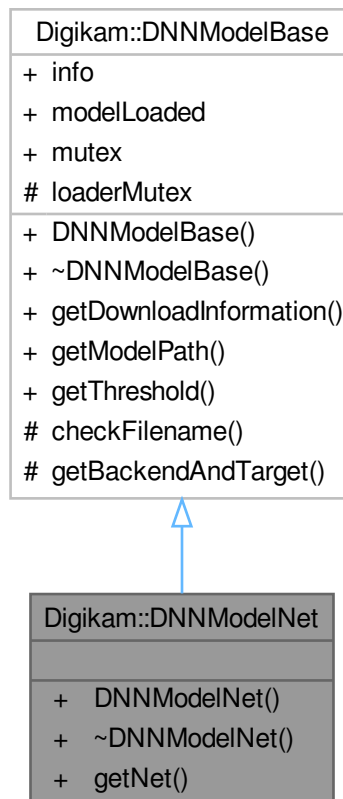
### 9.404.1.2 instance()

`DNNModelManager * Digikam::DNNModelManager::instance ( ) [static]`

All accessor methods are thread-safe.

## 9.405 Digikam::DNNModelNet Class Reference

Inheritance diagram for Digikam::DNNModelNet:



### Public Member Functions

- `DNNModelNet` (const `DNNModelInfoContainer` &\_info)
- `cv::dnn::Net` & `getNet` ()

### Public Member Functions inherited from `Digikam::DNNModelBase`

- `DNNModelBase` (const `DNNModelInfoContainer` &\_info)
- `DownloadInfo` `getDownloadInformation` () const
- const `QString` `getModelPath` () const  
*Return path to the model, or null string if path cannot be found.*
- float `getThreshold` (int uiThreshold=`DNN_MODEL_THRESHOLD_NOT_SET`) const  
*input: uiThreshold is the slider value from the UI.*



**Additional Inherited Members****Public Attributes inherited from [Digikam::DNNModelBase](#)**

- const [DNNModelInfoContainer](#) **info**  
*information about the model.*
- bool **modelLoaded** = false  
*check if the model has been loaded.*
- QMutex **mutex**  
*mutex to sigle-thread model during critical processing functions.*

**Protected Member Functions inherited from [Digikam::DNNModelBase](#)**

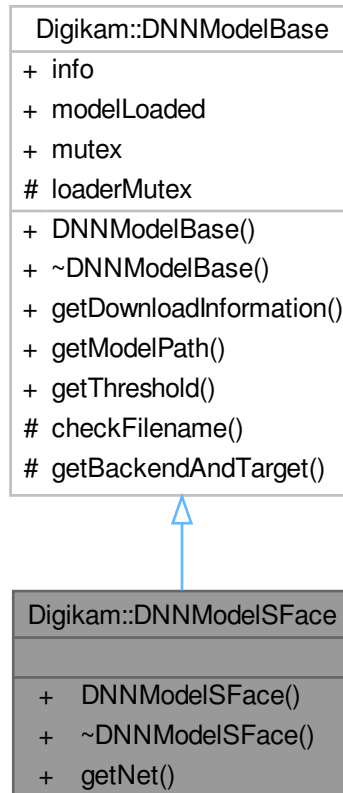
- bool **checkFilename** () const
- const QPair< int, int > **getBackendAndTarget** () const

**Protected Attributes inherited from [Digikam::DNNModelBase](#)**

- QMutex **loaderMutex**

**9.406 Digikam::DNNModelSFace Class Reference**

Inheritance diagram for Digikam::DNNModelSFace:



### Public Member Functions

- **DNNModelSFace** (const [DNNModelInfoContainer](#) &\_info)
- cv::Ptr< cv::FaceRecognizerSF > & **getNet** ()

### Public Member Functions inherited from [Digikam::DNNModelBase](#)

- **DNNModelBase** (const [DNNModelInfoContainer](#) &\_info)
- [DownloadInfo](#) **getDownloadInformation** () const
- const QString **getModelPath** () const  
*Return path to the model, or null string if path cannot be found.*
- float **getThreshold** (int uiThreshold=DNN\_MODEL\_THRESHOLD\_NOT\_SET) const  
*input: uiThreshold is the slider value from the UI.*

### Additional Inherited Members

### Public Attributes inherited from [Digikam::DNNModelBase](#)

- const [DNNModelInfoContainer](#) **info**  
*information about the model.*
- bool **modelLoaded** = false  
*check if the model has been loaded.*
- QMutex **mutex**  
*mutex to sigle-thread model during critical processing functions.*

### Protected Member Functions inherited from [Digikam::DNNModelBase](#)

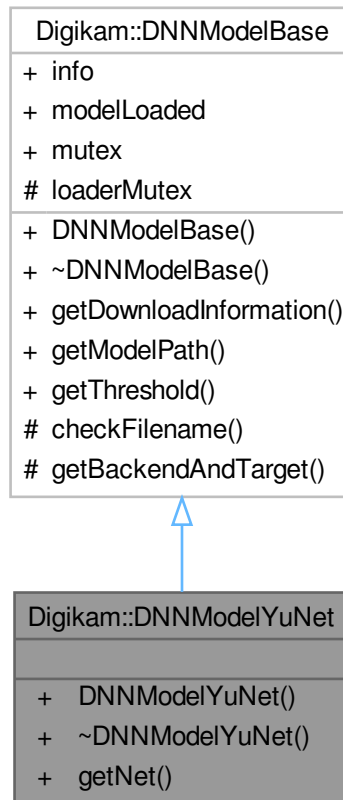
- bool **checkFilename** () const
- const QPair< int, int > **getBackendAndTarget** () const

### Protected Attributes inherited from [Digikam::DNNModelBase](#)

- QMutex **loaderMutex**

## 9.407 Digikam::DNNModelYuNet Class Reference

Inheritance diagram for Digikam::DNNModelYuNet:



### Public Member Functions

- **DNNModelYuNet** (const [DNNModelInfoContainer](#) &\_info)
- `cv::Ptr< cv::FaceDetectorYN >` & **getNet** ()

### Public Member Functions inherited from [Digikam::DNNModelBase](#)

- **DNNModelBase** (const [DNNModelInfoContainer](#) &\_info)
- [DownloadInfo](#) **getDownloadInformation** () const
- const `QString` **getModelPath** () const  
*Return path to the model, or null string if path cannot be found.*
- float **getThreshold** (int uiThreshold=DNN\_MODEL\_THRESHOLD\_NOT\_SET) const  
*input: uiThreshold is the slider value from the UI.*

### Additional Inherited Members

### Public Attributes inherited from [Digikam::DNNModelBase](#)

- const [DNNModelInfoContainer](#) **info**  
*information about the model.*
- bool **modelLoaded** = false  
*check if the model has been loaded.*
- QMutex **mutex**  
*mutex to sigle-thread model during critical processing functions.*

### Protected Member Functions inherited from [Digikam::DNNModelBase](#)

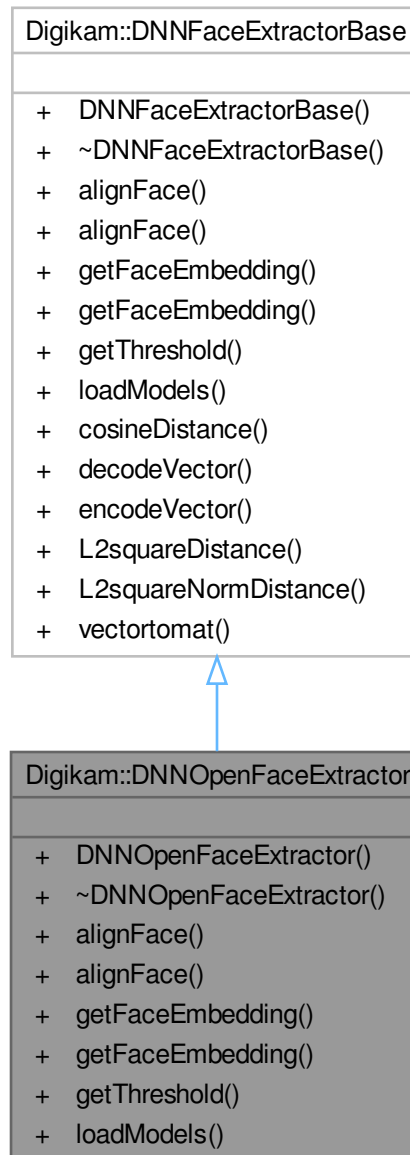
- bool **checkFilename** () const
- const QPair< int, int > **getBackendAndTarget** () const

### Protected Attributes inherited from [Digikam::DNNModelBase](#)

- QMutex **loaderMutex**

## 9.408 Digikam::DNNOpenFaceExtractor Class Reference

Inheritance diagram for Digikam::DNNOpenFaceExtractor:



### Public Member Functions

- virtual `cv::Mat alignFace` (`const cv::Mat &inputImage`) `const` override
- virtual `cv::UMat alignFace` (`const cv::UMat &inputImage`) `const` override
- virtual `cv::Mat getFaceEmbedding` (`const cv::Mat &faceImage`) `override`
- virtual `cv::Mat getFaceEmbedding` (`const cv::UMat &faceImage`) `override`
- float `getThreshold` (`int uiThreshold=DNN_MODEL_THRESHOLD_NOT_SET`) `const` override

cover the UI threshold to a float using the conversion factor built into the model

- bool `loadModels ()` override

Read pretrained neural network for face recognition.

## Additional Inherited Members

## Static Public Member Functions inherited from [Digikam::DNNFaceExtractorBase](#)

- static double **cosineDistance** (const std::vector< float > &v1, const std::vector< float > &v2)  
Calculate different between 2 vectors.
- static std::vector< float > **decodeVector** (const QJsonArray &json)
- static QJsonArray **encodeVector** (const std::vector< float > &vector)
- static double **L2squareDistance** (const std::vector< float > &v1, const std::vector< float > &v2)
- static double **L2squareNormDistance** (const std::vector< float > &v1, const std::vector< float > &v2)
- static cv::Mat **vectortomat** (const std::vector< float > &vector)

Convert face embedding between different formats.

## 9.408.1 Member Function Documentation

### 9.408.1.1 alignFace() [1/2]

```
cv::Mat Digikam::DNNOpenFaceExtractor::alignFace (
    const cv::Mat & inputImage ) const [override], [virtual]
```

Implements [Digikam::DNNFaceExtractorBase](#).

### 9.408.1.2 alignFace() [2/2]

```
virtual cv::UMat Digikam::DNNOpenFaceExtractor::alignFace (
    const cv::UMat & inputImage ) const [inline], [override], [virtual]
```

Implements [Digikam::DNNFaceExtractorBase](#).

### 9.408.1.3 getFaceEmbedding() [1/2]

```
cv::Mat Digikam::DNNOpenFaceExtractor::getFaceEmbedding (
    const cv::Mat & faceImage ) [override], [virtual]
```

Implements [Digikam::DNNFaceExtractorBase](#).

### 9.408.1.4 getFaceEmbedding() [2/2]

```
virtual cv::Mat Digikam::DNNOpenFaceExtractor::getFaceEmbedding (
    const cv::UMat & faceImage ) [inline], [override], [virtual]
```

Implements [Digikam::DNNFaceExtractorBase](#).

### 9.408.1.5 getThreshold()

```
float Digikam::DNNOpenFaceExtractor::getThreshold (
    int uiThreshold = DNN_MODEL_THRESHOLD_NOT_SET ) const [override], [virtual]
```

Implements [Digikam::DNNFaceExtractorBase](#).

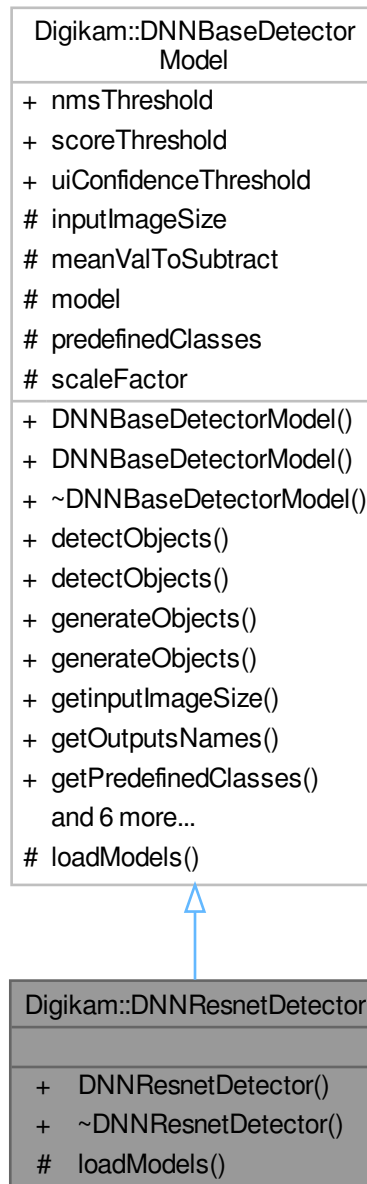
### 9.408.1.6 loadModels()

```
bool Digikam::DNNOpenFaceExtractor::loadModels ( ) [override], [virtual]
```

Implements [Digikam::DNNFaceExtractorBase](#).

## 9.409 Digikam::DNNResnetDetector Class Reference

Inheritance diagram for Digikam::DNNResnetDetector:



### Protected Member Functions

- bool `loadModels ()` override



## Additional Inherited Members

### Public Member Functions inherited from [Digikam::DNNBaseDetectorModel](#)

- **DNNBaseDetectorModel** (float scale, const cv::Scalar &val, const cv::Size &inputImgSize)
- virtual QHash< QString, QVector< QRect > > **detectObjects** (const cv::Mat &inputImage)  
*detectObjects return the predicted objects and localization as well (if we use deeplearning for object detection like YOLO, etc) otherwise the map whose the key is the objects name and their values are empty.*
- virtual QList< QHash< QString, QVector< QRect > > > **detectObjects** (const std::vector< cv::Mat > &inputBatchImages)  
*detectObjects in batch images (fixed batch size).*
- QList< QString > **generateObjects** (const cv::Mat &inputImage)  
*generateObjects in one image return just the predicted objects without locations of objects using for the assignment tagging names.*
- QList< QList< QString > > **generateObjects** (const std::vector< cv::Mat > &inputImage)  
*generateObjects in batch images return just the predicted objects without locations of objects using for the assignment tagging names.*
- cv::Size **getInputImageSize** () const  
*Return the input Image Size from Deep NN model.*
- std::vector< cv::String > **getOutputsNames** () const
- virtual QList< QString > **getPredefinedClasses** () const  
*Get predefined objects according to selected model.*
- QList< QString > **loadDetectionClasses** ()
- QList< QHash< QString, QVector< QRect > > > **postprocess** (const std::vector< cv::Mat > &inputBatchImages, const std::vector< cv::Mat > &outs) const
- std::vector< cv::Mat > **preprocess** (const cv::Mat &inputImage)
- std::vector< cv::Mat > **preprocess** (const std::vector< cv::Mat > &inputBatchImages)
- double **showInferenceTime** ()

### Static Public Attributes inherited from [Digikam::DNNBaseDetectorModel](#)

- static float **nmsThreshold** = 0.4F  
*Threshold for nms suppression.*
- static float **scoreThreshold** = 0.45F  
*Threshold for class detection score.*
- static int **uiConfidenceThreshold** = DNN\_MODEL\_THRESHOLD\_NOT\_SET  
*Threshold for bbox detection. It can be init and changed in the GUI.*

### Protected Attributes inherited from [Digikam::DNNBaseDetectorModel](#)

- cv::Size **inputImageSize**
- cv::Scalar **meanValToSubtract**
- [DNNModelBase](#) \* **model** = nullptr
- QList< QString > **predefinedClasses**
- float **scaleFactor** = 1.0F

## 9.409.1 Member Function Documentation

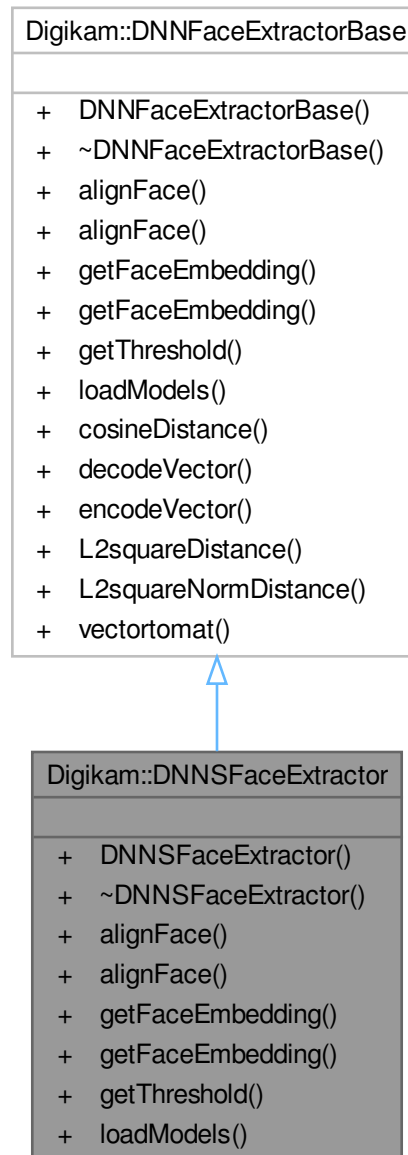
### 9.409.1.1 loadModels()

```
bool Digikam::DNNResnetDetector::loadModels ( ) [override], [protected], [virtual]
```

Implements [Digikam::DNNBaseDetectorModel](#).

## 9.410 Digikam::DNNSFaceExtractor Class Reference

Inheritance diagram for Digikam::DNNSFaceExtractor:



### Public Member Functions

- virtual cv::Mat [alignFace](#) (const cv::Mat &inputImage) const override
- virtual cv::UMat [alignFace](#) (const cv::UMat &inputImage) const override
- virtual cv::Mat [getFaceEmbedding](#) (const cv::Mat &facelImage) override
- virtual cv::Mat [getFaceEmbedding](#) (const cv::UMat &facelImage) override
- float [getThreshold](#) (int uiThreshold=DNN\_MODEL\_THRESHOLD\_NOT\_SET) const override

- cover the UI threshold to a float using the conversion factor built into the model
- bool `loadModels ()` override  
Read pretrained neural network for face recognition.

## Additional Inherited Members

## Static Public Member Functions inherited from [Digikam::DNNFaceExtractorBase](#)

- static double `cosineDistance` (const std::vector< float > &v1, const std::vector< float > &v2)  
*Calculate different between 2 vectors.*
- static std::vector< float > `decodeVector` (const QJsonArray &json)
- static QJsonArray `encodeVector` (const std::vector< float > &vector)
- static double `L2squareDistance` (const std::vector< float > &v1, const std::vector< float > &v2)
- static double `L2squareNormDistance` (const std::vector< float > &v1, const std::vector< float > &v2)
- static cv::Mat `vectortomat` (const std::vector< float > &vector)  
*Convert face embedding between different formats.*

## 9.410.1 Member Function Documentation

### 9.410.1.1 `alignFace()` [1/2]

```
cv::Mat Digikam::DNNSFaceExtractor::alignFace (
    const cv::Mat & inputImage ) const [override], [virtual]
```

Implements [Digikam::DNNFaceExtractorBase](#).

### 9.410.1.2 `alignFace()` [2/2]

```
cv::UMat Digikam::DNNSFaceExtractor::alignFace (
    const cv::UMat & inputImage ) const [override], [virtual]
```

Implements [Digikam::DNNFaceExtractorBase](#).

### 9.410.1.3 `getFaceEmbedding()` [1/2]

```
cv::Mat Digikam::DNNSFaceExtractor::getFaceEmbedding (
    const cv::Mat & faceImage ) [override], [virtual]
```

Implements [Digikam::DNNFaceExtractorBase](#).

### 9.410.1.4 `getFaceEmbedding()` [2/2]

```
cv::Mat Digikam::DNNSFaceExtractor::getFaceEmbedding (
    const cv::UMat & faceImage ) [override], [virtual]
```

Implements [Digikam::DNNFaceExtractorBase](#).

#### 9.410.1.5 getThreshold()

```
float Digikam::DNNSFaceExtractor::getThreshold (
    int uiThreshold = DNN_MODEL_THRESHOLD_NOT_SET ) const [override], [virtual]
```

Implements [Digikam::DNNFaceExtractorBase](#).

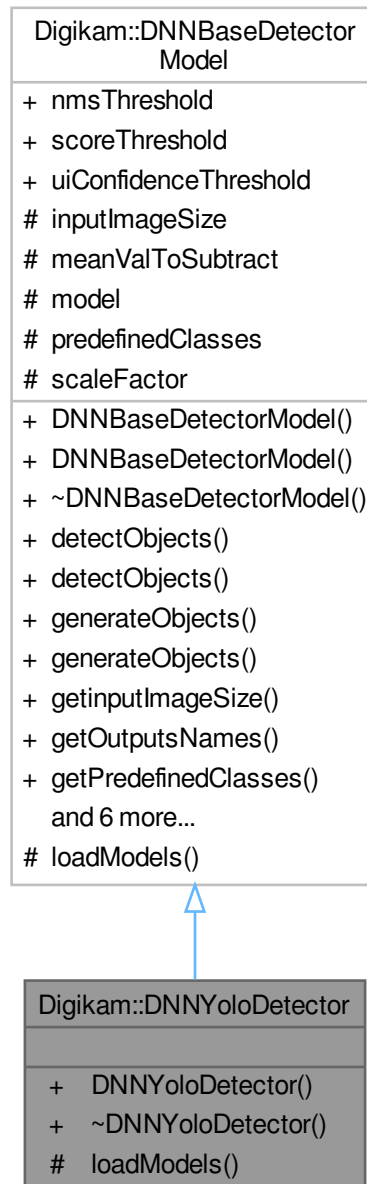
#### 9.410.1.6 loadModels()

```
bool Digikam::DNNSFaceExtractor::loadModels ( ) [override], [virtual]
```

Implements [Digikam::DNNFaceExtractorBase](#).

## 9.411 Digikam::DNNYoloDetector Class Reference

Inheritance diagram for Digikam::DNNYoloDetector:



### Public Member Functions

- **DNNYoloDetector** ([YoloVersions](#) modelVersion=[YoloVersions::YOLOV5NANO](#))

## Public Member Functions inherited from [Digikam::DNNBaseDetectorModel](#)

- **DNNBaseDetectorModel** (float scale, const cv::Scalar &val, const cv::Size &inputImgSize)
- virtual QHash< QString, QVector< QRect > > **detectObjects** (const cv::Mat &inputImage)  
*detectObjects return the predicted objects and localization as well (if we use deeplearning for object detection like YOLO, etc) otherwise the map whose the key is the objects name and their values are empty.*
- virtual QList< QHash< QString, QVector< QRect > > > **detectObjects** (const std::vector< cv::Mat > &inputBatchImages)  
*detectObjects in batch images (fixed batch size).*
- QList< QString > **generateObjects** (const cv::Mat &inputImage)  
*generateObjects in one image return just the predicted objects without locations of objects using for the assignment tagging names.*
- QList< QList< QString > > **generateObjects** (const std::vector< cv::Mat > &inputImage)  
*generateObjects in batch images return just the predicted objects without locations of objects using for the assignment tagging names.*
- cv::Size **getInputImageSize** () const  
*Return the input Image Size from Deep NN model.*
- std::vector< cv::String > **getOutputsNames** () const
- virtual QList< QString > **getPredefinedClasses** () const  
*Get predefined objects according to selected model.*
- QList< QString > **loadDetectionClasses** ()
- QList< QHash< QString, QVector< QRect > > > **postprocess** (const std::vector< cv::Mat > &inputBatchImages, const std::vector< cv::Mat > &outs) const
- std::vector< cv::Mat > **preprocess** (const cv::Mat &inputImage)
- std::vector< cv::Mat > **preprocess** (const std::vector< cv::Mat > &inputBatchImages)
- double **showInferenceTime** ()

## Protected Member Functions

- bool [loadModels](#) () override

## Additional Inherited Members

## Static Public Attributes inherited from [Digikam::DNNBaseDetectorModel](#)

- static float **nmsThreshold** = 0.4F  
*Threshold for nms suppression.*
- static float **scoreThreshold** = 0.45F  
*Threshold for class detection score.*
- static int **uiConfidenceThreshold** = DNN\_MODEL\_THRESHOLD\_NOT\_SET  
*Threshold for bbox detection. It can be init and changed in the GUI.*

## Protected Attributes inherited from [Digikam::DNNBaseDetectorModel](#)

- cv::Size **inputImageSize**
- cv::Scalar **meanValToSubtract**
- [DNNModelBase](#) \* **model** = nullptr
- QList< QString > **predefinedClasses**
- float **scaleFactor** = 1.0F

## 9.411.1 Member Function Documentation

### 9.411.1.1 loadModels()

```
bool Digikam::DNNYoloDetector::loadModels ( ) [override], [protected], [virtual]
```

Implements [Digikam::DNNBaseDetectorModel](#).

## 9.412 Digikam::DNotificationPopup Class Reference

A dialog-like popup that displays messages without interrupting the user.

Inheritance diagram for Digikam::DNotificationPopup:



## Public Types

- enum `PopupStyle` { `Boxed` , `Balloon` }  
*Styles that a `DNotificationPopup` can have.*

## Public Slots

- void `setPopupStyle` (int popupstyle)



*Sets the visual appearance of the popup.*

- void **setTimeout** (int delay)  
*Sets the delay for the popup is removed automatically.*
- void **setVisible** (bool visible) override
- void **show** (const QPoint &p)  
*Shows the popup in the given point.*

## Signals

- void **clicked** ()  
*Emitted when the popup is clicked.*
- void **clicked** (const QPoint &pos)  
*Emitted when the popup is clicked.*

## Public Member Functions

- **DNotificationPopup** (QWidget \*const parent=nullptr, Qt::WindowFlags f=Qt::WindowFlags())  
*Creates a popup for the specified widget.*
- **DNotificationPopup** (WId parent)  
*Creates a popup for the specified window.*
- **~DNotificationPopup** () override  
*Cleans up.*
- QPoint **anchor** () const  
*Returns the position to which this popup is anchored.*
- bool **autoDelete** () const  
*Returns whether the popup will be deleted when it is hidden.*
- void **setAnchor** (const QPoint &anchor)  
*Sets the anchor of this popup.*
- virtual void **setAutoDelete** (bool autoDelete)  
*Sets whether the popup will be deleted when it is hidden.*
- virtual void **setView** (const QString &caption, const QString &text, const QPixmap &icon)  
*Creates a standard view then calls [setView\(QWidget\\*\)](#) .*
- void **setView** (const QString &caption, const QString &text=QString())  
*Creates a standard view then calls [setView\(QWidget\\*\)](#) .*
- void **setView** (QWidget \*child)  
*Sets the main view to be the specified widget (which must be a child of the popup).*
- QWidget \* **standardView** (const QString &caption, const QString &text, const QPixmap &icon, QWidget \*parent=nullptr)  
*Returns a widget that is used as standard view if one of the [setView\(\)](#) methods taking the QString arguments is used.*
- int **timeout** () const  
*Returns the delay before the popup is removed automatically.*
- QWidget \* **view** () const  
*Returns the main view.*

## Static Public Member Functions

- static [DNotificationPopup](#) \* [message](#) (const QString &caption, const QString &text, const QPixmap &icon, QSystemTrayIcon \*parent, int timeout=-1)  
*Convenience method that displays popup with the specified icon, caption and message beside the icon of the specified QSystemTrayIcon.*
- static [DNotificationPopup](#) \* [message](#) (const QString &caption, const QString &text, const QPixmap &icon, QWidget \*parent, int timeout=-1, const QPoint &p=QPoint())  
*Convenience method that displays popup with the specified icon, caption and message beside the icon of the specified widget.*
- static [DNotificationPopup](#) \* [message](#) (const QString &caption, const QString &text, const QPixmap &icon, WId parent, int timeout=-1, const QPoint &p=QPoint())  
*Convenience method that displays popup with the specified icon, caption and message beside the icon of the specified window.*
- static [DNotificationPopup](#) \* [message](#) (const QString &caption, const QString &text, QSystemTrayIcon \*parent)  
*Convenience method that displays popup with the specified caption and message beside the icon of the specified QSystemTrayIcon.*
- static [DNotificationPopup](#) \* [message](#) (const QString &caption, const QString &text, QWidget \*parent, const QPoint &p=QPoint())  
*Convenience method that displays popup with the specified caption and message beside the icon of the specified widget.*
- static [DNotificationPopup](#) \* [message](#) (const QString &text, QSystemTrayIcon \*parent)  
*Convenience method that displays popup with the specified message beside the icon of the specified QSystemTrayIcon.*
- static [DNotificationPopup](#) \* [message](#) (const QString &text, QWidget \*parent, const QPoint &p=QPoint())  
*Convenience method that displays popup with the specified message beside the icon of the specified widget.*
- static [DNotificationPopup](#) \* [message](#) (int popupStyle, const QString &caption, const QString &text, const QPixmap &icon, QSystemTrayIcon \*parent, int timeout=-1)  
*Convenience method that displays popup with the specified popup-style, icon, caption and message beside the icon of the specified QSystemTrayIcon.*
- static [DNotificationPopup](#) \* [message](#) (int popupStyle, const QString &caption, const QString &text, const QPixmap &icon, QWidget \*parent, int timeout=-1, const QPoint &p=QPoint())  
*Convenience method that displays popup with the specified popup-style, icon, caption and message beside the icon of the specified widget.*
- static [DNotificationPopup](#) \* [message](#) (int popupStyle, const QString &caption, const QString &text, const QPixmap &icon, WId parent, int timeout=-1, const QPoint &p=QPoint())  
*Convenience method that displays popup with the specified popup-style, icon, caption and message beside the icon of the specified window.*
- static [DNotificationPopup](#) \* [message](#) (int popupStyle, const QString &caption, const QString &text, QSystemTrayIcon \*parent)  
*Convenience method that displays popup with the specified popup-style, caption and message beside the icon of the specified QSystemTrayIcon.*
- static [DNotificationPopup](#) \* [message](#) (int popupStyle, const QString &caption, const QString &text, QWidget \*parent, const QPoint &p=QPoint())  
*Convenience method that displays popup with the specified popup-style, caption and message beside the icon of the specified widget.*
- static [DNotificationPopup](#) \* [message](#) (int popupStyle, const QString &text, QSystemTrayIcon \*parent)  
*Convenience method that displays popup with the specified popup-style and message beside the icon of the specified QSystemTrayIcon.*
- static [DNotificationPopup](#) \* [message](#) (int popupStyle, const QString &text, QWidget \*parent, const QPoint &p=QPoint())  
*Convenience method that displays popup with the specified popup-style and message beside the icon of the specified widget.*

## Protected Member Functions

- virtual QPoint [defaultLocation](#) () const  
*Returns a default location for popups when a better placement cannot be found.*
- void **hideEvent** (QHideEvent \*) override
- void **mouseReleaseEvent** (QMouseEvent \*e) override
- void [moveNear](#) (const QRect &target)  
*Moves the popup to be adjacent to target.*
- void **paintEvent** (QPaintEvent \*pe) override
- virtual void [positionSelf](#) ()  
*Positions the popup.*

## Properties

- bool **autoDelete**
- int **timeout**

### 9.412.1 Detailed Description

The simplest uses of [DNotificationPopup](#) are by using the various [message\(\)](#) static methods. The position the popup appears at depends on the type of the parent window:

### 9.412.2 Member Enumeration Documentation

#### 9.412.2.1 PopupStyle

```
enum Digikam::DNotificationPopup::PopupStyle
```

#### Enumerator

Boxed	Information will appear in a framed box (default)
Balloon	Information will appear in a comic-alike balloon.

### 9.412.3 Member Function Documentation

#### 9.412.3.1 autoDelete()

```
bool Digikam::DNotificationPopup::autoDelete ( ) const
```

#### See also

[setAutoDelete](#)

### 9.412.3.2 defaultLocation()

```
QPoint Digikam::DNotificationPopup::defaultLocation ( ) const [protected], [virtual]
```

The default implementation returns the top-left corner of the available work area of the desktop (ie: minus panels, etc).

### 9.412.3.3 message() [1/14]

```
DNotificationPopup * Digikam::DNotificationPopup::message (
    const QString & caption,
    const QString & text,
    const QPixmap & icon,
    QSystemTrayIcon * parent,
    int timeout = -1 ) [static]
```

Note that the returned object is destroyed when it is hidden.

See also

[setAutoDelete](#)

### 9.412.3.4 message() [2/14]

```
DNotificationPopup * Digikam::DNotificationPopup::message (
    const QString & caption,
    const QString & text,
    const QPixmap & icon,
    QWidget * parent,
    int timeout = -1,
    const QPoint & p = QPoint() ) [static]
```

Note that the returned object is destroyed when it is hidden.

See also

[setAutoDelete](#)

### 9.412.3.5 message() [3/14]

```
DNotificationPopup * Digikam::DNotificationPopup::message (
    const QString & caption,
    const QString & text,
    const QPixmap & icon,
    WId parent,
    int timeout = -1,
    const QPoint & p = QPoint() ) [static]
```

Note that the returned object is destroyed when it is hidden.

See also

[setAutoDelete](#)

**9.412.3.6 message()** [4/14]

```
DNotificationPopup * Digikam::DNotificationPopup::message (
    const QString & caption,
    const QString & text,
    QSystemTrayIcon * parent ) [static]
```

Note that the returned object is destroyed when it is hidden.

See also

[setAutoDelete](#)

**9.412.3.7 message()** [5/14]

```
DNotificationPopup * Digikam::DNotificationPopup::message (
    const QString & caption,
    const QString & text,
    QWidget * parent,
    const QPoint & p = QPoint() ) [static]
```

Note that the returned object is destroyed when it is hidden.

See also

[setAutoDelete](#)

**9.412.3.8 message()** [6/14]

```
DNotificationPopup * Digikam::DNotificationPopup::message (
    const QString & text,
    QSystemTrayIcon * parent ) [static]
```

Note that the returned object is destroyed when it is hidden.

See also

[setAutoDelete](#)

**9.412.3.9 message()** [7/14]

```
DNotificationPopup * Digikam::DNotificationPopup::message (
    const QString & text,
    QWidget * parent,
    const QPoint & p = QPoint() ) [static]
```

Note that the returned object is destroyed when it is hidden.

See also

[setAutoDelete](#)

**9.412.3.10 message()** [8/14]

```
DNotificationPopup * Digikam::DNotificationPopup::message (
    int popupStyle,
    const QString & caption,
    const QString & text,
    const QPixmap & icon,
    QSystemTrayIcon * parent,
    int timeout = -1 ) [static]
```

Note that the returned object is destroyed when it is hidden.

See also

[setAutoDelete](#)

**9.412.3.11 message()** [9/14]

```
DNotificationPopup * Digikam::DNotificationPopup::message (
    int popupStyle,
    const QString & caption,
    const QString & text,
    const QPixmap & icon,
    QWidget * parent,
    int timeout = -1,
    const QPoint & p = QPoint() ) [static]
```

Note that the returned object is destroyed when it is hidden.

See also

[setAutoDelete](#)

**9.412.3.12 message()** [10/14]

```
DNotificationPopup * Digikam::DNotificationPopup::message (
    int popupStyle,
    const QString & caption,
    const QString & text,
    const QPixmap & icon,
    WId parent,
    int timeout = -1,
    const QPoint & p = QPoint() ) [static]
```

Note that the returned object is destroyed when it is hidden.

See also

[setAutoDelete](#)

**9.412.3.13 message()** [11/14]

```
DNotificationPopup * Digikam::DNotificationPopup::message (
    int popupStyle,
    const QString & caption,
    const QString & text,
    QSystemTrayIcon * parent ) [static]
```

Note that the returned object is destroyed when it is hidden.

See also

[setAutoDelete](#)

**9.412.3.14 message()** [12/14]

```
DNotificationPopup * Digikam::DNotificationPopup::message (
    int popupStyle,
    const QString & caption,
    const QString & text,
    QWidget * parent,
    const QPoint & p = QPoint() ) [static]
```

Note that the returned object is destroyed when it is hidden.

See also

[setAutoDelete](#)

**9.412.3.15 message()** [13/14]

```
DNotificationPopup * Digikam::DNotificationPopup::message (
    int popupStyle,
    const QString & text,
    QSystemTrayIcon * parent ) [static]
```

Note that the returned object is destroyed when it is hidden.

See also

[setAutoDelete](#)

**9.412.3.16 message()** [14/14]

```
DNotificationPopup * Digikam::DNotificationPopup::message (
    int popupStyle,
    const QString & text,
    QWidget * parent,
    const QPoint & p = QPoint() ) [static]
```

Note that the returned object is destroyed when it is hidden.

See also

[setAutoDelete](#)

### 9.412.3.17 `moveNear()`

```
void Digikam::DNotificationPopup::moveNear (
    const QRect & target ) [protected]
```

The popup will be placed adjacent to, but outside of, `target`, without going off the current desktop.

Reimplementations of `positionSelf()` can use this to actually position the popup.

### 9.412.3.18 `positionSelf()`

```
void Digikam::DNotificationPopup::positionSelf ( ) [protected], [virtual]
```

The default implementation attempts to place it by the taskbar entry; failing that it places it by the window of the associated widget; failing that it places it at the location given by `defaultLocation()`.

See also

[moveNear\(\)](#)

### 9.412.3.19 `setAnchor()`

```
void Digikam::DNotificationPopup::setAnchor (
    const QPoint & anchor )
```

The popup is placed near to the anchor.

### 9.412.3.20 `setAutoDelete()`

```
void Digikam::DNotificationPopup::setAutoDelete (
    bool autoDelete ) [virtual]
```

The default is false (unless created by one of the static `message()` overloads).

### 9.412.3.21 `setPopupStyle`

```
void Digikam::DNotificationPopup::setPopupStyle (
    int popupstyle ) [slot]
```

See also

[PopupStyle](#)



**9.412.3.22 setTimeout**

```
void Digikam::DNotificationPopup::setTimeout (
    int delay ) [slot]
```

Setting the delay to 0 disables the timeout, if you're doing this, you may want to connect the [clicked\(\)](#) signal to the `hide()` slot. Setting the delay to -1 makes it use the default value.

**See also**

[timeout](#)

**9.412.3.23 standardView()**

```
QWidget * Digikam::DNotificationPopup::standardView (
    const QString & caption,
    const QString & text,
    const QPixmap & icon,
    QWidget * parent = nullptr )
```

You can use the returned widget to customize the passivepopup while keeping the look similar to the "standard" passivepopups.

After customizing the widget, pass it to [setView\( QWidget\\* \)](#)

**Parameters**

<i>caption</i>	The window caption (title) on the popup
<i>text</i>	The text for the popup
<i>icon</i>	The icon to use for the popup
<i>parent</i>	The parent widget used for the returned widget. If left 0, then "this", i.e. the passive popup object will be used.

**Returns**

a QWidget containing the given arguments, looking like the standard passivepopups. The returned widget contains a QVBoxLayout, which is accessible through `layout()`.

**See also**

[setView\( QWidget \\* \)](#)  
[setView\( const QString&, const QString& \)](#)  
[setView\( const QString&, const QString&, const QPixmap& \)](#)

**9.413 Digikam::DNotificationWidget Class Reference**

This widget can be used to provide inline positive or negative feedback, or to implement opportunistic interactions.

Inheritance diagram for Digikam::DNotificationWidget:



## Public Types

- enum `MessageType` {  
**Positive** , **Notification** , **Information** , **Warning** ,  
**Error** }

*Available message types.*

## Public Slots

- void **animatedHide** ()  
*Hide the widget using an animation.*
- void **animatedShow** ()  
*Show the widget using an animation.*
- void **setCloseButtonVisible** (bool visible)  
*Set the visibility of the close button.*
- void **setIcon** (const QIcon &icon)  
*Define an icon to be shown on the left of the text.*
- void **setMessageType** (DNotificationWidget::MessageType type)  
*Set the message type to `type`.*
- void **setText** (const QString &text)  
*Set the text of the message widget to `text`.*
- void **setWordWrap** (bool wordWrap)  
*Set word wrap to `wordWrap`.*

## Signals

- void **hideAnimationFinished** ()  
*This signal is emitted when the hide animation is finished, started by calling [animatedHide\(\)](#).*
- void **linkActivated** (const QString &contents)  
*This signal is emitted when the user clicks a link in the text label.*
- void **linkHovered** (const QString &contents)  
*This signal is emitted when the user hovers over a link in the text label.*
- void **showAnimationFinished** ()  
*This signal is emitted when the show animation is finished, started by calling [animatedShow\(\)](#).*

## Public Member Functions

- **DNotificationWidget** (const QString &text, QWidget \*const parent=nullptr)  
*Constructs a [DNotificationWidget](#) with the specified `parent` and contents `text`.*
- **DNotificationWidget** (QWidget \*const parent=nullptr)  
*Constructs a [DNotificationWidget](#) with the specified `parent`.*
- **~DNotificationWidget** () override  
*Destructor.*
- void **addAction** (QAction \*action)  
*Add `action` to the message widget.*
- void **animatedShowTemporized** (int delay)  
*Show the widget using an animation.*
- void **clearAllActions** ()  
*clear all actions from the message widget.*
- int **heightForWidth** (int width) const override  
*Returns the required height for `width`.*
- QIcon **icon** () const  
*The icon shown on the left of the text.*
- bool **isCloseButtonVisible** () const  
*Check whether the close button is visible.*
- bool **isHideAnimationRunning** () const  
*Check whether the hide animation started by calling [animatedHide\(\)](#) is still running.*

- bool `isShowAnimationRunning` () const  
*Check whether the show animation started by calling `animatedShow()` is still running.*
- `MessageType` `messageType` () const  
*Get the type of this message.*
- QSize `minimumSizeHint` () const override  
*Returns the minimum size of the message widget.*
- void `removeAction` (QAction \*action)  
*Remove `action` from the message widget.*
- QSize `sizeHint` () const override  
*Returns the preferred size of the message widget.*
- QString `text` () const  
*Get the text of this message widget.*
- bool `wordWrap` () const  
*Check whether word wrap is enabled.*

### Protected Member Functions

- bool `event` (QEvent \*event) override
- void `paintEvent` (QPaintEvent \*event) override
- void `resizeEvent` (QResizeEvent \*event) override

### Properties

- bool `closeButtonVisible`
- QIcon `icon`
- `MessageType` `messageType`
- QString `text`
- bool `wordWrap`

### Friends

- class `Private`

## 9.413.1 Member Enumeration Documentation

### 9.413.1.1 MessageType

```
enum Digikam::DNotificationWidget::MessageType
```

The background colors are chosen depending on the message type.

## 9.413.2 Member Function Documentation

### 9.413.2.1 addAction()

```
void Digikam::DNotificationWidget::addAction (
    QAction * action )
```

For each action a button is added to the message widget in the order the actions were added.

## Parameters

<i>action</i>	the action to add
---------------	-------------------

## See also

[removeAction\(\)](#), [QWidget::actions\(\)](#)

### 9.413.2.2 animatedShowTemporized()

```
void Digikam::DNotificationWidget::animatedShowTemporized (
    int delay )
```

The widget is automatically hidden after the delay (in ms).

### 9.413.2.3 clearAllActions()

```
void Digikam::DNotificationWidget::clearAllActions ( )
```

## See also

[DNotificationWidget::MessageType](#), [addAction\(\)](#), [setMessageType\(\)](#)

### 9.413.2.4 heightForWidth()

```
int Digikam::DNotificationWidget::heightForWidth (
    int width ) const [override]
```

## Parameters

<i>width</i>	the width in pixels
--------------	---------------------

### 9.413.2.5 hideAnimationFinished

```
void Digikam::DNotificationWidget::hideAnimationFinished ( ) [signal]
```

If animations are disabled, this signal is emitted immediately after the message widget got hidden.

## Note

This signal is *not* emitted if the widget was hidden by calling [hide\(\)](#), so this signal is only useful in conjunction with [animatedHide\(\)](#).

## See also

[animatedHide\(\)](#)

### 9.413.2.6 icon()

```
QIcon Digikam::DNotificationWidget::icon ( ) const
```

By default, no icon is shown.

### 9.413.2.7 isCloseButtonVisible()

```
bool Digikam::DNotificationWidget::isCloseButtonVisible ( ) const
```

See also

[setCloseButtonVisible\(\)](#)

### 9.413.2.8 isHideAnimationRunning()

```
bool Digikam::DNotificationWidget::isHideAnimationRunning ( ) const
```

If animations are disabled, this function always returns *false*.

See also

[animatedHide\(\)](#), [hideAnimationFinished\(\)](#)

### 9.413.2.9 isShowAnimationRunning()

```
bool Digikam::DNotificationWidget::isShowAnimationRunning ( ) const
```

If animations are disabled, this function always returns *false*.

See also

[animatedShow\(\)](#), [showAnimationFinished\(\)](#)

### 9.413.2.10 linkActivated

```
void Digikam::DNotificationWidget::linkActivated (
    const QString & contents ) [signal]
```

The URL referred to by the href anchor is passed in contents.

Parameters

<i>contents</i>	text of the href anchor
-----------------	-------------------------

See also

[QLabel::linkActivated\(\)](#)

#### 9.413.2.11 linkHovered

```
void Digikam::DNotificationWidget::linkHovered (
    const QString & contents ) [signal]
```

The URL referred to by the href anchor is passed in contents.

Parameters

<i>contents</i>	text of the href anchor
-----------------	-------------------------

See also

[QLabel::linkHovered\(\)](#)

#### 9.413.2.12 messageType()

```
DNotificationWidget::MessageType Digikam::DNotificationWidget::messageType ( ) const
```

By default, the type is set to DNotificationWidget::Information.

See also

[DNotificationWidget::MessageType](#), [setMessageType\(\)](#)

#### 9.413.2.13 removeAction()

```
void Digikam::DNotificationWidget::removeAction (
    QAction * action )
```

Parameters

<i>action</i>	the action to remove
---------------	----------------------

See also

[DNotificationWidget::MessageType](#), [addAction\(\)](#), [setMessageType\(\)](#)

#### 9.413.2.14 setCloseButtonVisible

```
void Digikam::DNotificationWidget::setCloseButtonVisible (
    bool visible ) [slot]
```

If *visible* is *true*, a close button is shown that calls [animatedHide\(\)](#) if clicked.

**See also**

`closeButtonVisible()`, [animatedHide\(\)](#)

**9.413.2.15 setMessageType**

```
void Digikam::DNotificationWidget::setMessageType (
    DNotificationWidget::MessageType type ) [slot]
```

By default, the message type is set to `DNotificationWidget::Information`.

**See also**

`messageType()`, [DNotificationWidget::MessageType](#)

**9.413.2.16 setText**

```
void Digikam::DNotificationWidget::setText (
    const QString & text ) [slot]
```

If the message widget is already visible, the text changes on the fly.

**Parameters**

<i>text</i>	the text to display, rich text is allowed
-------------	---

**See also**

`text()`

**9.413.2.17 setWordWrap**

```
void Digikam::DNotificationWidget::setWordWrap (
    bool wordWrap ) [slot]
```

If word wrap is enabled, the `text()` of the message widget is wrapped to fit the available width. If word wrap is disabled, the message widget's minimum size is such that the entire text fits.

**Parameters**

<i>wordWrap</i>	disable/enable word wrap
-----------------	--------------------------

**See also**

`wordWrap()`



### 9.413.2.18 showAnimationFinished

```
void Digikam::DNotificationWidget::showAnimationFinished ( ) [signal]
```

If animations are disabled, this signal is emitted immediately after the message widget got shown.

#### Note

This signal is *not* emitted if the widget was shown by calling `show()`, so this signal is only useful in conjunction with [animatedShow\(\)](#).

#### See also

[animatedShow\(\)](#)

### 9.413.2.19 text()

```
QString Digikam::DNotificationWidget::text ( ) const
```

#### See also

[setText\(\)](#)

### 9.413.2.20 wordWrap()

```
bool Digikam::DNotificationWidget::wordWrap ( ) const
```

If word wrap is enabled, the message widget wraps the displayed text as required to the available width of the widget. This is useful to avoid breaking widget layouts.

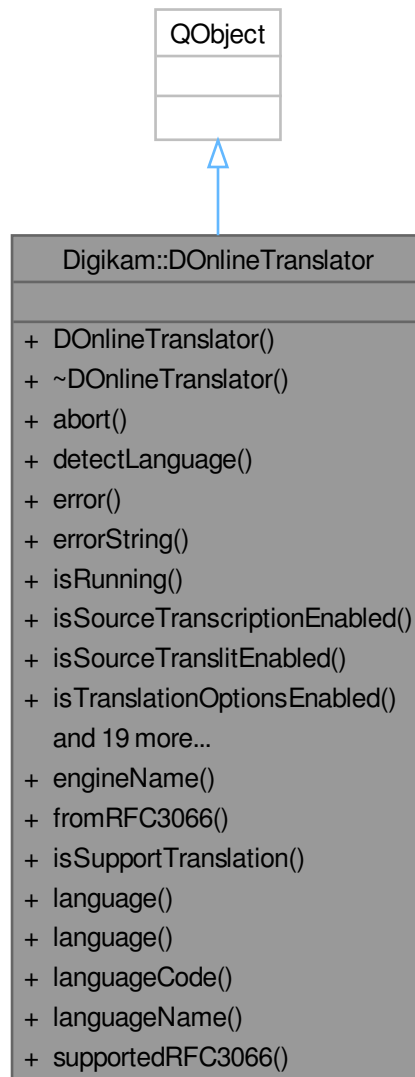
#### See also

[setWordWrap\(\)](#)

## 9.414 Digikam::DOnlineTranslator Class Reference

Provides translation data.

Inheritance diagram for Digikam::DOnlineTranslator:



## Public Types

- enum `Engine` {  
**Google** , **Yandex** , **Bing** , **LibreTranslate** ,  
**Lingva** }  
*Represents online engines.*
- enum `Language` {  
**NoLanguage** = -1 , **Auto** , **Afrikaans** , **Albanian** ,  
**Amharic** , **Arabic** , **Armenian** , **Azerbaijani** ,  
**Bashkir** , **Basque** , **Belarusian** , **Bengali** ,  
**Bosnian** , **Bulgarian** , **Cantonese** , **Catalan** ,  
**Cebuano** , **Chichewa** , **Corsican** , **Croatian** ,  
**Czech** , **Danish** , **Dutch** , **English** ,

Esperanto , Estonian , Fijian , Filipino ,  
 Finnish , French , Frisian , Galician ,  
 Georgian , German , Greek , Gujarati ,  
 HaitianCreole , Hausa , Hawaiian , Hebrew ,  
 HillMari , Hindi , Hmong , Hungarian ,  
 Icelandic , Igbo , Indonesian , Irish ,  
 Italian , Japanese , Javanese , Kannada ,  
 Kazakh , Khmer , Kinyarwanda , Klingon ,  
 KlingonPlqaD , Korean , Kurdish , Kyrgyz ,  
 Lao , Latin , Latvian , LevantineArabic ,  
 Lithuanian , Luxembourgish , Macedonian , Malagasy ,  
 Malay , Malayalam , Maltese , Maori ,  
 Marathi , Mari , Mongolian , Myanmar ,  
 Nepali , Norwegian , Oriya , Papiamentu ,  
 Pashto , Persian , Polish , Portuguese ,  
 Punjabi , QueretaroOtomi , Romanian , Russian ,  
 Samoan , ScotsGaelic , SerbianCyrillic , SerbianLatin ,  
 Sesotho , Shona , SimplifiedChinese , Sindhi ,  
 Sinhala , Slovak , Slovenian , Somali ,  
 Spanish , Sundanese , Swahili , Swedish ,  
 Tagalog , Tahitian , Tajik , Tamil ,  
 Tatar , Telugu , Thai , Tongan ,  
 TraditionalChinese , Turkish , Turkmen , Udmurt ,  
 Uighur , Ukrainian , Urdu , Uzbek ,  
 Vietnamese , Welsh , Xhosa , Yiddish ,  
 Yoruba , YucatecMaya , Zulu }

*Represents all languages for translation.*

- enum [TranslationError](#) {  
[NoError](#) , [ParametersError](#) , [NetworkError](#) , [ServiceError](#) ,  
[ParsingError](#) }

*Indicates all possible error conditions found during the processing of the translation.*

## Signals

- void [signalFinished](#) ()  
*Translation finished.*

## Public Member Functions

- [DOnlineTranslator](#) (QObject \*const parent=nullptr)  
*Create object.*
- void **abort** ()  
*Cancel translation operation (if any).*
- void [detectLanguage](#) (const QString &text, [Engine](#) engine=Google)  
*Detect language.*
- [TranslationError error](#) () const  
*Last error.*
- QString [errorString](#) () const  
*Last error string.*
- bool [isRunning](#) () const  
*Check translation progress.*
- bool [isSourceTranscriptionEnabled](#) () const  
*Check if source transcription is enabled.*

- bool `isSourceTranslitEnabled` () const  
*Check if source transliteration is enabled.*
- bool `isTranslationOptionsEnabled` () const  
*Check if translation options are enabled.*
- bool `isTranslationTranslitEnabled` () const  
*Check if translation transliteration is enabled.*
- void `setEngineApiKey` (`Engine` engine, const `QByteArray` &apiKey)  
*Set api key for engine.*
- void `setEngineUrl` (`Engine` engine, const `QString` &url)  
*Set the URL engine.*
- void `setSourceTranscriptionEnabled` (bool enable)  
*Enable or disable source transcription.*
- void `setSourceTranslitEnabled` (bool enable)  
*Enable or disable source transliteration.*
- void `setTranslationOptionsEnabled` (bool enable)  
*Enable or disable translation options.*
- void `setTranslationTranslitEnabled` (bool enable)  
*Enable or disable translation transliteration.*
- `QString` `source` () const  
*Source text.*
- `Language` `sourceLanguage` () const  
*Source language.*
- `QString` `sourceLanguageName` () const  
*Source language name.*
- `QString` `sourceTranscription` () const  
*Source transcription.*
- `QString` `sourceTranslit` () const  
*Source transliteration.*
- `QJsonDocument` `toJson` () const  
*Converts the object to JSON.*
- void `translate` (const `QString` &text, `Engine` engine=Google, `Language` translationLang=Auto, `Language` sourceLang=Auto, `Language` uiLang=Auto)  
*Translate text.*
- `QString` `translation` () const  
*Translated text.*
- `Language` `translationLanguage` () const  
*Translation language.*
- `QString` `translationLanguageName` () const  
*Translation language name.*
- `QMap`< `QString`, `QVector`< `DOnlineTranslatorOption` > > `translationOptions` () const  
*Translation options.*
- `QString` `translationTranslit` () const  
*Translation transliteration.*

## Static Public Member Functions

- static QString **engineName** ([Engine](#) engine)  
*Return the engine literal name.*
- static QString **fromRFC3066** ([Engine](#) engine, const QString &langCodeRFC3066)  
*Convert language RFC3066 to supported language code.*
- static bool **isSupportTranslation** ([Engine](#) engine, [Language](#) lang)  
*Check if transliteration is supported.*
- static [Language](#) **language** (const QLocale &locale)  
*Language.*
- static [Language](#) **language** (const QString &langCode)  
*Returns general language code.*
- static QString **languageCode** ([Language](#) lang)  
*Language code.*
- static QString **languageName** ([Language](#) lang)  
*Language name.*
- static QStringList **supportedRFC3066** ([Engine](#) engine)  
*Return a list of all supported language in RFC3066.*

## Friends

- class **DOnlineTts**

## 9.414.1 Member Enumeration Documentation

### 9.414.1.1 TranslationError

```
enum Digikam::DOnlineTranslator::TranslationError
```

#### Enumerator

NoError	No error condition.
ParametersError	Unsupported combination of parameters.
NetworkError	Network error.
ServiceError	Service unavailable or maximum number of requests.
ParsingError	The request could not be parsed (report a bug if you see this)

## 9.414.2 Constructor & Destructor Documentation

### 9.414.2.1 DOnlineTranslator()

```
Digikam::DOnlineTranslator::DOnlineTranslator (
    QObject *const parent = nullptr ) [explicit]
```

Constructs an object with empty data and with parent. You can use [translate\(\)](#) to send text to object.

## Parameters

<i>parent</i>	the parent object
---------------	-------------------

### 9.414.3 Member Function Documentation

#### 9.414.3.1 detectLanguage()

```
void Digikam::DOnlineTranslator::detectLanguage (
    const QString & text,
    Engine engine = Google )
```

## Parameters

<i>text</i>	the text for language detection
<i>engine</i>	the engine to use

#### 9.414.3.2 error()

```
DOnlineTranslator::TranslationError Digikam::DOnlineTranslator::error ( ) const
```

Error that was found during the processing of the last translation. If no error was found, returns [DOnlineTranslator::NoError](#). The text of the error can be obtained by [errorString\(\)](#).

## Returns

last error

#### 9.414.3.3 errorString()

```
QString Digikam::DOnlineTranslator::errorString ( ) const
```

A human-readable description of the last translation error that occurred.

## Returns

last error string

#### 9.414.3.4 isRunning()

```
bool Digikam::DOnlineTranslator::isRunning ( ) const
```

## Returns

`true` when the translation is still processing and has not finished or was aborted yet.

### 9.414.3.5 isSourceTranscriptionEnabled()

```
bool Digikam::DOnlineTranslator::isSourceTranscriptionEnabled ( ) const
```

#### Returns

true if source transcription is enabled

### 9.414.3.6 isSourceTranslitEnabled()

```
bool Digikam::DOnlineTranslator::isSourceTranslitEnabled ( ) const
```

#### Returns

true if source transliteration is enabled

### 9.414.3.7 isSupportTranslation()

```
bool Digikam::DOnlineTranslator::isSupportTranslation (
    Engine engine,
    Language lang ) [static]
```

#### Parameters

<i>engine</i>	the engine to use
<i>lang</i>	language

#### Returns

true if the specified engine supports transliteration for specified language

### 9.414.3.8 isTranslationOptionsEnabled()

```
bool Digikam::DOnlineTranslator::isTranslationOptionsEnabled ( ) const
```

#### Returns

true if translation options are enabled

#### See also

[DOnlineTranslatorOption](#)

### 9.414.3.9 isTranslationTranslitEnabled()

```
bool Digikam::DOnlineTranslator::isTranslationTranslitEnabled ( ) const
```

#### Returns

true if translation transliteration is enabled

### 9.414.3.10 language() [1/2]

```
DOnlineTranslator::Language Digikam::DOnlineTranslator::language (
    const QLocale & locale ) [static]
```

#### Parameters

<i>locale</i>	the locale to use
---------------	-------------------

#### Returns

language

### 9.414.3.11 language() [2/2]

```
DOnlineTranslator::Language Digikam::DOnlineTranslator::language (
    const QString & langCode ) [static]
```

#### Parameters

<i>langCode</i>	code
-----------------	------

#### Returns

language

### 9.414.3.12 languageCode()

```
QString Digikam::DOnlineTranslator::languageCode (
    Language lang ) [static]
```

#### Parameters

<i>lang</i>	language
-------------	----------



**Returns**

language code

**9.414.3.13 languageName()**

```
QString Digikam::DOnlineTranslator::languageName (
    Language lang ) [static]
```

**Parameters**

<i>lang</i>	language
-------------	----------

**Returns**

language name

**9.414.3.14 setEngineApiKey()**

```
void Digikam::DOnlineTranslator::setEngineApiKey (
    Engine engine,
    const QByteArray & apiKey )
```

Affects only LibreTranslate.

**Parameters**

<i>engine</i>	the engine to use
<i>apiKey</i>	your key for this particular instance

**9.414.3.15 setEngineUrl()**

```
void Digikam::DOnlineTranslator::setEngineUrl (
    Engine engine,
    const QString & url )
```

Only affects LibreTranslate and Lingva because these engines have multiple instances. You need to call this function to specify the URL of an instance for them.

**Parameters**

<i>engine</i>	the engine to use
<i>url</i>	engine url

**9.414.3.16 setSourceTranscriptionEnabled()**

```
void Digikam::DOnlineTranslator::setSourceTranscriptionEnabled (
```

```
bool enable )
```

**Parameters**

<i>enable</i>	whether to enable source transcription
---------------	--

**9.414.3.17 setSourceTranslitEnabled()**

```
void Digikam::DOnlineTranslator::setSourceTranslitEnabled (
    bool enable )
```

**Parameters**

<i>enable</i>	whether to enable source transliteration
---------------	--

**9.414.3.18 setTranslationOptionsEnabled()**

```
void Digikam::DOnlineTranslator::setTranslationOptionsEnabled (
    bool enable )
```

**Parameters**

<i>enable</i>	whether to enable translation options
---------------	---------------------------------------

**See also**

[DOnlineTranslatorOption](#)

**9.414.3.19 setTranslationTranslitEnabled()**

```
void Digikam::DOnlineTranslator::setTranslationTranslitEnabled (
    bool enable )
```

**Parameters**

<i>enable</i>	whether to enable translation transliteration
---------------	---

**9.414.3.20 signalFinished**

```
void Digikam::DOnlineTranslator::signalFinished ( ) [signal]
```

This signal is emitted when the translation is complete.

### 9.414.3.21 source()

```
QString Digikam::DOnlineTranslator::source ( ) const
```

#### Returns

source text

### 9.414.3.22 sourceLanguage()

```
DOnlineTranslator::Language Digikam::DOnlineTranslator::sourceLanguage ( ) const
```

#### Returns

language of the source text

### 9.414.3.23 sourceLanguageName()

```
QString Digikam::DOnlineTranslator::sourceLanguageName ( ) const
```

#### Returns

language name of the source text

### 9.414.3.24 sourceTranscription()

```
QString Digikam::DOnlineTranslator::sourceTranscription ( ) const
```

#### Returns

transcription of the source text

### 9.414.3.25 sourceTranslit()

```
QString Digikam::DOnlineTranslator::sourceTranslit ( ) const
```

#### Returns

transliteration of the source text

### 9.414.3.26 toJson()

```
QJsonDocument Digikam::DOnlineTranslator::toJson ( ) const
```

#### Returns

JSON representation

### 9.414.3.27 translate()

```
void Digikam::DOnlineTranslator::translate (
    const QString & text,
    Engine engine = Google,
    Language translationLang = Auto,
    Language sourceLang = Auto,
    Language uiLang = Auto )
```

## Parameters

<i>text</i>	the text to translate
<i>engine</i>	online engine to use
<i>translationLang</i>	language to translation
<i>sourceLang</i>	language of the passed text
<i>uiLang</i>	ui language to use for display

**9.414.3.28 translation()**

```
QString Digikam::DOnlineTranslator::translation ( ) const
```

## Returns

translated text.

**9.414.3.29 translationLanguage()**

```
DOnlineTranslator::Language Digikam::DOnlineTranslator::translationLanguage ( ) const
```

## Returns

language of the translated text

**9.414.3.30 translationLanguageName()**

```
QString Digikam::DOnlineTranslator::translationLanguageName ( ) const
```

## Returns

language name of the translated text

**9.414.3.31 translationOptions()**

```
QMap< QString, QVector< DOnlineTranslatorOption > > Digikam::DOnlineTranslator::translation↔  
Options ( ) const
```

## Returns

QMap whose key represents the type of speech, and the value is a QVector of translation options

## See also

[DOnlineTranslatorOption](#)

### 9.414.3.32 translationTranslit()

```
QString Digikam::DOnlineTranslator::translationTranslit ( ) const
```

#### Returns

transliteration of the translated text

## 9.415 Digikam::DOnlineTranslatorOption Struct Reference

Contains translation options for a single word.

### Public Member Functions

- QObject `toJson ()` const  
*Converts the object to JSON.*

### Public Attributes

- QString `gender`  
*Gender of the word.*
- QStringList `translations`  
*Associated translations for the word.*
- QString `word`  
*Word that specified for translation options.*

### 9.415.1 Detailed Description

Can be obtained from the QOnlineTranslator object.

#### Example:

```
QOnlineTranslator translator;
// Obtain translation

QTextStream out(stdout);

for (auto it = translator.translationOptions().cbegin() ; it != translator.translationOptions().cend() ;
     ++it)
{
    out << it.key() << ":" << endl; // Output the type of speech with a colon

    for (const auto &[word, gender, translations] : it.value())
    {
        out << " " << word << " "; // Print the word
        out << translations;      // Print translations
        out << endl;
    }

    out << endl;
}

```

#### Possible output:

```
// verb:
// sagen: say, tell, speak, mean, utter
// sprechen: speak, talk, say, pronounce, militate, discourse
// meinen: think, mean, believe, say, opine, fancy
// heißen: mean, be called, be named, bid, tell, be titled
// äußern: express, comment, speak, voice, say, utter
// aussprechen: express, pronounce, say, speak, voice, enunciate
// vorbringen: make, put forward, raise, say, put, bring forward
// aufsagen: recite, say, speak

// noun:
// Sagen: say
// Mitspracherecht: say

```

## 9.415.2 Member Function Documentation

### 9.415.2.1 toJson()

```
QJsonObject Digikam::DOnlineTranslatorOption::toJson ( ) const [inline]
```

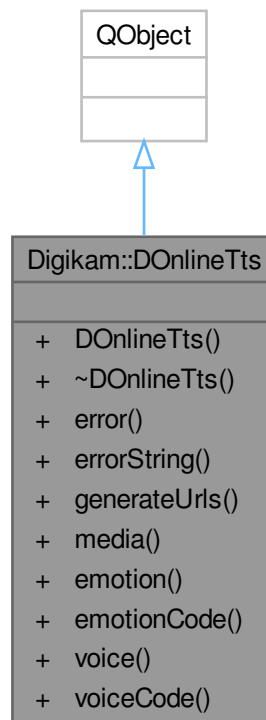
#### Returns

JSON representation

## 9.416 Digikam::DOnlineTts Class Reference

Provides TTS URL generation.

Inheritance diagram for Digikam::DOnlineTts:



### Public Types

- enum [Emotion](#) { **NoEmotion** = -1 , **Neutral** , **Good** , **Evil** }

*Defines emotion to use.*

- enum `TtsError` {  
`NoError` , `UnsupportedEngine` , `UnsupportedLanguage` , `UnsupportedVoice` ,  
`UnsupportedEmotion` }  
*Indicates all possible error conditions found during the processing of the URLs generation.*
- enum `Voice` {  
`NoVoice` = -1 , `Zahar` , `Ermil` , `Jane` ,  
`Oksana` , `Alyss` , `Omazh` }  
*Defines voice to use.*

### Public Member Functions

- `DOnlineTts` (QObject \*const parent=nullptr)  
*Create object.*
- `TtsError error` () const  
*Last error.*
- `QString errorString` () const  
*Last error string.*
- void `generateUrls` (const `QString` &text, `DOnlineTranslator::Engine` engine, `DOnlineTranslator::Language` lang, `Voice voice`=NoVoice, `Emotion emotion`=NoEmotion)  
*Create TTS urls.*
- `QList< QUrl > media` () const  
*Generated media.*

### Static Public Member Functions

- static `Emotion emotion` (const `QString` &emotionCode)  
*Emotion from code.*
- static `QString emotionCode` (`Emotion emotion`)  
*Code of the emotion.*
- static `Voice voice` (const `QString` &voiceCode)  
*Voice from code.*
- static `QString voiceCode` (`Voice voice`)  
*Code of the voice.*

## 9.416.1 Detailed Description

### Example:

```
DOnlineTts tts;
tts.generateUrls(QLatin1String("Hello World!"), DOnlineTranslator::Google, DOnlineTranslator::English);

// Get list of Urls to play with media player.
QList<QUrl> urls = tts.media();
```

## 9.416.2 Member Enumeration Documentation

### 9.416.2.1 Emotion

```
enum Digikam::DOnlineTts::Emotion
```

Used only by Yandex.

### 9.416.2.2 TtsError

```
enum Digikam::DOnlineTts::TtsError
```

**Enumerator**

NoError	No error condition.
UnsupportedEngine	Specified engine does not support TTS.
UnsupportedLanguage	Unsupported language by specified engine.
UnsupportedVoice	Unsupported voice by specified engine.
UnsupportedEmotion	Unsupported emotion by specified engine.

**9.416.2.3 Voice**

```
enum Digikam::DOnlineTts::Voice
```

Used only by Yandex.

**9.416.3 Constructor & Destructor Documentation****9.416.3.1 DOnlineTts()**

```
Digikam::DOnlineTts::DOnlineTts (
    QObject *const parent = nullptr ) [explicit]
```

Constructs an object with empty data and with parent. You can use [generateUrls\(\)](#) to create URLs for use in QMediaPlayer.

**Parameters**

<i>parent</i>	the parent object
---------------	-------------------

**9.416.4 Member Function Documentation****9.416.4.1 emotion()**

```
DOnlineTts::Emotion Digikam::DOnlineTts::emotion (
    const QString & emotionCode ) [static]
```

Used only by Yandex.

**Parameters**

<i>emotionCode</i>	emotion code
--------------------	--------------

**Returns**

corresponding emotion



### 9.416.4.2 emotionCode()

```
QString Digikam::DOnlineTts::emotionCode (
    Emotion emotion ) [static]
```

Used only by Yandex.

#### Parameters

<i>emotion</i>	the emotion to use
----------------	--------------------

#### Returns

code for emotion

### 9.416.4.3 error()

```
DOnlineTts::TtsError Digikam::DOnlineTts::error ( ) const
```

Error that was found during the generating tts. If no error was found, returns [TtsError::NoError](#). The text of the error can be obtained by [errorString\(\)](#).

#### Returns

last error

### 9.416.4.4 errorString()

```
QString Digikam::DOnlineTts::errorString ( ) const
```

A human-readable description of the last tts URL generation error that occurred.

#### Returns

last error string

### 9.416.4.5 generateUrls()

```
void Digikam::DOnlineTts::generateUrls (
    const QString & text,
    DOnlineTranslator::Engine engine,
    DOnlineTranslator::Language lang,
    Voice voice = NoVoice,
    Emotion emotion = NoEmotion )
```

Splits text into parts (engines have a limited number of characters per request) and returns list with the generated API URLs to play.

**Parameters**

<i>text</i>	the text to speak
<i>engine</i>	online translation engine
<i>lang</i>	text language
<i>voice</i>	the voice to use (used only by Yandex)
<i>emotion</i>	the emotion to use (used only by Yandex)

**9.416.4.6 media()**

```
QList< QUrl > Digikam::DOnlineTts::media ( ) const
```

**Returns**

List of generated URLs

**9.416.4.7 voice()**

```
DOnlineTts::Voice Digikam::DOnlineTts::voice (
    const QString & voiceCode ) [static]
```

Used only by Yandex.

**Parameters**

<i>voiceCode</i>	voice code
------------------	------------

**Returns**

corresponding voice

**9.416.4.8 voiceCode()**

```
QString Digikam::DOnlineTts::voiceCode (
    Voice voice ) [static]
```

**Parameters**

<i>voice</i>	the voice to use
--------------	------------------

## Returns

code for voice

## 9.417 Digikam::DownloadInfo Class Reference

### Public Member Functions

- **DownloadInfo** (const [DownloadInfo](#) &other)
- **DownloadInfo** (const QString &\_path, const QString &\_name, const QString &\_hash, const qint64 &\_size)
- **DownloadInfo** & **operator=** (const [DownloadInfo](#) &other)

### Public Attributes

- QString **hash**  
*The file hash as SHA256.*
- QString **name**  
*The file name on the server.*
- QString **path**  
*The file path on the server.*
- qint64 **size** = 0  
*The file size.*

## 9.418 Digikam::DownloadSettings Class Reference

### Public Attributes

- bool **autoRotate** = true  
*Settings from [AdvancedSettings](#) widget.*
- bool **backupRaw** = false
- int **colorLabel** = NoColorLabel  
*Pre-colorLabel of each camera file.*
- bool **compressDng** = true
- bool **convertDng** = false  
*Settings from DNG convert widget.*
- bool **convertJpeg** = false
- QString **dest**
- bool **documentName** = false
- QString **file**
- bool **fixDateTime** = false
- QString **folder**  
*File path to download.*
- QString **losslessFormat**  
*New format to convert Jpeg files.*
- QString **mime**  
*Mime type from file to download.*
- QDateTime **newDateTime**
- int **pickLabel** = NoPickLabel  
*Pre-pickLabel of each camera file.*

- int **previewMode** = [DNGWriter::FULL\\_SIZE](#)
- int **rating** = NoRating  
*Pre-rating of each camera file.*
- QString **script**  
*Settings from [ScriptingSettings](#) widget.*
- QList< int > **tagIds**  
*Pre-tags of each camera file.*
- QString **templateTitle**  
*Metadata template title.*

## 9.419 Digikam::DPixelsAliasFilter Class Reference

### Public Member Functions

- void [pixelAntiAliasing](#) (uchar \*const data, int Width, int Height, double X, double Y, uchar \*const A, uchar \*const R, uchar \*const G, uchar \*const B)  
*Function to perform pixel antialiasing with 8 bits/color/pixel images.*
- void [pixelAntiAliasing16](#) (unsigned short \*const data, int Width, int Height, double X, double Y, unsigned short \*const A, unsigned short \*const R, unsigned short \*const G, unsigned short \*const B)  
*Function to perform pixel antialiasing with 16 bits/color/pixel images.*

### 9.419.1 Member Function Documentation

#### 9.419.1.1 pixelAntiAliasing()

```
void Digikam::DPixelsAliasFilter::pixelAntiAliasing (
    uchar *const data,
    int Width,
    int Height,
    double X,
    double Y,
    uchar *const A,
    uchar *const R,
    uchar *const G,
    uchar *const B )
```

This method is used to smooth target image in transformation method like free rotation or shear tool.

#### 9.419.1.2 pixelAntiAliasing16()

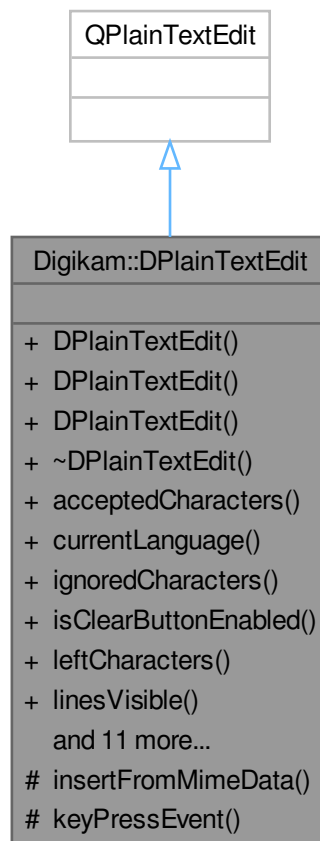
```
void Digikam::DPixelsAliasFilter::pixelAntiAliasing16 (
    unsigned short *const data,
    int Width,
    int Height,
    double X,
    double Y,
    unsigned short *const A,
    unsigned short *const R,
    unsigned short *const G,
    unsigned short *const B )
```

This method is used to smooth target image in transformation method like free rotation or shear tool.

## 9.420 Digikam::DPlainTextEdit Class Reference

A text edit widget based on QPlainTextEdit with spell checker capabilities based on Sonnet (optional).

Inheritance diagram for Digikam::DPlainTextEdit:



### Signals

- void `returnPressed()`  
*Emitted only when mimic `QLineEdit` mode is enabled.*
- void `textEdited(const QString &)`

### Public Member Functions

- **`DPlainTextEdit`** (const `QString` &contents, `QWidget` \*const parent=nullptr)  
*Constructor with text contents to use.*
- **`DPlainTextEdit`** (`QWidget` \*const parent=nullptr)  
*Default constructor.*
- **`DPlainTextEdit`** (unsigned int lines, `QWidget` \*const parent=nullptr)

- Constructor with a number of lines.
- `~DPlainTextEdit ()` override
  - Standard destructor.
- `QString acceptedCharacters ()` const
  - This property holds whether the edit widget handle the mask of accepted characters in text editor.
- `QString currentLanguage ()` const
- `QString ignoredCharacters ()` const
  - This property holds whether the edit widget handle the mask of ignored characters in text editor.
- `bool isClearButtonEnabled ()` const
  - This property holds whether the edit widget displays a clear button when it is not empty.
- `int leftCharacters ()` const
  - Return the left characters that user can enter if a limit have been previously set with `setMaxLeght()`.
- `unsigned int linesVisible ()` const
- `int maxLength ()` const
- `void setAcceptedCharacters (const QString &mask)`
- `void setClearButtonEnabled (bool enable)`
- `void setCurrentLanguage (const QString &lang)`
  - This property holds whether the edit widget handle a specific spell-checker language (2 letters code based as "en", "fr", "es", etc.).
- `void setIgnoredCharacters (const QString &mask)`
- `void setLinesVisible (unsigned int lines)`
  - This property holds whether the edit widget handle visible lines used by the widget to show text.
- `void setLocalizeSettings (const LocalizeContainer &settings)`
- `void setMaxLength (int length)`
  - This property holds whether the edit widget handle the maximum of characters that user can enter in editor.
- `void setText (const QString &text)`
- `LocalizeContainer spellCheckSettings ()` const
  - This property holds whether the edit widget handle the Spellcheck settings.
- `QString text ()` const
  - This property holds whether the edit widget handle text contents as plain text.

## Protected Member Functions

- `void insertFromMimeData (const QMimeData *source)` override
- `void keyPressEvent (QKeyEvent *e)` override

### 9.420.1 Detailed Description

Widget size can be constrained with the number of visible lines. A single line constraint will emulate `QLineEdit`. See `setLinesVisible()` for details. The maximum number of characters can be limited with `setMaxLenght()`. The characters can be limited in editor by `setIgnoredCharacters()` and `setAcceptedCharacters()`. Implementation: `dplaintextedit.cpp`

### 9.420.2 Constructor & Destructor Documentation

#### 9.420.2.1 DPlainTextEdit()

```
Digikam::DPlainTextEdit::DPlainTextEdit (
    unsigned int lines,
    QWidget *const parent = nullptr ) [explicit]
```

Zero lines do not apply a size constraint.

## 9.420.3 Member Function Documentation

### 9.420.3.1 `acceptedCharacters()`

```
QString Digikam::DPlainTextEdit::acceptedCharacters ( ) const
```

The mask of characters is passed as string (ex: "abcABC"). By default the mask is empty.

### 9.420.3.2 `ignoredCharacters()`

```
QString Digikam::DPlainTextEdit::ignoredCharacters ( ) const
```

The mask of characters is passed as string (ex: "+/!()"). By default the mask is empty.

### 9.420.3.3 `isClearButtonEnabled()`

```
bool Digikam::DPlainTextEdit::isClearButtonEnabled ( ) const
```

If enabled, the edit widget displays a trailing clear button when it contains some text, otherwise the edit widget does not show a clear button. This option only take effect in QLineEdit emulation mode when lines visible is set to 1. See [setLinesVisible\(\)](#) for details.

### 9.420.3.4 `returnPressed`

```
void Digikam::DPlainTextEdit::returnPressed ( ) [signal]
```

See [setLinesVisible\(\)](#) for details.

### 9.420.3.5 `setCurrentLanguage()`

```
void Digikam::DPlainTextEdit::setCurrentLanguage (
    const QString & lang )
```

If this property is not set, spell-checker will try to auto-detect language by parsing the text. To reset this setting, pass a empty string as language. If Sonnet dependencies is not resolved, these method do nothing.

### 9.420.3.6 `setLinesVisible()`

```
void Digikam::DPlainTextEdit::setLinesVisible (
    unsigned int lines )
```

Lines must be superior or egal to 1 to apply a size constraint. Notes: if a single visible line is used, the widget emulate QLineEdit. a null value do not apply a size constraint.

### 9.420.3.7 setMaxLength()

```
void Digikam::DPlainTextEdit::setMaxLength (
    int length )
```

By default no limit is set. A zero length reset a limit.

### 9.420.3.8 spellCheckSettings()

```
LocalizeContainer Digikam::DPlainTextEdit::spellCheckSettings ( ) const
```

See [LocalizeContainer](#) class for details.

### 9.420.3.9 text()

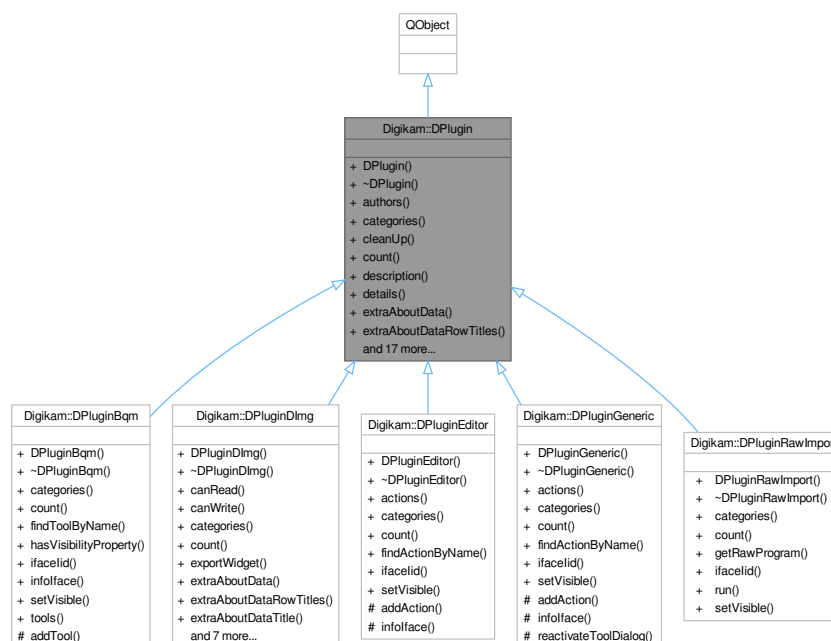
```
QString Digikam::DPlainTextEdit::text ( ) const
```

If ignored or accepted characters masks are set, text is filtered accordingly.

## 9.421 Digikam::DPlugin Class Reference

A digiKam external plugin abstract class.

Inheritance diagram for Digikam::DPlugin:





## Public Member Functions

- **DPlugin** (QObject \*const parent=nullptr)  
*Constructor with optional parent object.*
- **~DPlugin** () override  
*Destructor.*
- virtual QList< **DPluginAuthor** > **authors** () const =0  
*Returns authors list for the plugin.*
- virtual QStringList **categories** () const =0  
*Return a list of categories as strings registered in this plugin.*
- virtual void **cleanUp** ()  
*Plugin method to clean up internal created objects.*
- virtual int **count** () const =0  
*Return the amount of tools registered to all parents.*
- virtual QString **description** () const =0  
*Returns a short description about the plugin.*
- virtual QString **details** () const =0  
*Returns a long description about the plugin.*
- virtual QMap< QString, QStringList > **extraAboutData** () const  
*Returns a map of extra data to show in plugin about dialog.*
- virtual QStringList **extraAboutDataRowTitles** () const  
*Returns a list of extra data row titles to show in tab of plugin about dialog.*
- virtual QString **extraAboutDataTitle** () const  
*Returns the tab title of data returned by [extraAboutData\(\)](#).*
- virtual QString **handbookChapter** () const  
*Return the online handbook chapter from an handbook section corresponding to this plugin.*
- virtual QString **handbookReference** () const  
*Return the online handbook reference from an handbook chapter corresponding to this plugin.*
- virtual QString **handbookSection** () const  
*Return the online handbook section corresponding to this plugin.*
- virtual bool **hasVisibilityProperty** () const  
*Return true if plugin can be configured in setup dialog about the visibility property.*
- virtual QIcon **icon** () const  
*Returns an icon for the plugin.*
- virtual QString **ifacelid** () const =0  
*Returns the unique top level internal identification property of the plugin interface.*
- virtual QString **iid** () const =0  
*Returns the unique internal identification property of the plugin.*
- QString **libraryFileName** () const  
*Returns the file name of the library for this plugin.*
- virtual QString **name** () const =0  
*Returns the user-visible name of the plugin.*
- QStringList **pluginAuthors** () const  
*Return a list of authors as strings registered in this plugin.*
- void **setLibraryFileName** (const QString &)  
*Sets the file name of the library for this plugin.*
- void **setShouldLoaded** (bool b)  
*Accessor to adjust the should loaded plugin property.*
- virtual void **setup** (QObject \*const parent)=0  
*Plugin factory method to create all internal object instances for a given parent.*
- virtual void **setVisible** (bool b)=0

*Holds whether the plugin can be seen in parent view.*

- bool [shouldLoaded](#) () const

*Return the should loaded property.*

- QString [version](#) () const

*Return the internal version used to check the binary compatibility at run-time.*

## 9.421.1 Member Function Documentation

### 9.421.1.1 categories()

```
virtual QStringList Digikam::DPlugin::categories ( ) const [pure virtual]
```

Implemented in [Digikam::DPluginDImg](#), [Digikam::DPluginEditor](#), [Digikam::DPluginGeneric](#), [Digikam::DPluginRawImport](#), and [Digikam::DPluginBqm](#).

### 9.421.1.2 cleanUp()

```
virtual void Digikam::DPlugin::cleanUp ( ) [inline], [virtual]
```

This method is called by plugin loader.

### 9.421.1.3 count()

```
virtual int Digikam::DPlugin::count ( ) const [pure virtual]
```

Implemented in [Digikam::DPluginDImg](#), [Digikam::DPluginEditor](#), [Digikam::DPluginGeneric](#), [Digikam::DPluginRawImport](#), and [Digikam::DPluginBqm](#).

### 9.421.1.4 extraAboutData()

```
virtual QMap< QString, QStringList > Digikam::DPlugin::extraAboutData ( ) const [inline], [virtual]
```

Reimplemented in [Digikam::DPluginDImg](#).

### 9.421.1.5 extraAboutDataRowTitles()

```
virtual QStringList Digikam::DPlugin::extraAboutDataRowTitles ( ) const [inline], [virtual]
```

Reimplemented in [Digikam::DPluginDImg](#).

### 9.421.1.6 extraAboutDataTitle()

```
virtual QString Digikam::DPlugin::extraAboutDataTitle ( ) const [inline], [virtual]
```

Reimplemented in [Digikam::DPluginDImg](#).

### 9.421.1.7 handbookChapter()

```
QString Digikam::DPlugin::handbookChapter ( ) const [virtual]
```

It's used in plugin dialog Help button. By default, no chapter is defined, and root page of the section is loaded by Help Button in this case. Note: a chapter is always included in a section. See [handbookSection\(\)](#) for details.

### 9.421.1.8 handbookReference()

```
QString Digikam::DPlugin::handbookReference ( ) const [virtual]
```

It's used in plugin dialog Help button. By default, no reference is defined, and root page of the chapter is loaded by Help Button in this case. Note: a reference is always included in a chapter. See [handbookChapter\(\)](#) for details.

### 9.421.1.9 handbookSection()

```
QString Digikam::DPlugin::handbookSection ( ) const [virtual]
```

It's used in plugin dialog Help button. By default, no section is defined, and root page of the documentation is loaded by Help Button in this case.

### 9.421.1.10 hasVisibilityProperty()

```
bool Digikam::DPlugin::hasVisibilityProperty ( ) const [virtual]
```

Default implementation return true.

Reimplemented in [Digikam::DPluginDImg](#), and [Digikam::DPluginBqm](#).

### 9.421.1.11 icon()

```
QIcon Digikam::DPlugin::icon ( ) const [virtual]
```

Default implementation return the system plugin icon.

### 9.421.1.12 ifaceId()

```
virtual QString Digikam::DPlugin::ifaceId ( ) const [pure virtual]
```

Must be formatted as "org.kde.digikam.\_NAME\_OF\_INTERFACE\_/\_VERSION\_". Examples: "org.kde.digikam.↔  
DPluginGeneric/1.1.0" "org.kde.digikam.DPluginEditor/1.1.0" "org.kde.digikam.DPluginBqm/1.1.0"

Implemented in [Digikam::DPluginDImg](#), [Digikam::DPluginEditor](#), [Digikam::DPluginGeneric](#), [Digikam::DPluginRawImport](#), and [Digikam::DPluginBqm](#).

#### 9.421.1.13 iid()

```
virtual QString Digikam::DPlugin::iid ( ) const [pure virtual]
```

Must be formatted as "org.kde.digikam.plugin.\_PLUGIN\_TYPE\_.\_NAME\_OF\_PLUGIN\_". Examples: "org.kde.digikam.plugin.generic.Calendar" "org.kde.digikam.plugin.editor.AdjustCurvesTool" "org.kde.digikam.plugin.bqm.NoiseReduction"

#### 9.421.1.14 libraryFileName()

```
QString Digikam::DPlugin::libraryFileName ( ) const
```

This string is filled at run-time by plugin loader.

#### 9.421.1.15 name()

```
virtual QString Digikam::DPlugin::name ( ) const [pure virtual]
```

The user-visible name should be context free, i.e. the name should provide enough information as to what the plugin is about in the context of digiKam.

#### 9.421.1.16 setLibraryFileName()

```
void Digikam::DPlugin::setLibraryFileName (
    const QString & name )
```

This string is filled at run-time by plugin loader.

#### 9.421.1.17 setShouldLoaded()

```
void Digikam::DPlugin::setShouldLoaded (
    bool b )
```

This property is adjusted by plugin loader at start-up.

#### 9.421.1.18 setVisible()

```
virtual void Digikam::DPlugin::setVisible (
    bool b ) [pure virtual]
```

Implemented in [Digikam::DPluginEditor](#), [Digikam::DPluginGeneric](#), [Digikam::DPluginBqm](#), [Digikam::DPluginDImg](#), and [Digikam::DPluginRawImport](#).

#### 9.421.1.19 shouldLoaded()

```
bool Digikam::DPlugin::shouldLoaded ( ) const
```

If it's true, the plugin must be loaded in application GUI at startup by plugin loader.

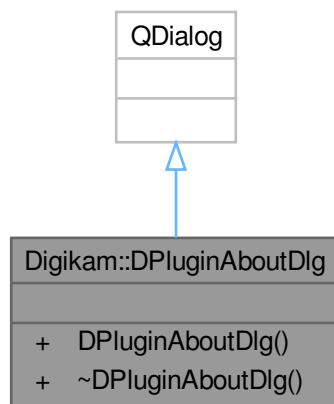
### 9.421.1.20 version()

```
QString Digikam::DPlugin::version ( ) const
```

This is typically the same version of digiKam core used at compilation time.

## 9.422 Digikam::DPluginAboutDlg Class Reference

Inheritance diagram for Digikam::DPluginAboutDlg:

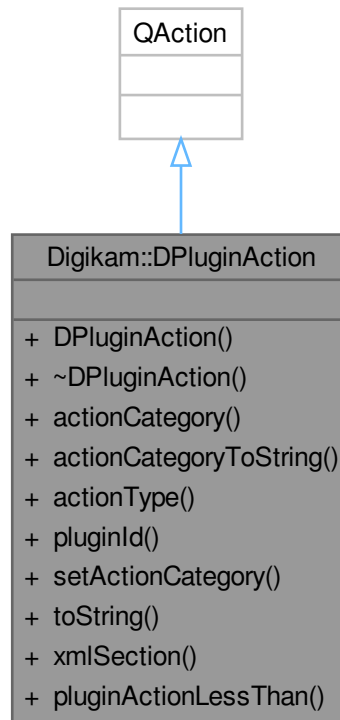


### Public Member Functions

- `DPluginAboutDlg` (`DPlugin` \*const tool, `QWidget` \*const parent=nullptr)

## 9.423 Digikam::DPluginAction Class Reference

Inheritance diagram for Digikam::DPluginAction:



### Public Types

- enum `ActionCategory` { `InvalidCat` = -1 , `GenericExport` = 0 , `GenericImport` , `GenericTool` , `GenericMetadata` , `GenericView` , `EditorFile` , `EditorColors` , `EditorEnhance` , `EditorTransform` , `EditorDecorate` , `EditorFilters` }  
*Plugin action categories.*
- enum `ActionType` { `InvalidType` = -1 , `Generic` = 0 , `Editor` }  
*Plugin action types to resume where they can be used.*
- enum `PluginActionData` { `NoData` = 0 , `AlbumData` }  
*Plugin action types via QAction data container.*

### Public Member Functions

- `DPluginAction` (`QObject *const parent=nullptr`)
- `ActionCategory` `actionCategory` () const
- `QString` `actionCategoryToString` () const
- `ActionType` `actionType` () const  
*Return the action type depending of category.*

- QString **pluginId** () const  
*Return the plugin id string hosting this action.*
- void **setActionCategory** (ActionCategory cat)  
*Manage the internal action category.*
- QString **toString** () const  
*Return details as string about action properties.*
- QString **xmlSection** () const  
*Return the XML section to merge in KXMLGUIClient host XML definition.*

### Static Public Member Functions

- static bool **pluginActionLessThan** (DPluginAction \*const a, DPluginAction \*const b)

## 9.423.1 Member Enumeration Documentation

### 9.423.1.1 ActionCategory

```
enum Digikam::DPluginAction::ActionCategory
```

#### Enumerator

GenericExport	Generic export action.
GenericImport	Generic import action.
GenericTool	Generic processing action.
GenericMetadata	Generic Metadata adjustment action.
GenericView	Generic View action (as Slideshow).
EditorFile	Image Editor file action.
EditorColors	Image Editor color correction action.
EditorEnhance	Image Editor enhance action.
EditorTransform	Image Editor transform action.
EditorDecorate	Image Editor decorate action.
EditorFilters	Image Editor special effects action.

### 9.423.1.2 ActionType

```
enum Digikam::DPluginAction::ActionType
```

#### Enumerator

InvalidType	An invalid action category.
Generic	Generic action available everywhere (AlbumView, Editor, and LightTable).
Editor	Specific action for Image Editor and Showfoto.

## 9.423.2 Member Function Documentation

### 9.423.2.1 toString()

```
QString Digikam::DPluginAction::toString ( ) const
```

For debug purpose only.

## 9.424 Digikam::DPluginAuthor Class Reference

### Public Member Functions

- **DPluginAuthor** (const QString &\_name, const QString &\_email, const QString &\_year, const QString &\_role)
- **DPluginAuthor** (const QString &\_name, const QString &\_email, const QString &\_year)
- QString [toString](#) ( ) const

*Return author details as string.*

### Public Attributes

- QString **email**  
*Email anti-spammed.*
- QString **name**  
*Author name and surname.*
- QString **roles**  
*Author roles, as "Developer", "Designer", "Translator", etc.*
- QString **years**  
*Copyrights years.*

## 9.424.1 Member Function Documentation

### 9.424.1.1 toString()

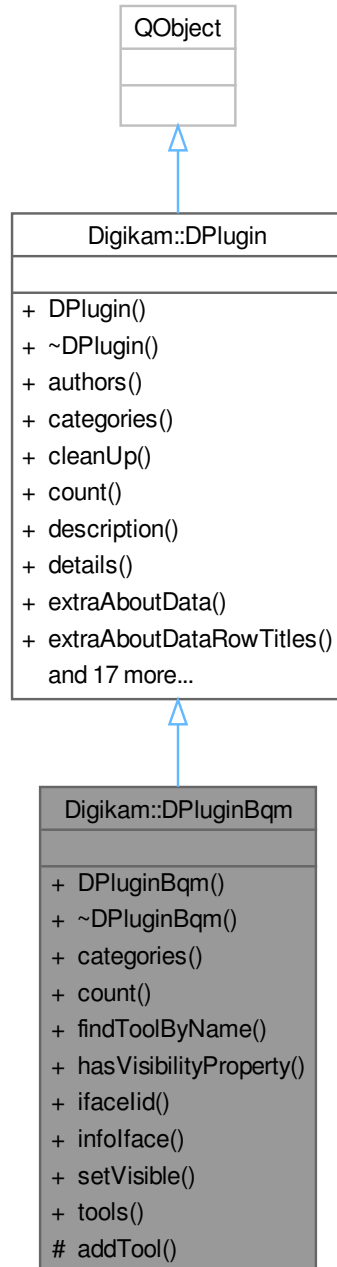
```
QString Digikam::DPluginAuthor::toString ( ) const
```

For debug purpose only.



## 9.425 Digikam::DPluginBqm Class Reference

Inheritance diagram for Digikam::DPluginBqm:



### Signals

- void **signalVisible** (bool)

## Public Member Functions

- **DPluginBqm** (QObject \*const parent=nullptr)  
*Constructor with optional parent object.*
- **~DPluginBqm** () override  
*Destructor.*
- QStringList **categories** () const override  
*Return a list of batch tool group categories as strings registered in this plugin.*
- int **count** () const override  
*Return the amount of tools registered.*
- **BatchTool** \* **findToolByName** (const QString &name, QObject \*const parent) const  
*Return a plugin tool instance found by name in plugin tools list for a given parent.*
- bool **hasVisibilityProperty** () const override  
*Return true if plugin can be configured in setup dialog about the visibility property.*
- QString **ifacelid** () const override  
*Return the plugin interface identifier.*
- **BqmlInfoface** \* **infoface** () const  
*Return the info interface instance.*
- void **setVisible** (bool b) override  
*Holds whether the plugin can be seen in parent view.*
- QList< **BatchTool** \* > **tools** (QObject \*const parent) const  
*Return all plugin tools registered in **setup()** method with **addTool()** for a given parent.*

## Public Member Functions inherited from **Digikam::DPlugin**

- **DPlugin** (QObject \*const parent=nullptr)  
*Constructor with optional parent object.*
- **~DPlugin** () override  
*Destructor.*
- virtual QList< **DPluginAuthor** > **authors** () const =0  
*Returns authors list for the plugin.*
- virtual void **cleanUp** ()  
*Plugin method to clean up internal created objects.*
- virtual QString **description** () const =0  
*Returns a short description about the plugin.*
- virtual QString **details** () const =0  
*Returns a long description about the plugin.*
- virtual QMap< QString, QStringList > **extraAboutData** () const  
*Returns a map of extra data to show in plugin about dialog.*
- virtual QStringList **extraAboutDataRowTitles** () const  
*Returns a list of extra data row titles to show in tab of plugin about dialog.*
- virtual QString **extraAboutDataTitle** () const  
*Returns the tab title of data returned by **extraAboutData()**.*
- virtual QString **handbookChapter** () const  
*Return the online handbook chapter from an handbook section corresponding to this plugin.*
- virtual QString **handbookReference** () const  
*Return the online handbook reference from an handbook chapter corresponding to this plugin.*
- virtual QString **handbookSection** () const  
*Return the online handbook section corresponding to this plugin.*
- virtual QIcon **icon** () const

- Returns an icon for the plugin.*

  - virtual QString `iid` () const =0

*Returns the unique internal identification property of the plugin.*
- QString `libraryFileName` () const

*Returns the file name of the library for this plugin.*
- virtual QString `name` () const =0

*Returns the user-visible name of the plugin.*
- QStringList `pluginAuthors` () const

*Return a list of authors as strings registered in this plugin.*
- void `setLibraryFileName` (const QString &)

*Sets the file name of the library for this plugin.*
- void `setShouldLoaded` (bool b)

*Accessor to adjust the should loaded plugin property.*
- virtual void `setup` (QObject \*const parent)=0

*Plugin factory method to create all internal object instances for a given parent.*
- bool `shouldLoaded` () const

*Return the should loaded property.*
- QString `version` () const

*Return the internal version used to check the binary compatibility at run-time.*

### Protected Member Functions

- void `addTool` ([BatchTool](#) \*const t)

## 9.425.1 Member Function Documentation

### 9.425.1.1 categories()

```
QStringList Digikam::DPluginBqm::categories ( ) const [override], [virtual]
```

Implements [Digikam::DPlugin](#).

### 9.425.1.2 count()

```
int Digikam::DPluginBqm::count ( ) const [override], [virtual]
```

Implements [Digikam::DPlugin](#).

### 9.425.1.3 hasVisibilityProperty()

```
bool Digikam::DPluginBqm::hasVisibilityProperty ( ) const [override], [virtual]
```

Default implementation return true.

Reimplemented from [Digikam::DPlugin](#).

### 9.425.1.4 ifaceId()

```
QString Digikam::DPluginBqm::ifaceId ( ) const [override], [virtual]
```

Implements [Digikam::DPlugin](#).

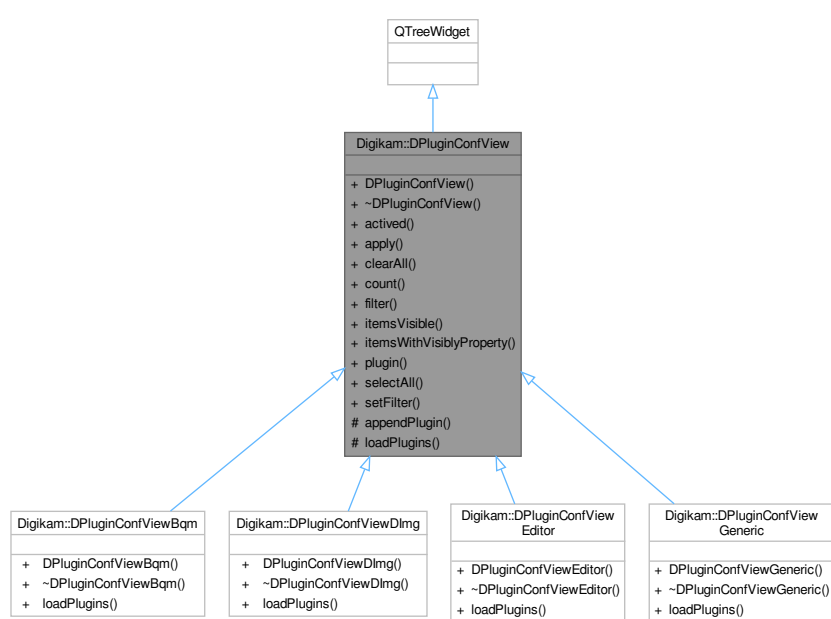
### 9.425.1.5 setVisible()

```
void Digikam::DPluginBqm::setVisible (
    bool b ) [override], [virtual]
```

Implements [Digikam::DPlugin](#).

## 9.426 Digikam::DPluginConfView Class Reference

Inheritance diagram for Digikam::DPluginConfView:



### Signals

- void [signalSearchResult](#) (int)  
*Signal emitted when filtering is done through `slotSetFilter()`.*

## Public Member Functions

- **DPluginConfView** (QWidget \*const parent=nullptr)  
*Default constructor.*
- int **activated** () const  
*Return the number of plugins active in the list.*
- void **apply** ()  
*Apply all changes about plugins selected to be hosted in host application.*
- void **clearAll** ()  
*Clear all selected plugins in the list.*
- int **count** () const  
*Return the total number of plugins in the list.*
- QString **filter** () const  
*Return the current string used to filter the plugins list.*
- int **itemsVisible** () const  
*Return the number of visible plugins in the list.*
- int **itemsWithVisiblyProperty** () const  
*Return the number of plugins in the list with visibly properties available.*
- **DPlugin** \* **plugin** (QTreeWidgetItem \*const item) const
- void **selectAll** () override  
*Select all plugins in the list.*
- void **setFilter** (const QString &filter, Qt::CaseSensitivity cs)  
*Set the string used to filter the plugins list.*

## Protected Member Functions

- QTreeWidgetItem \* **appendPlugin** (DPlugin \*const)
- virtual void **loadPlugins** ()=0

## 9.426.1 Member Function Documentation

### 9.426.1.1 setFilter()

```
void Digikam::DPluginConfView::setFilter (
    const QString & filter,
    Qt::CaseSensitivity cs )
```

[signalSearchResult\(\)](#) is emitted when all is done.

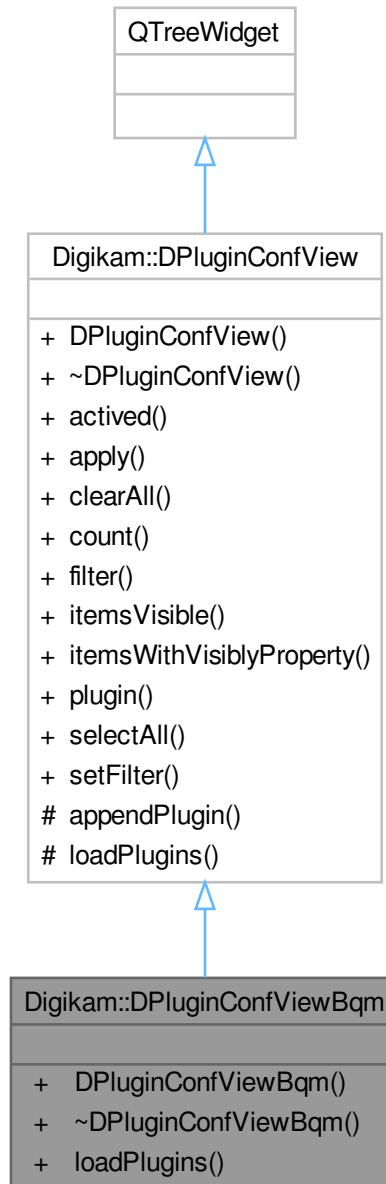
### 9.426.1.2 signalSearchResult

```
void Digikam::DPluginConfView::signalSearchResult (
    int ) [signal]
```

Number of plugins found is sent when item relevant of filtering match the query.

## 9.427 Digikam::DPluginConfViewBqm Class Reference

Inheritance diagram for Digikam::DPluginConfViewBqm:



### Public Member Functions

- **DPluginConfViewBqm** (`QWidget *const parent=nullptr`)
- void `loadPlugins ()` override

## Public Member Functions inherited from Digikam::DPluginConfView

- **DPluginConfView** (QWidget \*const parent=nullptr)  
*Default constructor.*
- int **activated** () const  
*Return the number of plugins active in the list.*
- void **apply** ()  
*Apply all changes about plugins selected to be hosted in host application.*
- void **clearAll** ()  
*Clear all selected plugins in the list.*
- int **count** () const  
*Return the total number of plugins in the list.*
- QString **filter** () const  
*Return the current string used to filter the plugins list.*
- int **itemsVisible** () const  
*Return the number of visible plugins in the list.*
- int **itemsWithVisiblyProperty** () const  
*Return the number of plugins in the list with visibly properties available.*
- **DPlugin** \* **plugin** (QTreeWidgetItem \*const item) const
- void **selectAll** () override  
*Select all plugins in the list.*
- void **setFilter** (const QString &filter, Qt::CaseSensitivity cs)  
*Set the string used to filter the plugins list.*

## Additional Inherited Members

## Signals inherited from Digikam::DPluginConfView

- void **signalSearchResult** (int)  
*Signal emitted when filtering is done through slotSetFilter().*

## Protected Member Functions inherited from Digikam::DPluginConfView

- QTreeWidgetItem \* **appendPlugin** (DPlugin \*const)

## 9.427.1 Member Function Documentation

### 9.427.1.1 loadPlugins()

```
void Digikam::DPluginConfViewBqm::loadPlugins ( ) [override], [virtual]
```

Implements [Digikam::DPluginConfView](#).

## 9.428 Digikam::DPluginConfViewDImg Class Reference

Inheritance diagram for Digikam::DPluginConfViewDImg:



### Public Member Functions

- **DPluginConfViewDImg** (`QWidget *const parent=nullptr`)
- void `loadPlugins ()` override



## Public Member Functions inherited from Digikam::DPluginConfView

- **DPluginConfView** (QWidget \*const parent=nullptr)  
*Default constructor.*
- int **activated** () const  
*Return the number of plugins active in the list.*
- void **apply** ()  
*Apply all changes about plugins selected to be hosted in host application.*
- void **clearAll** ()  
*Clear all selected plugins in the list.*
- int **count** () const  
*Return the total number of plugins in the list.*
- QString **filter** () const  
*Return the current string used to filter the plugins list.*
- int **itemsVisible** () const  
*Return the number of visible plugins in the list.*
- int **itemsWithVisiblyProperty** () const  
*Return the number of plugins in the list with visibly properties available.*
- **DPlugin** \* **plugin** (QTreeWidgetItem \*const item) const
- void **selectAll** () override  
*Select all plugins in the list.*
- void **setFilter** (const QString &filter, Qt::CaseSensitivity cs)  
*Set the string used to filter the plugins list.*

## Additional Inherited Members

## Signals inherited from Digikam::DPluginConfView

- void **signalSearchResult** (int)  
*Signal emitted when filtering is done through slotSetFilter().*

## Protected Member Functions inherited from Digikam::DPluginConfView

- QTreeWidgetItem \* **appendPlugin** (DPlugin \*const)

## 9.428.1 Member Function Documentation

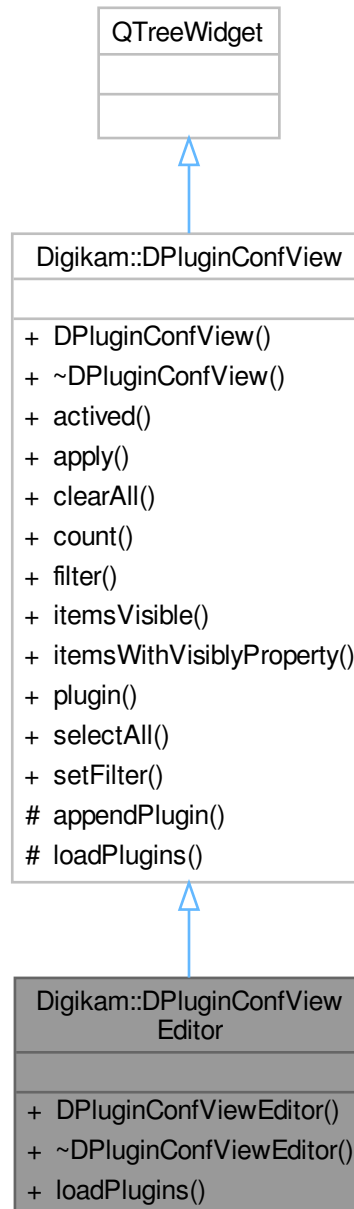
### 9.428.1.1 loadPlugins()

```
void Digikam::DPluginConfViewDImg::loadPlugins ( ) [override], [virtual]
```

Implements [Digikam::DPluginConfView](#).

## 9.429 Digikam::DPluginConfViewEditor Class Reference

Inheritance diagram for Digikam::DPluginConfViewEditor:



### Public Member Functions

- **DPluginConfViewEditor** (`QWidget *const parent=nullptr`)
- void `loadPlugins ()` override

## Public Member Functions inherited from [Digikam::DPluginConfView](#)

- **DPluginConfView** (QWidget \*const parent=nullptr)  
*Default constructor.*
- int **activated** () const  
*Return the number of plugins active in the list.*
- void **apply** ()  
*Apply all changes about plugins selected to be hosted in host application.*
- void **clearAll** ()  
*Clear all selected plugins in the list.*
- int **count** () const  
*Return the total number of plugins in the list.*
- QString **filter** () const  
*Return the current string used to filter the plugins list.*
- int **itemsVisible** () const  
*Return the number of visible plugins in the list.*
- int **itemsWithVisiblyProperty** () const  
*Return the number of plugins in the list with visibly properties available.*
- [DPlugin](#) \* **plugin** (QTreeWidgetItem \*const item) const
- void **selectAll** () override  
*Select all plugins in the list.*
- void **setFilter** (const QString &filter, Qt::CaseSensitivity cs)  
*Set the string used to filter the plugins list.*

## Additional Inherited Members

## Signals inherited from [Digikam::DPluginConfView](#)

- void [signalSearchResult](#) (int)  
*Signal emitted when filtering is done through slotSetFilter().*

## Protected Member Functions inherited from [Digikam::DPluginConfView](#)

- QTreeWidgetItem \* **appendPlugin** ([DPlugin](#) \*const)

## 9.429.1 Member Function Documentation

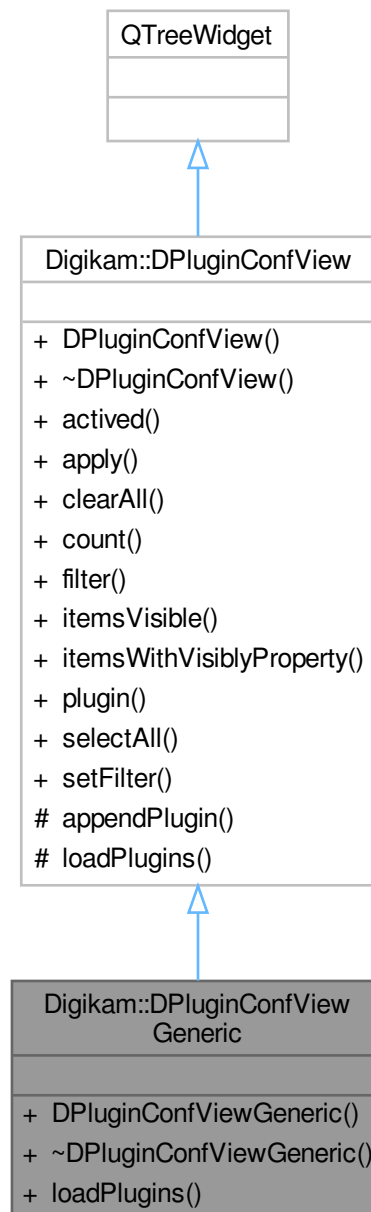
### 9.429.1.1 loadPlugins()

```
void Digikam::DPluginConfViewEditor::loadPlugins ( ) [override], [virtual]
```

Implements [Digikam::DPluginConfView](#).

## 9.430 Digikam::DPluginConfViewGeneric Class Reference

Inheritance diagram for Digikam::DPluginConfViewGeneric:



### Public Member Functions

- **DPluginConfViewGeneric** (QWidget \*const parent=nullptr)
- void [loadPlugins](#) () override

## Public Member Functions inherited from [Digikam::DPluginConfView](#)

- **DPluginConfView** (QWidget \*const parent=nullptr)  
*Default constructor.*
- int **activated** () const  
*Return the number of plugins active in the list.*
- void **apply** ()  
*Apply all changes about plugins selected to be hosted in host application.*
- void **clearAll** ()  
*Clear all selected plugins in the list.*
- int **count** () const  
*Return the total number of plugins in the list.*
- QString **filter** () const  
*Return the current string used to filter the plugins list.*
- int **itemsVisible** () const  
*Return the number of visible plugins in the list.*
- int **itemsWithVisiblyProperty** () const  
*Return the number of plugins in the list with visibly properties available.*
- [DPlugin](#) \* **plugin** (QTreeWidgetItem \*const item) const
- void **selectAll** () override  
*Select all plugins in the list.*
- void **setFilter** (const QString &filter, Qt::CaseSensitivity cs)  
*Set the string used to filter the plugins list.*

## Additional Inherited Members

## Signals inherited from [Digikam::DPluginConfView](#)

- void [signalSearchResult](#) (int)  
*Signal emitted when filtering is done through slotSetFilter().*

## Protected Member Functions inherited from [Digikam::DPluginConfView](#)

- QTreeWidgetItem \* **appendPlugin** ([DPlugin](#) \*const)

## 9.430.1 Member Function Documentation

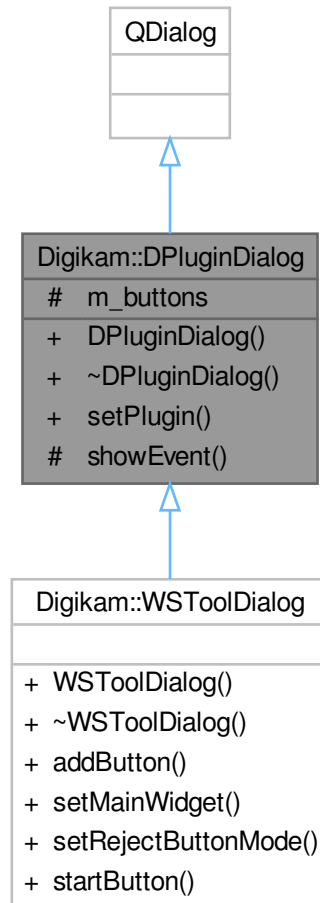
### 9.430.1.1 loadPlugins()

```
void Digikam::DPluginConfViewGeneric::loadPlugins ( ) [override], [virtual]
```

Implements [Digikam::DPluginConfView](#).

## 9.431 Digikam::DPluginDialog Class Reference

Inheritance diagram for Digikam::DPluginDialog:



### Public Member Functions

- **DPluginDialog** (QWidget \*const parent, const QString &objName)
- void **setPlugin** (DPlugin \*const tool)

### Protected Member Functions

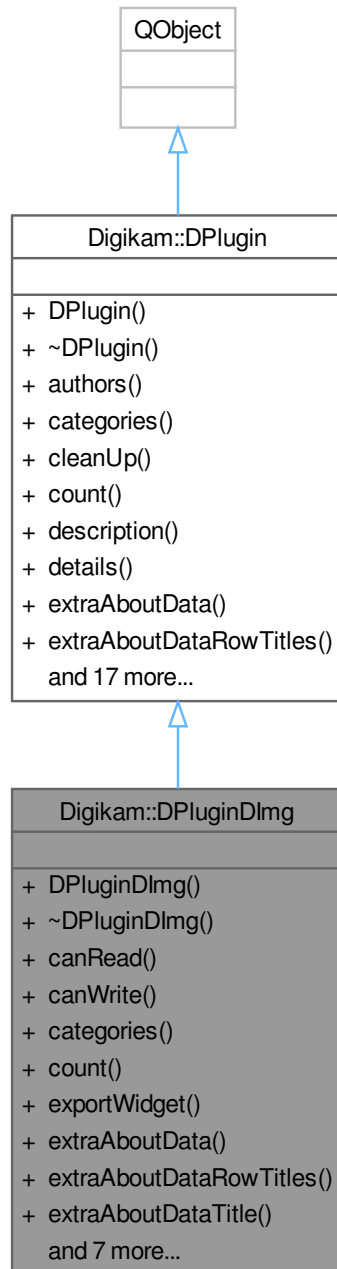
- void **showEvent** (QShowEvent \*) override

### Protected Attributes

- QDialogButtonBox \* **m\_buttons** = nullptr

## 9.432 Digikam::DPluginDImg Class Reference

Inheritance diagram for Digikam::DPluginDImg:



### Public Member Functions

- **DPluginDImg** (QObject \*const parent=nullptr)  
*Constructor with optional parent object.*

- `~DPluginDImg ()` override=default  
*Destructor.*
- virtual int `canRead (const QFileInfo &fileInfo, bool magic) const =0`  
*Return > 0 if source file path is supported by the loader and contents can be loaded.*
- virtual int `canWrite (const QString &format) const =0`  
*Return > 0 if target file format is supported by the loader and contents can be written.*
- QStringList `categories ()` const override  
*This kind of plugin do not use a category.*
- int `count ()` const override  
*This kind of plugin only provide one tool.*
- virtual `DImgLoaderSettings * exportWidget (const QString &format) const =0`  
*Return a new widget instance to show settings while exporting image to specified format.*
- QMap< QString, QStringList > `extraAboutData ()` const override  
*With this kind of plugin, we will display the type-mimes list on about dialog.*
- QStringList `extraAboutDataRowTitles ()` const override  
*Returns a list of extra data row titles to show in tab of plugin about dialog.*
- QString `extraAboutDataTitle ()` const override  
*Returns the tab title of data returned by `extraAboutData()`.*
- bool `hasVisibilityProperty ()` const override  
*This kind of plugin do not need to be configurable.*
- QString `ifacelid ()` const override  
*Return the plugin interface identifier.*
- virtual `DImgLoader * loader (DImg *const image, const DRawDecoding &rawSettings=DRawDecoding()) const =0`  
*Return the image loader instance for the `DImg` instance.*
- virtual QString `loaderName ()` const =0  
*Return a single capitalized word to identify the format supported by the loader.*
- virtual bool `previewSupported ()` const  
*Return true if the loader can read a preview image.*
- void `setVisible (bool)` override  
*This kind of plugin do not have GUI visibility attribute.*
- virtual QString `typeMimes ()` const =0  
*Return the list of white-listed type-mimes supported by the loader, as a string of file-name suffix separated by spaces.*

## Public Member Functions inherited from `Digikam::DPlugin`

- `DPlugin (QObject *const parent=nullptr)`  
*Constructor with optional parent object.*
- `~DPlugin ()` override  
*Destructor.*
- virtual QList< `DPluginAuthor` > `authors ()` const =0  
*Returns authors list for the plugin.*
- virtual void `cleanUp ()`  
*Plugin method to clean up internal created objects.*
- virtual QString `description ()` const =0  
*Returns a short description about the plugin.*
- virtual QString `details ()` const =0  
*Returns a long description about the plugin.*
- virtual QString `handbookChapter ()` const  
*Return the online handbook chapter from an handbook section corresponding to this plugin.*



- virtual QString [handbookReference](#) () const  
*Return the online handbook reference from an handbook chapter corresponding to this plugin.*
- virtual QString [handbookSection](#) () const  
*Return the online handbook section corresponding to this plugin.*
- virtual QIcon [icon](#) () const  
*Returns an icon for the plugin.*
- virtual QString [iid](#) () const =0  
*Returns the unique internal identification property of the plugin.*
- QString [libraryFileName](#) () const  
*Returns the file name of the library for this plugin.*
- virtual QString [name](#) () const =0  
*Returns the user-visible name of the plugin.*
- QStringList [pluginAuthors](#) () const  
*Return a list of authors as strings registered in this plugin.*
- void [setLibraryFileName](#) (const QString &)  
*Sets the file name of the library for this plugin.*
- void [setShouldLoaded](#) (bool b)  
*Accessor to adjust the should loaded plugin property.*
- virtual void [setup](#) (QObject \*const parent)=0  
*Plugin factory method to create all internal object instances for a given parent.*
- bool [shouldLoaded](#) () const  
*Return the should loaded property.*
- QString [version](#) () const  
*Return the internal version used to check the binary compatibility at run-time.*

## 9.432.1 Member Function Documentation

### 9.432.1.1 canRead()

```
virtual int Digikam::DPluginDImg::canRead (
    const QFileInfo & fileInfo,
    bool magic ) const [pure virtual]
```

The return value (1 - 100) is a priority. digiKam default loaders have a priority of 10, the QImage loader has a priority of 80 and the ImageMagick loader has a priority of 90. If the loader is to be used before the default loader, the value must be less than 10.

### 9.432.1.2 canWrite()

```
virtual int Digikam::DPluginDImg::canWrite (
    const QString & format ) const [pure virtual]
```

The return value (1 - 100) is a priority.

### 9.432.1.3 categories()

```
QStringList Digikam::DPluginDImg::categories ( ) const [inline], [override], [virtual]
```

Implements [Digikam::DPlugin](#).

#### 9.432.1.4 count()

```
int Digikam::DPluginDImg::count ( ) const [inline], [override], [virtual]
```

Implements [Digikam::DPlugin](#).

#### 9.432.1.5 exportWidget()

```
virtual DImgLoaderSettings * Digikam::DPluginDImg::exportWidget (
    const QString & format ) const [pure virtual]
```

Return nullptr if format is not supported or if no settings widget is available for this format.

#### 9.432.1.6 extraAboutData()

```
QMap< QString, QStringList > Digikam::DPluginDImg::extraAboutData ( ) const [override], [virtual]
```

Reimplemented from [Digikam::DPlugin](#).

#### 9.432.1.7 extraAboutDataRowTitles()

```
QStringList Digikam::DPluginDImg::extraAboutDataRowTitles ( ) const [override], [virtual]
```

Reimplemented from [Digikam::DPlugin](#).

#### 9.432.1.8 extraAboutDataTitle()

```
QString Digikam::DPluginDImg::extraAboutDataTitle ( ) const [override], [virtual]
```

Reimplemented from [Digikam::DPlugin](#).

#### 9.432.1.9 hasVisibilityProperty()

```
bool Digikam::DPluginDImg::hasVisibilityProperty ( ) const [inline], [override], [virtual]
```

Reimplemented from [Digikam::DPlugin](#).

#### 9.432.1.10 ifaceId()

```
QString Digikam::DPluginDImg::ifaceId ( ) const [inline], [override], [virtual]
```

Implements [Digikam::DPlugin](#).

### 9.432.1.11 loaderName()

```
virtual QString Digikam::DPluginDImg::loaderName ( ) const [pure virtual]
```

Ex: jpeg => "JPG" ; tiff => "TIF", etc.

### 9.432.1.12 setVisible()

```
void Digikam::DPluginDImg::setVisible (
    bool ) [inline], [override], [virtual]
```

Implements [Digikam::DPlugin](#).

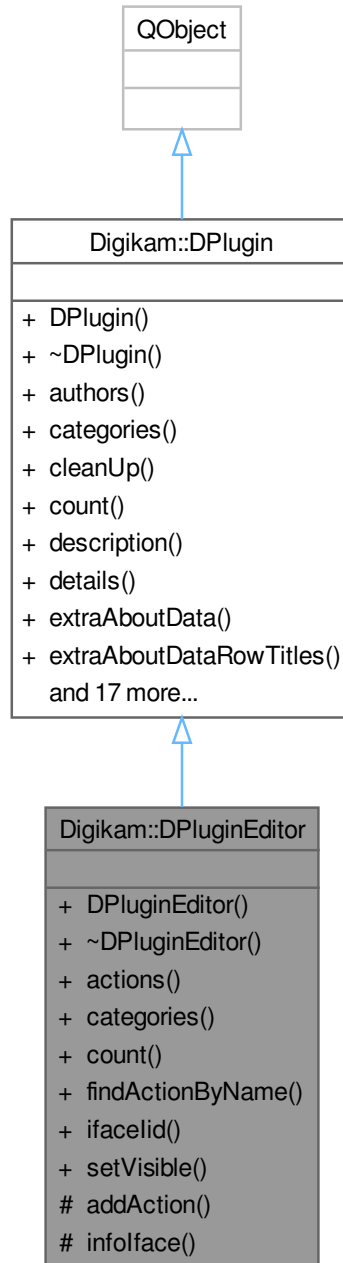
### 9.432.1.13 typeMimes()

```
virtual QString Digikam::DPluginDImg::typeMimes ( ) const [pure virtual]
```

Ex: "jpeg jpg thm"

## 9.433 Digikam::DPluginEditor Class Reference

Inheritance diagram for Digikam::DPluginEditor:



### Public Member Functions

- **DPluginEditor** (`QObject *const parent=nullptr`)  
*Constructor with optional parent object.*

- `~DPluginEditor ()` override  
*Destructor.*
- `QList< DPluginAction * > actions (QObject *const parent) const`  
*Return all plugin actions registered in `setup()` method with `addAction()` for a given parent.*
- `QStringList categories ()` const override  
*Return a list of categories as strings registered in this plugin.*
- `int count ()` const override  
*Return the amount of tools registered to all parents.*
- `DPluginAction * findActionByName (const QString &name, QObject *const parent) const`  
*Return a plugin action instance found by name in plugin action list for a given parent.*
- `QString ifacelid ()` const override  
*Return the plugin interface identifier.*
- `void setVisible (bool b)` override  
*Holds whether the plugin can be seen in parent view.*

## Public Member Functions inherited from Digikam::DPlugin

- `DPlugin (QObject *const parent=nullptr)`  
*Constructor with optional parent object.*
- `~DPlugin ()` override  
*Destructor.*
- `virtual QList< DPluginAuthor > authors () const =0`  
*Returns authors list for the plugin.*
- `virtual void cleanUp ()`  
*Plugin method to clean up internal created objects.*
- `virtual QString description () const =0`  
*Returns a short description about the plugin.*
- `virtual QString details () const =0`  
*Returns a long description about the plugin.*
- `virtual QMap< QString, QStringList > extraAboutData () const`  
*Returns a map of extra data to show in plugin about dialog.*
- `virtual QStringList extraAboutDataRowTitles () const`  
*Returns a list of extra data row titles to show in tab of plugin about dialog.*
- `virtual QString extraAboutDataTitle () const`  
*Returns the tab title of data returned by `extraAboutData()`.*
- `virtual QString handbookChapter () const`  
*Return the online handbook chapter from an handbook section corresponding to this plugin.*
- `virtual QString handbookReference () const`  
*Return the online handbook reference from an handbook chapter corresponding to this plugin.*
- `virtual QString handbookSection () const`  
*Return the online handbook section corresponding to this plugin.*
- `virtual bool hasVisibilityProperty () const`  
*Return true if plugin can be configured in setup dialog about the visibility property.*
- `virtual QIcon icon () const`  
*Returns an icon for the plugin.*
- `virtual QString iid () const =0`  
*Returns the unique internal identification property of the plugin.*
- `QString libraryFileName () const`  
*Returns the file name of the library for this plugin.*
- `virtual QString name () const =0`

- Returns the user-visible name of the plugin.*
- QStringList **pluginAuthors** () const
  - Return a list of authors as strings registered in this plugin.*
- void **setLibraryFileName** (const QString &)
  - Sets the file name of the library for this plugin.*
- void **setShouldLoaded** (bool b)
  - Accessor to adjust the should loaded plugin property.*
- virtual void **setup** (QObject \*const parent)=0
  - Plugin factory method to create all internal object instances for a given parent.*
- bool **shouldLoaded** () const
  - Return the should loaded property.*
- QString **version** () const
  - Return the internal version used to check the binary compatibility at run-time.*

### Protected Member Functions

- void **addAction** (DPluginAction \*const ac)
- **InfoInterface** \* **infoface** (QObject \*const ac) const
- Return the info interface instance for the given action.*

## 9.433.1 Member Function Documentation

### 9.433.1.1 categories()

```
QStringList Digikam::DPluginEditor::categories ( ) const [override], [virtual]
```

Implements [Digikam::DPlugin](#).

### 9.433.1.2 count()

```
int Digikam::DPluginEditor::count ( ) const [override], [virtual]
```

Implements [Digikam::DPlugin](#).

### 9.433.1.3 ifaceId()

```
QString Digikam::DPluginEditor::ifaceId ( ) const [inline], [override], [virtual]
```

Implements [Digikam::DPlugin](#).

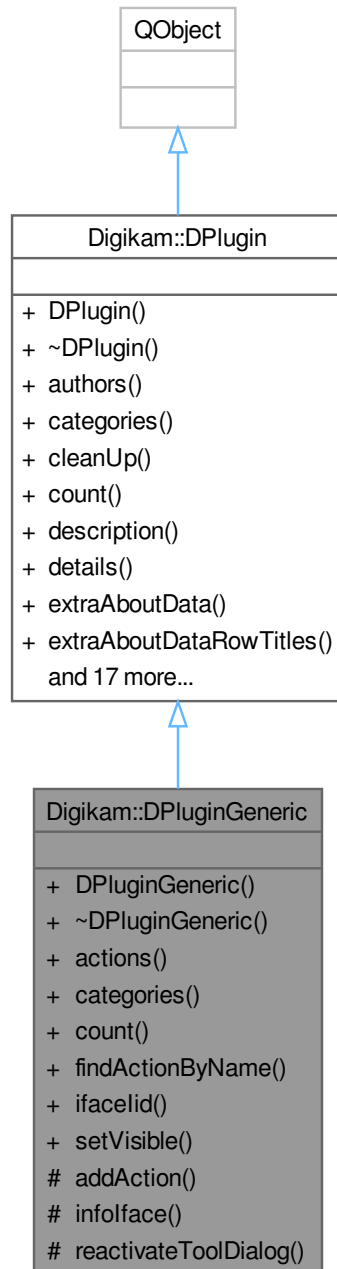
### 9.433.1.4 setVisible()

```
void Digikam::DPluginEditor::setVisible (
    bool b ) [override], [virtual]
```

Implements [Digikam::DPlugin](#).

## 9.434 Digikam::DPluginGeneric Class Reference

Inheritance diagram for Digikam::DPluginGeneric:



### Public Member Functions

- **DPluginGeneric** (QObject \*const parent=nullptr)  
*Constructor with optional parent object.*

- `~DPluginGeneric ()` override  
*Destructor.*
- `QList< DPluginAction * > actions (QObject *const parent) const`  
*Return all plugin actions registered in `setup()` method with `addAction()` for a given parent.*
- `QStringList categories ()` const override  
*Return a list of categories as strings registered in this plugin.*
- `int count ()` const override  
*Return the amount of tools registered to all parents.*
- `DPluginAction * findActionByName (const QString &name, QObject *const parent) const`  
*Return a plugin action instance found by name in plugin action list for a given parent.*
- `QString ifacelid ()` const override  
*Return the plugin interface identifier.*
- `void setVisible (bool b)` override  
*Holds whether the plugin can be seen in parent view.*

## Public Member Functions inherited from `Digikam::DPlugin`

- `DPlugin (QObject *const parent=nullptr)`  
*Constructor with optional parent object.*
- `~DPlugin ()` override  
*Destructor.*
- `virtual QList< DPluginAuthor > authors () const =0`  
*Returns authors list for the plugin.*
- `virtual void cleanUp ()`  
*Plugin method to clean up internal created objects.*
- `virtual QString description () const =0`  
*Returns a short description about the plugin.*
- `virtual QString details () const =0`  
*Returns a long description about the plugin.*
- `virtual QMap< QString, QStringList > extraAboutData () const`  
*Returns a map of extra data to show in plugin about dialog.*
- `virtual QStringList extraAboutDataRowTitles () const`  
*Returns a list of extra data row titles to show in tab of plugin about dialog.*
- `virtual QString extraAboutDataTitle () const`  
*Returns the tab title of data returned by `extraAboutData()`.*
- `virtual QString handbookChapter () const`  
*Return the online handbook chapter from an handbook section corresponding to this plugin.*
- `virtual QString handbookReference () const`  
*Return the online handbook reference from an handbook chapter corresponding to this plugin.*
- `virtual QString handbookSection () const`  
*Return the online handbook section corresponding to this plugin.*
- `virtual bool hasVisibilityProperty () const`  
*Return true if plugin can be configured in setup dialog about the visibility property.*
- `virtual QIcon icon () const`  
*Returns an icon for the plugin.*
- `virtual QString iid () const =0`  
*Returns the unique internal identification property of the plugin.*
- `QString libraryFileName () const`  
*Returns the file name of the library for this plugin.*
- `virtual QString name () const =0`



- Returns the user-visible name of the plugin.*
- QStringList **pluginAuthors** () const
 

*Return a list of authors as strings registered in this plugin.*
  - void **setLibraryFileName** (const QString &)
 

*Sets the file name of the library for this plugin.*
  - void **setShouldLoaded** (bool b)
 

*Accessor to adjust the should loaded plugin property.*
  - virtual void **setup** (QObject \*const parent)=0
 

*Plugin factory method to create all internal object instances for a given parent.*
  - bool **shouldLoaded** () const
 

*Return the should loaded property.*
  - QString **version** () const
 

*Return the internal version used to check the binary compatibility at run-time.*

### Protected Member Functions

- void **addAction** (DPluginAction \*const ac)
- DInfoInterface \* **infoInterface** (QObject \*const ac) const
 

*Return the info interface instance for the given action object.*
- bool **reactivateToolDialog** (QWidget \*const dlg) const
 

*Helper function to reactivate the desktop visibility of tool widget.*

## 9.434.1 Member Function Documentation

### 9.434.1.1 categories()

```
QStringList Digikam::DPluginGeneric::categories ( ) const [override], [virtual]
```

Implements [Digikam::DPlugin](#).

### 9.434.1.2 count()

```
int Digikam::DPluginGeneric::count ( ) const [override], [virtual]
```

Implements [Digikam::DPlugin](#).

### 9.434.1.3 ifaceId()

```
QString Digikam::DPluginGeneric::ifaceId ( ) const [inline], [override], [virtual]
```

Implements [Digikam::DPlugin](#).

### 9.434.1.4 setVisible()

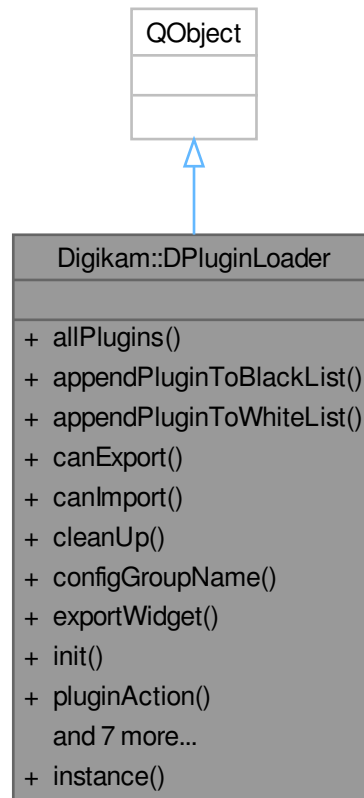
```
void Digikam::DPluginGeneric::setVisible (
    bool b ) [override], [virtual]
```

Implements [Digikam::DPlugin](#).

## 9.435 Digikam::DPluginLoader Class Reference

The class that handles digiKam's external plugins.

Inheritance diagram for Digikam::DPluginLoader:



### Public Member Functions

- `QList< DPlugin * > allPlugins ()` const  
*Returns all available plugins.*
- void `appendPluginToBlackList (const QString &filename)`  
*appendPluginToBlackList Prevent that a plugin is loaded from the given filename*
- void `appendPluginToWhiteList (const QString &filename)`  
*appendPluginToWhiteList Add a plugin to the whitelist of tools.*
- bool `canExport (const QString &format)` const  
*Return true if format is supported by a [DPluginDImg](#) to export image.*
- bool `canImport (const QString &format)` const  
*Return true if format is supported by a [DPluginDImg](#) to import image.*
- void `cleanUp ()`  
*Unload all loaded plugins.*
- `QString configGroupName ()` const

- Return the config group name used to store the list of plugins to load at startup.*

  - [DImgLoaderSettings](#) \* [exportWidget](#) (const QString &format) const

*Return a new widget instance from a [DPluginDImg](#) to show settings while exporting image to specified format.*
- void [init](#) ()
  - Init plugin loader.*
- [DPluginAction](#) \* [pluginAction](#) (const QString &actionName, QObject \*const parent) const
  - Returns the plugin action corresponding to a action name for a given parent.*
- QList< [DPluginAction](#) \* > [pluginActions](#) (const QString &pluginIID, QObject \*const parent) const
  - Returns the plugin actions corresponding to a plugin internal ID string for a given parent.*
- QList< [DPluginAction](#) \* > [pluginsActions](#) ([DPluginAction::ActionCategory](#) cat, QObject \*const parent) const
  - Returns a list of plugin actions set as category for a given parent.*
- QList< [DPluginAction](#) \* > [pluginsActions](#) ([DPluginAction::ActionType](#) type, QObject \*const parent) const
  - Returns a list of plugin actions set as type for a given parent.*
- QString [pluginXmlSections](#) ([DPluginAction::ActionCategory](#) cat, QObject \*const parent) const
  - Returns all xml sections as string of plugin actions set with a kind of category for a given parent.*
- void [registerEditorPlugins](#) (QObject \*const parent)
  - Register all Editor plugin actions to parent object.*
- void [registerGenericPlugins](#) (QObject \*const parent)
  - Register all Generic plugin actions to parent object.*
- void [registerRawImportPlugins](#) (QObject \*const parent)
  - Register all Raw Import plugin to parent object.*

### Static Public Member Functions

- static [DPluginLoader](#) \* [instance](#) ()
  - instance: returns the singleton of plugin loader*

### Friends

- class [DPluginLoaderCreator](#)

## 9.435.1 Detailed Description

Ownership policy for plugins:

The [DPluginLoader](#) creates new objects and transfer ownership. In order to create the objects, the [DPluginLoader](#) internally has a list of the tools which are owned by the [DPluginLoader](#) and destroyed by it.

## 9.435.2 Member Function Documentation

### 9.435.2.1 [appendPluginToBlackList\(\)](#)

```
void Digikam::DPluginLoader::appendPluginToBlackList (
    const QString & filename )
```

## Parameters

<i>filename</i>	The name of the file excluding file extension to blacklist. E.g. to ignore "HtmlGalleryPlugin.so" on Linux and "HtmlGalleryPlugin.dll" on Windows, pass "HtmlGalleryPlugin"
-----------------	---

**9.435.2.2 appendPluginToWhiteList()**

```
void Digikam::DPluginLoader::appendPluginToWhiteList (
    const QString & filename )
```

If the whitelist is not empty, only whitelisted tools are loaded. If a tool is both whitelisted and blacklisted, it will not be loaded.

## Parameters

<i>filename</i>	The name of the file excluding file extension to whitelist. E.g. to not ignore "HtmlGalleryPlugin.so" on Linux and "HtmlGalleryPlugin.dll" on Windows, pass "HtmlGalleryPlugin"
-----------------	---

**9.435.2.3 cleanUp()**

```
void Digikam::DPluginLoader::cleanUp ( )
```

Call this method before the main instance is closed.

**9.435.2.4 exportWidget()**

```
DImgLoaderSettings * Digikam::DPluginLoader::exportWidget (
    const QString & format ) const
```

Return nullptr if format is not supported or if no settings widget is available for this format.

**9.435.2.5 init()**

```
void Digikam::DPluginLoader::init ( )
```

Call this method to parse and load relevant plugins installed on your system.

**9.435.2.6 instance()**

```
DPluginLoader * Digikam::DPluginLoader::instance ( ) [static]
```

## Returns

[DPluginLoader](#) global instance

### 9.435.2.7 pluginAction()

```
DPluginAction * Digikam::DPluginLoader::pluginAction (
    const QString & actionName,
    QObject *const parent ) const
```

If not found, this returns a null pointer.

### 9.435.2.8 pluginActions()

```
QList< DPluginAction * > Digikam::DPluginLoader::pluginActions (
    const QString & pluginIID,
    QObject *const parent ) const
```

If not found, this returns an empty list.

### 9.435.2.9 pluginsActions() [1/2]

```
QList< DPluginAction * > Digikam::DPluginLoader::pluginsActions (
    DPluginAction::ActionCategory cat,
    QObject *const parent ) const
```

If no plugin have found in this category, this returns an empty list.

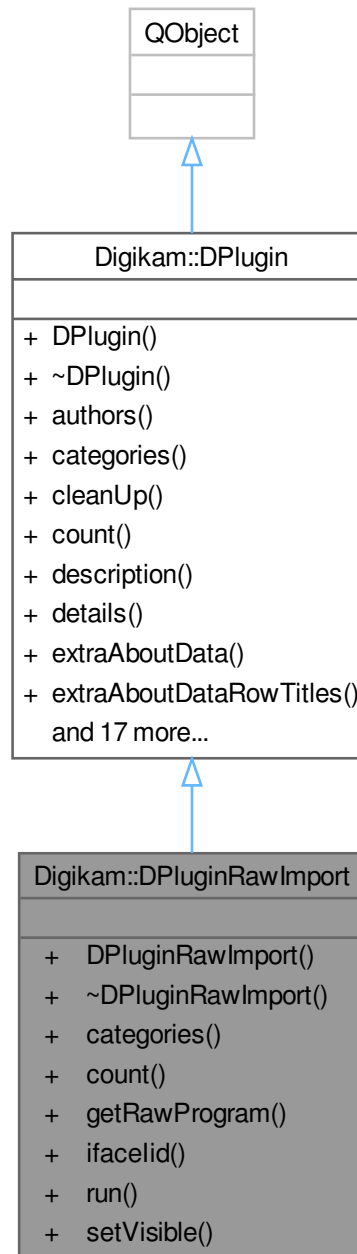
### 9.435.2.10 pluginsActions() [2/2]

```
QList< DPluginAction * > Digikam::DPluginLoader::pluginsActions (
    DPluginAction::ActionType type,
    QObject *const parent ) const
```

If no plugin have found in this category, this returns an empty list.

## 9.436 Digikam::DPluginRawImport Class Reference

Inheritance diagram for Digikam::DPluginRawImport:



### Signals

- void **signalDecodedImage** (const [Digikam::LoadingDescription](#) &, const [Digikam::DImg](#) &)  
*Signal emitted to notify host application to load pre-decoded Raw preprocessed with these decoding settings.*
- void **signalLoadRaw** (const [Digikam::LoadingDescription](#) &)  
*Signal emitted to notify host application to load Raw with these decoding settings.*

## Public Member Functions

- **DPluginRawImport** (QObject \*const parent=nullptr)  
*Constructor with optional parent object.*
- **~DPluginRawImport** () override=default  
*Destructor.*
- QStringList **categories** () const override  
*This kind of plugin do not use a category.*
- int **count** () const override  
*This kind of plugin only provide one tool.*
- virtual QString **getRawProgram** () const  
*Return the path to the raw program, or empty if not found.*
- QString **ifaceId** () const override  
*Return the plugin interface identifier.*
- virtual bool **run** (const QString &path, const DRawDecoding &def)=0  
*Function to re-implement used to invoke Raw processor for a Raw file path and a Default Raw decoding settings.*
- void **setVisible** (bool) override  
*This kind of plugin do not have GUI visibility attribute.*

## Public Member Functions inherited from Digikam::DPlugin

- **DPlugin** (QObject \*const parent=nullptr)  
*Constructor with optional parent object.*
- **~DPlugin** () override  
*Destructor.*
- virtual QList< **DPluginAuthor** > **authors** () const =0  
*Returns authors list for the plugin.*
- virtual void **cleanUp** ()  
*Plugin method to clean up internal created objects.*
- virtual QString **description** () const =0  
*Returns a short description about the plugin.*
- virtual QString **details** () const =0  
*Returns a long description about the plugin.*
- virtual QMap< QString, QStringList > **extraAboutData** () const  
*Returns a map of extra data to show in plugin about dialog.*
- virtual QStringList **extraAboutDataRowTitles** () const  
*Returns a list of extra data row titles to show in tab of plugin about dialog.*
- virtual QString **extraAboutDataTitle** () const  
*Returns the tab title of data returned by [extraAboutData\(\)](#).*
- virtual QString **handbookChapter** () const  
*Return the online handbook chapter from an handbook section corresponding to this plugin.*
- virtual QString **handbookReference** () const  
*Return the online handbook reference from an handbook chapter corresponding to this plugin.*
- virtual QString **handbookSection** () const  
*Return the online handbook section corresponding to this plugin.*
- virtual bool **hasVisibilityProperty** () const  
*Return true if plugin can be configured in setup dialog about the visibility property.*
- virtual QIcon **icon** () const  
*Returns an icon for the plugin.*
- virtual QString **iid** () const =0

- Returns the unique internal identification property of the plugin.*

  - QString [libraryFileName](#) () const
    - Returns the file name of the library for this plugin.*
  - virtual QString [name](#) () const =0
    - Returns the user-visible name of the plugin.*
  - QStringList [pluginAuthors](#) () const
    - Return a list of authors as strings registered in this plugin.*
  - void [setLibraryFileName](#) (const QString &)
    - Sets the file name of the library for this plugin.*
  - void [setShouldLoaded](#) (bool b)
    - Accessor to adjust the should loaded plugin property.*
  - virtual void [setup](#) (QObject \*const parent)=0
    - Plugin factory method to create all internal object instances for a given parent.*
  - bool [shouldLoaded](#) () const
    - Return the should loaded property.*
  - QString [version](#) () const
    - Return the internal version used to check the binary compatibility at run-time.*

## 9.436.1 Member Function Documentation

### 9.436.1.1 categories()

```
QStringList Digikam::DPluginRawImport::categories ( ) const [inline], [override], [virtual]
```

Implements [Digikam::DPlugin](#).

### 9.436.1.2 count()

```
int Digikam::DPluginRawImport::count ( ) const [inline], [override], [virtual]
```

Implements [Digikam::DPlugin](#).

### 9.436.1.3 ifaceId()

```
QString Digikam::DPluginRawImport::ifaceId ( ) const [inline], [override], [virtual]
```

Implements [Digikam::DPlugin](#).

### 9.436.1.4 setVisible()

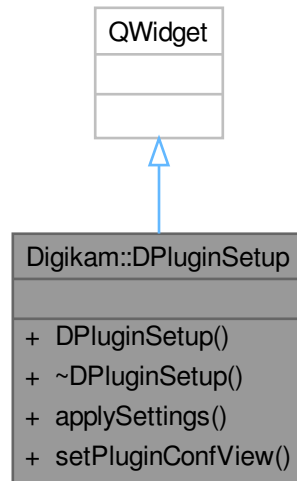
```
void Digikam::DPluginRawImport::setVisible (
    bool ) [inline], [override], [virtual]
```

Implements [Digikam::DPlugin](#).



## 9.437 Digikam::DPluginSetup Class Reference

Inheritance diagram for Digikam::DPluginSetup:

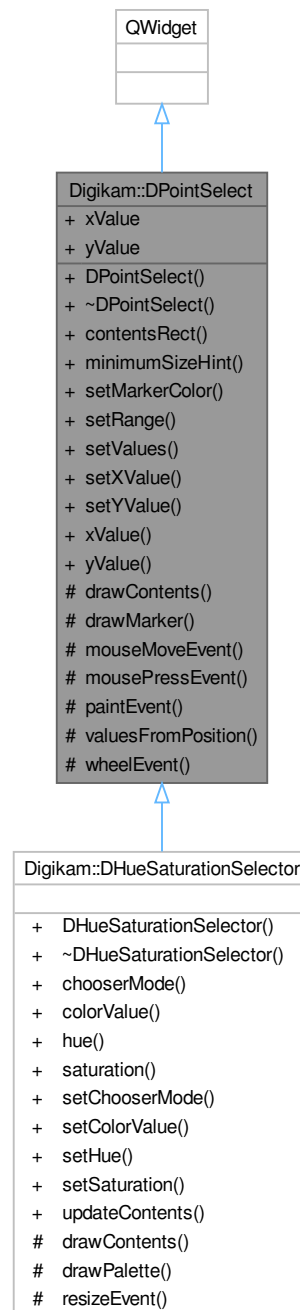


### Public Member Functions

- **DPluginSetup** (`QWidget *const parent=nullptr`)
- void **applySettings** ()
- void **setPluginConfView** ([DPluginConfView](#) \*const view)

## 9.438 Digikam::DPointSelect Class Reference

Inheritance diagram for Digikam::DPointSelect:



### Signals

- void `valueChanged` (int x, int y)

*This signal is emitted whenever the user chooses a value, e.g.*

## Public Member Functions

- **DPointSelect** (QWidget \*const parent)  
*Constructs a two-dimensional selector widget which has a value range of [0..100] in both directions.*
- QRect **contentsRect** () const
- QSize **minimumSizeHint** () const override  
*Reimplemented to give the widget a minimum size.*
- void **setMarkerColor** (const QColor &col)  
*Sets the color used to draw the marker.*
- void **setRange** (int minX, int minY, int maxX, int maxY)  
*Sets the range of possible values.*
- void **setValues** (int xPos, int yPos)  
*Sets the current values in horizontal and vertical direction.*
- void **setXValue** (int xPos)  
*Sets the current horizontal value.*
- void **setYValue** (int yPos)  
*Sets the current vertical value.*
- int **xValue** () const
- int **yValue** () const

## Protected Member Functions

- virtual void **drawContents** (QPainter \*)  
*Override this function to draw the contents of the widget.*
- virtual void **drawMarker** (QPainter \*p, int xp, int yp)  
*Override this function to draw the marker which indicates the currently selected value pair.*
- void **mouseMoveEvent** (QMouseEvent \*e) override
- void **mousePressEvent** (QMouseEvent \*e) override
- void **paintEvent** (QPaintEvent \*e) override
- void **valuesFromPosition** (int x, int y, int &xVal, int &yVal) const  
*Converts a pixel position to its corresponding values.*
- void **wheelEvent** (QWheelEvent \*) override

## Properties

- int **xValue**
- int **yValue**

## Friends

- class **Private**

## 9.438.1 Member Function Documentation

### 9.438.1.1 contentsRect()

QRect Digikam::DPointSelect::contentsRect ( ) const

#### Returns

the rectangle on which subclasses should draw.

### 9.438.1.2 drawContents()

```
virtual void Digikam::DPointSelect::drawContents (
    QPainter * ) [inline], [protected], [virtual]
```

The default implementation does nothing.

Draw within [contentsRect\(\)](#) only.

Reimplemented in [Digikam::DHueSaturationSelector](#).

### 9.438.1.3 setMarkerColor()

```
void Digikam::DPointSelect::setMarkerColor (
    const QColor & col )
```

#### Parameters

<i>col</i>	the color
------------	-----------

### 9.438.1.4 setValues()

```
void Digikam::DPointSelect::setValues (
    int xPos,
    int yPos )
```

#### Parameters

<i>xPos</i>	the horizontal value
<i>yPos</i>	the vertical value

### 9.438.1.5 setXValue()

```
void Digikam::DPointSelect::setXValue (
    int xPos )
```

#### Parameters

<i>xPos</i>	the horizontal value
-------------	----------------------

### 9.438.1.6 setYValue()

```
void Digikam::DPointSelect::setYValue (
    int yPos )
```

**Parameters**

<i>yPos</i>	the vertical value
-------------	--------------------

**9.438.1.7 valueChanged**

```
void Digikam::DPointSelect::valueChanged (
    int x,
    int y ) [signal]
```

by clicking with the mouse on the widget.

**9.438.1.8 xValue()**

```
int Digikam::DPointSelect::xValue ( ) const
```

**Returns**

the current value in horizontal direction.

**9.438.1.9 yValue()**

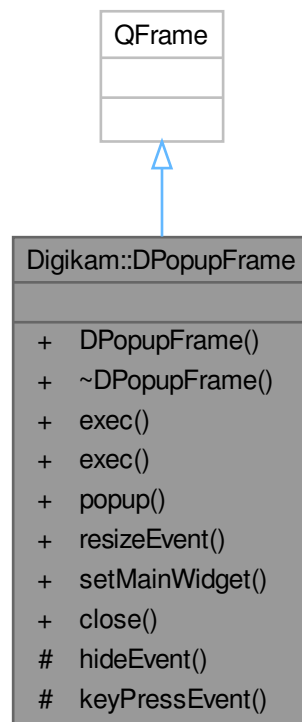
```
int Digikam::DPointSelect::yValue ( ) const
```

**Returns**

the current value in vertical direction.

## 9.439 Digikam::DPopupFrame Class Reference

Inheritance diagram for Digikam::DPopupFrame:

**Public Slots**

- void `close` (int r)  
*Close the popup window.*

**Signals**

- void `leaveModality` ()

## Public Member Functions

- [DPopupFrame](#) (QWidget \*const parent=nullptr)  
*The constructor.*
- [~DPopupFrame](#) () override  
*The destructor.*
- int **exec** (const QPoint &p)  
*Execute the popup window.*
- int **exec** (int x, int y)  
*Execute the popup window.*
- void **popup** (const QPoint &p)  
*Open the popup window at position pos.*
- void [resizeEvent](#) (QResizeEvent \*e) override  
*The resize event.*
- void [setMainWidget](#) (QWidget \*const m)  
*Set the main widget.*

## Protected Member Functions

- void **hideEvent** (QHideEvent \*e) override  
*Catch hide events.*
- void **keyPressEvent** (QKeyEvent \*e) override  
*Catch key press events.*

## Friends

- class **Private**

## 9.439.1 Constructor & Destructor Documentation

### 9.439.1.1 DPopupFrame()

```
Digikam::DPopupFrame::DPopupFrame (  
    QWidget *const parent = nullptr ) [explicit]
```

Creates a dialog without buttons.

## 9.439.2 Member Function Documentation

### 9.439.2.1 close

```
void Digikam::DPopupFrame::close (  
    int r ) [slot]
```

This is called from the main widget, usually. `r` is the result returned from [exec\(\)](#).

### 9.439.2.2 `resizeEvent()`

```
void Digikam::DPopupFrame::resizeEvent (
    QResizeEvent * e ) [override]
```

Simply resizes the main widget to the whole widgets client size.

### 9.439.2.3 `setMainWidget()`

```
void Digikam::DPopupFrame::setMainWidget (
    QWidget *const m )
```

You cannot set the main widget from the constructor, since it must be a child of the frame itself. Be careful: the size is set to the main widgets size. It is up to you to set the main widgets correct size before setting it as the main widget.



## 9.440 Digikam::DPreviewImage Class Reference

Inheritance diagram for Digikam::DPreviewImage:



### Public Slots

- void **slotClearActiveSelection** ()
- void **slotClearHighlight** ()

*This function removes the highlight area.*

- void **slotSetHighlightArea** (float tl\_x, float tl\_y, float br\_x, float br\_y)

*This function is used to darken everything except what is inside the given area.*

- void **slotSetHighlightShown** (int percentage, const QColor &highlightColor=Qt::white)

*This function sets the percentage of the highlighted area that is visible.*

- void **slotSetSelection** (float tl\_x, float tl\_y, float br\_x, float br\_y)

*This function is used to set a selection without the user setting it.*

- void **slotZoom2Fit** ()
- void **slotZoomIn** ()
- void **slotZoomOut** ()

- void **slotSetTLX** (float ratio)

*Selection area specific slots (TL = TopLeft, BR = BottomRight)*

- void **slotSetTLY** (float ratio)
- void **slotSetBRX** (float ratio)
- void **slotSetBRY** (float ratio)

## Public Member Functions

- **DPreviewImage** (QWidget \*const parent)
- void **enableSelectionArea** (bool b)
- QRectF **getSelectionArea** () const
- bool **load** (const QUrl &file) const
- bool **setImage** (const QImage &img) const
- void **setSelectionArea** (const QRectF &rectangle)

*Sets a selection area and show it.*

## Protected Member Functions

- void **enterEvent** (QEnterEvent \*) override
- bool **eventFilter** (QObject \*, QEvent \*) override
- void **leaveEvent** (QEvent \*) override
- void **mouseMoveEvent** (QMouseEvent \*) override
- void **mousePressEvent** (QMouseEvent \*) override
- void **mouseReleaseEvent** (QMouseEvent \*) override
- void **resizeEvent** (QResizeEvent \*) override
- void **updateHighlight** ()
- void **updateSelVisibility** ()
- void **wheelEvent** (QWheelEvent \*) override

## 9.440.1 Member Function Documentation

### 9.440.1.1 setSelectionArea()

```
void Digikam::DPreviewImage::setSelectionArea (
    const QRectF & rectangle )
```

## Parameters

<i>rectangle</i>	This rectangle should have height and width of 1.0
------------------	--

## 9.440.1.2 slotSetHighlightArea

```
void Digikam::DPreviewImage::slotSetHighlightArea (
    float tl_x,
    float tl_y,
    float br_x,
    float br_y ) [slot]
```

## Note

all parameters must be in the range 0.0 -> 1.0.

## Parameters

<i>tl</i> <sub>↔</sub> <i>_x</i>	is the x coordinate of the top left corner 0=0 1=image with.
<i>tl</i> <sub>↔</sub> <i>_y</i>	is the y coordinate of the top left corner 0=0 1=image height.
<i>br</i> <sub>↔</sub> <i>_x</i>	is the x coordinate of the bottom right corner 0=0 1=image with.
<i>br</i> <sub>↔</sub> <i>_y</i>	is the y coordinate of the bottom right corner 0=0 1=image height.

## 9.440.1.3 slotSetHighlightShown

```
void Digikam::DPreviewImage::slotSetHighlightShown (
    int percentage,
    const QColor & highlightColor = Qt::white ) [slot]
```

The rest is hidden. This stacks with the previous highlight area.

## Parameters

<i>percentage</i>	is the percentage of the highlighted area that is shown.
<i>highlightColor</i>	is the color to use to hide the highlighted area of the image.

## 9.440.1.4 slotSetSelection

```
void Digikam::DPreviewImage::slotSetSelection (
    float tl_x,
    float tl_y,
    float br_x,
    float br_y ) [slot]
```

**Note**

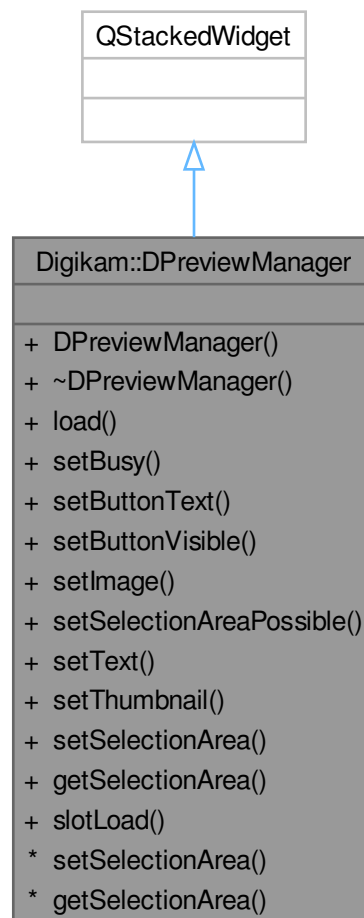
all parameters must be in the range 0.0 -> 1.0.

**Parameters**

<i>tl</i> ↔ _x	is the x coordinate of the top left corner 0=0 1=image with.
<i>tl</i> ↔ _y	is the y coordinate of the top left corner 0=0 1=image height.
<i>br</i> ↔ _x	is the x coordinate of the bottom right corner 0=0 1=image with.
<i>br</i> ↔ _y	is the y coordinate of the bottom right corner 0=0 1=image height.

**9.441 Digikam::DPreviewManager Class Reference**

Inheritance diagram for Digikam::DPreviewManager:



## Public Types

- enum **DisplayMode** { **MessageMode** = 0 , **PreviewMode** }

## Public Slots

- void **slotLoad** (const QUrl &url)

## Signals

- void **signalButtonClicked** ()

## Public Member Functions

- **DPreviewManager** (QWidget \*const parent)
- bool **load** (const QUrl &file, bool fit=true)
- void **setBusy** (bool b, const QString &text=QString())
- void **setButtonText** (const QString &text)
- void **setButtonVisible** (bool b)
- void **setImage** (const QImage &img, bool fit=true)
- void **setSelectionAreaPossible** (bool b)
- void **setText** (const QString &text, const QColor &color=Qt::white)
- void **setThumbnail** (const QPixmap &preview=QPixmap())
  
- void **setSelectionArea** (const QRectF &rectangle)  
*Manage a selection area and show it.*
- QRectF **getSelectionArea** () const

## 9.441.1 Member Function Documentation

### 9.441.1.1 setSelectionArea()

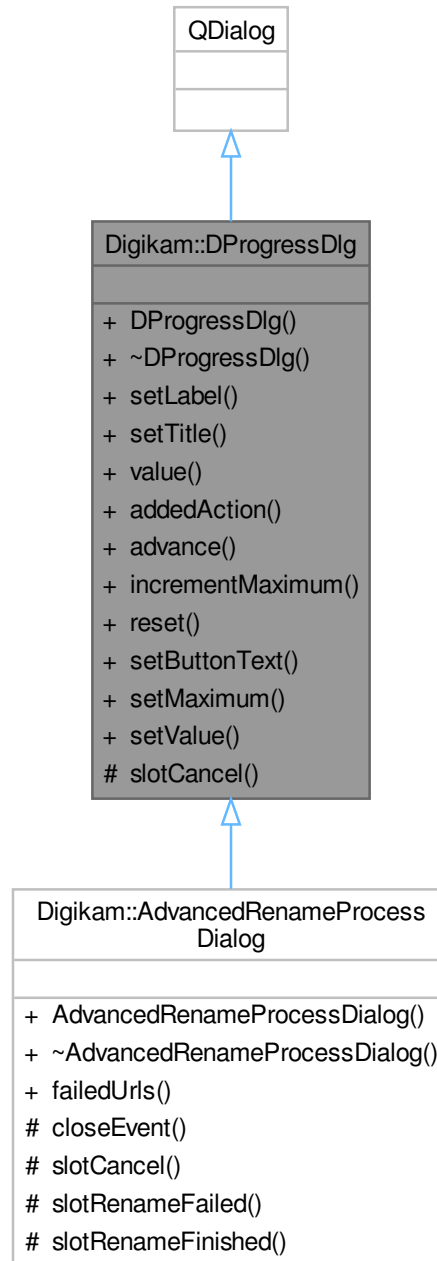
```
void Digikam::DPreviewManager::setSelectionArea (  
    const QRectF & rectangle )
```

#### Parameters

<i>rectangle</i>	This rectangle should have height and width of 1.0
------------------	--

## 9.442 Digikam::DProgressDlg Class Reference

Inheritance diagram for Digikam::DProgressDlg:



### Public Slots

- void **addedAction** (const QPixmap &icon, const QString &text)
- void **advance** (int offset)

- void **incrementMaximum** (int added)
- void **reset** ()
- void **setButtonText** (const QString &text)
- void **setMaximum** (int max)
- void **setValue** (int value)

### Signals

- void **signalCancelPressed** ()

### Public Member Functions

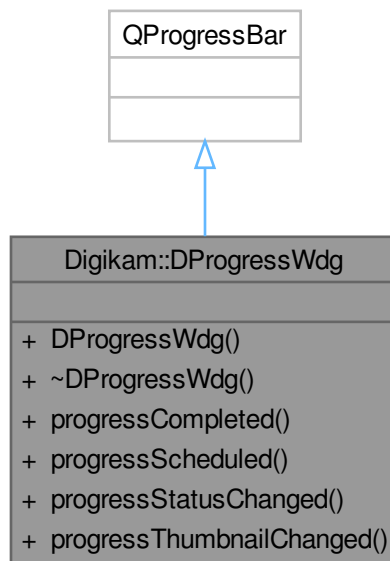
- **DProgressDlg** (QWidget \*const parent=nullptr, const QString &caption=QString())
- void **setLabel** (const QString &text)
- void **setTitle** (const QString &text)
- int **value** () const

### Protected Slots

- void **slotCancel** ()

## 9.443 Digikam::DProgressWdg Class Reference

Inheritance diagram for Digikam::DProgressWdg:



## Signals

- void **signalProgressCanceled** ()  
*Fired when user press cancel button from progress manager.*

## Public Member Functions

- **DProgressWdg** (QWidget \*const parent)
- void **progressCompleted** ()  
*Call this method to query progress manager that process is done.*
- void **progressScheduled** (const QString &title, bool canBeCanceled, bool hasThumb)  
*Call this method to start a new instance of progress notification into progress manager You can pass title string to name progress item, and set it as cancelable.*
- void **progressStatusChanged** (const QString &status)  
*Change status string in progress manager.*
- void **progressThumbnailChanged** (const QPixmap &thumb)  
*Change thumbnail in progress manager.*

## 9.443.1 Member Function Documentation

### 9.443.1.1 progressScheduled()

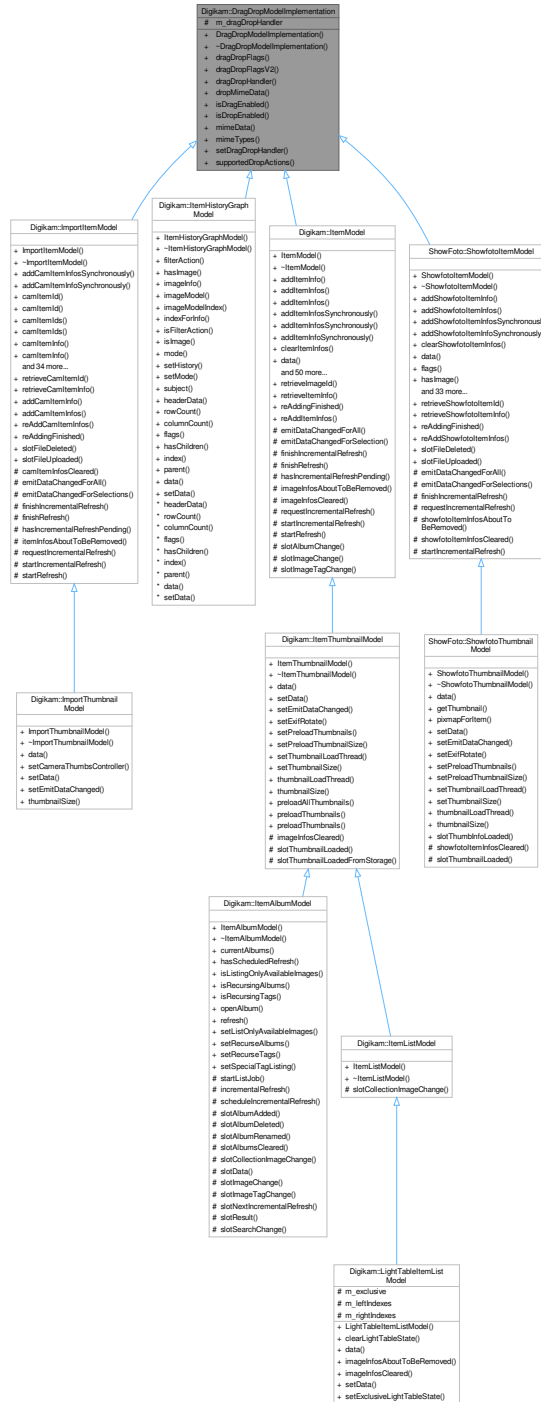
```
void DigiKam::DProgressWdg::progressScheduled (
    const QString & title,
    bool canBeCanceled,
    bool hasThumb )
```

In this case, [signalProgressCanceled\(\)](#) is fired when user press cancel button from progress manager. This item can also accept a thumbnail that you can change through [progressThumbnailChanged\(\)](#).



# 9.444 Digikam::DragDropModelImplementation Class Reference

Inheritance diagram for Digikam::DragDropModelImplementation:



## Public Member Functions

- [DragDropModelImplementation\(\)](#) = default

A class providing a sample implementation for a QAbstractItemModel redirecting drag-and-drop support to a handler.

- virtual Qt::ItemFlags [dragDropFlags](#) (const QModelIndex &index) const  
*Call from your flags() method, adding the relevant drag drop flags.*
- Qt::ItemFlags [dragDropFlagsV2](#) (const QModelIndex &index) const  
*This is an alternative approach to [dragDropFlags\(\)](#).*
- [AbstractItemDragDropHandler](#) \* **dragDropHandler** () const
- bool **dropMimeData** (const QMimeData \*, Qt::DropAction, int, int, const QModelIndex &)
- virtual bool **isDragEnabled** (const QModelIndex &index) const
- virtual bool **isDropEnabled** (const QModelIndex &index) const
- QMimeData \* **mimeData** (const QModelIndexList &indexes) const
- QStringList **mimeTypes** () const
- void **setDragDropHandler** ([AbstractItemDragDropHandler](#) \*handler)  
*Set a drag drop handler.*
- Qt::DropActions [supportedDropActions](#) () const  
*Implements the relevant QAbstractItemModel methods for drag and drop.*

### Protected Attributes

- [AbstractItemDragDropHandler](#) \* **m\_dragDropHandler** = nullptr

## 9.444.1 Constructor & Destructor Documentation

### 9.444.1.1 DragDropModelImplementation()

```
Digikam::DragDropModelImplementation::DragDropModelImplementation ( ) [default]
```

Include the macro `DECLARE_Model_DRAG_DROP_METHODS` in your derived `QAbstractItemModel` class.

## 9.444.2 Member Function Documentation

### 9.444.2.1 dragDropFlags()

```
Qt::ItemFlags Digikam::DragDropModelImplementation::dragDropFlags (
    const QModelIndex & index ) const [virtual]
```

Default implementation enables both drag and drop on the index if a drag drop handler is set. Reimplement to fine-tune. Note: There is an alternative below.

### 9.444.2.2 dragDropFlagsV2()

```
Qt::ItemFlags Digikam::DragDropModelImplementation::dragDropFlagsV2 (
    const QModelIndex & index ) const
```

`dragDropFlagsV2` calls the virtual methods `isDragEnabled()` and `isDropEnabled()` which you then reimplement. Use simple [dragDropFlags\(\)](#) if you need not customization, or reimplement [dragDropFlags\(\)](#) if you fine-tune it yourself.



## Public Member Functions

- virtual void **copy** ()
- virtual void **cut** ()
- virtual void **paste** ()

## Protected Member Functions

- virtual QAbstractItemView \* **asView** ()=0  
*This one is implemented by DECLARE\_VIEW\_DRAG\_DROP\_METHODS.*
- bool **decodelsCutSelection** (const QMimeData \*mimeData)
- virtual [AbstractItemDragDropHandler](#) \* **dragDropHandler** () const =0  
*You need to implement these three methods Returns the drag drop handler.*
- void **dragEnterEvent** (QDragEnterEvent \*event)  
*Implements the relevant QAbstractItemView methods for drag and drop.*
- void **dragMoveEvent** (QDragMoveEvent \*e)
- void **dropEvent** (QDropEvent \*e)
- void **encodelsCutSelection** (QMimeData \*mime, bool isCutSelection)
- virtual QModelIndex **mapIndexForDragDrop** (const QModelIndex &index) const =0  
*Maps the given index of the view's model to an index of the handler's model, which can be a source model of the view's model.*
- virtual QPixmap **pixmapForDrag** (const QList< QModelIndex > &indexes) const =0  
*Creates a pixmap for dragging the given indexes.*
- void **startDrag** (Qt::DropActions supportedActions)

## 9.445.1 Member Function Documentation

### 9.445.1.1 dragDropHandler()

```
virtual AbstractItemDragDropHandler * Digikam::DragDropViewImplementation::dragDropHandler ( )
const [protected], [pure virtual]
```

Implemented in [Digikam::ItemCategorizedView](#), [Digikam::TableViewTreeView](#), [Digikam::VersionsTreeView](#), [ShowFoto::ShowfotoCategorizedView](#), and [Digikam::ImportCategorizedView](#).

### 9.445.1.2 mapIndexForDragDrop()

```
virtual QModelIndex Digikam::DragDropViewImplementation::mapIndexForDragDrop (
    const QModelIndex & index ) const [protected], [pure virtual]
```

Implemented in [Digikam::TableViewTreeView](#), [Digikam::VersionsTreeView](#), and [Digikam::ItemViewCategorized](#).

### 9.445.1.3 pixmapForDrag()

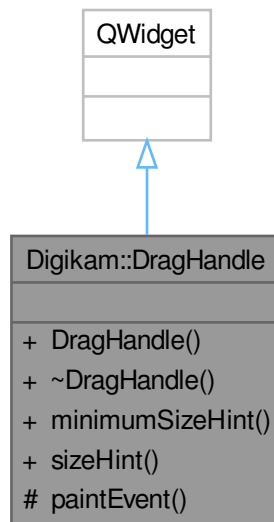
```
virtual QPixmap Digikam::DragDropViewImplementation::pixmapForDrag (
    const QList< QModelIndex > & indexes ) const [protected], [pure virtual]
```

Implemented in [Digikam::TableViewTreeView](#), [Digikam::VersionsTreeView](#), and [Digikam::ItemViewCategorized](#).

## 9.446 Digikam::DragHandle Class Reference

An alternative handle for QDockWidget's that looks like a toolbar handle.

Inheritance diagram for Digikam::DragHandle:



### Public Member Functions

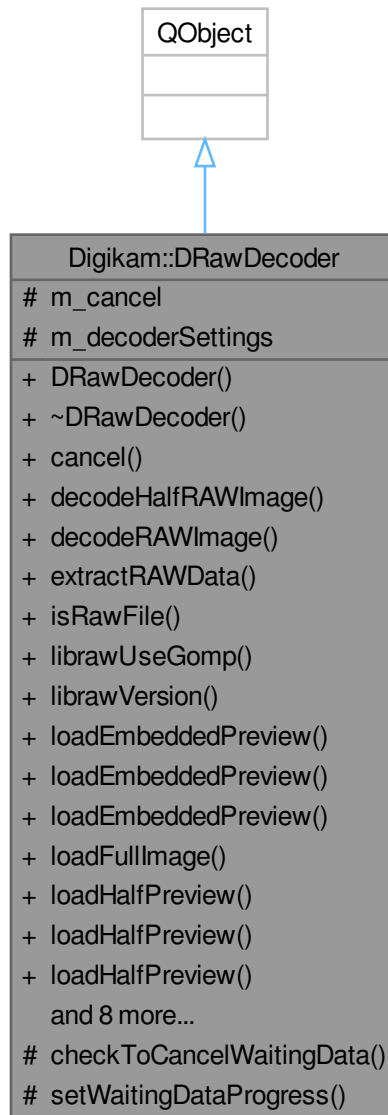
- **DragHandle** (QDockWidget \*const)
- QSize **minimumSizeHint** () const override
- QSize **sizeHint** () const override

### Protected Member Functions

- void **paintEvent** (QPaintEvent \*) override

## 9.447 Digikam::DRawDecoder Class Reference

Inheritance diagram for Digikam::DRawDecoder:



### Public Member Functions

- **DRawDecoder ()**  
*Standard constructor.*
- **~DRawDecoder ()** override  
*Standard destructor.*
- void **cancel ()**  
*To cancel 'decodeHalfRAWImage' and 'decodeRAWImage' methods running in a separate thread.*

- bool [decodeHalfRAWImage](#) (const QString &filePath, const [DRawDecoderSettings](#) &[DRawDecoderSettings](#), QByteArray &imageData, int &width, int &height, int &rgbmax)  
*Extract a small size of decode RAW data from 'filePath' picture file using 'DRawDecoderSettings' settings.*
- bool [decodeRAWImage](#) (const QString &filePath, const [DRawDecoderSettings](#) &[DRawDecoderSettings](#), QByteArray &imageData, int &width, int &height, int &rgbmax)  
*Extract a full size of RAW data from 'filePath' picture file using 'DRawDecoderSettings' settings.*
- bool [extractRAWData](#) (const QString &filePath, QByteArray &rawData, [DRawInfo](#) &identify, unsigned int shotSelect=0)  
*Extract Raw image data undemosaiced and without post processing from 'filePath' picture file.*

### Static Public Member Functions

- static bool [isRawFile](#) (const QUrl &url)
- static int [librawUseGomp](#) ()  
*Return true or false if LibRaw use parallel demosaicing or not (libgomp support).*
- static QString [librawVersion](#) ()  
*Return LibRaw version string.*
- static bool [loadEmbeddedPreview](#) (QByteArray &imgData, const QBuffer &inBuffer)  
*Get the embedded JPEG preview image from RAW image passed in QBuffer as a QByteArray which will include Exif Data.*
- static bool [loadEmbeddedPreview](#) (QByteArray &imgData, const QString &path)  
*Get the embedded JPEG preview image from RAW picture as a QByteArray which will include Exif Data.*
- static bool [loadEmbeddedPreview](#) (QImage &image, const QString &path)  
*Get the embedded JPEG preview image from RAW picture as a QImage.*
- static bool [loadFullImage](#) (QImage &image, const QString &path, const [DRawDecoderSettings](#) &settings=[DRawDecoderSettings](#)())  
*Get the full decoded RAW picture.*
- static bool [loadHalfPreview](#) (QByteArray &imgData, const QBuffer &inBuffer)  
*Get the half decoded RAW picture passed in QBuffer as JPEG data in QByteArray.*
- static bool [loadHalfPreview](#) (QByteArray &imgData, const QString &path)  
*Get the half decoded RAW picture as JPEG data in QByteArray.*
- static bool [loadHalfPreview](#) (QImage &image, const QString &path, bool rotate=true)  
*Get the half decoded RAW picture.*
- static bool [loadRawPreview](#) (QByteArray &imgData, const QBuffer &inBuffer)  
*Get the preview of RAW picture passed in QBuffer as a QByteArray holding JPEG data.*
- static bool [loadRawPreview](#) (QByteArray &imgData, const QString &path)  
*Get the preview of RAW picture as a QByteArray holding JPEG data.*
- static bool [loadRawPreview](#) (QImage &image, const QString &path)  
*Get the preview of RAW picture as a QImage.*
- static bool [rawFileIdentify](#) ([DRawInfo](#) &identify, const QString &path)  
*Get the camera settings which have taken RAW file.*
- static QString [rawFiles](#) ()  
*Return the string of all RAW file type mime supported.*
- static QStringList [rawFilesList](#) ()  
*Return the list of all RAW file type mime supported, as a QStringList, without wildcard and suffix dot.*
- static int [rawFilesVersion](#) ()  
*Returns a version number for the list of supported RAW file types.*
- static QStringList [supportedCamera](#) ()  
*Provide a list of supported RAW Camera name.*

## Protected Member Functions

- virtual bool [checkToCancelWaitingData](#) ()  
*Re-implement this method to control the cancelisation of loop which wait data from RAW decoding process with your proper environment.*
- virtual void [setWaitingDataProgress](#) (double value)  
*Re-implement this method to control the pseudo progress value during RAW decoding (when ddraw run with an internal loop without feedback) with your proper environment.*

## Protected Attributes

- bool [m\\_cancel](#) = false  
*Used internally to cancel RAW decoding operation.*
- [DRawDecoderSettings m\\_decoderSettings](#)  
*The settings container used to perform RAW pictures decoding.*

## Friends

- class **Private**

## 9.447.1 Member Function Documentation

### 9.447.1.1 checkToCancelWaitingData()

```
bool Digikam::DRawDecoder::checkToCancelWaitingData ( ) [protected], [virtual]
```

By default, this method check if `m_cancel` is true.

### 9.447.1.2 decodeHalfRAWImage()

```
bool Digikam::DRawDecoder::decodeHalfRAWImage (
    const QString & filePath,
    const DRawDecoderSettings & DRawDecoderSettings,
    QByteArray & imageData,
    int & width,
    int & height,
    int & rgbmax )
```

This is a cancelable method which require a class instance to run because RAW pictures decoding can take a while.

This method return:

- A byte array container 'imageData' with picture data. Pixels order is RGB. Color depth can be 8 or 16. In 8 bits you can access to color component using (uchar\*), in 16 bits using (ushort\*).
- Size size of image in number of pixels ('width' and 'height').
- The max average of RGB components from decoded picture.
- 'false' is returned if decoding failed, else 'true'.



### 9.447.1.3 decodeRAWImage()

```
bool Digikam::DRawDecoder::decodeRAWImage (
    const QString & filePath,
    const DRawDecoderSettings & DRawDecoderSettings,
    QByteArray & imageData,
    int & width,
    int & height,
    int & rgbmax )
```

This is a cancelable method which require a class instance to run because RAW pictures decoding can take a while.

This method return:

- A byte array container 'imageData' with picture data. Pixels order is RGB. Color depth can be 8 or 16. In 8 bits you can access to color component using (uchar\*), in 16 bits using (ushort\*).
- Size size of image in number of pixels ('width' and 'height').
- The max average of RGB components from decoded picture.
- 'false' is returned if decoding failed, else 'true'.

### 9.447.1.4 extractRAWData()

```
bool Digikam::DRawDecoder::extractRAWData (
    const QString & filePath,
    QByteArray & rawData,
    DRawInfo & identify,
    unsigned int shotSelect = 0 )
```

This is a cancelable method which require a class instance to run because RAW pictures loading can take a while.

This method return:

- A byte array container 'rawData' with raw data.
- All info about Raw image into 'identify' container.
- 'false' is returned if loading failed, else 'true'.

### 9.447.1.5 librawUseGomp()

```
int Digikam::DRawDecoder::librawUseGomp ( ) [static]
```

Return -1 if undefined.

### 9.447.1.6 loadEmbeddedPreview() [1/3]

```
bool Digikam::DRawDecoder::loadEmbeddedPreview (
    QByteArray & imgData,
    const QBuffer & inBuffer ) [static]
```

This is fast and non cancelable. This method does not require a class instance to run.

#### 9.447.1.7 loadEmbeddedPreview() [2/3]

```
bool Digikam::DRawDecoder::loadEmbeddedPreview (
    QByteArray & imgData,
    const QString & path ) [static]
```

This is fast and non cancelable. This method does not require a class instance to run.

#### 9.447.1.8 loadEmbeddedPreview() [3/3]

```
bool Digikam::DRawDecoder::loadEmbeddedPreview (
    QImage & image,
    const QString & path ) [static]
```

This is fast and non cancelable This method does not require a class instance to run.

#### 9.447.1.9 loadFullImage()

```
bool Digikam::DRawDecoder::loadFullImage (
    QImage & image,
    const QString & path,
    const DRawDecoderSettings & settings = DRawDecoderSettings() ) [static]
```

This is a more slower than [loadHalfPreview\(\)](#) method and non cancelable. This method does not require a class instance to run.

#### 9.447.1.10 loadHalfPreview() [1/3]

```
bool Digikam::DRawDecoder::loadHalfPreview (
    QByteArray & imgData,
    const QBuffer & inBuffer ) [static]
```

This is slower than [loadEmbeddedPreview\(\)](#) method and non cancelable. This method does not require a class instance to run.

#### 9.447.1.11 loadHalfPreview() [2/3]

```
bool Digikam::DRawDecoder::loadHalfPreview (
    QByteArray & imgData,
    const QString & path ) [static]
```

This is slower than [loadEmbeddedPreview\(\)](#) method and non cancelable. This method does not require a class instance to run.

#### 9.447.1.12 loadHalfPreview() [3/3]

```
bool Digikam::DRawDecoder::loadHalfPreview (
    QImage & image,
    const QString & path,
    bool rotate = true ) [static]
```

This is slower than [loadEmbeddedPreview\(\)](#) method and non cancelable. This method does not require a class instance to run.

**9.447.1.13 loadRawPreview()** [1/3]

```
static bool Digikam::DRawDecoder::loadRawPreview (
    QByteArray & imgData,
    const QBuffer & inBuffer ) [static]
```

It tries [loadEmbeddedPreview\(\)](#) first and if it fails, calls [loadHalfPreview\(\)](#).

**9.447.1.14 loadRawPreview()** [2/3]

```
static bool Digikam::DRawDecoder::loadRawPreview (
    QByteArray & imgData,
    const QString & path ) [static]
```

It tries [loadEmbeddedPreview\(\)](#) first and if it fails, calls [loadHalfPreview\(\)](#).

**9.447.1.15 loadRawPreview()** [3/3]

```
bool Digikam::DRawDecoder::loadRawPreview (
    QImage & image,
    const QString & path ) [static]
```

It tries [loadEmbeddedPreview\(\)](#) first and if it fails, calls [loadHalfPreview\(\)](#).

**9.447.1.16 rawFileIdentify()**

```
bool Digikam::DRawDecoder::rawFileIdentify (
    DRawInfo & identify,
    const QString & path ) [static]
```

Look into `rawinfo.h` for more details. This is a fast and non cancelable method which do not require a class instance to run.

**9.447.1.17 rawFilesVersion()**

```
int Digikam::DRawDecoder::rawFilesVersion ( ) [static]
```

This version is incremented if the list of supported formats has changed between library releases.

**9.447.1.18 setWaitingDataProgress()**

```
void Digikam::DRawDecoder::setWaitingDataProgress (
    double value ) [protected], [virtual]
```

By default, this method does nothing. Progress value average for this stage is 0%-n%, with 'n' == 40% max (see [setWaitingDataProgress\(\)](#) method).

## 9.447.2 Member Data Documentation

### 9.447.2.1 m\_cancel

```
bool Digikam::DRawDecoder::m_cancel = false [protected]
```

Normally, you don't need to use it directly, excepted if you derivated this class. Usual way is to use [cancel\(\)](#) method

### 9.447.2.2 m\_decoderSettings

```
DRawDecoderSettings Digikam::DRawDecoder::m_decoderSettings [protected]
```

See 'drawdecodingsetting.h' for details.

## 9.448 Digikam::DRawDecoderSettings Class Reference

### Public Types

- enum [DecodingQuality](#) {  
**BILINEAR** = 0 , **VNG** = 1 , **PPG** = 2 , **AHD** = 3 ,  
**DCB** = 4 , **DHT** = 11 , **AAHD** = 12 }  
*RAW decoding Interpolation methods.*
- enum [InputColorSpace](#) { **NOINPUTCS** = 0 , **EMBEDDED** , **CUSTOMINPUTCS** }  
*Input color profile used to decoded image NOINPUTCS: No input color profile.*
- enum [NoiseReduction](#) { **NONR** = 0 , **WAVELETSNR** , **FBDDNR** }  
*Noise Reduction method to apply before demosaicing NONR: No noise reduction.*
- enum [OutputColorSpace](#) {  
**RAWCOLOR** = 0 , **SRGB** , **ADOBERGB** , **WIDEGAMMUT** ,  
**PROPHOTO** , **CUSTOMOUTPUTCS** }  
*Output RGB color space used to decoded image RAWCOLOR: No output color profile (Linear RAW).*
- enum [WhiteBalance](#) {  
**NONE** = 0 , **CAMERA** = 1 , **AUTO** = 2 , **CUSTOM** = 3 ,  
**AERA** = 4 }  
*White balances alternatives NONE: no white balance used : reverts to standard daylight D65 WB.*

### Public Member Functions

- [DRawDecoderSettings](#) ()=default  
*Standard constructor with default settings.*
- [DRawDecoderSettings](#) (const [DRawDecoderSettings](#) &o)  
*Equivalent to the copy constructor.*
- [~DRawDecoderSettings](#) ()=default  
*Standard destructor.*
- [DRawDecoderSettings](#) & **operator=** (const [DRawDecoderSettings](#) &o)
- bool **operator==** (const [DRawDecoderSettings](#) &o) const  
*Compare for equality.*
- void **optimizeTimeLoading** ()  
*Method to use a settings to optimize time loading, for example to compute image histogram.*

## Public Attributes

- bool **autoBrightness** = true  
*If false, use a fixed white level, ignoring the image histogram.*
- int **blackPoint** = 0  
*Black Point value of output image.*
- double **brightness** = 1.0  
*Brightness of output image.*
- int **customWhiteBalance** = 6500  
*The temperature and the green multiplier of the custom white balance.*
- double **customWhiteBalanceGreen** = 1.0
- bool **dcbEnhanceFI** = false  
*Turn on the DCB interpolation with enhance interpolated colors.*
- int **dcbIterations** = -1  
*For DCB interpolation.*
- QString **deadPixelMap**  
*Path to text file including dead pixel list.*
- bool **DontStretchPixels** = false  
*For cameras with non-square pixels, do not stretch the image to its correct aspect ratio.*
- bool **enableBlackPoint** = false  
*Turn on the black point setting to decode RAW image.*
- bool **enableWhitePoint** = false  
*Turn on the white point setting to decode RAW image.*
- bool **expoCorrection** = false  
*Turn on the Exposure Correction before interpolation.*
- double **expoCorrectionHighlight** = 0.0  
*Amount of highlight preservation for exposure correction before interpolation in E.V.*
- double **expoCorrectionShift** = 1.0  
*Shift of Exposure Correction before interpolation in linear scale.*
- bool **fixColorsHighlights** = false  
*If true, images with overblown channels are processed much more accurate, without 'pink clouds' (and blue highlights under tungsten lamps).*
- bool **halfSizeColorImage** = false  
*Half-size color image decoding (twice as fast as "enableRAWQuality").*
- InputColorSpace **inputColorSpace** = NOINPUTCS  
*The input color profile used to decoded RAW data.*
- QString **inputProfile**  
*Path to custom input ICC profile to define the camera's raw colorspace.*
- int **medianFilterPasses** = 0  
*After interpolation, clean up color artifacts by repeatedly applying a 3x3 median filter to the R-G and B-G channels.*
- int **NRThreshold** = 0  
*Noise reduction threshold value.*
- NoiseReduction **NRType** = NONR  
*Noise reduction method to apply before demosaicing.*
- OutputColorSpace **outputColorSpace** = SRGB  
*The output color profile used to decoded RAW data.*
- QString **outputProfile**  
*Path to custom output ICC profile to define the color workspace.*
- DecodingQuality **RAWQuality** = BILINEAR  
*RAW quality decoding factor value.*
- bool **RGBInterpolate4Colors** = false

- Turn on RAW file decoding using RGB interpolation as four colors.*

  - bool **sixteenBitsImage** = false
    - Turn on RAW file decoding in 16 bits per color per pixel instead 8 bits.*
  - int **unclipColors** = 0
    - Unclip Highlight color level: 0 = Clip all highlights to solid white.*
  - **WhiteBalance whiteBalance** = CAMERA
    - White balance type to use.*
  - QRect **whiteBalanceArea**
    - Rectangle used to calculate the white balance by averaging the region of image.*
  - int **whitePoint** = 0
    - White Point value of output image.*

## 9.448.1 Member Enumeration Documentation

### 9.448.1.1 DecodingQuality

```
enum Digikam::DRawDecoderSettings::DecodingQuality
```

#### Note

from original dcraw demosaic

Bilinear: use high-speed but low-quality bilinear interpolation (default - for slow computer). In this method, the red value of a non-red pixel is computed as the average of the adjacent red pixels, and similar for blue and green. VNG: use Variable Number of Gradients interpolation. This method computes gradients near the pixel of interest and uses the lower gradients (representing smoother and more similar parts of the image) to make an estimate. PPG↔: use Patterned Pixel Grouping interpolation. Pixel Grouping uses assumptions about natural scenery in making estimates. It has fewer color artifacts on natural images than the Variable Number of Gradients method. AHD: use Adaptive Homogeneity-Directed interpolation. This method selects the direction of interpolation so as to maximize a homogeneity metric, thus typically minimizing color artifacts. DCB: DCB interpolation (see [www.linuxphoto.org/html/dcb.html](http://www.linuxphoto.org/html/dcb.html) for details) DHT: DHT interpolation. AAHD: Enhanced Adaptive AHD interpolation.

### 9.448.1.2 InputColorSpace

```
enum Digikam::DRawDecoderSettings::InputColorSpace
```

EMBEDDED: Use the camera profile embedded in RAW file if exist. CUSTOMINPUTCS: Use a custom input color space profile.

### 9.448.1.3 NoiseReduction

```
enum Digikam::DRawDecoderSettings::NoiseReduction
```

WAVELETSNR: wavelets correction to erase noise while preserving real detail. It's applied after interpolation. FBDDNR: Fake Before Demosaicing Denoising noise reduction. It's applied before interpolation.

### 9.448.1.4 OutputColorSpace

enum `Digikam::DRawDecoderSettings::OutputColorSpace`

SRGB: Use standard sRGB color space. ADOBERGB: Use standard Adobe RGB color space. WIDEGAMMUT↔ : Use standard RGB Wide Gamut color space. PROPHOTO: Use standard RGB Pro Photo color space. CUSTOMOUTPUTCS: Use a custom workspace color profile.

### 9.448.1.5 WhiteBalance

enum `Digikam::DRawDecoderSettings::WhiteBalance`

CAMERA: Use the camera embedded WB if available. Reverts to NONE if not. AUTO: Averages an auto WB on the entire image. CUSTOM: Let use set it's own temperature and green factor (later converted to RGBG factors). AERA: Let use an area from image to average white balance (see `whiteBalanceArea` for details).

## 9.448.2 Member Data Documentation

### 9.448.2.1 dcbIterations

```
int Digikam::DRawDecoderSettings::dcbIterations = -1
```

Number of DCB median filtering correction passes. -1 : disable (default) 1-10 : DCB correction passes

### 9.448.2.2 DontStretchPixels

```
bool Digikam::DRawDecoderSettings::DontStretchPixels = false
```

In any case, this option guarantees that each output pixel corresponds to one RAW pixel.

### 9.448.2.3 expoCorrectionHighlight

```
double Digikam::DRawDecoderSettings::expoCorrectionHighlight = 0.0
```

Usable range is from 0.0 (linear exposure shift, highlights may blow) to 1.0 (maximum highlights preservation) This settings can only take effect if `expoCorrectionShift > 1.0`.

### 9.448.2.4 expoCorrectionShift

```
double Digikam::DRawDecoderSettings::expoCorrectionShift = 1.0
```

Usable range is from 0.25 (darken image 1 stop : -2EV) to 8.0 (lighten ~1.5 photographic stops : +3EV).

### 9.448.2.5 halfSizeColorImage

```
bool Digikam::DRawDecoderSettings::halfSizeColorImage = false
```

Turn on this option to reduce time loading to render histogram for example, no to render an image to screen.

### 9.448.2.6 inputColorSpace

`InputColorSpace` Digikam::DRawDecoderSettings::inputColorSpace = NOINPUTCS

See OutputColorProfile values for details.

### 9.448.2.7 NRThreshold

`int` Digikam::DRawDecoderSettings::NRThreshold = 0

Null value disable NR. Range is between 100 and 1000. For IMPULSENK : set the amount of Luminance impulse denoise.

### 9.448.2.8 outputColorSpace

`OutputColorSpace` Digikam::DRawDecoderSettings::outputColorSpace = SRGB

See OutputColorProfile values for details.

### 9.448.2.9 RAWQuality

`DecodingQuality` Digikam::DRawDecoderSettings::RAWQuality = BILINEAR

See DecodingQuality values for details.

### 9.448.2.10 unclipColors

`int` Digikam::DRawDecoderSettings::unclipColors = 0

1 = Leave highlights unclipped in various shades of pink. 2 = Blend clipped and unclipped values together for a gradual fade to white. 3-9 = Reconstruct highlights. Low numbers favor whites; high numbers favor colors.

### 9.448.2.11 whiteBalance

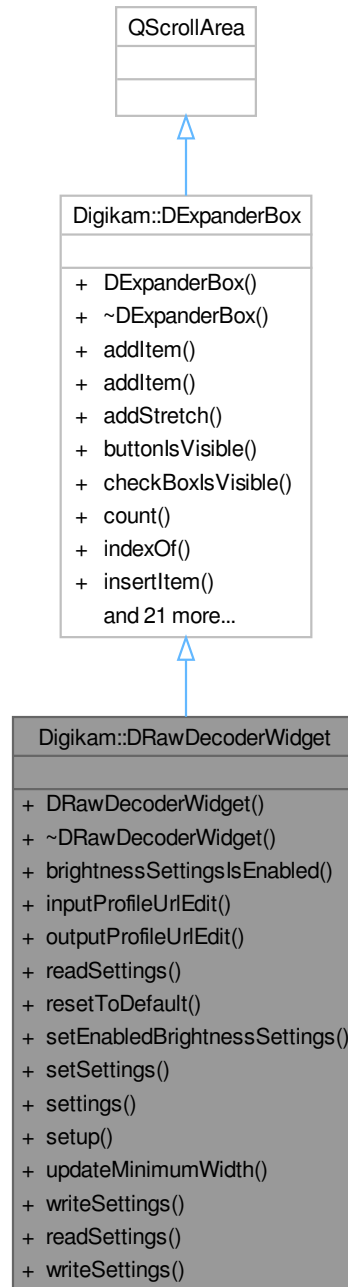
`WhiteBalance` Digikam::DRawDecoderSettings::whiteBalance = CAMERA

See WhiteBalance values for detail



## 9.449 Digikam::DRawDecoderWidget Class Reference

Inheritance diagram for Digikam::DRawDecoderWidget:



### Public Types

- enum **AdvancedSettingsOptions** { **SIXTEENBITS** = 0x00000001 , **COLORSPACE** = 0x00000002 , **POST-PROCESSING** = 0x00000004 , **BLACKWHITEPOINTS** = 0x00000008 }
- enum **SettingsTabs** { **DEMOSAICING** = 0 , **WHITEBALANCE** , **CORRECTIONS** , **COLORMANAGEMENT** }

## Signals

- void **signalSettingsChanged** ()
- void **signalSixteenBitsImageToggled** (bool)

## Signals inherited from [Digikam::DExpanderBox](#)

- void **signalItemButtonPressed** (int index)
- void **signalItemExpanded** (int index, bool b)
- void **signalItemToggled** (int index, bool b)

## Public Member Functions

- [DRawDecoderWidget](#) (QWidget \*const parent, int advSettings=COLORSPACE)  
*The widget to host the Raw Decoder settings.*
- bool **brightnessSettingsIsEnabled** () const
- [DFileSelector](#) \* **inputProfileUrlEdit** () const
- [DFileSelector](#) \* **outputProfileUrlEdit** () const
- void **readSettings** (KConfigGroup &group) override
- void **resetToDefault** ()
- void **setEnabledBrightnessSettings** (bool b)
- void **setSettings** (const [DRawDecoderSettings](#) &settings)
- [DRawDecoderSettings](#) **settings** () const
- void **setup** (int advSettings)
- void **updateMinimumWidth** ()
- void **writeSettings** (KConfigGroup &group) override

## Public Member Functions inherited from [Digikam::DExpanderBox](#)

- **DExpanderBox** (QWidget \*const parent=nullptr)
- void **addItem** (QWidget \*const w, const QIcon &icon, const QString &txt, const QString &objName, bool expandBydefault)  
*Add [DLabelExpander](#) item at end of box layout with these settings : 'w' : the widget hosted by [DLabelExpander](#).*
- void **addItem** (QWidget \*const w, const QString &txt, const QString &objName, bool expandBydefault)
- void **addStretch** ()
- bool **buttonsVisible** (int index) const
- bool **checkboxesVisible** (int index) const
- int **count** () const
- int **indexOf** ([DLabelExpander](#) \*const widget) const
- void **insertItem** (int index, QWidget \*const w, const QIcon &icon, const QString &txt, const QString &objName, bool expandBydefault)  
*Insert [DLabelExpander](#) item at box layout index with these settings : 'w' : the widget hosted by [DLabelExpander](#).*
- void **insertItem** (int index, QWidget \*const w, const QString &txt, const QString &objName, bool expandBydefault)
- void **insertStretch** (int index)
- bool **isChecked** (int index) const
- bool **isItemEnabled** (int index) const
- bool **isItemExpanded** (int index) const
- QIcon **itemIcon** (int index) const
- QString **itemText** (int index) const
- QString **itemToolTip** (int index) const
- void **removeItem** (int index)

- void **setButtonIcon** (int index, const QIcon &icon)
- void **setButtonVisible** (int index, bool b)
- void **setCheckBoxVisible** (int index, bool b)
- void **setChecked** (int index, bool b)
- void **setItemEnabled** (int index, bool enabled)
- void **setItemExpanded** (int index, bool b)
- void **setItemIcon** (int index, const QIcon &icon)
- void **setItemText** (int index, const QString &txt)
- void **setItemToolTip** (int index, const QString &tip)
- [DLabelExpander](#) \* **widget** (int index) const

### Static Public Member Functions

- static void **readSettings** ([DRawDecoderSettings](#) &setting, const KConfigGroup &group)
- static void **writeSettings** (const [DRawDecoderSettings](#) &setting, KConfigGroup &group)

## 9.449.1 Constructor & Destructor Documentation

### 9.449.1.1 DRawDecoderWidget()

```
Digikam::DRawDecoderWidget::DRawDecoderWidget (
    QWidget *const parent,
    int advSettings = COLORSPACE ) [explicit]
```

#### Parameters

<i>parent</i>	the parent widget instance
<i>advSettings</i>	the default value is COLORSPACE

## 9.449.2 Member Function Documentation

### 9.449.2.1 readSettings()

```
void Digikam::DRawDecoderWidget::readSettings (
    KConfigGroup & group ) [override], [virtual]
```

Reimplemented from [Digikam::DExpanderBox](#).

### 9.449.2.2 writeSettings()

```
void Digikam::DRawDecoderWidget::writeSettings (
    KConfigGroup & group ) [override], [virtual]
```

Reimplemented from [Digikam::DExpanderBox](#).

## 9.450 Digikam::DRawDecoding Class Reference

### Public Member Functions

- **DRawDecoding** ()  
*Standard constructor with default settings.*
- **DRawDecoding** (const [DRawDecoderSettings](#) &prm)  
*Copy constructor.*
- **~DRawDecoding** ()=default  
*Standard destructor.*
- bool **operator==** (const [DRawDecoding](#) &other) const  
*Equality operator.*
- void **optimizeTimeLoading** ()  
*Method to use a settings to optimize time loading, for example to compute image histogram.*
- bool **postProcessingSettingsIsDirty** () const  
*Method to check is a post-processing setting have been changed.*
- void **resetPostProcessingSettings** ()  
*Method to reset to default values all Raw processing settings.*
- void **writeToFilterAction** ([FilterAction](#) &action, const QString &prefix=QString()) const

### Static Public Member Functions

- static void **decodingSettingsFromXml** (const QDomElement &elm, [DRawDecoderSettings](#) &prm)
- static void **decodingSettingsToXml** (const [DRawDecoderSettings](#) &prm, QDomElement &elm)  
*Used by BQM to read/store Queue Raw decoding settings from/to configuration file.*
- static [DRawDecoding](#) **fromFilterAction** (const [FilterAction](#) &action, const QString &prefix=QString())

### Public Attributes

- [BCGContainer](#) **bcg**  
*Post Processing settings -----.*
- [CurvesContainer](#) **curvesAdjust**  
*Curve adjustments.*
- [DRawDecoderSettings](#) **rawPrm**  
*All Raw decoding settings provided by RawEngine.*
- [WBContainer](#) **wb**  
*White Balance correction values.*

### 9.450.1 Constructor & Destructor Documentation

#### 9.450.1.1 DRawDecoding()

```
Digikam::DRawDecoding::DRawDecoding (
    const DRawDecoderSettings & prm ) [explicit]
```

Creates a copy of a [DRawDecoderSettings](#) object.

## 9.450.2 Member Data Documentation

### 9.450.2.1 bcg

`BCGContainer` Digikam::DRawDecoding::bcg

BCG correction values.

## 9.451 Digikam::DRawInfo Class Reference

### Public Types

- enum `ImageOrientation` {  
**ORIENTATION\_NONE** = 0 , **ORIENTATION\_180** = 3 , **ORIENTATION\_Mirror90CCW** = 4 , **ORIENTATION\_90CCW** = 5 ,  
**ORIENTATION\_90CW** = 6 }

*The RAW image orientation values.*

### Public Member Functions

- `DRawInfo` ()  
*Standard constructor.*
- `~DRawInfo` ()=default  
*Standard destructor.*

### Public Attributes

- double **altitude** = 0.0F
- float `ambientAcceleration` = -1000.0F  
*Directionless camera acceleration in units of mGal, or 10-5 m/s<sup>2</sup>.*
- float `ambientElevationAngle` = -1000.0F  
*Camera elevation angle in degrees.*
- float `ambientHumidity` = -1000.0F  
*Ambient relative humidity in percent.*
- float `ambientPressure` = -1000.0F  
*Ambient air pressure in hPa or mbar.*
- float `ambientTemperature` = -1000.0F  
*Ambient temperature in Celsius degrees.*
- float `ambientWaterDepth` = 1000.0F  
*Depth under water in metres, negative for above water.*
- float **aperture** = -1.0F  
*Aperture value in APEX.*
- float `baselineExposure` = -999.0F  
*Exposure compensation to be applied during raw conversion.*
- unsigned int **blackPoint** = 0  
*Black level from Raw histogram.*
- unsigned int **blackPointCh** [4] = { 0 }  
*Channel black levels from Raw histogram.*

- float **cameraColorMatrix1** [3][4]  
*Camera Color [Matrix](#).*
- float **cameraColorMatrix2** [3][4]
- double **cameraMult** [4] = { 0.0 }  
*Camera multipliers used for White Balance adjustments.*
- float **cameraXYZMatrix** [4][3]
- QString **colorKeys**  
*The used Color Keys.*
- QDateTime **dateTime**  
*Date & time when the picture has been taken.*
- double **daylightMult** [3] = { 0.0 }  
*White color balance settings.*
- QString **description**  
*The image description of raw image.*
- QString **DNGVersion**  
*The DNG version.*
- float **exposureIndex** = -1.0F  
*Exposure Index from the camera.*
- int **exposureProgram** = -1  
*The exposure program used by camera.*
- float **exposureTime** = -1.0F  
*1/exposureTime = exposure time in seconds.*
- QString **filterPattern**  
*The demosaicing filter pattern.*
- QString **firmware**  
*The Firmware name or version which create raw image.*
- int **flashUsed** = -1  
*Describe how flash has been used by camera.*
- float **focalLength** = -1.0F  
*Focal Length value in mm.*
- int **focalLengthIn35mmFilm** = -1  
*Valid value is unsigned.*
- QSize **fullSize**  
*The full RAW image dimensions in pixels.*
- bool **hasGpsInfo** = false  
*true if GPS info are parsed from RAW file.*
- bool **hasIccProfile** = false  
*True if RAW file include an ICC color profile.*
- QByteArray **iccData**  
*ICC color profilr container extracted from RAW file, if present.*
- QString **imageID**  
*An unique image ID generated by camera.*
- QSize **imageSize**  
*The image dimensions in pixels.*
- bool **isDecodable** = false  
*True is RAW file is decodable by dcrw.*
- double **latitude** = 0.0F  
*GPS information.*
- unsigned int **leftMargin** = 0  
*Left margin of raw image.*
- QString **lensMake**

- QString **lensModel**  
*Description of lens properties.*
- QString **lensSerial**
- QString **localizedCameraModel**  
*Localized name for the camera model that created the raw file.*
- double **longitude** = 0.0F
- QString **make**  
*The camera maker.*
- float **maxAperture** = -1.0F  
*Valid value is unsigned.*
- int **meteringMode** = -1  
*The metering mode used by camera.*
- QString **model**  
*The camera model.*
- **ImageOrientation orientation** = ORIENTATION\_NONE  
*The raw image orientation.*
- QString **originalRawFileName**  
*The original RAW file name.*
- QSize **outputSize**  
*The output dimensions in pixels.*
- QString **owner**  
*The artist name who have picture owner.*
- float **pixelAspectRatio** = 1.0F  
*The pixel Aspect Ratio if != 1.0.*
- int **rawColors** = -1  
*The number of RAW colors.*
- QString **rawDataUniqueID**  
*An unique RAW data ID.*
- int **rawImages** = -1  
*The number of RAW images.*
- float **sensitivity** = -1.0F  
*The sensitivity in ISO used by camera to take the picture.*
- unsigned int **serialNumber** = 0  
*Serial number of raw image.*
- QString **software**  
*The software name which process raw image.*
- QByteArray **thumbnail**  
*Thumbnail image data extracted from raw file.*
- QSize **thumbSize**  
*The thumb dimensions in pixels.*
- unsigned int **topMargin** = 0  
*Top margin of raw image.*
- QString **uniqueCameraModel**  
*Non-localized name for the camera model that created the raw file.*
- unsigned int **whitePoint** = 0  
*White level from Raw histogram.*
- QByteArray **xmpData**  
*Xmp metadata container extracted from RAW file, if present.*

## 9.451.1 Constructor & Destructor Documentation

### 9.451.1.1 DRawInfo()

```
Digikam::DRawInfo::DRawInfo ( ) [explicit]
```

< NOTE: see bug #253911 : [y][x] not [x][y]

## 9.451.2 Member Data Documentation

### 9.451.2.1 ambientAcceleration

```
float Digikam::DRawInfo::ambientAcceleration = -1000.0F
```

-1000 is an invalid acceleration.

### 9.451.2.2 ambientElevationAngle

```
float Digikam::DRawInfo::ambientElevationAngle = -1000.0F
```

-1000 is an invalid angle.

### 9.451.2.3 ambientHumidity

```
float Digikam::DRawInfo::ambientHumidity = -1000.0F
```

-1000 is an invalid humidity.

### 9.451.2.4 ambientPressure

```
float Digikam::DRawInfo::ambientPressure = -1000.0F
```

-1000 is an invalid pressure.

### 9.451.2.5 ambientTemperature

```
float Digikam::DRawInfo::ambientTemperature = -1000.0F
```

-1000 is an invalid temperature.

### 9.451.2.6 ambientWaterDepth

```
float Digikam::DRawInfo::ambientWaterDepth = 1000.0F
```

1000 is an invalid water depth.



### 9.451.2.7 baselineExposure

```
float Digikam::DRawInfo::baselineExposure = -999.0F
```

-999 is an invalid exposure.

### 9.451.2.8 DNGVersion

```
QString Digikam::DRawInfo::DNGVersion
```

NOTE: it is only shown with DNG RAW files.

### 9.451.2.9 exposureIndex

```
float Digikam::DRawInfo::exposureIndex = -1.0F
```

Valid value is unsigned.

### 9.451.2.10 exposureProgram

```
int Digikam::DRawInfo::exposureProgram = -1
```

Valid value is unsigned.

### 9.451.2.11 flashUsed

```
int Digikam::DRawInfo::flashUsed = -1
```

Valid value is unsigned.

### 9.451.2.12 meteringMode

```
int Digikam::DRawInfo::meteringMode = -1
```

Valid value is unsigned.

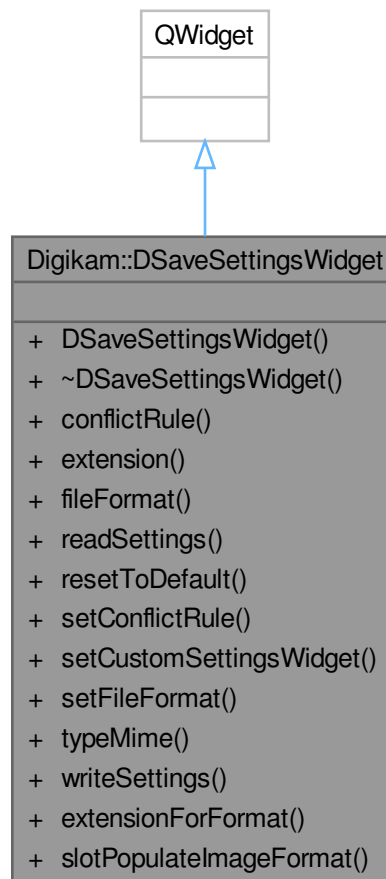
### 9.451.2.13 pixelAspectRatio

```
float Digikam::DRawInfo::pixelAspectRatio = 1.0F
```

NOTE: if == 1.0, libraw CLI tool do not show this value. Default value = 1.0. This can be unavailable (depending of camera model).

## 9.452 Digikam::DSaveSettingsWidget Class Reference

Inheritance diagram for Digikam::DSaveSettingsWidget:



### Public Types

- enum `OutputFormat` { `OUTPUT_PNG = 0` , `OUTPUT_TIFF` , `OUTPUT_JPEG` , `OUTPUT_PPM` }

### Public Slots

- void `slotPopulateImageFormat` (bool sixteenBits)

### Signals

- void `signalConflictButtonChanged` (int)
- void `signalSaveFormatChanged` ()

### Public Member Functions

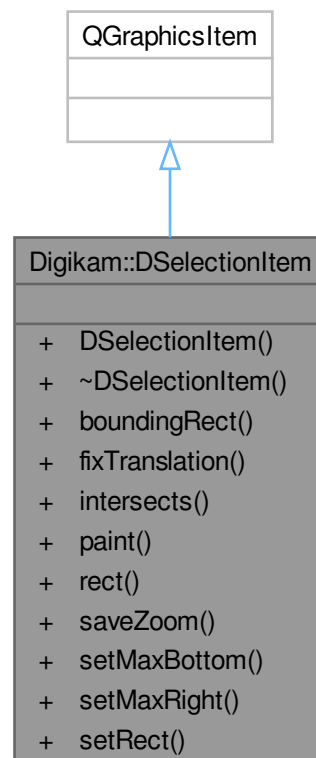
- **DSaveSettingsWidget** (QWidget \*const parent)
- FileSaveConflictBox::ConflictRule **conflictRule** () const
- QString **extension** () const
- OutputFormat **fileFormat** () const
- void **readSettings** (KConfigGroup &group)
- void **resetToDefault** ()
- void **setConflictRule** (FileSaveConflictBox::ConflictRule r)
- void **setCustomSettingsWidget** (QWidget \*const custom)
- void **setFileFormat** (OutputFormat f)
- QString **typeMime** () const
- void **writeSettings** (KConfigGroup &group)

### Static Public Member Functions

- static QString **extensionForFormat** (OutputFormat format)

## 9.453 Digikam::DSelectedItem Class Reference

Inheritance diagram for Digikam::DSelectedItem:



## Public Types

- enum **Intersects** {  
    **None** , **Top** , **TopRight** , **Right** ,  
    **BottomRight** , **Bottom** , **BottomLeft** , **Left** ,  
    **TopLeft** , **Move** }

## Public Member Functions

- **DSelectionItem** (const QRectF &rect)
- QRectF **boundingRect** () const override
- QPointF **fixTranslation** (QPointF dp) const
- Intersects **intersects** (QPointF &point)
- void **paint** (QPainter \*painter, const QStyleOptionGraphicsItem \*option, QWidget \*widget) override
- QRectF **rect** () const
- void **saveZoom** (qreal zoom)
- void **setMaxBottom** (qreal maxBottom)
- void **setMaxRight** (qreal maxRight)
- void **setRect** (const QRectF &rect)

## 9.454 Digikam::DSelector Class Reference

[DSelector](#) is the base class for other widgets which provides the ability to choose from a one-dimensional range of values.

Inheritance diagram for Digikam::DSelector:



## Public Member Functions

- **DSelector** (Qt::Orientation o, QWidget \*const parent=nullptr)
- **DSelector** (QWidget \*const parent=nullptr)
- Qt::ArrowType **arrowDirection** () const
- QRect **contentsRect** () const
- bool **indent** () const

- void **setArrowDirection** (Qt::ArrowType direction)  
*Sets the arrow direction.*
- void **setIndent** (bool i)  
*Sets the indent option of the widget to i.*

### Protected Member Functions

- virtual void **drawArrow** (QPainter \*painter, const QPoint &pos)  
*Override this function to draw the cursor which indicates the current value.*
- virtual void **drawContents** (QPainter \*)  
*Override this function to draw the contents of the control.*
- void **mouseMoveEvent** (QMouseEvent \*e) override
- void **mousePressEvent** (QMouseEvent \*e) override
- void **mouseReleaseEvent** (QMouseEvent \*e) override
- void **paintEvent** (QPaintEvent \*) override
- void **wheelEvent** (QWheelEvent \*) override

### Properties

- Qt::ArrowType **arrowDirection**
- bool **indent**
- int **maxValue**
- int **minValue**
- int **value**

### Friends

- class **Private**

## 9.454.1 Detailed Description

An example is the KGradientSelector which allows to choose from a range of colors.

A custom drawing routine for the widget surface has to be provided by the subclass.

## 9.454.2 Member Function Documentation

### 9.454.2.1 arrowDirection()

```
Qt::ArrowType Digikam::DSelector::arrowDirection ( ) const
```

#### Returns

the current arrow direction

### 9.454.2.2 contentsRect()

```
QRect Digikam::DSelector::contentsRect ( ) const
```

#### Returns

the rectangle on which subclasses should draw.

### 9.454.2.3 drawContents()

```
virtual void Digikam::DSelector::drawContents (
    QPainter * ) [inline], [protected], [virtual]
```

The default implementation does nothing.

Draw only within [contentsRect\(\)](#).

Reimplemented in [Digikam::DColorValueSelector](#).

### 9.454.2.4 indent()

```
bool Digikam::DSelector::indent ( ) const
```

#### Returns

whether the indent option is set.

### 9.454.2.5 setIndent()

```
void Digikam::DSelector::setIndent (
    bool i )
```

This determines whether a shaded frame is drawn.

## 9.455 Digikam::DServiceInfo Class Reference

### Public Member Functions

- **DServiceInfo** (const [DServiceInfo](#) &other)
- **DServiceInfo** (const QString &\_name, const QString &\_exec, const QString &\_icon, const QString &\_topt, bool \_term)
- bool **isEmpty** () const
- [DServiceInfo](#) & **operator=** (const [DServiceInfo](#) &other)

## Public Attributes

- QString **exec**
- QString **icon**
- QString **name**
- bool **term** = false
- QString **topt**

## 9.456 Digikam::DServiceMenu Class Reference

### Static Public Member Functions

- static QIcon **getIconFromService** (const [DServiceInfo](#) &serviceInfo)  
*Return the QIcon depending on the operating system.*
- static bool **runFiles** (const [DServiceInfo](#) &serviceInfo, const QList< QUrl > &urls)
- static bool **runFiles** (const KService::Ptr &service, const QList< QUrl > &urls)  
*Linux only: open file urls with the service.*
- static bool **runFiles** (const QString &appCmd, const QList< QUrl > &urls, const KService::Ptr &service=KService::Ptr(), const [DServiceInfo](#) &serviceInfo=[DServiceInfo](#)())  
*Linux only: open file urls with the application command.*
- static QList< [DServiceInfo](#) > **servicesForOpen** (const QList< QUrl > &urls)
- static KService::List **servicesForOpenWith** (const QList< QUrl > &urls)  
*Linux only: return list of service available on desktop to open files.*



## 9.457 Digikam::DSliderSpinBox Class Reference

Inheritance diagram for Digikam::DSliderSpinBox:



### Public Slots

- void **setValue** (int [value](#))  
Set the value, don't use [setValue\(\)](#)

## Signals

- void **valueChanged** (int [value](#))

## Public Member Functions

- **DSliderSpinBox** (QWidget \*const parent=nullptr)
- int **fastSliderStep** () const
- int **maximum** () const
- int **minimum** () const
- void **setFastSliderStep** (int step)
- void **setMaximum** (int maximum)
- void **setMinimum** (int minimum)
- void **setPageStep** (int [value](#))
- void **setRange** (int minimum, int maximum)
- void **setSingleStep** (int [value](#))
- int **value** ()

*Get the value, don't use [value\(\)](#)*

## Public Member Functions inherited from [Digikam::DAbstractSliderSpinBox](#)

- void **hideEdit** ()
- bool **isDragging** () const
- virtual QSize **minimumSize** () const
- QSize **minimumSizeHint** () const override
- void **setBlockUpdateSignalOnDrag** (bool block)

*If set to block, it informs inheriting classes that they shouldn't emit signals if the update comes from a mouse dragging the slider.*

- void **setExponentRatio** (double dbl)
- void **setPrefix** (const QString &prefix)
- void **setSuffix** (const QString &suffix)
- void **showEdit** ()
- QSize **sizeHint** () const override

## Protected Member Functions

- void **setInternalValue** (int [value](#), bool blockUpdateSignal) override

*Sets the slider internal value.*

- QString **valueString** () const override

## Protected Member Functions inherited from [Digikam::DAbstractSliderSpinBox](#)

- **DAbstractSliderSpinBox** (QWidget \*const parent, DAbstractSliderSpinBoxPrivate \*const q)
- void **changeEvent** (QEvent \*e) override
- QRect **downButtonRect** (const QStyleOptionSpinBox &spinBoxOptions) const
- bool **eventFilter** (QObject \*recv, QEvent \*e) override
- void **focusInEvent** (QFocusEvent \*e) override
- void **keyPressEvent** (QKeyEvent \*e) override
- void **mouseMoveEvent** (QMouseEvent \*e) override
- void **mousePressEvent** (QMouseEvent \*e) override
- void **mouseReleaseEvent** (QMouseEvent \*e) override
- void **paint** (QPainter &painter)
- void **paintBreeze** (QPainter &painter)
- void **paintEvent** (QPaintEvent \*e) override
- void **paintFusion** (QPainter &painter)
- void **paintPlastique** (QPainter &painter)
- QStyleOptionProgressBar **progressBarOptions** () const
- QRect **progressRect** (const QStyleOptionSpinBox &spinBoxOptions) const
- QStyleOptionSpinBox **spinBoxOptions** () const
- QRect **upButtonRect** (const QStyleOptionSpinBox &spinBoxOptions) const
- int **valueForX** (int x, Qt::KeyboardModifiers modifiers=Qt::NoModifier) const
- void **wheelEvent** (QWheelEvent \*e) override

## Properties

- int **maximum**
- int **minimum**

## Additional Inherited Members

## Protected Slots inherited from [Digikam::DAbstractSliderSpinBox](#)

- void **contextMenuEvent** (QContextMenuEvent \*event) override
- void **editLostFocus** ()

## Protected Attributes inherited from [Digikam::DAbstractSliderSpinBox](#)

- DAbstractSliderSpinBoxPrivate \*const **d\_ptr**

## 9.457.1 Member Function Documentation

### 9.457.1.1 setInternalValue()

```
void Digikam::DSliderSpinBox::setInternalValue (
    int value,
    bool blockUpdateSignal ) [override], [protected], [virtual]
```

Inheriting classes should respect blockUpdateSignal so that, in specific cases, we have a performance improvement. See setIgnoreMouseMoveEvents.

Implements [Digikam::DAbstractSliderSpinBox](#).

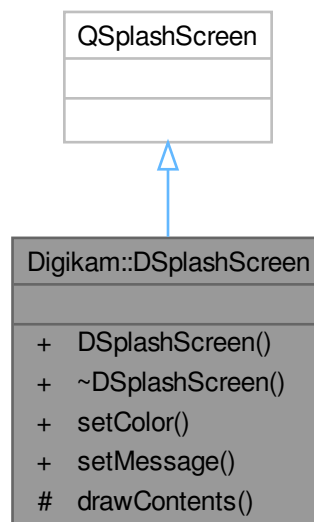
### 9.457.1.2 valueString()

```
QString Digikam::DSliderSpinBox::valueString ( ) const [override], [protected], [virtual]
```

Implements [Digikam::DAbstractSliderSpinBox](#).

## 9.458 Digikam::DSplashScreen Class Reference

Inheritance diagram for Digikam::DSplashScreen:



### Public Member Functions

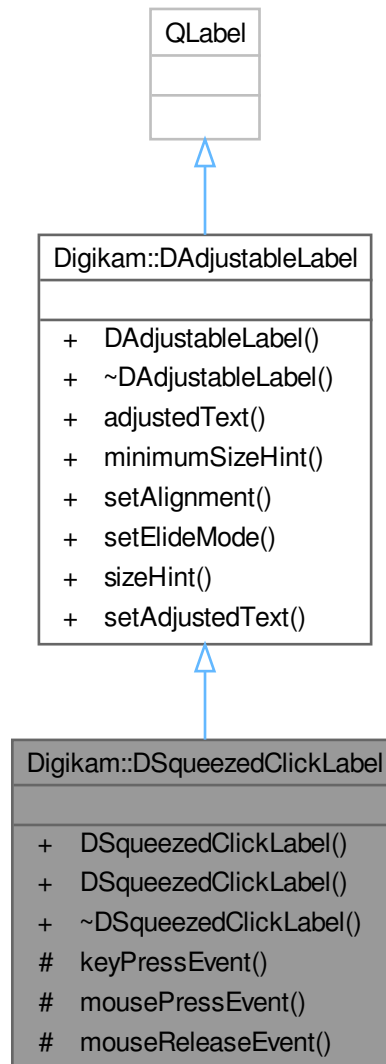
- void **setColor** (const QColor &color)
- void **setMessage** (const QString &message)

### Protected Member Functions

- void **drawContents** (QPainter \*) override

## 9.459 Digikam::DSqueezedClickLabel Class Reference

Inheritance diagram for Digikam::DSqueezedClickLabel:



### Signals

- void **activated** ()
- void **leftClicked** ()

### Public Member Functions

- **DSqueezedClickLabel** (const QString &text, QWidget \*const parent=nullptr)
- **DSqueezedClickLabel** (QWidget \*const parent=nullptr)

## Public Member Functions inherited from [Digikam::DAdjustableLabel](#)

- **DAdjustableLabel** (QWidget \*const parent=nullptr)
- QString **adjustedText** () const
- QSize **minimumSizeHint** () const override
- void **setAlignment** (Qt::Alignment align)
- void **setElideMode** (Qt::TextElideMode mode)
- QSize **sizeHint** () const override

## Protected Member Functions

- void **keyPressEvent** (QKeyEvent \*event) override
- void **mousePressEvent** (QMouseEvent \*event) override
- void **mouseReleaseEvent** (QMouseEvent \*event) override

## Additional Inherited Members

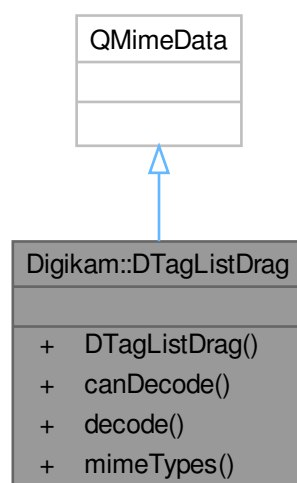
## Public Slots inherited from [Digikam::DAdjustableLabel](#)

- void **setAdjustedText** (const QString &text=QString())

## 9.460 Digikam::DTagListDrag Class Reference

Provides a drag object for a list of tags.

Inheritance diagram for Digikam::DTagListDrag:



### Public Member Functions

- **DTagListDrag** (const QList< int > &tagIDs)

### Static Public Member Functions

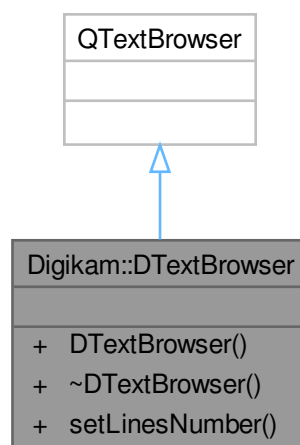
- static bool **canDecode** (const QMimeData \*e)
- static bool **decode** (const QMimeData \*e, QList< int > &tagIDs)
- static QStringList **mimeTypes** ()

## 9.460.1 Detailed Description

When a tag is moved through drag'n'drop an object of this class is created.

## 9.461 Digikam::DTextBrowser Class Reference

Inheritance diagram for Digikam::DTextBrowser:



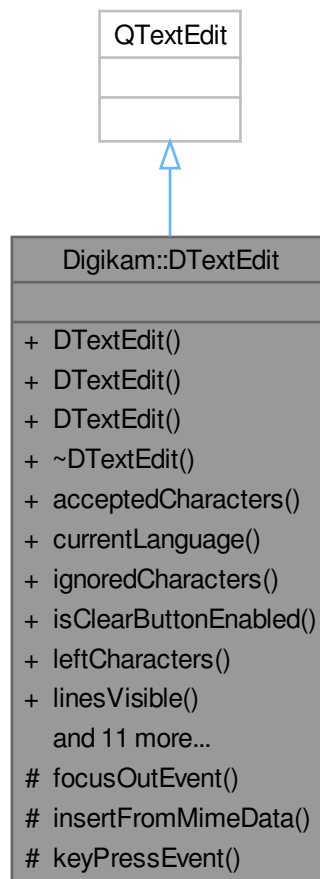
### Public Member Functions

- **DTextBrowser** (const QString &text, QWidget \*const parent=nullptr)
- void **setLinesNumber** (int l)

## 9.462 Digikam::DTextEdit Class Reference

A text edit widget based on QTextEdit with spell checker capabilities based on Sonnet (optional).

Inheritance diagram for Digikam::DTextEdit:



### Signals

- void **editingFinished** ()
- void **returnPressed** ()
  - Emitted only when mimic QLineEdit mode is enabled.*
- void **textEdited** (const QString &)

### Public Member Functions

- **DTextEdit** (const QString &contents, QWidget \*const parent=nullptr)
  - Constructor with text contents to use.*
- **DTextEdit** (QWidget \*const parent=nullptr)



- Default constructor.*
- [DTextEdit](#) (unsigned int lines, QWidget \*const parent=nullptr)
  - Constructor with a number of lines.*
- [~DTextEdit](#) () override
  - Standard destructor.*
- QString [acceptedCharacters](#) () const
  - This property holds whether the edit widget handle the mask of accepted characters in text editor.*
- QString [currentLanguage](#) () const
- QString [ignoredCharacters](#) () const
  - This property holds whether the edit widget handle the mask of ignored characters in text editor.*
- bool [isClearButtonEnabled](#) () const
  - This property holds whether the edit widget displays a clear button when it is not empty.*
- int [leftCharacters](#) () const
  - Return the left characters that user can enter if a limit have been previously set with [setMaxLeght\(\)](#).*
- unsigned int [linesVisible](#) () const
- int [maxLength](#) () const
- void [setAcceptedCharacters](#) (const QString &mask)
- void [setClearButtonEnabled](#) (bool enable)
- void [setCurrentLanguage](#) (const QString &lang)
  - This property holds whether the edit widget handle a specific spell-checker language (2 letters code based as "en", "fr", "es", etc.).*
- void [setIgnoredCharacters](#) (const QString &mask)
- void [setLinesVisible](#) (unsigned int lines)
  - This property holds whether the edit widget handle visible lines used by the widget to show text.*
- void [setLocalizeSettings](#) (const [LocalizeContainer](#) &settings)
- void [setMaxLength](#) (int length)
  - This property holds whether the edit widget handle the maximum of characters that user can enter in editor.*
- void [setText](#) (const QString &text)
- [LocalizeContainer](#) [spellCheckSettings](#) () const
  - This property holds whether the edit widget handle the Spellcheck settings.*
- QString [text](#) () const
  - This property holds whether the edit widget handle text contents as plain text.*

## Protected Member Functions

- void [focusOutEvent](#) (QFocusEvent \*e) override
- void [insertFromMimeData](#) (const QMimeData \*source) override
- void [keyPressEvent](#) (QKeyEvent \*e) override

### 9.462.1 Detailed Description

Widget size can be constrained with the number of visible lines. A single line constraint will emulate [QLineEdit](#). See [setLinesVisible\(\)](#) for details. The maximum number of characters can be limited with [setMaxLenght\(\)](#). The characters can be limited in editor by [setIgnoredCharacters\(\)](#) and [setAcceptedCharacters\(\)](#). Implementation: [dtextedit.cpp](#)

## 9.462.2 Constructor & Destructor Documentation

### 9.462.2.1 DTextEdit()

```
Digikam::DTextEdit::DTextEdit (
    unsigned int lines,
    QWidget *const parent = nullptr ) [explicit]
```

Zero lines do not apply a size constraint.

## 9.462.3 Member Function Documentation

### 9.462.3.1 acceptedCharacters()

```
QString Digikam::DTextEdit::acceptedCharacters ( ) const
```

The mask of characters is passed as string (ex: "abcABC"). By default the mask is empty.

### 9.462.3.2 ignoredCharacters()

```
QString Digikam::DTextEdit::ignoredCharacters ( ) const
```

The mask of characters is passed as string (ex: "+/!()"). By default the mask is empty.

### 9.462.3.3 isClearButtonEnabled()

```
bool Digikam::DTextEdit::isClearButtonEnabled ( ) const
```

If enabled, the edit widget displays a trailing clear button when it contains some text, otherwise the edit widget does not show a clear button. This option only take effect in QLineEdit emulation mode when lines visible is set to 1. See [setLinesVisible\(\)](#) for details.

### 9.462.3.4 returnPressed

```
void Digikam::DTextEdit::returnPressed ( ) [signal]
```

See [setLinesVisible\(\)](#) for details.

### 9.462.3.5 setCurrentLanguage()

```
void Digikam::DTextEdit::setCurrentLanguage (
    const QString & lang )
```

If this property is not set, spell-checker will try to auto-detect language by parsing the text. To reset this setting, pass a empty string as language. If Sonnet dependencies is not resolved, these method do nothing.

### 9.462.3.6 setLinesVisible()

```
void Digikam::DTextEdit::setLinesVisible (
    unsigned int lines )
```

Lines must be superior or equal to 1 to apply a size constraint. Notes: if a single visible line is used, the widget will emulate QLineEdit. a null value do not apply a size constraint.

### 9.462.3.7 setMaxLength()

```
void Digikam::DTextEdit::setMaxLength (
    int length )
```

By default no limit is set. A zero length reset a limit.

### 9.462.3.8 spellCheckSettings()

```
LocalizeContainer Digikam::DTextEdit::spellCheckSettings ( ) const
```

See [LocalizeContainer](#) class for details.

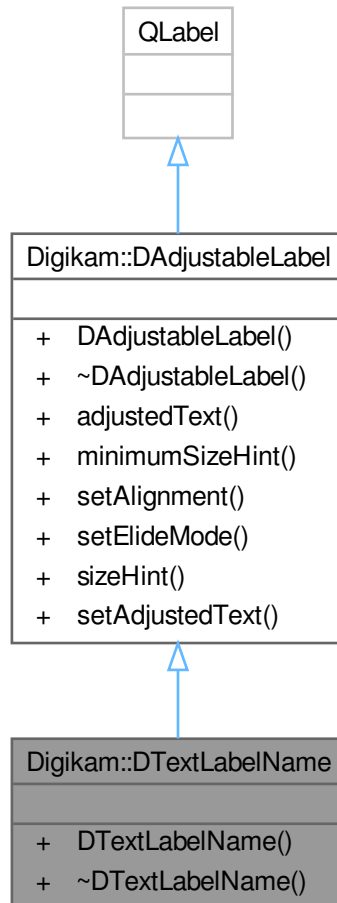
### 9.462.3.9 text()

```
QString Digikam::DTextEdit::text ( ) const
```

If ignored or accepted characters masks are set, text is filtered accordingly.

## 9.463 Digikam::DTextLabelName Class Reference

Inheritance diagram for Digikam::DTextLabelName:



### Public Member Functions

- **DTextLabelName** (const QString &name, QWidget \*const parent=nullptr)

### Public Member Functions inherited from [Digikam::DAdjustableLabel](#)

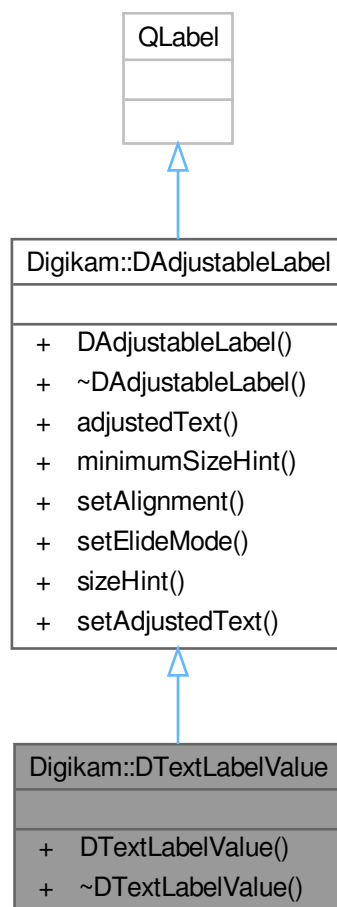
- **DAdjustableLabel** (QWidget \*const parent=nullptr)
- QString **adjustedText** () const
- QSize **minimumSizeHint** () const override
- void **setAlignment** (Qt::Alignment align)
- void **setElideMode** (Qt::TextElideMode mode)
- QSize **sizeHint** () const override

**Additional Inherited Members****Public Slots inherited from [Digikam::DAdjustableLabel](#)**

- void **setAdjustedText** (const QString &text=QString())

**9.464 Digikam::DTextLabelValue Class Reference**

Inheritance diagram for Digikam::DTextLabelValue:

**Public Member Functions**

- **DTextLabelValue** (const QString &value, QWidget \*const parent=nullptr)

### Public Member Functions inherited from [Digikam::DAdjustableLabel](#)

- **DAdjustableLabel** (QWidget \*const parent=nullptr)
- QString **adjustedText** () const
- QSize **minimumSizeHint** () const override
- void **setAlignment** (Qt::Alignment align)
- void **setElideMode** (Qt::TextElideMode mode)
- QSize **sizeHint** () const override

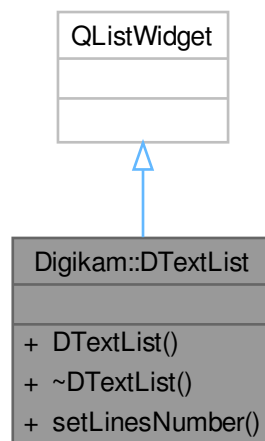
### Additional Inherited Members

### Public Slots inherited from [Digikam::DAdjustableLabel](#)

- void **setAdjustedText** (const QString &text=QString())

## 9.465 Digikam::DTextList Class Reference

Inheritance diagram for Digikam::DTextList:



### Public Member Functions

- **DTextList** (const QStringList &list, QWidget \*const parent=nullptr)
- void **setLinesNumber** (int l)

## 9.466 Digikam::DToolTipStyleSheet Class Reference

### Public Member Functions

- **DToolTipStyleSheet** (const QFont &font=QFontDatabase::systemFont(QFontDatabase::GeneralFont))
- QString **breakString** (const QString &input) const
- QString **elidedText** (const QString &input, Qt::TextElideMode mode) const
- QString **imageAsBase64** (const QImage &img) const

**Public Attributes**

- QString **cellBeg**
- QString **cellEnd**
- QString **cellMid**
- QString **cellSpecBeg**
- QString **cellSpecEnd**
- QString **cellSpecMid**
- QString **headBeg**
- QString **headEnd**
- const int **maxStringLength**
- QString **tipFooter**
- QString **tipHeader**
- QString **unavailable**

**9.467 Digikam::DTrash Class Reference****Static Public Member Functions**

- static bool **deleteDirRecursivley** (const QString &dirToDelete, const QDateTime &deleteTime)  
*Deletes a whole folder from the collection.*
- static bool **deleteImage** (const QString &imagePath, const QDateTime &deleteTime)  
*Deletes image to the trash of a particular collection.*
- static void **extractJsonForItem** (const QString &collPath, const QString &baseName, [DTrashItemInfo](#) &item↔Info)  
*Extracts the data from json file and gives it to [DTrashItemInfo](#).*

**Static Public Attributes**

- static const QString **DELETIONTIMESTAMP\_JSON\_KEY** = QLatin1String("deletiontimestamp")
- static const QString **FILES\_FOLDER** = QLatin1String("files")
- static const QString **IMAGEID\_JSON\_KEY** = QLatin1String("imageid")
- static const QString **INFO\_FILE\_EXTENSION** = QLatin1String(".dtrashinfo")
- static const QString **INFO\_FOLDER** = QLatin1String("info")
- static const QString **PATH\_JSON\_KEY** = QLatin1String("path")
- static const QString **TRASH\_FOLDER** = QLatin1String(".dtrash")

**9.467.1 Member Function Documentation****9.467.1.1 deleteDirRecursivley()**

```
bool Digikam::DTrash::deleteDirRecursivley (
    const QString & dirToDelete,
    const QDateTime & deleteTime ) [static]
```

**Parameters**

<i>dirToDelete</i>	path to folder
<i>deleteTime</i>	delete time from the image

**Returns**

true if folder was deleted

**9.467.1.2 deleteImage()**

```
bool Digikam::DTrash::deleteImage (
    const QString & imagePath,
    const QDateTime & deleteTime ) [static]
```

**Parameters**

<i>imagePath</i>	path to image
<i>deleteTime</i>	delete time from the image

**Returns**

true if the image was deleted

**9.467.1.3 extractJsonForItem()**

```
void Digikam::DTrash::extractJsonForItem (
    const QString & collPath,
    const QString & baseName,
    DTrashItemInfo & itemInfo ) [static]
```

**Parameters**

<i>collPath</i>	path to collection
<i>baseName</i>	name of the file inside the trash
<i>itemInfo</i>	item to extract data to it

**9.468 Digikam::DTrashItemInfo Class Reference****Public Member Functions**

- bool **isNull** () const
- bool **operator==** (const [DTrashItemInfo](#) &itemInfo) const

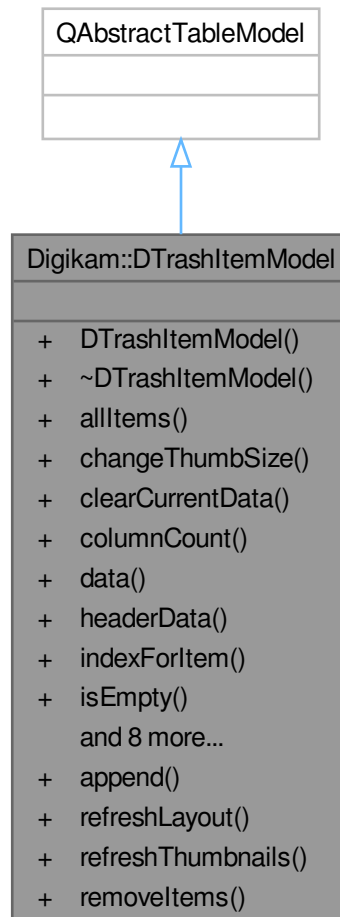
**Public Attributes**

- QString **collectionPath**
- QString **collectionRelativePath**
- QDateTime **deletionTimestamp**
- qlonglong **imageId** = -1
- QString **jsonFilePath**
- QString **trashPath**



## 9.469 Digikam::DTrashItemModel Class Reference

Inheritance diagram for Digikam::DTrashItemModel:



### Public Types

- enum `DTrashColumn` { `DTrashThumb = 0` , `DTrashRelPath` , `DTrashTimeStamp` , `DTrashNumCol` }

### Public Slots

- void `append` (const `DTrashItemInfo` &itemInfo)  
*appends item to model data and informs the view*
- void `refreshLayout` ()  
*refreshes the view layout*
- void `refreshThumbnails` (const `LoadingDescription` &desc, const `QPixmap` &pix)  
*refreshes the thumbnails*
- void `removeItems` (const `QModelIndexList` &indexes)  
*removes list of items for given indexes from model data and informs the view*

## Signals

- void **dataChange** ()
- void **signalLoadingFinished** ()
- void **signalLoadingStarted** ()

## Public Member Functions

- **DTrashItemModel** (QObject \*const parent, QWidget \*const widget)
- DTrashItemInfoList **allItems** ()  
*returns a list of all items in model*
- void **changeThumbSize** (int size)  
*Changes the thumbnail size.*
- void **clearCurrentData** ()  
*Clears all data from model and informs the view.*
- int **columnCount** (const QModelIndex &) const override
- QVariant **data** (const QModelIndex &index, int role) const override
- QVariant **headerData** (int section, Qt::Orientation orientation, int role) const override
- QModelIndex **indexForItem** (const DTrashItemInfo &itemInfo) const  
*returns the index for the DTrashItemInfo in model*
- bool **isEmpty** ()
- DTrashItemInfo **itemForIndex** (const QModelIndex &index)  
*returns DTrashItemInfo for specific index in model*
- DTrashItemInfoList **itemsForIndexes** (const QList< QModelIndex > &indexes)  
*returns DTrashItemInfoList for given indexes in model*
- void **loadItemsForCollection** (const QString &colPath)  
*Runs a thread to list all items from a collection trash.*
- bool  **pixmapForItem** (const QString &path, QPixmap &pix) const  
*loads a thumbnail for item in trash for showing*
- int **rowCount** (const QModelIndex &) const override  
*QAbstractItemModel interface.*
- void **sort** (int column, Qt::SortOrder order=Qt::AscendingOrder) override
- void **stopLoadingTrash** ()  
*Stop loading of trash.*
- QString **trashAlbumPath** () const

## 9.469.1 Member Function Documentation

### 9.469.1.1 append

```
void Digikam::DTrashItemModel::append (
    const DTrashItemInfo & itemInfo ) [slot]
```

#### Parameters

<i>itemInfo</i>	item to append
-----------------	----------------

**9.469.1.2 changeThumbSize()**

```
void Digikam::DTrashItemModel::changeThumbSize (
    int size )
```

**Parameters**

<i>size</i>	size to change to
-------------	-------------------

**9.469.1.3 isEmpty()**

```
bool Digikam::DTrashItemModel::isEmpty ( )
```

**Returns**

true if there is no data in the model

**9.469.1.4 loadItemsForCollection()**

```
void Digikam::DTrashItemModel::loadItemsForCollection (
    const QString & colPath )
```

**Parameters**

<i>colPath</i>	path to collection to load items for
----------------	--------------------------------------

**9.469.1.5 pixmapForItem()**

```
bool Digikam::DTrashItemModel::pixmapForItem (
    const QString & path,
    QPixmap & pix ) const
```

**Parameters**

<i>path</i>	path of image in trash
<i>pix</i>	Pixmap to fill

**Returns**

true if there is an available thumbnail

**9.469.1.6 refreshThumbnails**

```
void Digikam::DTrashItemModel::refreshThumbnails (
    const LoadingDescription & desc,
    const QPixmap & pix ) [slot]
```

**Parameters**

<i>desc</i>	loading description from thumbnail load thread
<i>pix</i>	pixmap from thumbnail load thread

**9.469.1.7 removeItems**

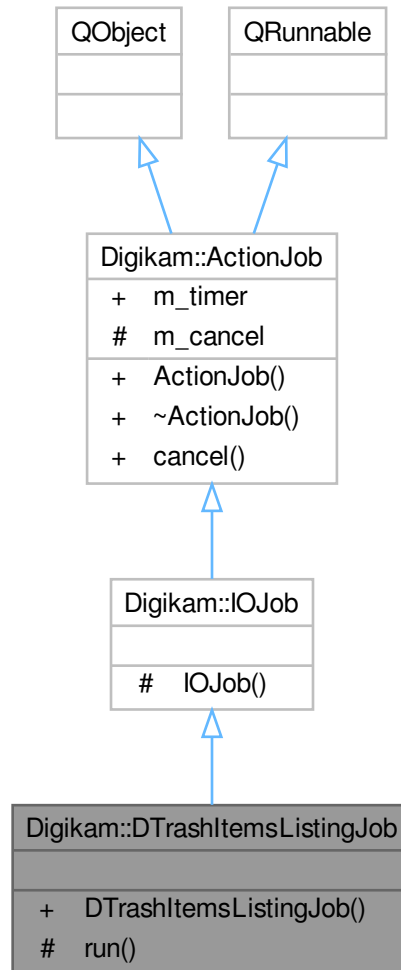
```
void Digikam::DTrashItemModel::removeItems (  
    const QModelIndexList & indexes ) [slot]
```

**Parameters**

<i>indexes</i>	indexes to remove
----------------	-------------------

## 9.470 Digikam::DTrashItemsListingJob Class Reference

Inheritance diagram for Digikam::DTrashItemsListingJob:



### Signals

- void `trashItemInfo` (const [DTrashItemInfo](#) &info)

### Signals inherited from [Digikam::IOJob](#)

- void `signalError` (const `QString` &errMsg)
- void `signalOneProcessed` (const `QUrl` &url)

## Signals inherited from [Digikam::ActionJob](#)

- void **signalDone** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job is done.*
- void **signalProgress** (int)  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job progress.*
- void **signalStarted** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager that job is started.*

## Public Member Functions

- **DTrashItemsListingJob** (const QString &collectionPath)

## Public Member Functions inherited from [Digikam::ActionJob](#)

- **ActionJob** (QObject \*const parent=nullptr)  
*Constructor which delegate deletion of QRunnable instance to [ActionThreadBase](#), not QThreadPool.*
- **~ActionJob** () override  
*Re-implement destructor in you implementation.*

## Protected Member Functions

- void **run** () override

## Additional Inherited Members

## Public Slots inherited from [Digikam::ActionJob](#)

- void **cancel** ()  
*Call this method to cancel job.*

## Public Attributes inherited from [Digikam::ActionJob](#)

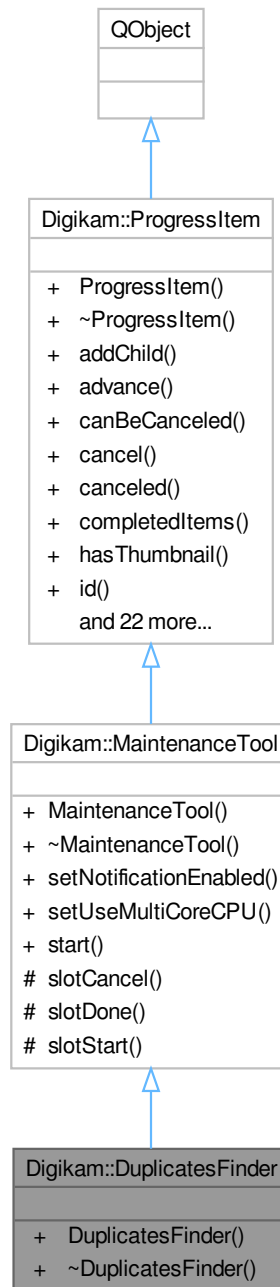
- QElapsedTimer **m\_timer**  
*Timer to determine the running time of the job.*

## Protected Attributes inherited from [Digikam::ActionJob](#)

- bool **m\_cancel** = false  
*You can use this boolean in your implementation to know if job must be canceled.*

## 9.471 Digikam::DuplicatesFinder Class Reference

Inheritance diagram for Digikam::DuplicatesFinder:



### Signals

- void **signalScanNotification** (const QString &msg, int type)

## Signals inherited from [Digikam::MaintenanceTool](#)

- void **signalCanceled** ()  
*Emit when process is canceled.*
- void **signalComplete** ()  
*Emit when process is done (not canceled).*

## Signals inherited from [Digikam::ProgressItem](#)

- void [progressItemAdded](#) ([ProgressItem](#) \*item)  
*Emitted when a new [ProgressItem](#) is added.*
- void [progressItemCanceled](#) ([ProgressItem](#) \*item)  
*Emitted when an item was canceled.*
- void **progressItemCanceledById** (const QString &id)
- void [progressItemCompleted](#) ([ProgressItem](#) \*item)  
*Emitted when a progress item was completed.*
- void [progressItemLabel](#) ([ProgressItem](#) \*item, const QString &label)  
*Emitted when the label of an item changed.*
- void [progressItemProgress](#) ([ProgressItem](#) \*item, unsigned int v)  
*Emitted when the progress value of an item changes.*
- void [progressItemStatus](#) ([ProgressItem](#) \*item, const QString &mess)  
*Emitted when the status message of an item changed.*
- void [progressItemThumbnail](#) ([ProgressItem](#) \*item, const QPixmap &thumb)  
*Emitted when the thumbnail data must be set in item.*
- void [progressItemUsesBusyIndicator](#) ([ProgressItem](#) \*item, bool value)  
*Emitted when the busy indicator state of an item changes.*

## Public Member Functions

- **DuplicatesFinder** (const AlbumList &albums, const AlbumList &tags, int albumTagRelation=0, int min← Similarity=90, int maxSimilarity=100, int searchResultRestriction=0, [Haarface::RefImageSelMethod](#) method=[Haarface::RefImageSelMethod::OlderOrLarger](#), const AlbumList &referenceImageAlbum={}, [ProgressItem](#) \*const parent=nullptr)  
*Version to find all duplicates over a specific list to PAlbums and TAlbums.*

## Public Member Functions inherited from [Digikam::MaintenanceTool](#)

- **MaintenanceTool** (const QString &id, [ProgressItem](#) \*const parent=nullptr)
- void **setNotificationEnabled** (bool b)  
*If true, show a notification message on desktop notification manager with time elapsed to run process.*
- virtual void [setUseMultiCoreCPU](#) (bool)  
*Re-implement this method if your tool is able to use multi-core CPU to process item in parallel.*



## Public Member Functions inherited from Digikam::ProgressItem

- **ProgressItem** ([ProgressItem](#) \*const [parent](#), const QString &[id](#), const QString &[label](#), const QString &[status](#), bool [canBeCanceled](#), bool [hasThumb](#))
- void **addChild** ([ProgressItem](#) \*const [kiddo](#))
- bool [advance](#) (unsigned int [v](#))
  - Advance total items processed by n values and update percentage in progressbar.*
- bool [canBeCanceled](#) () const
- void **cancel** ()
- bool **canceled** () const
- unsigned int **completedItems** () const
- bool [hasThumbnail](#) () const
- const QString & [id](#) () const
- bool **incCompletedItems** (unsigned int [v](#)=1)
- void **incTotalItems** (unsigned int [v](#)=1)
- const QString & [label](#) () const
- [ProgressItem](#) \* [parent](#) () const
- unsigned int [progress](#) () const
- void **removeChild** ([ProgressItem](#) \*const [kiddo](#))
- void **reset** ()
  - Reset the progress value of this item to 0 and the status string to the empty string.*
- void [setComplete](#) ()
  - Tell the item it has finished.*
- bool **setCompletedItems** (unsigned int [v](#))
- void [setLabel](#) (const QString &[v](#))
- void [setProgress](#) (unsigned int [v](#))
  - Set the progress (percentage of completion) value of this item.*
- void [setShowAtStart](#) (bool [showAtStart](#))
  - Set the property to pop-up item when it's added in progress manager.*
- void [setStatus](#) (const QString &[v](#))
  - Set the string to be used for showing this item's current status.*
- void [setThumbnail](#) (const QIcon &[icon](#))
  - Sets whether this item has a thumbnail.*
- void **setTotalItems** (unsigned int [v](#))
- void [setUsesBusyIndicator](#) (bool [useBusyIndicator](#))
  - Sets whether this item uses a busy indicator instead of real progress for its progress bar.*
- bool [showAtStart](#) () const
- const QString & [status](#) () const
- bool **totalCompleted** () const
- unsigned int **totalItems** () const
- void **updateProgress** ()
  - Recalculate progress according to total/completed items and update.*
- bool [usesBusyIndicator](#) () const

## Additional Inherited Members

## Public Slots inherited from Digikam::MaintenanceTool

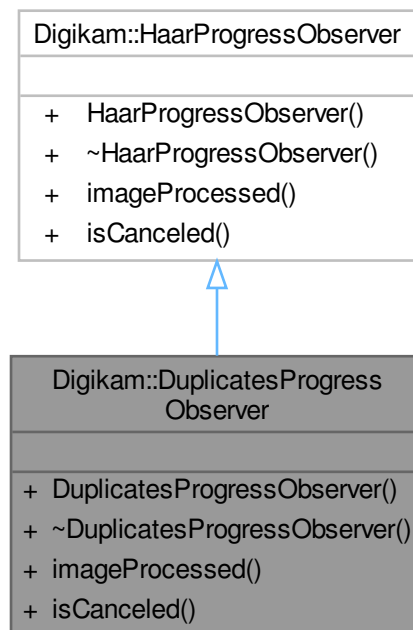
- void **start** ()

## Protected Slots inherited from [Digikam::MaintenanceTool](#)

- virtual void `slotCancel` ()
- virtual void `slotDone` ()
- virtual void `slotStart` ()

## 9.472 [Digikam::DuplicatesProgressObserver](#) Class Reference

Inheritance diagram for [Digikam::DuplicatesProgressObserver](#):



### Public Member Functions

- `DuplicatesProgressObserver` ([SearchesJob](#) \*const thread)
- void `imageProcessed` (const [ItemInfo](#) &inf, const QImage &img, int dup) override
- bool `isCanceled` () override

### 9.472.1 Member Function Documentation

#### 9.472.1.1 `imageProcessed()`

```
void Digikam::DuplicatesProgressObserver::imageProcessed (
    const ItemInfo & inf,
    const QImage & img,
    int dup ) [override], [virtual]
```

Implements [Digikam::HaarProgressObserver](#).

## 9.472.1.2 isCanceled()

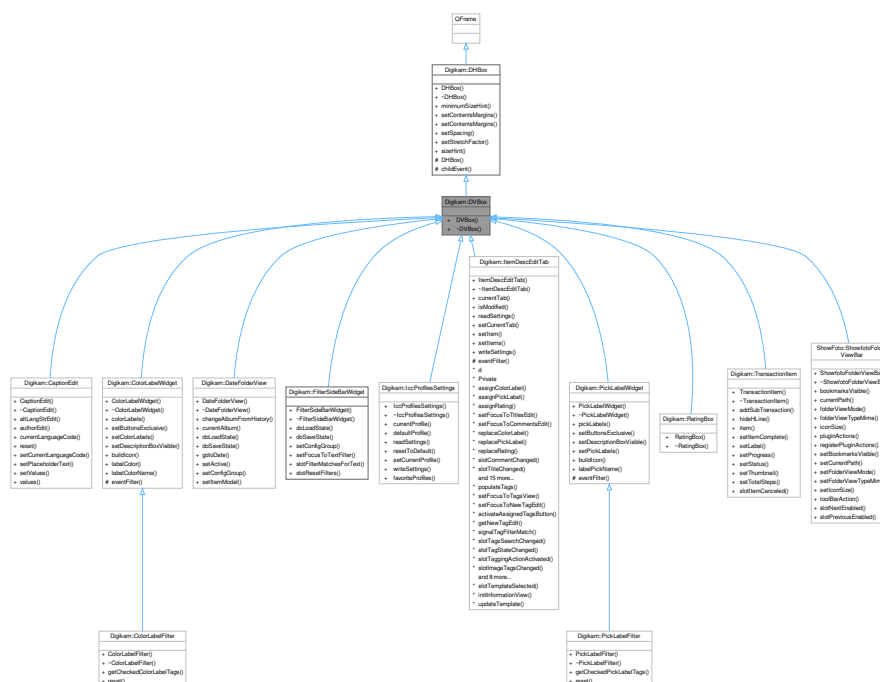
```
bool Digikam::DuplicatesProgressObserver::isCanceled ( ) [override], [virtual]
```

Reimplemented from [Digikam::HaarProgressObserver](#).

## 9.473 Digikam::DVBox Class Reference

A Vertical widget to host children widgets.

Inheritance diagram for Digikam::DVBox:



## Public Member Functions

- **DVBox** (QWidget \*const parent=nullptr)

Public Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (QWidget \*const parent=nullptr)
- QSize **minimumSizeHint** () const override
- void **setContentsMargins** (const QMargins &margins)
- void **setContentsMargins** (int left, int top, int right, int bottom)
- void **setSpacing** (int space)
- void **setStretchFactor** (QWidget \*const widget, int stretch)
- QSize **sizeHint** () const override

### Additional Inherited Members

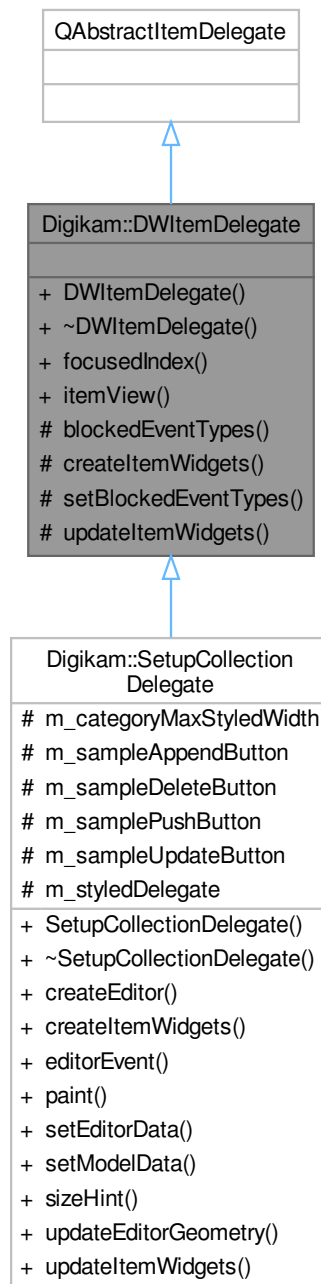
### Protected Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (bool vertical, QWidget \*const parent)
- void **childEvent** (QChildEvent \*e) override

## 9.474 Digikam::DWItemDelegate Class Reference

This class allows to create item delegates embedding simple widgets to interact with items.

Inheritance diagram for Digikam::DWItemDelegate:



### Public Member Functions

- `DWItemDelegate` (`QAbstractItemView *const itemView`, `QObject *const parent=nullptr`)  
*Creates a new `ItemDelegate` to be used with a given `itemview`.*
- `QPersistentModelIndex focusedIndex` () const  
*Retrieves the currently focused index.*
- `QAbstractItemView * itemView` () const  
*Retrieves the item view this delegate is monitoring.*

## Protected Member Functions

- `QList< QEvent::Type > blockedEventTypes` (`QWidget *const widget`) `const`  
*Retrieves the list of blocked event types for the given widget.*
- virtual `QList< QWidget * > createItemWidgets` (`const QModelIndex &index`) `const =0`  
*Creates the list of widgets needed for an item.*
- void `setBlockedEventTypes` (`QWidget *const widget`, `const QList< QEvent::Type > &types`) `const`  
*Sets the list of event types that a widget will block.*
- virtual void `updateItemWidgets` (`const QList< QWidget * > &widgets`, `const QStyleOptionViewItem &option`, `const QModelIndex &index`) `const =0`  
*Updates a list of widgets for its use inside of the delegate (painting or event handling).*

## Friends

- class `DWItemDelegateEventListener`
- class `DWItemDelegatePool`

## 9.474.1 Detailed Description

For instance you can add push buttons, line edits, etc. to your delegate and use them to modify the state of your model.

## 9.474.2 Constructor & Destructor Documentation

### 9.474.2.1 DWItemDelegate()

```
Digikam::DWItemDelegate::DWItemDelegate (
    QAbstractItemView *const itemView,
    QObject *const parent = nullptr ) [explicit]
```

#### Parameters

<i>itemView</i>	the item view the new delegate will monitor
<i>parent</i>	the parent of this delegate

## 9.474.3 Member Function Documentation

### 9.474.3.1 blockedEventTypes()

```
QList< QEvent::Type > Digikam::DWItemDelegate::blockedEventTypes (
    QWidget *const widget ) const [protected]
```

#### Parameters

<i>widget</i>	the specified widget.
---------------	-----------------------

**Returns**

the list of blocked event types, can be empty if no events are blocked.

**9.474.3.2 createItemWidgets()**

```
virtual QList< QWidget * > Digikam::DWItemDelegate::createItemWidgets (
    const QModelIndex & index ) const [protected], [pure virtual]
```

**Note**

No initialization of the widgets is supposed to happen here. The widgets will be initialized based on needs for a given item.

If you want to connect some widget signals to any slot, you should do it here.

- index the index to create widgets for.

**Note**

If you want to know the index for which you are creating widgets, it is available as a QModelIndex [Q\\_↔](#) PROPERTY called "goya:creatingWidgetsForIndex". Ensure to add Q\_DECLARE\_METATYPE(QModelIndex) before your method definition to tell QVariant about QModelIndex.

**Returns**

the list of newly created widgets which will be used to interact with an item.

**See also**

[updateItemWidgets\(\)](#)

Implemented in [Digikam::SetupCollectionDelegate](#).

**9.474.3.3 focusedIndex()**

```
QPersistentModelIndex Digikam::DWItemDelegate::focusedIndex ( ) const
```

An invalid index if none is focused.

**Returns**

the current focused index, or QPersistentModelIndex() if none is focused.

**9.474.3.4 itemView()**

```
QAbstractItemView * Digikam::DWItemDelegate::itemView ( ) const
```

**Returns**

the item view this delegate is monitoring

**9.474.3.5 setBlockedEventTypes()**

```
void Digikam::DWItemDelegate::setBlockedEventTypes (
    QWidget *const widget,
    const QList< QEvent::Type > & types ) const [protected]
```

Blocked events are not passed to the view. This way you can prevent an item from being selected when a button is clicked for instance.

## Parameters

<i>widget</i>	the widget which must block events
<i>types</i>	the list of event types the widget must block

**9.474.3.6 updateItemWidgets()**

```
virtual void Digikam::DWItemDelegate::updateItemWidgets (
    const QList< QWidget * > & widgets,
    const QStyleOptionViewItem & option,
    const QPersistentModelIndex & index ) const [protected], [pure virtual]
```

## Note

All the positioning and sizing should be done in item coordinates.

## Warning

Do not make widget connections in here, since this method will be called very regularly.

## Parameters

<i>widgets</i>	the widgets to update
<i>option</i>	the current set of style options for the view.
<i>index</i>	the model index of the item currently manipulated.

Implemented in [Digikam::SetupCollectionDelegate](#).

**9.475 Digikam::DWItemDelegatePool Class Reference**

## Public Types

- enum **UpdateWidgetsEnum** { **UpdateWidgets** = 0 , **NotUpdateWidgets** }

## Public Member Functions

- [DWItemDelegatePool](#) ([DWItemDelegate](#) \*const delegate)  
*Creates a new ItemDelegatePool.*
- [QList< QWidget \\* >](#) [findWidgets](#) (const [QPersistentModelIndex](#) &index, const [QStyleOptionViewItem](#) &option, [UpdateWidgetsEnum](#) updateWidgets=[UpdateWidgets](#)) const  
*Returns the widget associated to index and widget.*
- void **fullClear** ()
- [QList< QWidget \\* >](#) **invalidIndexesWidgets** () const



## Friends

- class **DWItemDelegate**
- class **DWItemDelegatePrivate**

## 9.475.1 Constructor & Destructor Documentation

### 9.475.1.1 DWItemDelegatePool()

```
Digikam::DWItemDelegatePool::DWItemDelegatePool (
    DWItemDelegate *const delegate ) [explicit]
```

#### Parameters

<i>delegate</i>	the <a href="#">ItemDelegate</a> for this pool.
-----------------	---

## 9.475.2 Member Function Documentation

### 9.475.2.1 findWidgets()

```
QList< QWidget * > Digikam::DWItemDelegatePool::findWidgets (
    const QPersistentModelIndex & index,
    const QStyleOptionViewItem & option,
    UpdateWidgetsEnum updateWidgets = UpdateWidgets ) const
```

#### Parameters

<i>index</i>	The index to search into.
<i>option</i>	a QStyleOptionViewItem.
<i>updateWidgets</i>	a flag to force to update widgets.

#### Returns

A QList of the pointers to the widgets found.

## 9.476 Digikam::DWItemDelegatePoolPrivate Class Reference

### Public Member Functions

- **DWItemDelegatePoolPrivate** ([DWItemDelegate](#) \*const dd)

### Public Attributes

- bool **clearing** = false
- [DWItemDelegate](#) \* **delegate** = nullptr
- DWItemDelegateEventListener \* **eventListener** = nullptr
- QHash< QPersistentModelIndex, QList< QWidget \* > > **usedWidgets**
- QHash< QWidget \*, QPersistentModelIndex > **widgetInIndex**

## 9.477 Digikam::DWizardDlg Class Reference

Inheritance diagram for Digikam::DWizardDlg:



### Public Member Functions

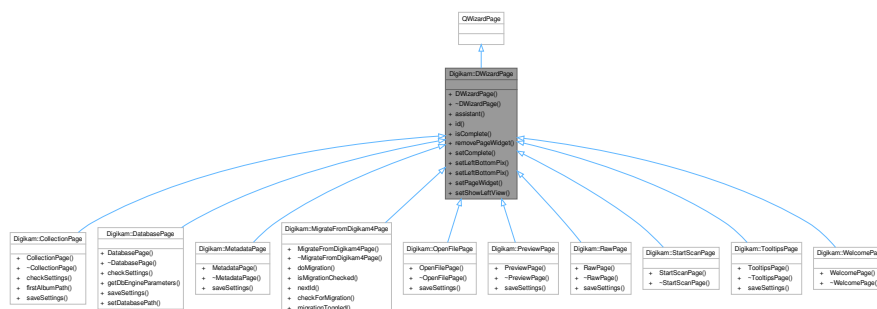
- **DWizardDlg** (QWidget \*const parent, const QString &objName)
- void **setPlugin** (DPlugin \*const tool)

### Protected Member Functions

- void **restoreDialogSize** ()
- void **saveDialogSize** ()
- void **showEvent** (QShowEvent \*) override

## 9.478 Digikam::DWizardPage Class Reference

Inheritance diagram for Digikam::DWizardPage:



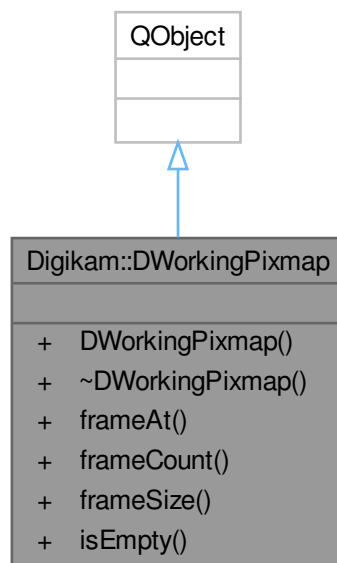
**Public Member Functions**

- **DWizardPage** (QWizard \*const dlg, const QString &title)
- QWizard \* **assistant** () const
- int **id** () const
- bool **isComplete** () const override
- void **removePageWidget** (QWidget \*const w)
- void **setComplete** (bool b)
- void **setLeftBottomPix** (const QIcon &icon)
- void **setLeftBottomPix** (const QPixmap &pix)
- void **setPageWidget** (QWidget \*const w)
- void **setShowLeftView** (bool v)

**9.479 Digikam::DWorkingPixmap Class Reference**

A widget to draw progress wheel indicator over thumbnails.

Inheritance diagram for Digikam::DWorkingPixmap:

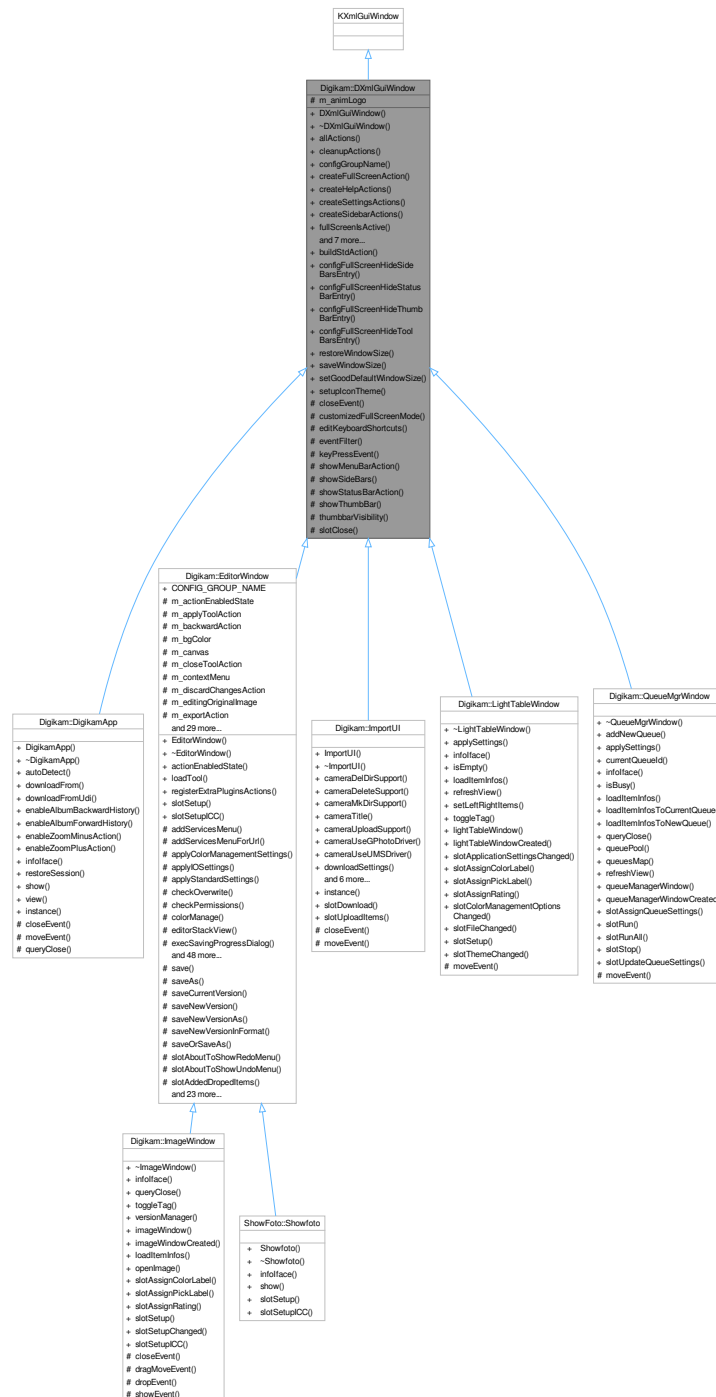
**Public Member Functions**

- **DWorkingPixmap** (QObject \*const parent=nullptr)
- QPixmap **frameAt** (int index) const
- int **frameCount** () const
- QSize **frameSize** () const
- bool **isEmpty** () const

## 9.480 Digikam::DXmlGuiWindow Class Reference

Generic class to use with all main window.

Inheritance diagram for Digikam::DXmlGuiWindow:



### Public Member Functions

- **DXmlGuiWindow** (QWidget \*const parent=nullptr, Qt::WindowFlags f=Qt::WindowFlags())

- `QList< QAction * > allActions () const`  
*Return all actions from internal collection.*
- `void cleanupActions ()`  
*Cleanup unwanted actions from action collection.*
- `QString configGroupName () const`
- `void createFullScreenAction (const QString &name)`  
*Create Full-screen action to action collection instance from managed window set through `setManagedWindow()`.*
- `void createHelpActions (const QString &handbookSection, bool coreOptions=true)`  
*Create common actions from Help menu for all digiKam main windows.*
- `void createSettingsActions ()`  
*Create common actions to setup all digiKam main windows.*
- `void createSidebarActions ()`  
*Create common actions to handle side-bar through keyboard shortcuts.*
- `bool fullScreensActive () const`  
*Return true if managed window is currently in Full Screen Mode.*
- `virtual DInfoInterface * interface (DPluginAction *const ac)=0`  
*Return the interface instance to access to items information.*
- `void readFullScreenSettings (const KConfigGroup &group)`  
*Read full-screen settings from KDE config file.*
- `virtual void registerExtraPluginsActions (QString &)`
- `void registerPluginsActions ()`  
*Register all generic plugins action to this instance.*
- `void setConfigGroupName (const QString &name)`  
*Manage config group name used by window instance to get/set settings from config file.*
- `void setFullScreenOptions (int options)`  
*Set full-screen options to managed window.*
- `void unminimizeAndActivateWindow ()`

### Static Public Member Functions

- `static QAction * buildStdAction (StdActionType type, const QObject *const recvr, const char *const slot, QObject *const parent)`
- `static QString configFullScreenHideSideBarsEntry ()`
- `static QString configFullScreenHideStatusBarEntry ()`
- `static QString configFullScreenHideThumbBarEntry ()`
- `static QString configFullScreenHideToolBarsEntry ()`  
*Shared with [FullScreenSettings](#).*
- `static void restoreWindowSize (QWindow *const win, const KConfigGroup &group)`
- `static void saveWindowSize (QWindow *const win, KConfigGroup &group)`
- `static void setGoodDefaultWindowSize (QWindow *const win)`
- `static void setupIconTheme ()`  
*If we have some local breeze icon resource, prefer it.*

### Protected Slots

- `bool slotClose ()`

## Protected Member Functions

- void **closeEvent** (QCloseEvent \*e) override
- virtual void **customizedFullScreenMode** (bool set)
 

*Re-implement this method if you want to manage customized view visibility in full-screen mode.*
- void **editKeyboardShortcuts** (KActionCollection \*const extraac=nullptr, const QString &actitle=QString())
 

*Call this method from your main window to show keyboard shortcut config dialog with an extra action collection to configure.*
- bool **eventFilter** (QObject \*obj, QEvent \*ev) override
- void **keyPressEvent** (QKeyEvent \*e) override
- QAction \* **showMenuBarAction** () const
- virtual void **showSideBars** (bool visible)
 

*Re-implement this method if you want to manage sidebars visibility in full-screen mode.*
- QAction \* **showStatusBarAction** () const
- virtual void **showThumbBar** (bool visible)
 

*Re-implement this method if you want to manage thumbbar visibility in full-screen mode.*
- virtual bool **thumbbarVisibility** () const
 

*Re-implement this method if managed window has a thumbbar.*

## Protected Attributes

- **DLogoAction** \* **m\_animLogo** = nullptr

## 9.480.1 Member Function Documentation

### 9.480.1.1 createFullScreenAction()

```
void Digikam::DXmlGuiWindow::createFullScreenAction (
    const QString & name )
```

This action is connected to slotToggleFullScreen() slot. 'name' is action name used in KDE UI rc file.

### 9.480.1.2 customizedFullScreenMode()

```
void Digikam::DXmlGuiWindow::customizedFullScreenMode (
    bool set ) [protected], [virtual]
```

This method is called by switchWindowToFullScreen(). By default this method do nothing.

### 9.480.1.3 editKeyboardShortcuts()

```
void Digikam::DXmlGuiWindow::editKeyboardShortcuts (
    KActionCollection *const extraac = nullptr,
    const QString & actitle = QString() ) [protected]
```

This method is called by slotEditKeys() which can be re-implement in child class for customization.

#### 9.480.1.4 infoface()

```
virtual DInfoInterface * Digikam::DXmlGuiWindow::infoIface (
    DPluginAction *const ac ) [pure virtual]
```

Implemented in [Digikam::DigikamApp](#), [ShowFoto::Showfoto](#), [Digikam::ImageWindow](#), [Digikam::LightTableWindow](#), [Digikam::ImportUI](#), and [Digikam::QueueMgrWindow](#).

#### 9.480.1.5 registerPluginsActions()

```
void Digikam::DXmlGuiWindow::registerPluginsActions ( )
```

Call `registerExtraPluginsActions()` to plug other kind of plugins in GUI.

#### 9.480.1.6 showSideBars()

```
void Digikam::DXmlGuiWindow::showSideBars (
    bool visible ) [protected], [virtual]
```

By default this method do nothing.

#### 9.480.1.7 showThumbBar()

```
void Digikam::DXmlGuiWindow::showThumbBar (
    bool visible ) [protected], [virtual]
```

By default this method do nothing.

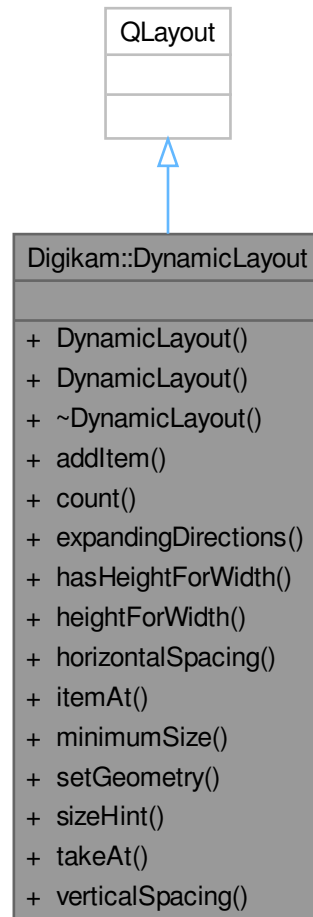
#### 9.480.1.8 thumbbarVisibility()

```
bool Digikam::DXmlGuiWindow::thumbbarVisibility ( ) const [protected], [virtual]
```

This must return visibility state of it.

## 9.481 Digikam::DynamicLayout Class Reference

Inheritance diagram for Digikam::DynamicLayout:



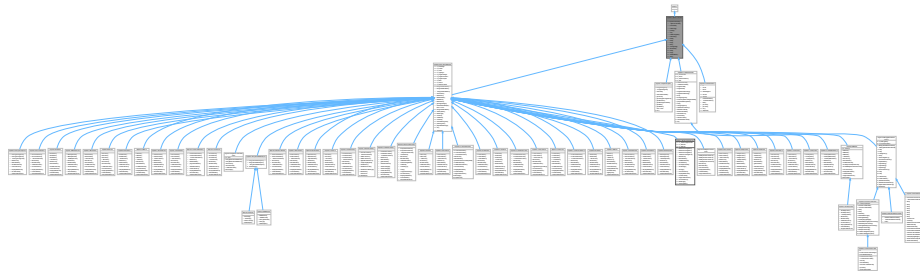
### Public Member Functions

- `DynamicLayout` (int margin=-1, int hSpacing=0, int vSpacing=0)
- `DynamicLayout` (QWidget \*const parent, int margin=-1, int hSpacing=0, int vSpacing=0)
- void `addItem` (QLayoutItem \*item) override
- int `count` () const override
- Qt::Orientations `expandingDirections` () const override
- bool `hasHeightForWidth` () const override
- int `heightForWidth` (int index) const override
- int `horizontalSpacing` () const
- QLayoutItem \* `itemAt` (int index) const override
- QSize `minimumSize` () const override
- void `setGeometry` (const QRect &rect) override
- QSize `sizeHint` () const override
- QLayoutItem \* `takeAt` (int index) override
- int `verticalSpacing` () const



## 9.482 Digikam::DynamicThread Class Reference

Inheritance diagram for Digikam::DynamicThread:



### Public Types

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

### Public Member Functions

- **DynamicThread** (QObject \*const parent=nullptr)  
*This class extends QRunnable, so you have to reimplement virtual void [run\(\)](#).*
- **~DynamicThread** () override  
*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool **isFinished** () const
- bool **isRunning** () const
- QThread::Priority **priority** () const
- virtual void **run** ()=0  
*Implement this pure virtual function in your subclass.*
- void **setEmitSignals** (bool emitThem)
- void **setPriority** (QThread::Priority priority)  
*Sets the priority for this dynamic thread.*
- State **state** () const

## Protected Member Functions

- bool **runningFlag** () const volatile  
*In you [run\(\)](#) method, you shall regularly check for [runningFlag\(\)](#) and cleanup and return if false.*
- void **shutDown** ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call [stop\(\)](#) and [wait\(\)](#), knowing that nothing will call [start\(\)](#) anymore after this 3) Be sure the thread will never be running at destruction.*
- void **start** (QMutexLocker< QMutex > &locker)  
*Doing the same as [start\(\)](#), [stop\(\)](#) and [wait](#) above, provide it with a locked QMutexLocker on mutex().*
- void **stop** (const QMutexLocker< QMutex > &locker)
- QMutex \* **threadMutex** () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void **wait** (QMutexLocker< QMutex > &locker)

## Friends

- class **DynamicThreadPriv**

## 9.482.1 Constructor & Destructor Documentation

### 9.482.1.1 DynamicThread()

```
Digikam::DynamicThread::DynamicThread (
    QObject *const parent = nullptr ) [explicit]
```

In all aspects the class will act similar to a QThread.

## 9.482.2 Member Function Documentation

### 9.482.2.1 run()

```
virtual void Digikam::DynamicThread::run ( ) [pure virtual]
```

Implemented in [Digikam::DImgThreadedFilter](#), [Digikam::ImageHistogram](#), [Digikam::LoadSaveThread](#), and [Digikam::ScanStateFilter](#).

### 9.482.2.2 setPriority()

```
void Digikam::DynamicThread::setPriority (
    QThread::Priority priority )
```

Can be set anytime. If the thread is currently not running, the priority will be set when it is run next time. When you set QThread::InheritPriority (default), the priority is not changed but inherited from the thread pool.

### 9.482.2.3 shutDown()

```
void Digikam::DynamicThread::shutDown ( ) [protected]
```

Note: This irrevocably stops this object. Note: It is not sufficient that your parent class does this. Calling this method, or providing one of the above mentioned equivalent guarantees, must be done by every single last class in the hierarchy with an implemented destructor deleting data. (the base class destructor is always called after the derived class)

### 9.482.2.4 start()

```
void Digikam::DynamicThread::start (
    QMutexLocker< QMutex > & locker ) [protected]
```

Note the start() will unlock and relock for scheduling once, after state change.

### 9.482.2.5 threadMutex()

```
QMutex * Digikam::DynamicThread::threadMutex ( ) const [protected]
```

You can use it if you want to protect your memory in the same scope as calling start, stop or wait, then using the QMutexLocker variants below. Note that when you have locked this mutex, you must use these variants, as the mutex is non-recursive.

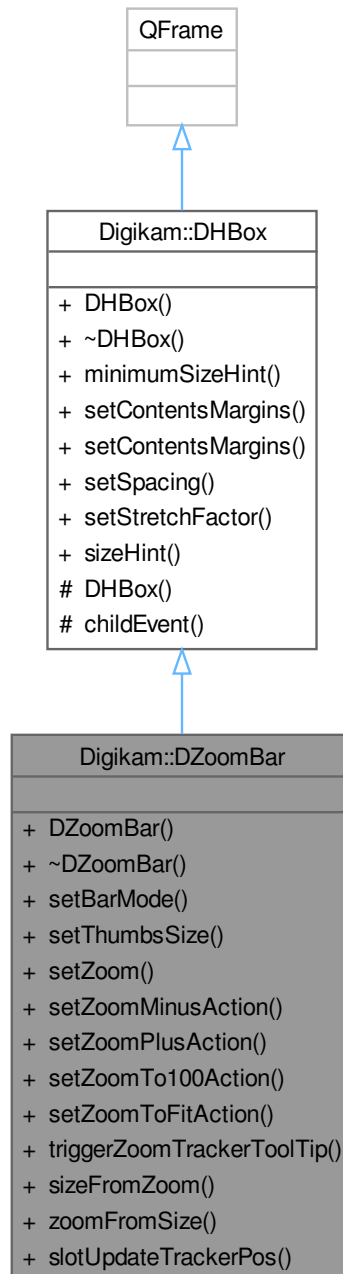
### 9.482.2.6 wait

```
void Digikam::DynamicThread::wait ( ) [slot]
```

Typically, call [stop\(\)](#) before.

## 9.483 Digikam::DZoomBar Class Reference

Inheritance diagram for Digikam::DZoomBar:



### Public Types

- enum `BarMode` { `PreviewZoomCtrl` = 0 , `ThumbsSizeCtrl` , `NoPreviewZoomCtrl` }

## Public Slots

- void **slotUpdateTrackerPos** ()

## Signals

- void **signalDelayedZoomSliderChanged** (int)
- void **signalZoomSliderChanged** (int)
- void **signalZoomSliderReleased** (int)
- void **signalZoomValueEdited** (double)

## Public Member Functions

- **DZoomBar** (QWidget \*const parent=nullptr)
- void **setBarMode** ([BarMode](#) mode)
- void **setThumbsSize** (int size)
- void **setZoom** (double zoom, double zmin, double zmax)
- void **setZoomMinusAction** (QAction \*const action)
- void **setZoomPlusAction** (QAction \*const action)
- void **setZoomTo100Action** (QAction \*const action)
- void **setZoomToFitAction** (QAction \*const action)
- void **triggerZoomTrackerToolTip** ()

## Public Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (QWidget \*const parent=nullptr)
- QSize **minimumSizeHint** () const override
- void **setContentsMargins** (const QMargins &margins)
- void **setContentsMargins** (int left, int top, int right, int bottom)
- void **setSpacing** (int space)
- void **setStretchFactor** (QWidget \*const widget, int stretch)
- QSize **sizeHint** () const override

## Static Public Member Functions

- static int **sizeFromZoom** (double zoom, double zmin, double zmax)
- static double **zoomFromSize** (int size, double zmin, double zmax)

## Additional Inherited Members

## Protected Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (bool vertical, QWidget \*const parent)
- void **childEvent** (QChildEvent \*e) override

## 9.483.1 Member Enumeration Documentation

### 9.483.1.1 BarMode

enum [Digikam::DZoomBar::BarMode](#)

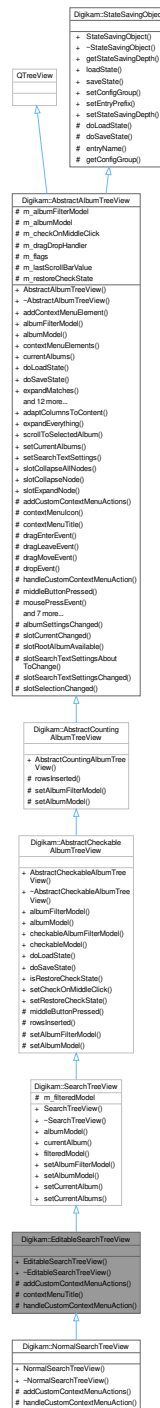
## Enumerator

PreviewZoomCtrl	Preview Zoom controller.
ThumbsSizeCtrl	Thumb Size controller. Preview zoom controller still visible but disabled.
NoPreviewZoomCtrl	Thumb Size controller alone. Preview Zoom controller is hidden.

## 9.484 Digikam::EditableSearchTreeView Class Reference

This tree view for searches adds basic editing functionality via the context menu.

Inheritance diagram for Digikam::EditableSearchTreeView:



## Public Member Functions

- [EditableSearchTreeView](#) (QWidget \*const parent, [searchModel](#) \*const searchModel, [SearchModificationHelper](#) \*const searchModificationHelper)

*Constructor.*

- [~EditableSearchTreeView](#) () override

*Destructor.*

## Public Member Functions inherited from [Digikam::SearchTreeView](#)

- **SearchTreeView** (QWidget \*const parent=nullptr, Flags flags=DefaultFlags)
- **searchModel** \* **albumModel** () const  
*Note: not filtered by search type.*
- **SAlbum** \* **currentAlbum** () const
- **SearchFilterModel** \* **filteredModel** () const  
*Contains only the searches with appropriate type - prefer to [albumModel\(\)](#)*
- void **setAlbumFilterModel** ([SearchFilterModel](#) \*const **filteredModel**, [CheckableAlbumFilterModel](#) \*const model)
- void **setAlbumModel** ([searchModel](#) \*const model)

## Public Member Functions inherited from [Digikam::AbstractCheckableAlbumTreeView](#)

- **AbstractCheckableAlbumTreeView** (QWidget \*const parent, Flags flags)  
*Models of these view can be checkable, they need not.*
- **CheckableAlbumFilterModel** \* **albumFilterModel** () const
- **AbstractCheckableAlbumModel** \* **albumModel** () const  
*Manage check state through the model directly.*
- **CheckableAlbumFilterModel** \* **checkableAlbumFilterModel** () const
- **AbstractCheckableAlbumModel** \* **checkableModel** () const
- void **doLoadState** () override  
*Implements state loading for the album tree view in a somewhat clumsy procedure because the model may not be fully loaded when this method is called.*
- void **doSaveState** () override  
*Implement this hook method for state saving.*
- bool **isRestoreCheckState** () const  
*Tells if the check state is restored while loading / saving state.*
- void **setCheckOnMiddleClick** (bool doThat)  
*Enable checking on middle mouse button click (default: on).*
- void **setRestoreCheckState** (bool restore)  
*Set whether to restore check state or not.*

## Public Member Functions inherited from [Digikam::AbstractCountingAlbumTreeView](#)

- **AbstractCountingAlbumTreeView** (QWidget \*const parent, Flags flags)

## Public Member Functions inherited from [Digikam::AbstractAlbumTreeView](#)

- **AbstractAlbumTreeView** (QWidget \*const parent, Flags flags)  
*Constructs an album tree view.*
- void **addContextMenuElement** ([ContextMenuElement](#) \*const element)
- **AlbumFilterModel** \* **albumFilterModel** () const
- **AbstractSpecificAlbumModel** \* **albumModel** () const
- QList< [ContextMenuElement](#) \* > **contextMenuElements** () const
- template<class A >  
QList< A \* > **currentAlbums** ()
- bool **expandMatches** (const QModelIndex &index)  
*Ensures that every current match is visible by expanding all parent entries.*
- QModelIndex **indexVisuallyAt** (const QPoint &p)



*This is a combination of `indexAt()` checked with `visualRect()`.*

- void **removeContextMenuElement** ([ContextMenuElement](#) \*const element)
- `QList< Album * > selectedItems ()`  
*selectedItems() -*
- void **setAlbumManagerCurrentAlbum** (const bool setCurrentAlbum)  
*Some treeviews shall control the global current album kept by [AlbumManager](#).*
- void **setContextMenuIcon** (const QPixmap &pixmap)  
*Set the context menu title and icon.*
- void **setContextMenuTitle** (const QString &title)
- void **setEnabledContextMenu** (const bool enable)  
*Determines the global decision to show a popup menu or not.*
- void **setExpandNewCurrentItem** (const bool doThat)  
*Expand an item when making it the new current item.*
- void **setExpandOnSingleClick** (const bool doThat)  
*Enable expanding of tree items on single click on the item (default: off)*
- void **setSelectAlbumOnClick** (const bool selectOnClick)  
*Sets whether to select an album on click via the album manager or not.*
- void **setSelectOnContextMenu** (const bool select)  
*Sets whether to select the album under the mouse cursor on a context menu request (so that the album is shown using the album manager) or not.*
- bool **viewportEvent** (QEvent \*event) override  
*For internal use only.*

## Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (QObject \*const host)  
*Constructor.*
- virtual `~StateSavingObject ()`  
*Destructor.*
- [StateSavingDepth](#) **getStateSavingDepth ()** const  
*Returns the depth used for state saving or loading.*
- void **loadState ()**  
*Invokes loading the class' state.*
- void **saveState ()**  
*Invokes saving the class' state.*
- virtual void **setConfigGroup** (const KConfigGroup &group)  
*Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void **setEntryPrefix** (const QString &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void **setStateSavingDepth** (const [StateSavingDepth](#) depth)  
*Sets the depth used for state saving or loading.*

## Protected Member Functions

- void **addCustomContextMenuActions** ([ContextMenuHelper](#) &cmh, [Album](#) \*album) override  
*Adds actions to delete or rename existing searches.*
- `QString contextMenuTitle ()` const override  
*implemented hook methods for context menus.*
- void **handleCustomContextMenuAction** (QAction \*action, const [AlbumPointer](#)< [Album](#) > &album) override  
*Handles deletion and renaming actions.*

### Protected Member Functions inherited from [Digikam::AbstractCheckableAlbumTreeView](#)

- void [middleButtonPressed](#) ([Album \\*a](#)) override
- void [rowsInserted](#) (const [QModelIndex &parent](#), int start, int end) override
- void [setAlbumFilterModel](#) ([CheckableAlbumFilterModel \\*const filterModel](#))
- void [setAlbumModel](#) ([AbstractCheckableAlbumModel \\*const model](#))

### Protected Member Functions inherited from [Digikam::AbstractCountingAlbumTreeView](#)

- void [rowsInserted](#) (const [QModelIndex &parent](#), int start, int end) override
- void [setAlbumFilterModel](#) ([AlbumFilterModel \\*const filterModel](#))
- void [setAlbumModel](#) ([AbstractCountingAlbumModel \\*const model](#))

### Protected Member Functions inherited from [Digikam::AbstractAlbumTreeView](#)

- virtual [QPixmap contextMenuIcon](#) () const  
*Hook method that can be implemented to return a special icon used for the context menu.*
- void [dragEnterEvent](#) ([QDragEnterEvent \\*e](#)) override
- void [dragLeaveEvent](#) ([QDragLeaveEvent \\*e](#)) override
- void [dragMoveEvent](#) ([QDragMoveEvent \\*e](#)) override
- void [dropEvent](#) ([QDropEvent \\*e](#)) override
- void [mousePressEvent](#) ([QMouseEvent \\*e](#)) override  
*Other helper methods.*
- virtual [QPixmap pixmapForDrag](#) (const [QStyleOptionViewItem &option](#), [QList< QModelIndex > indexes](#))  
*TODO: Move to delegate, when we have one.*
- void [rowsAboutToBeRemoved](#) (const [QModelIndex &parent](#), int start, int end) override
- void [rowsInserted](#) (const [QModelIndex &index](#), int start, int end) override
- void [setAlbumFilterModel](#) ([AlbumFilterModel \\*const filterModel](#))
- void [setAlbumModel](#) ([AbstractSpecificAlbumModel \\*const model](#))
- virtual bool [showContextMenuAt](#) ([QContextMenuEvent \\*event](#), [Album \\*albumForEvent](#))  
*Hook method to implement that determines if a context menu shall be displayed for the given event at the position coded in the event.*
- void [startDrag](#) ([Qt::DropActions supportedActions](#)) override

### Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- [QString entryName](#) (const [QString &base](#)) const  
*Always use this method to create config group entry names.*
- [KConfigGroup getConfigGroup](#) () const  
*Returns the config group that must be used for state saving and loading.*

### Additional Inherited Members

### Public Types inherited from [Digikam::AbstractAlbumTreeView](#)

- enum [Flag](#) {  
[CreateDefaultModel](#), [CreateDefaultFilterModel](#), [CreateDefaultDelegate](#), [ShowCountAccordingToSettings](#),  
[AlwaysShowInclusiveCounts](#), [DefaultFlags](#) = [CreateDefaultFilterModel](#) | [CreateDefaultDelegate](#) | [ShowCountAccordingToSettings](#) }
- typedef [QFlags< Flag >](#) [Flags](#)

## Public Types inherited from Digikam::StateSavingObject

- enum [StateSavingDepth](#) { `INSTANCE` , `DIRECT_CHILDREN` , `RECURSIVE` }

*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

## Public Slots inherited from Digikam::SearchTreeView

- void **setCurrentAlbum** (int searchId, bool selectInAlbumManager=true)
- void **setCurrentAlbums** (const QList< [Album](#) \* > &albums, bool selectInAlbumManager=true)

## Public Slots inherited from Digikam::AbstractAlbumTreeView

- void **adaptColumnsToContent** ()  
*Adapt the column sizes to the contents of the tree view.*
- void **expandEverything** (const QModelIndex &index)  
*Expands the complete tree under the given index.*
- void **scrollToSelectedAlbum** ()  
*Scrolls to the first selected album if there is one.*
- void **setCurrentAlbums** (const QList< [Album](#) \* > &albums, bool selectInAlbumManager=true)  
*Selects the given album.*
- void **setSearchTextSettings** (const [SearchTextSettings](#) &settings)
- void **slotCollapseAllNodes** ()  
*slotCollapseAllNodes - collapse all nodes without root node*
- void **slotCollapseNode** ()  
*slotCollapseNode - collapse recursively selected nodes*
- void **slotExpandNode** ()  
*slotExpandNode - expands recursively selected nodes*

## Signals inherited from Digikam::AbstractAlbumTreeView

- void **currentAlbumChanged** ([Album](#) \*currentAlbum)  
*Emitted when the currently selected album changes.*
- void **selectedAlbumsChanged** (const QList< [Album](#) \* > &selectedAlbums)  
*Emitted when the current selection changes.*

## Protected Slots inherited from Digikam::AbstractAlbumTreeView

- void **albumSettingsChanged** ()
- void **slotCurrentChanged** ()
- virtual void **slotRootAlbumAvailable** ()  
*override if implemented behavior is not as intended*
- void **slotSearchTextSettingsAboutToChange** (bool searched, bool willSearch)
- void **slotSearchTextSettingsChanged** (bool wasSearching, bool searching)
- void **slotSelectionChanged** ()

## Protected Attributes inherited from Digikam::SearchTreeView

- [SearchFilterModel](#) \* **m\_filteredModel** = nullptr

## Protected Attributes inherited from [Digikam::AbstractAlbumTreeView](#)

- [AlbumFilterModel](#) \* `m_albumFilterModel` = nullptr
- [AbstractSpecificAlbumModel](#) \* `m_albumModel` = nullptr
- bool `m_checkOnMiddleClick` = false
- [AlbumModelDragDropHandler](#) \* `m_dragDropHandler` = nullptr
- Flags `m_flags` = DefaultFlags
- int `m_lastScrollBarValue` = 0
- bool `m_restoreCheckState` = false

### 9.484.1 Detailed Description

This is in detail deleting and renaming existing searches.

#### Author

jwienke

### 9.484.2 Constructor & Destructor Documentation

#### 9.484.2.1 EditableSearchTreeView()

```
Digikam::EditableSearchTreeView::EditableSearchTreeView (
    QWidget *const parent,
    SearchModel *const searchModel,
    SearchModificationHelper *const searchModificationHelper )
```

#### Parameters

<i>parent</i>	qt parent
<i>searchModel</i>	the model this view should act on
<i>searchModificationHelper</i>	the modification helper object used to perform operations on the displayed searches

### 9.484.3 Member Function Documentation

#### 9.484.3.1 addCustomContextMenuActions()

```
void Digikam::EditableSearchTreeView::addCustomContextMenuActions (
    ContextMenuHelper & cmh,
    Album * album ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::AbstractAlbumTreeView](#).

Reimplemented in [Digikam::NormalSearchTreeView](#).

#### 9.484.3.2 contextMenuTitle()

```
QString Digikam::EditableSearchTreeView::contextMenuTitle ( ) const [override], [protected], [virtual]
```

Reimplemented from [Digikam::AbstractAlbumTreeView](#).

### 9.484.3.3 handleCustomContextMenuAction()

```
void Digikam::EditableSearchTreeView::handleCustomContextMenuAction (
    QAction * action,
    const AlbumPointer< Album > & album ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::AbstractAlbumTreeView](#).

Reimplemented in [Digikam::NormalSearchTreeView](#).

## 9.485 Digikam::EditorCore Class Reference

Inheritance diagram for Digikam::EditorCore:



## Signals

- void **signalFileOriginChanged** (const QString &filePath)
- void **signalImageLoaded** (const QString &filePath, bool success)
- void **signalImageSaved** (const QString &filePath, bool success)
- void **signalLoadingProgress** (const QString &filePath, float progress)
- void **signalLoadingStarted** (const QString &filename)
- void **signalModified** ()
- void **signalSavingProgress** (const QString &filePath, float progress)
- void **signalSavingStarted** (const QString &filename)
- void **signalUndoStateChanged** ()

## Public Member Functions

- void **abortSaving** ()
- void **applyTransform** (const [IccTransform](#) &transform)
- int **availableRedoSteps** () const
- int **availableUndoSteps** () const
- int **bytesDepth** () const
- void **clearUndoManager** ()
- void **convertDepth** (int depth)
- QPixmap **convertToPixmap** (const [DImg](#) &img) const  
*Convert a [DImg](#) image to a pixmap for screen using color managed view if necessary.*
- void **crop** (const QRect &rect)
- QString **ensureHasCurrentUuid** () const
- bool **exifRotated** () const
- void **flipHoriz** ()
- void **flipVert** ()
- [IccProfile](#) **getEmbeddedICC** () const
- [ExposureSettingsContainer](#) \* **getExposureSettings** () const
- [ICCSettingsContainer](#) **getICCSettings** () const
- QString **getImageFileName** () const
- QString **getImageFilePath** () const
- QString **getImageFormat** () const
- [DImageHistory](#) **getImageHistoryOfFullRedo** () const
- [DImg](#) \* **getImg** () const
- [DImg](#) **getImgSelection** () const  
*Image properties.*
- [DImageHistory](#) **getInitialImageHistory** () const
- [DImageHistory](#) **getItemHistory** () const
- [MetaEngineData](#) **getMetadata** () const
- QStringList **getRedoHistory** () const
- [DImageHistory](#) **getResolvedInitialHistory** () const
- QRect **getSelectedArea** () const
- QStringList **getUndoHistory** () const
- bool **hasAlpha** () const
- int **height** () const
- void **imageUndoChanged** (const [UndoMetadataContainer](#) &c)
- bool **isReadOnly** () const
- bool **isValid** () const
- void **load** (const QString &filename, [IOFileSettings](#) \*const ioFileSettings)
- QSize **loadedSize** () const
- int **origHeight** () const
- int **origWidth** () const

- void **provideCurrentUuid** (const QString &uuid)
- void **putIccProfile** (const [IccProfile](#) &profile)
- void **putImg** (const QString &caller, const [FilterAction](#) &action, const [DImg](#) &img)
- void **putImgSelection** (const QString &caller, const [FilterAction](#) &action, const [DImg](#) &img)
- void **readMetadataFromFile** (const QString &file)
- void **redo** ()
- void **resetImage** ()
- void **restore** ()
- void **rollbackToOrigin** ()
- void **rotate180** ()
- void **rotate270** ()
- void **rotate90** ()

*Image transforms.*

- void **saveAs** (const QString &file, [IOFileSettings](#) \*const iofileSettings, bool setExifOrientationTag, const QString &givenMimeType, const QString &intendedFilePath)
- void **saveAs** (const QString &file, [IOFileSettings](#) \*const iofileSettings, bool setExifOrientationTag, const QString &givenMimeType, const [VersionFileOperation](#) &operation)
- void **setDisplayingWidget** (QWidget \*const widget)
- void **setExifOrient** (bool exifOrient)
- void **setExposureSettings** ([ExposureSettingsContainer](#) \*const expoSettings)
- void **setFileOriginData** (const QVariant &data)
- void **setHistoryIsBranch** (bool isBranching)
- void **setIccSettings** (const [IccSettingsContainer](#) &cmSettings)
- void **setLastSaved** (const QString &filePath)
- void **setModified** ()
- void **setResolvedInitialHistory** (const [DImageHistory](#) &history)
- void **setSelectedArea** (const QRect &rect)
- void **setSoftProofingEnabled** (bool enabled)
- void **setUndoImg** (const [UndoMetadataContainer](#) &c, const [DImg](#) &img)

*For internal usage by [UndoManager](#).*

- void **setUndoManagerOrigin** ()
- bool **sixteenBit** () const
- bool **softProofingEnabled** () const
- void **switchToLastSaved** (const [DImageHistory](#) &resolvedCurrentHistory=[DImageHistory](#)())
- void **undo** ()
- [UndoState](#) **undoState** () const
- int **width** () const
- void **zoom** (double val)

### Static Public Member Functions

- static [EditorCore](#) \* **defaultInstance** ()
- static void **setDefaultInstance** ([EditorCore](#) \*const instance)

### Protected Slots

- void **slotImageLoaded** (const [LoadingDescription](#) &loadingDescription, const [DImg](#) &img)
- void **slotImageSaved** (const QString &filePath, bool success)
- void **slotLoadingProgress** (const [LoadingDescription](#) &loadingDescription, float progress)
- void **slotSavingProgress** (const QString &filePath, float progress)

## 9.486 Digikam::EditorStackView Class Reference

Inheritance diagram for Digikam::EditorStackView:



### Public Types

- enum `StackViewMode` { `CanvasMode = 0` , `ToolViewMode` }

### Public Slots

- void `setZoomFactor` (double)
- void `slotZoomSliderChanged` (int)

### Signals

- void `signalToggleOffFitToWindow` ()
- void `signalZoomChanged` (bool isMax, bool isMin, double zoom)



## Public Member Functions

- **EditorStackView** (QWidget \*const parent=nullptr)
- **Canvas** \* **canvas** () const
- void **decreaseZoom** ()
- void **fitToSelect** ()
- void **increaseZoom** ()
- bool **isZoomablePreview** () const
- void **setCanvas** (**Canvas** \*const canvas)
- void **setToolView** (QWidget \*const view)
- void **setViewMode** (int mode)
- void **toggleFitToWindow** ()
- QWidget \* **toolView** () const
- int **viewMode** () const
- double **zoomMax** () const
- double **zoomMin** () const
- void **zoomTo100Percent** ()

## 9.487 Digikam::EditorTool Class Reference

Inheritance diagram for Digikam::EditorTool:



### Public Slots

- virtual void **slotApplyTool** ()
- virtual void **slotCloseTool** ()
- void **slotPreviewModeChanged** ()
- void **slotUpdateSpotInfo** (const [Digikam::DColor](#) &col, const QPoint &point)

## Signals

- void **cancelClicked** ()
- void **okClicked** ()

## Public Member Functions

- **EditorTool** (QObject \*const parent)
- virtual void **exposureSettingsChanged** ()
- virtual void **ICCSettingsChanged** ()
- void **init** ()
 

*Called by editor tool interface to initialize tool when all is ready, through slotInit().*
- **DPlugin** \* **plugin** () const
- virtual void **setBackground** (const QColor &bg)
- void **setInitPreview** (bool b)
 

*Set this option to on if you want to call slotPreview() in slotInit() at tool startup.*
- void **setPlugin** (DPlugin \*const plugin)
- **FilterAction::Category** **toolCategory** () const
- QString **toolHelp** () const
- QIcon **toolIcon** () const
- QString **toolName** () const
- **EditorToolSettings** \* **toolSettings** () const
- int **toolVersion** () const
- QWidget \* **toolView** () const

## Protected Slots

- virtual void **slotCancel** ()
- virtual void **slotInit** ()
- virtual void **slotLoadSettings** ()
- virtual void **slotOk** ()
- virtual void **slotResetSettings** ()
- void **slotTimer** ()

## Protected Member Functions

- virtual void **finalRendering** ()
- virtual void **readSettings** ()
- virtual void **setBusy** (bool)
- void **setPreviewModeMask** (int mask)
- void **setToolCategory** (const FilterAction::Category category)
- void **setToolHelp** (const QString &anchor)
- void **setToolIcon** (const QIcon &icon)
- void **setToolInfoMessage** (const QString &txt)
- void **setToolName** (const QString &name)
- virtual void **setToolSettings** (EditorToolSettings \*const settings)
- void **setToolVersion** (const int version)
- virtual void **setToolView** (QWidget \*const view)
- virtual void **slotChannelChanged** ()
- virtual void **slotPreview** ()
- virtual void **slotSaveAsSettings** ()
- virtual void **slotScaleChanged** ()
- virtual void **writeSettings** ()

## 9.488 Digikam::EditorToolface Class Reference

Inheritance diagram for Digikam::EditorToolface:



### Public Slots

- void **slotApplyTool** ()
- void **slotCloseTool** ()
- void **slotToolAborted** ()
- void **slotToolApplied** ()

### Signals

- void **signalPreviewModeChanged** ()

**Public Member Functions**

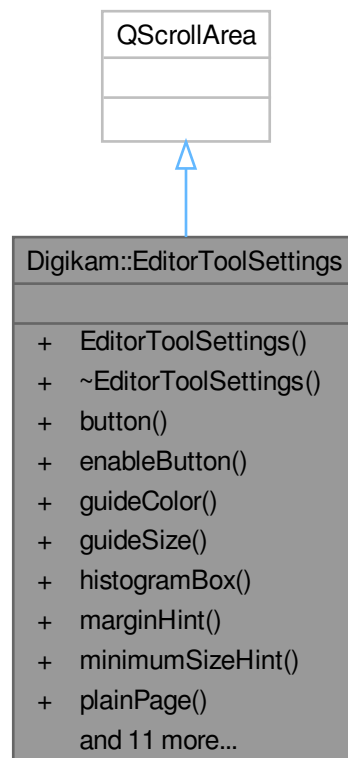
- **EditorTooliface** ([EditorWindow](#) \*const editor)
- [EditorTool](#) \* **currentTool** () const
- void **loadTool** ([EditorTool](#) \*const tool)
- void **setPreviewModeMask** (int mask)
- void **setToolInfoMessage** (const QString &txt)
- void **setToolProgress** (int progress)
- void **setToolsIconView** ([DCategorizedView](#) \*const view)
- void **setToolStartProgress** (const QString &toolName)
- void **setToolStopProgress** ()
- void **setupICC** ()
- void **themeChanged** ()
- void **unLoadTool** ()
- void **updateExposureSettings** ()
- void **updateICCSettings** ()

**Static Public Member Functions**

- static [EditorTooliface](#) \* **editorTooliface** ()

## 9.489 Digikam::EditorToolSettings Class Reference

Inheritance diagram for Digikam::EditorToolSettings:



## Public Types

- enum **ButtonCode** {  
**Default** = 0x00000001 , **Try** = 0x00000002 , **Ok** = 0x00000004 , **Cancel** = 0x00000008 ,  
**SaveAs** = 0x00000010 , **Load** = 0x00000020 }
- typedef QFlags< ButtonCode > **Buttons**
- enum **ToolCode** { **NoTool** = 0x00000000 , **ColorGuide** = 0x00000001 , **Histogram** = 0x00000002 }
- typedef QFlags< ToolCode > **Tools**

## Signals

- void **signalCancelClicked** ()
- void **signalChannelChanged** ()
- void **signalColorGuideChanged** ()
- void **signalDefaultClicked** ()
- void **signalLoadClicked** ()
- void **signalOkClicked** ()
- void **signalSaveAsClicked** ()
- void **signalScaleChanged** ()
- void **signalTryClicked** ()

## Public Member Functions

- **EditorToolSettings** (QWidget \*const parent)
- QPushButton \* **button** (int buttonCode) const
- void **enableButton** (int buttonCode, bool state)
- QColor **guideColor** () const
- int **guideSize** () const
- [HistogramBox](#) \* **histogramBox** () const
- int **marginHint** ()
- QSize **minimumSizeHint** () const override
- QWidget \* **plainPage** () const
- virtual void **readSettings** ()
- virtual void **resetSettings** ()
- virtual void **setBusy** (bool)
- void **setButtons** (Buttons buttonMask)
- void **setGuideColor** (const QColor &color)
- void **setGuideSize** (int size)
- void **setHistogramType** (HistogramBoxType type)
- void **setTool** ([EditorTool](#) \*const tool)
- void **setTools** (Tools toolMask)
- int **spacingHint** ()
- virtual void **writeSettings** ()

## 9.490 Digikam::EditorToolThreaded Class Reference

Inheritance diagram for Digikam::EditorToolThreaded:



### Public Types

- enum **RenderingMode** { **NoneRendering** = 0 , **PreviewRendering** , **FinalRendering** }

## Public Slots

- virtual void **slotAbort** ()

## Public Slots inherited from [Digikam::EditorTool](#)

- virtual void **slotApplyTool** ()
- virtual void **slotCloseTool** ()
- void **slotPreviewModeChanged** ()
- void **slotUpdateSpotInfo** (const [Digikam::DColor](#) &col, const QPoint &point)

## Public Member Functions

- **EditorToolThreaded** (QObject \*const parent)
- RenderingMode **renderingMode** () const  
*return the current tool rendering mode.*
- void **setProgressMessage** (const QString &mess)  
*Set the small text to show in editor status progress bar during tool computation.*

## Public Member Functions inherited from [Digikam::EditorTool](#)

- **EditorTool** (QObject \*const parent)
- virtual void **exposureSettingsChanged** ()
- virtual void **ICCSettingsChanged** ()
- void **init** ()  
*Called by editor tool interface to initialized tool when all is ready, through slotInit().*
- [DPlugin](#) \* **plugin** () const
- virtual void **setBackgroundcolor** (const QColor &bg)
- void **setInitPreview** (bool b)  
*Set this option to on if you want to call slotPreview() in slotInit() at tool startup.*
- void **setPlugin** ([DPlugin](#) \*const plugin)
- [FilterAction::Category](#) **toolCategory** () const
- QString **toolHelp** () const
- QIcon **toolIcon** () const
- QString **toolName** () const
- [EditorToolSettings](#) \* **toolSettings** () const
- int **toolVersion** () const
- QWidget \* **toolView** () const

## Protected Slots

- void **slotAnalyserFinished** (bool success)
- void **slotAnalyserStarted** ()  
*Manage start and end events from analyser.*
- void **slotCancel** () override
- void **slotFilterFinished** (bool success)
- void **slotFilterStarted** ()  
*Manage start and end events from filter.*
- void **slotInit** () override
- void **slotOk** () override
- void **slotPreview** () override
- void **slotProgress** (int progress)  
*Dispatch progress event from filter and analyser.*



## Protected Slots inherited from [Digikam::EditorTool](#)

- virtual void **slotCancel** ()
- virtual void **slotInit** ()
- virtual void **slotLoadSettings** ()
- virtual void **slotOk** ()
- virtual void **slotResetSettings** ()
- void **slotTimer** ()

## Protected Member Functions

- [DImgThreadedAnalyser](#) \* **analyser** () const  
*Manage analyser instance plugged in tool interface.*
- virtual void **analyserCompleted** ()
- void [deleteFilterInstance](#) (bool b=true)  
*If true, delete filter instance when preview or final rendering is processed.*
- [DImgThreadedFilter](#) \* **filter** () const  
*Manage filter instance plugged in tool interface.*
- virtual void **prepareFinal** ()
- virtual void **preparePreview** ()
- virtual void **renderingFinished** ()
- void **setAnalyser** ([DImgThreadedAnalyser](#) \*const analyser)
- void **setFilter** ([DImgThreadedFilter](#) \*const filter)
- virtual void **setFinalImage** ()
- virtual void **setPreviewImage** ()

## Protected Member Functions inherited from [Digikam::EditorTool](#)

- virtual void **finalRendering** ()
- virtual void **readSettings** ()
- virtual void **setBusy** (bool)
- void **setPreviewModeMask** (int mask)
- void **setToolCategory** (const [FilterAction::Category](#) category)
- void **setToolHelp** (const QString &anchor)
- void **setToolIcon** (const QIcon &icon)
- void **setToolInfoMessage** (const QString &txt)
- void **setToolName** (const QString &name)
- virtual void **setToolSettings** ([EditorToolSettings](#) \*const settings)
- void **setToolVersion** (const int version)
- virtual void **setToolView** (QWidget \*const view)
- virtual void **slotChannelChanged** ()
- virtual void **slotPreview** ()
- virtual void **slotSaveAsSettings** ()
- virtual void **slotScaleChanged** ()
- virtual void **writeSettings** ()

## Additional Inherited Members

## Signals inherited from [Digikam::EditorTool](#)

- void **cancelClicked** ()
- void **okClicked** ()

## 9.490.1 Member Function Documentation

### 9.490.1.1 deleteFilterInstance()

```
void Digikam::EditorToolThreaded::deleteFilterInstance (
    bool b = true ) [protected]
```

If false, filter instance will be managed outside for ex. with ContentAwareResizing tool.

### 9.490.1.2 setProgressMessage()

```
void Digikam::EditorToolThreaded::setProgressMessage (
    const QString & mess )
```

If it's not set, tool name is used instead.

## 9.491 Digikam::EditorWindow Class Reference

Inheritance diagram for Digikam::EditorWindow:



### Public Types

- enum **TransformType** { **RotateLeft** , **RotateRight** , **FlipHorizontal** , **FlipVertical** }

## Public Slots

- void **slotSetup** () override=0
- virtual void **slotSetupICC** ()=0

## Signals

- void **signalNoCurrentItem** ()
- void **signalPreviewModeChanged** (int)
- void **signalSelectionChanged** (const QRect &)
- void **signalToolApplied** ()

## Public Member Functions

- **EditorWindow** (const QString &name, QWidget \*const parent=nullptr)
- bool **actionEnabledState** () const
- void **loadTool** ([EditorTool](#) \*const tool)
- void [registerExtraPluginsActions](#) (QString &dom) override

## Public Member Functions inherited from [Digikam::DXmlGuiWindow](#)

- **DXmlGuiWindow** (QWidget \*const parent=nullptr, Qt::WindowFlags f=Qt::WindowFlags())
- QList< QAction \* > **allActions** () const  
*Return all actions from internal collection.*
- void **cleanupActions** ()  
*Cleanup unwanted actions from action collection.*
- QString **configGroupName** () const
- void [createFullScreenAction](#) (const QString &name)  
*Create Full-screen action to action collection instance from managed window set through setManagedWindow().*
- void **createHelpActions** (const QString &handbookSection, bool coreOptions=true)  
*Create common actions from Help menu for all digiKam main windows.*
- void **createSettingsActions** ()  
*Create common actions to setup all digiKam main windows.*
- void **createSidebarActions** ()  
*Create common actions to handle side-bar through keyboard shortcuts.*
- bool **fullScreensActive** () const  
*Return true if managed window is currently in Full Screen Mode.*
- virtual [DInfoInterface](#) \* **interface** ([DPluginAction](#) \*const ac)=0  
*Return the interface instance to access to items information.*
- void **readFullScreenSettings** (const KConfigGroup &group)  
*Read full-screen settings from KDE config file.*
- void [registerPluginsActions](#) ()  
*Register all generic plugins action to this instance.*
- void **setConfigGroupName** (const QString &name)  
*Manage config group name used by window instance to get/set settings from config file.*
- void **setFullScreenOptions** (int options)  
*Set full-screen options to managed window.*
- void **unminimizeAndActivateWindow** ()

## Static Public Attributes

- static const QString **CONFIG\_GROUP\_NAME**

## Protected Types

- enum **SaveAskMode** {  
**AskIfNeeded** , **OverwriteWithoutAsking** , **AlwaysSaveAs** , **SaveVersionWithoutAsking** = Overwrite↔  
WithoutAsking ,  
**AlwaysNewVersion** = AlwaysSaveAs }

## Protected Slots

- virtual bool **save** ()=0
- virtual bool **saveAs** ()=0
- virtual bool **saveCurrentVersion** ()=0
- virtual bool **saveNewVersion** ()=0
- virtual bool **saveNewVersionAs** ()=0
- virtual bool **saveNewVersionInFormat** (const QString &)=0
- virtual bool **saveOrSaveAs** ()
- void **slotAboutToShowRedoMenu** ()
- void **slotAboutToShowUndoMenu** ()
- virtual void **slotAddedDroppedItems** (QDropEvent \*e)=0
- virtual void **slotBackward** ()=0
- virtual void **slotChanged** ()=0
- void **slotComponentsInfo** () override
- virtual void **slotContextMenu** ()=0
- virtual void **slotDeleteCurrentItem** ()=0
- virtual void **slotDiscardChanges** ()
- virtual void **slotFileOriginChanged** (const QString &filePath)
- virtual void **slotFileWithDefaultApplication** ()=0
- virtual void **slotFirst** ()=0
- virtual void **slotForward** ()=0
- virtual void **slotLast** ()=0
- virtual void **slotLoadingFinished** (const QString &filename, bool success)
- void **slotLoadingProgress** (const QString &filePath, float progress)
- virtual void **slotLoadingStarted** (const QString &filename)
- void **slotNameLabelCancelButtonPressed** ()
- virtual void **slotOpenOriginal** ()
- virtual void **slotOpenWith** (QAction \*action=nullptr)=0
- virtual void **slotPrepareToLoad** ()
- virtual void **slotRevert** ()=0
- void **slotSavingProgress** (const QString &filePath, float progress)
- virtual void **slotSavingStarted** (const QString &filename)
- void **slotSelected** (bool)
- virtual void **slotUpdateItemInfo** ()=0

## Protected Slots inherited from [Digikam::DXmlGuiWindow](#)

- bool **slotClose** ()

## Protected Member Functions

- virtual void **addServicesMenu** ()=0
- void **addServicesMenuForUrl** (const QUrl &url)
- void **applyColorManagementSettings** ()
- void **applyIOSettings** ()
- void **applyStandardSettings** ()
- bool **checkOverwrite** (const QUrl &url)
- bool **checkPermissions** (const QUrl &url)
- void **colorManage** ()
- [EditorStackView](#) \* **editorStackView** () const
- void **execSavingProgressDialog** ()
- [ExposureSettingsContainer](#) \* **exposureSettings** () const
- virtual void **finishSaving** (bool success)
- virtual bool **hasOriginalToRestore** ()
- virtual void **moveFile** ()
- bool **moveLocalFile** (const QString &src, const QString &dest)
- void **movingSaveFileFinished** (bool successful)
- void **openWith** (const QUrl &url, QAction \*action)
- bool **promptForOverWrite** ()
- bool **promptUserDelete** (const QUrl &url)
- bool **promptUserSave** (const QUrl &url, SaveAskMode mode=AskIfNeeded, bool allowCancel=true)
- virtual void **readSettings** ()
- void **readStandardSettings** ()
- void **resetOrigin** ()
- void **resetOriginSwitchFile** ()
- virtual [DImageHistory](#) **resolvedImageHistory** (const [DImageHistory](#) &history)
- virtual [Sidebar](#) \* **rightSideBar** () const =0
- virtual void **saveAsIsComplete** ()=0
- [VersionFileOperation](#) **saveAsVersionFileOperation** (const QUrl &url, const QUrl &saveLocation, const QString &format)
- virtual QUrl **saveDestinationUrl** ()=0
  - Hook method that subclasses must implement to return the destination url of the image to save.*
- [VersionFileOperation](#) **saveInFormatVersionFileOperation** (const QUrl &url, const QString &format)
- virtual void **savelsComplete** ()=0
- virtual void **saveSettings** ()
- void **saveStandardSettings** ()
- [VersionFileOperation](#) **saveVersionFileOperation** (const QUrl &url, bool fork)
- virtual void **saveVersionIsComplete** ()=0
- virtual void **setupActions** ()=0
- virtual void **setupConnections** ()=0
- void **setupContextMenu** ()
- void **setupSelectToolsAction** ()
- void **setupStandardActions** ()
- void **setupStandardConnections** ()
- void **setupStatusBar** ()
- virtual void **setupUserArea** ()=0
- [SidebarSplitter](#) \* **sidebarSplitter** () const
- void **startingSave** (const QUrl &url)
- bool **startingSaveAs** (const QUrl &url)
- bool **startingSaveCurrentVersion** (const QUrl &url)
- bool **startingSaveNewVersion** (const QUrl &url)
- bool **startingSaveNewVersionAs** (const QUrl &url)
- bool **startingSaveNewVersionInFormat** (const QUrl &url, const QString &format)
- virtual [ThumbBarDock](#) \* **thumbBar** () const =0

- virtual void **toggleActions** (bool val)
- void **toggleNonDestructiveActions** ()
- void **toggleStandardActions** (bool val)
- void **toggleToolActions** ([EditorTool](#) \*tool=nullptr)
- void **toggleZoomActions** (bool val)
  - Method used by Editor Tools.*
- virtual [VersionManager](#) \* **versionManager** () const
- bool **waitForSavingToComplete** ()

## Protected Member Functions inherited from [Digikam::DXmlGuiWindow](#)

- void **closeEvent** (QCloseEvent \*e) override
- void **editKeyboardShortcuts** (KActionCollection \*const extraac=nullptr, const QString &actitle=QString())
  - Call this method from your main window to show keyboard shortcut config dialog with an extra action collection to configure.*
- bool **eventFilter** (QObject \*obj, QEvent \*ev) override
- void **keyPressEvent** (QKeyEvent \*e) override
- QAction \* **showMenuBarAction** () const
- QAction \* **showStatusBarAction** () const

## Protected Attributes

- bool **m\_actionEnabledState** = false
- QAction \* **m\_applyToolAction** = nullptr
- QAction \* **m\_backwardAction** = nullptr
- QColor **m\_bgColor**
- [Canvas](#) \* **m\_canvas** = nullptr
- QAction \* **m\_closeToolAction** = nullptr
- QMenu \* **m\_contextMenu** = nullptr
- QAction \* **m\_discardChangesAction** = nullptr
- bool **m\_editingOriginalImage** = true
- QAction \* **m\_exportAction** = nullptr
- QAction \* **m\_fileDeleteAction** = nullptr
- QAction \* **m\_firstAction** = nullptr
- QString **m\_formatForRAWVersioning**
- QString **m\_formatForSubversions**
- QAction \* **m\_forwardAction** = nullptr
- [IOFileSettings](#) \* **m\_IOFileSettings** = nullptr
- QAction \* **m\_lastAction** = nullptr
- [StatusProgressBar](#) \* **m\_nameLabel** = nullptr
- bool **m\_nonDestructive** = true
- QAction \* **m\_openVersionAction** = nullptr
- [KToolBarPopupAction](#) \* **m\_redoAction** = nullptr
- [DAdjustableLabel](#) \* **m\_resLabel** = nullptr
- QAction \* **m\_revertAction** = nullptr
- QAction \* **m\_saveAction** = nullptr
- QAction \* **m\_saveAsAction** = nullptr
- QAction \* **m\_saveCurrentVersionAction** = nullptr
- [KToolBarPopupAction](#) \* **m\_saveNewVersionAction** = nullptr
- QAction \* **m\_saveNewVersionAsAction** = nullptr
- QMenu \* **m\_saveNewVersionInFormatAction** = nullptr
- [SavingContext](#) **m\_savingContext**
- [QPointer](#)< [QProgressDialog](#) > **m\_savingProgressDialog** = nullptr

- QAction \* **m\_serviceAction** = nullptr
- QMenu \* **m\_servicesMenu** = nullptr
- bool **m\_setExifOrientationTag** = true
- QAction \* **m\_showBarAction** = nullptr
- [SidebarSplitter](#) \* **m\_splitter** = nullptr
- [EditorStackView](#) \* **m\_stackView** = nullptr
- QVector< TransformType > **m\_transformQue**
- KToolBarPopupAction \* **m\_undoAction** = nullptr

### Protected Attributes inherited from [Digikam::DXmlGuiWindow](#)

- [DLogoAction](#) \* **m\_animLogo** = nullptr

### Friends

- class **EditorToolface**

### Additional Inherited Members

### Static Public Member Functions inherited from [Digikam::DXmlGuiWindow](#)

- static QAction \* **buildStdAction** (StdActionType type, const QObject \*const recvr, const char \*const slot, QObject \*const parent)
  - static QString **configFullScreenHideSideBarsEntry** ()
  - static QString **configFullScreenHideStatusBarEntry** ()
  - static QString **configFullScreenHideThumbBarEntry** ()
  - static QString **configFullScreenHideToolBarsEntry** ()
- Shared with [FullScreenSettings](#).*
- static void **restoreWindowSize** (QWindow \*const win, const KConfigGroup &group)
  - static void **saveWindowSize** (QWindow \*const win, KConfigGroup &group)
  - static void **setGoodDefaultWindowSize** (QWindow \*const win)
  - static void **setupIconTheme** ()

*If we have some local breeze icon resource, prefer it.*

## 9.491.1 Member Function Documentation

### 9.491.1.1 registerExtraPluginsActions()

```
void Digikam::EditorWindow::registerExtraPluginsActions (
    QString & dom ) [override], [virtual]
```

Reimplemented from [Digikam::DXmlGuiWindow](#).



### 9.491.1.2 saveDestinationUrl()

```
virtual QUrl Digikam::EditorWindow::saveDestinationUrl ( ) [protected], [pure virtual]
```

This may also be a remote url.

This method will only be called while saving.

#### Returns

destination for the file that is currently being saved.

### 9.491.1.3 toggleZoomActions()

```
void Digikam::EditorWindow::toggleZoomActions (
    bool val ) [protected]
```

Only tools based on imageregionwidget support zooming. TODO: Fix this behavior when editor tool preview widgets will be factored.

## 9.491.2 Member Data Documentation

### 9.491.2.1 m\_transformQue

```
QVector<TransformType> Digikam::EditorWindow::m_transformQue [protected]
```

#### Note

using QVector to store transforms

## 9.492 Digikam::EffectMngr Class Reference

### Public Types

- enum [EffectType](#) {
 [None](#) = 0 , [KenBurnsZoomIn](#) , [KenBurnsZoomOut](#) , [KenBurnsPanLR](#) ,
 [KenBurnsPanRL](#) , [KenBurnsPanTB](#) , [KenBurnsPanBT](#) , [Random](#) }

See KEn Burns effect description: [https://en.wikipedia.org/wiki/Ken\\_Burns\\_effect](https://en.wikipedia.org/wiki/Ken_Burns_effect).

### Public Member Functions

- QImage [currentFrame](#) (int &tmout)
- void [setEffect](#) ([EffectType](#) eff)
- void [setFrames](#) (int ifrms)
- void [setImage](#) (const QImage &img)
- void [setOutputSize](#) (const QSize &size)

### Static Public Member Functions

- static QMap< [EffectType](#), QString > [effectNames](#) ()

## 9.492.1 Member Enumeration Documentation

### 9.492.1.1 EffectType

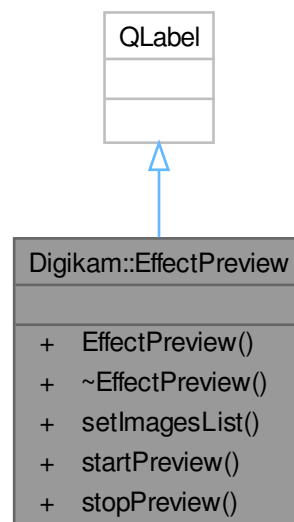
```
enum Digikam::EffectMngr::EffectType
```

Enumerator

None	Static camera.
------	----------------

## 9.493 Digikam::EffectPreview Class Reference

Inheritance diagram for Digikam::EffectPreview:



### Public Member Functions

- **EffectPreview** (`QWidget *const parent=nullptr`)
- void **setImagesList** (`const QList< QUrl > &images`)
- void **startPreview** ([EffectMgr::EffectType](#) `eff`)
- void **stopPreview** (`()`)

## 9.494 Digikam::Ellipsoid Class Reference

Geometric figure that can be used to describe the approximate shape of the earth.

## Public Member Functions

- double [eccentricity](#) () const  
*The ratio of the distance between the center and a focus of the ellipse to the length of its semimajor axis.*
- double [inverseFlattening](#) () const  
*Returns the value of the inverse of the flattening constant.*
- bool [isIvfDefinitive](#) () const  
*Indicates if the inverse flattening is definitive for this ellipsoid.*
- bool [isSphere](#) () const  
*true if the ellipsoid is degenerate and is actually a sphere.*
- double [orthodromicDistance](#) (double x1, double y1, double x2, double y2)  
*Returns the orthodromic distance between two geographic coordinates.*
- double [radiusOfCurvature](#) (double latitude)  
*Returns the Radius Of Curvature for the given latitude, using the geometric mean of two radii of curvature for all azimuths.*
- double [semiMajorAxis](#) () const  
*Length of the semi-major axis of the ellipsoid.*
- double [semiMinorAxis](#) () const  
*Length of the semi-minor axis of the ellipsoid.*

## Static Public Member Functions

- static [Ellipsoid CLARKE\\_1866](#) ()  
*Clarke 1866 ellipsoid with axis in metres.*
- static [Ellipsoid createEllipsoid](#) (const QString &name, double [semiMajorAxis](#), double [semiMinorAxis](#))  
*Constructs a new ellipsoid using the specified axis length.*
- static [Ellipsoid createFlattenedSphere](#) (const QString &name, double [semiMajorAxis](#), double [inverseFlattening](#))  
*Constructs a new ellipsoid using the specified axis length and inverse flattening value.*
- static [Ellipsoid GRS80](#) ()  
*GRS 80 ellipsoid with axis in metres.*
- static [Ellipsoid INTERNATIONAL\\_1924](#) ()  
*International 1924 ellipsoid with axis in metres.*
- static [Ellipsoid SPHERE](#) ()  
*A sphere with a radius of 6371000 metres.*
- static [Ellipsoid WGS84](#) ()  
*WGS 1984 ellipsoid with axis in metres.*

## Protected Member Functions

- [Ellipsoid](#) (const QString &name, double radius, bool ivfDefinitive)
- [Ellipsoid](#) (const QString &name, double [semiMajorAxis](#), double [semiMinorAxis](#), double [inverseFlattening](#), bool ivfDefinitive)  
*Constructs a new ellipsoid using the specified axis length.*

## Protected Attributes

- double `m_inverseFlattening` = 0.0  
*The inverse of the flattening value, or DBL\_MAX if the ellipsoid is a sphere.*
- bool `m_isSphere` = false
- bool `m_ivfDefinitive` = false  
*Tells if the Inverse Flattening definitive for this ellipsoid.*
- double `m_semiMajorAxis` = 0.0  
*The equatorial radius.*
- double `m_semiMinorAxis` = 0.0  
*The polar radius.*
- QString `name`

### 9.494.1 Detailed Description

In mathematical terms, it is a surface formed by the rotation of an ellipse about its minor axis. An ellipsoid requires two defining parameters:

- semi-major axis and inverse flattening, or
- semi-major axis and semi-minor axis.

### 9.494.2 Constructor & Destructor Documentation

#### 9.494.2.1 Ellipsoid()

```

Digikam::Ellipsoid::Ellipsoid (
    const QString & name,
    double semiMajorAxis,
    double semiMinorAxis,
    double inverseFlattening,
    bool ivfDefinitive ) [protected]

```

The properties map is given unchanged to the AbstractIdentifiedObjectAbstractIdentifiedObject(Map) super-class constructor.

#### Parameters

<i>name</i>	The ellipsoid name.
<i>semiMajorAxis</i>	The equatorial radius.
<i>semiMinorAxis</i>	The polar radius.
<i>inverseFlattening</i>	The inverse of the flattening value.
<i>ivfDefinitive</i>	true if the inverse flattening is definitive.

#### See also

[createEllipsoid](#)

[createFlattenedSphere](#)

## 9.494.3 Member Function Documentation

### 9.494.3.1 createEllipsoid()

```
Ellipsoid Digikam::Ellipsoid::createEllipsoid (
    const QString & name,
    double semiMajorAxis,
    double semiMinorAxis ) [static]
```

#### Parameters

<i>name</i>	The ellipsoid name.
<i>semiMajorAxis</i>	The equatorial radius.
<i>semiMinorAxis</i>	The polar radius.

### 9.494.3.2 createFlattenedSphere()

```
Ellipsoid Digikam::Ellipsoid::createFlattenedSphere (
    const QString & name,
    double semiMajorAxis,
    double inverseFlattening ) [static]
```

#### Parameters

<i>name</i>	The ellipsoid name.
<i>semiMajorAxis</i>	The equatorial radius.
<i>inverseFlattening</i>	The inverse flattening value. values.

### 9.494.3.3 eccentricity()

```
double Digikam::Ellipsoid::eccentricity ( ) const
```

The eccentricity can alternately be computed from the equation:  $e = \sqrt{2f - f^2}$ .

### 9.494.3.4 inverseFlattening()

```
double Digikam::Ellipsoid::inverseFlattening ( ) const
```

Flattening is a value used to indicate how closely an ellipsoid approaches a spherical shape. The inverse flattening is related to the equatorial/polar radius by the formula

$$ivf = r_e / (r_e - r_p).$$

For perfect spheres (i.e. if `isSphere` returns `true`), the `DoublePOSITIVE_INFINITY` value is used.

#### Returns

The inverse flattening value.

### 9.494.3.5 isIvfDefinitive()

```
bool Digikam::Ellipsoid::isIvfDefinitive ( ) const
```

Some ellipsoids use the IVF as the defining value, and calculate the polar radius whenever asked. Other ellipsoids use the polar radius to calculate the IVF whenever asked. This distinction can be important to avoid floating-point rounding errors.

#### Returns

`true` if the inverse flattening is definitive, or `false` if the polar radius is definitive.

### 9.494.3.6 isSphere()

```
bool Digikam::Ellipsoid::isSphere ( ) const
```

The sphere is completely defined by the semi-major axis, which is the radius of the sphere.

#### Returns

`true` if the ellipsoid is degenerate and is actually a sphere.

### 9.494.3.7 orthodromicDistance()

```
double Digikam::Ellipsoid::orthodromicDistance (
    double x1,
    double y1,
    double x2,
    double y2 )
```

The orthodromic distance is the shortest distance between two points on a sphere's surface. The orthodromic path is always on a great circle. This is different from the loxodromic distance, which is a longer distance on a path with a constant direction on the compass.

#### Parameters

<code>x1</code>	Longitude of first point (in decimal degrees).
<code>y1</code>	Latitude of first point (in decimal degrees).
<code>x2</code>	Longitude of second point (in decimal degrees).
<code>y2</code>	Latitude of second point (in decimal degrees).

#### Returns

The orthodromic distance (in the units of this ellipsoid's axis).

### 9.494.3.8 radiusOfCurvature()

```
double Digikam::Ellipsoid::radiusOfCurvature (
    double latitude )
```

## Parameters

<i>latitude</i>	in degrees
-----------------	------------

**9.494.3.9 semiMajorAxis()**

```
double Digikam::Ellipsoid::semiMajorAxis ( ) const
```

This is the equatorial radius in axis linear unit.

## Returns

Length of semi-major axis.

**9.494.3.10 semiMinorAxis()**

```
double Digikam::Ellipsoid::semiMinorAxis ( ) const
```

This is the polar radius in axis linear unit.

## Returns

Length of semi-minor axis.

**9.494.3.11 SPHERE()**

```
Ellipsoid Digikam::Ellipsoid::SPHERE ( ) [static]
```

Spheres use a simpler algorithm for orthodromic distance computation, which may be faster and more robust.

**9.494.3.12 WGS84()**

```
Ellipsoid Digikam::Ellipsoid::WGS84 ( ) [static]
```

This ellipsoid is used in GPS systems and is the default for most `org.geotools` packages.

**9.494.4 Member Data Documentation****9.494.4.1 m\_inverseFlattening**

```
double Digikam::Ellipsoid::m_inverseFlattening = 0.0 [protected]
```

## See also

`getInverseFlattening`

#### 9.494.4.2 m\_ivfDefinitive

```
bool Digikam::Ellipsoid::m_ivfDefinitive = false [protected]
```

See also

[isIvfDefinitive](#)

#### 9.494.4.3 m\_semiMajorAxis

```
double Digikam::Ellipsoid::m_semiMajorAxis = 0.0 [protected]
```

See also

[getSemiMajorAxis](#)

#### 9.494.4.4 m\_semiMinorAxis

```
double Digikam::Ellipsoid::m_semiMinorAxis = 0.0 [protected]
```

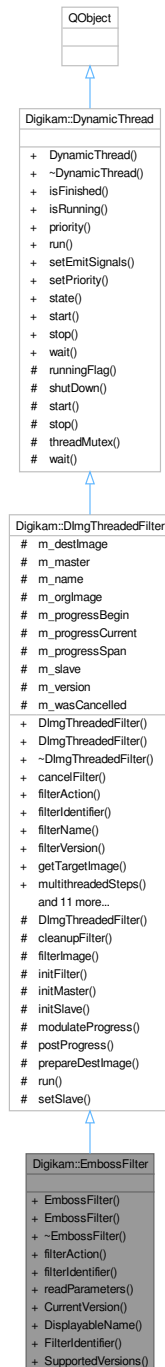


See also

[getSemiMinorAxis](#)

## 9.495 Digikam::EmbossFilter Class Reference

Inheritance diagram for Digikam::EmbossFilter:



## Public Member Functions

- **EmbossFilter** ([DImg](#) \*const orgImage, QObject \*const parent=nullptr, int depth=30)
- **EmbossFilter** (QObject \*const parent=nullptr)
- [FilterAction](#) filterAction () override
 

*Returns the action description corresponding to currently set options.*
- QString filterIdentifier () const override
 

*Return the identifier for this filter in the image history.*
- void readParameters (const [FilterAction](#) &action) override

## Public Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImg](#) \*const orgImage, QObject \*const parent, const QString &name=QString())
 

*Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter](#) (QObject \*const parent=nullptr, const QString &name=QString())
 

*Constructs a filter without argument.*
- virtual void cancelFilter ()
 

*Cancel the threaded computation.*
- const QString & filterName ()
- int filterVersion () const
- [DImg](#) getTargetImage ()
- QList< int > multithreadedSteps (int stop, int start=0) const
 

*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool parametersSuccessfullyRead () const
 

*Optional: error handling for readParameters.*
- virtual QString readParametersError (const [FilterAction](#) &actionThatFailed) const
- void setFilterName (const QString &name)
- void setFilterVersion (int version)
 

*Replaying a filter action: Set the filter version.*
- void setOriginalImage (const [DImg](#) &orgImage)
- void setupAndStartDirectly (const [DImg](#) &orgImage, [DImgThreadedFilter](#) \*const master, int progress←Begin=0, int progressEnd=100)
 

*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void setupFilter (const [DImg](#) &orgImage)
 

*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void startFilter ()
 

*Start the threaded computation.*
- virtual void startFilterDirectly ()
 

*Start computation of this filter, directly in this thread.*
- virtual QList< int > supportedVersions () const

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread](#) (QObject \*const parent=nullptr)
 

*This class extends QRunnable, so you have to reimplement virtual void run().*
- ~[DynamicThread](#) () override
 

*The destructor calls stop() and wait(), but if you, in your destructor, delete any data that is accessed by your run() method, you must call stop() and wait() before yourself.*
- bool isFinished () const
- bool isRunning () const
- QThread::Priority priority () const
- void setEmitSignals (bool emitThem)
- void setPriority (QThread::Priority priority)
 

*Sets the priority for this dynamic thread.*
- State state () const

### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from [Digikam::DImgThreadedFilter](#)

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.495.1 Member Function Documentation

### 9.495.1.1 filterAction()

`FilterAction` Digikam::EmbossFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.495.1.2 filterIdentifier()

`QString` Digikam::EmbossFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

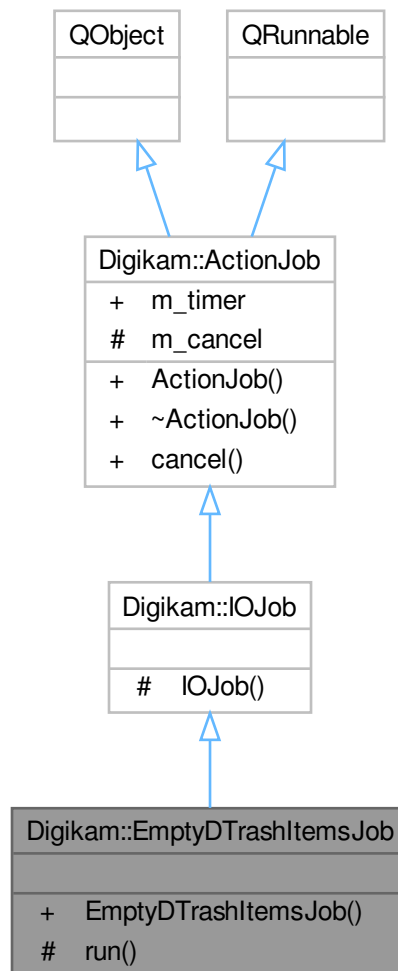
### 9.495.1.3 readParameters()

```
void Digikam::EmbossFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

## 9.496 Digikam::EmptyDTrashItemsJob Class Reference

Inheritance diagram for Digikam::EmptyDTrashItemsJob:



### Public Member Functions

- `EmptyDTrashItemsJob` (`IOJobData *const data`)

### Public Member Functions inherited from `Digikam::ActionJob`

- `ActionJob` (`QObject *const parent=nullptr`)  
*Constructor which delegate deletion of `QRunnable` instance to `ActionThreadBase`, not `QThreadPool`.*
- `~ActionJob` () override  
*Re-implement destructor in you implementation.*

### Protected Member Functions

- void **run** () override

### Additional Inherited Members

### Public Slots inherited from [Digikam::ActionJob](#)

- void **cancel** ()  
*Call this method to cancel job.*

### Signals inherited from [Digikam::IOJob](#)

- void **signalError** (const QString &errMsg)
- void **signalOneProcessed** (const QUrl &url)

### Signals inherited from [Digikam::ActionJob](#)

- void **signalDone** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job is done.*
- void **signalProgress** (int)  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job progress.*
- void **signalStarted** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager that job is started.*

### Public Attributes inherited from [Digikam::ActionJob](#)

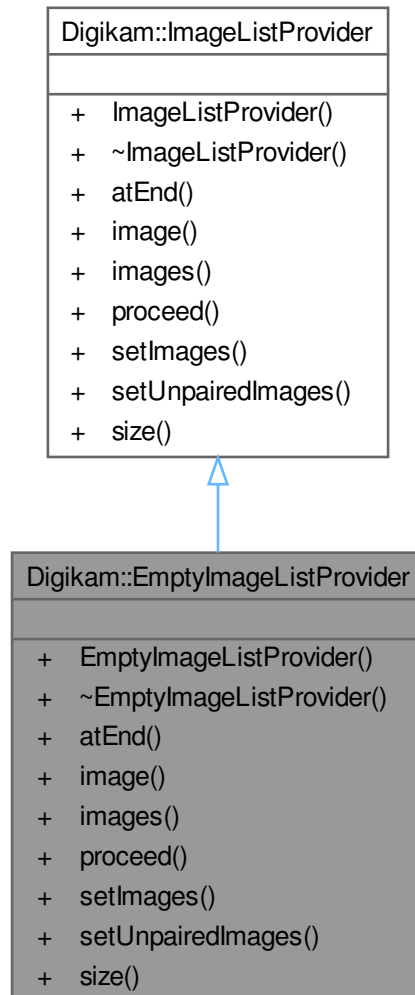
- QElapsedTimer **m\_timer**  
*Timer to determine the running time of the job.*

### Protected Attributes inherited from [Digikam::ActionJob](#)

- bool **m\_cancel** = false  
*You can use this boolean in your implementation to know if job must be canceled.*

## 9.497 Digikam::EmptyImageListProvider Class Reference

Inheritance diagram for Digikam::EmptyImageListProvider:



### Public Member Functions

- bool `atEnd()` const override
- `QPair< QImage *, QString >` `image()` override
- `QList< QPair< QImage *, QString > >` `images()` override
- void `proceed` (int steps=1) override
- void `setImages` (const `QList< QPair< QImage *, QString > >` &) override
- void `setUnpairedImages` (const `QList< QImage * >` &) override
- int `size()` const override



## 9.497.1 Member Function Documentation

### 9.497.1.1 atEnd()

```
bool Digikam::EmptyImageListProvider::atEnd ( ) const [override], [virtual]
```

Implements [Digikam::ImageListProvider](#).

### 9.497.1.2 image()

```
QPair< QImage *, QString > Digikam::EmptyImageListProvider::image ( ) [override], [virtual]
```

Implements [Digikam::ImageListProvider](#).

### 9.497.1.3 images()

```
QList< QPair< QImage *, QString > > Digikam::EmptyImageListProvider::images ( ) [override], [virtual]
```

Implements [Digikam::ImageListProvider](#).

### 9.497.1.4 proceed()

```
void Digikam::EmptyImageListProvider::proceed (
    int steps = 1 ) [override], [virtual]
```

Implements [Digikam::ImageListProvider](#).

### 9.497.1.5 setImages()

```
void Digikam::EmptyImageListProvider::setImages (
    const QList< QPair< QImage *, QString > > & ) [override], [virtual]
```

Implements [Digikam::ImageListProvider](#).

### 9.497.1.6 setUnpairedImages()

```
void Digikam::EmptyImageListProvider::setUnpairedImages (
    const QList< QImage * > & ) [override], [virtual]
```

Implements [Digikam::ImageListProvider](#).

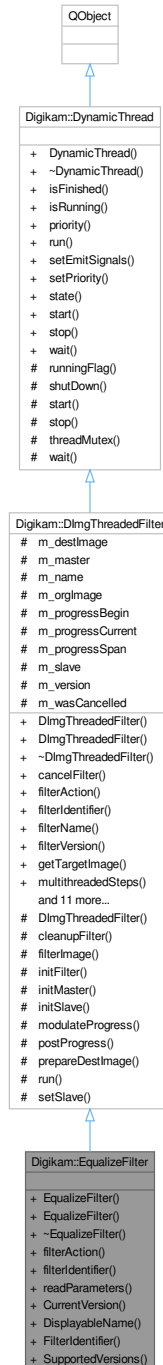
### 9.497.1.7 size()

```
int Digikam::EmptyImageListProvider::size ( ) const [override], [virtual]
```

Implements [Digikam::ImageListProvider](#).

## 9.498 Digikam::EqualizeFilter Class Reference

Inheritance diagram for Digikam::EqualizeFilter:



### Public Member Functions

- **EqualizeFilter** (*DImg* \*const orgImage, const *DImg* \*const reflImage, QObject \*const parent=nullptr)
- **EqualizeFilter** (QObject \*const parent=nullptr)

- [FilterAction filterAction](#) () override  
*Returns the action description corresponding to currently set options.*
- [QString filterIdentifier](#) () const override  
*Return the identifier for this filter in the image history.*
- void [readParameters](#) (const [FilterAction](#) &action) override

## Public Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImg](#) \*const orgImage, [QObject](#) \*const parent, const [QString](#) &name=[QString](#)())  
*Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter](#) ([QObject](#) \*const parent=nullptr, const [QString](#) &name=[QString](#)())  
*Constructs a filter without argument.*
- virtual void [cancelFilter](#) ()  
*Cancel the threaded computation.*
- const [QString](#) & [filterName](#) ()
- int [filterVersion](#) () const
- [DImg](#) [getTargetImage](#) ()
- [QList](#)< int > [multithreadedSteps](#) (int stop, int start=0) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool [parametersSuccessfullyRead](#) () const  
*Optional: error handling for readParameters.*
- virtual [QString](#) [readParametersError](#) (const [FilterAction](#) &actionThatFailed) const
- void [setFilterName](#) (const [QString](#) &name)
- void [setFilterVersion](#) (int version)  
*Replaying a filter action: Set the filter version.*
- void [setOriginalImage](#) (const [DImg](#) &orgImage)
- void [setupAndStartDirectly](#) (const [DImg](#) &orgImage, [DImgThreadedFilter](#) \*const master, int progress←Begin=0, int progressEnd=100)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void [setupFilter](#) (const [DImg](#) &orgImage)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void [startFilter](#) ()  
*Start the threaded computation.*
- virtual void [startFilterDirectly](#) ()  
*Start computation of this filter, directly in this thread.*
- virtual [QList](#)< int > [supportedVersions](#) () const

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread](#) ([QObject](#) \*const parent=nullptr)  
*This class extends [QRunnable](#), so you have to reimplement virtual void [run\(\)](#).*
- [~DynamicThread](#) () override  
*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool [isFinished](#) () const
- bool [isRunning](#) () const
- [QThread::Priority](#) [priority](#) () const
- void [setEmitSignals](#) (bool emitThem)
- void [setPriority](#) ([QThread::Priority](#) priority)  
*Sets the priority for this dynamic thread.*
- State [state](#) () const

### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from Digikam::DImgThreadedFilter

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from Digikam::DynamicThread

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from Digikam::DImgThreadedFilter

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.498.1 Member Function Documentation

### 9.498.1.1 filterAction()

`FilterAction` Digikam::EqualizeFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.498.1.2 filterIdentifier()

`QString` Digikam::EqualizeFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

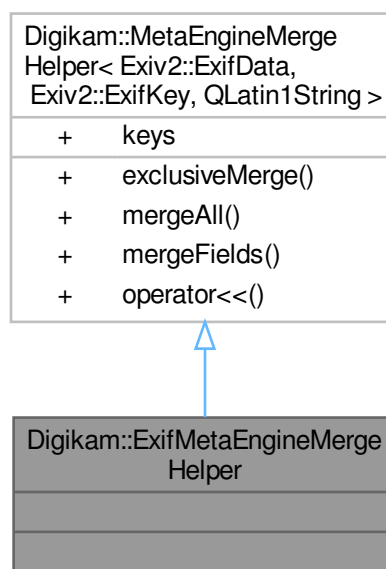
### 9.498.1.3 readParameters()

```
void Digikam::EqualizeFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

## 9.499 Digikam::ExifMetaEngineMergeHelper Class Reference

Inheritance diagram for Digikam::ExifMetaEngineMergeHelper:



## Additional Inherited Members

### Public Member Functions inherited from

[Digikam::MetaEngineMergeHelper](#)< [Exiv2::ExifData](#), [Exiv2::ExifKey](#), [QLatin1String](#) >

- void [exclusiveMerge](#) (const [Exiv2::ExifData](#) &src, [Exiv2::ExifData](#) &dest)  
*Merge two (Exif,IPTC,Xmp) Data packages, the result is stored in dest.*
- void [mergeAll](#) (const [Exiv2::ExifData](#) &src, [Exiv2::ExifData](#) &dest)  
*Merge two (Exif,IPTC,Xmp) Data packages, where the result is stored in dest and fields from src take precedence over existing data from dest.*
- void [mergeFields](#) (const [Exiv2::ExifData](#) &src, [Exiv2::ExifData](#) &dest)  
*Merge two (Exif,IPTC,Xmp) Data packages, the result is stored in dest.*
- [MetaEngineMergeHelper](#) & [operator](#)<< (const [QLatin1String](#) &key)

### Public Attributes inherited from

[Digikam::MetaEngineMergeHelper](#)< [Exiv2::ExifData](#), [Exiv2::ExifKey](#), [QLatin1String](#) >

- [QList](#)< [QLatin1String](#) > **keys**

## 9.500 Digikam::ExifToolBinary Class Reference

Inheritance diagram for Digikam::ExifToolBinary:



### Public Member Functions

- `ExifToolBinary` (`QObject *const parent=nullptr`)



## Public Member Functions inherited from [Digikam::DBinaryIface](#)

- **DBinaryIface** (const QString &binaryName, const QString &minimalVersion, const QString &header, const int headerLine, const QString &projectName, const QString &url, const QString &pluginName, const QStringList &args=QStringList(), const QString &desc=QString())
- **DBinaryIface** (const QString &binaryName, const QString &projectName, const QString &url, const QString &pluginName, const QStringList &args=QStringList(), const QString &desc=QString())
- virtual QString **baseName** () const
- virtual bool **checkDir** ()
- virtual bool **checkDirForPath** (const QString &path)
- const QString & **description** () const
- bool **developmentVersion** () const
- virtual QString **directory** () const
- bool **hasError** () const
- bool **isFound** () const
- bool **isValid** () const
- virtual QString **minimalVersion** () const
- virtual QString **path** () const
- virtual QString **path** (const QString &dir) const
- virtual QString **projectName** () const
- virtual bool **recheckDirectories** ()
- virtual void **setup** (const QString &prev=QString())
- virtual QUrl **url** () const
- const QString & **version** () const
- bool **versionsRight** () const
- bool **versionsRight** (const float) const

## Additional Inherited Members

## Public Slots inherited from [Digikam::DBinaryIface](#)

- virtual void **slotAddPossibleSearchDirectory** (const QString &dir)
- virtual void **slotAddSearchDirectory** (const QString &dir)
- virtual void **slotNavigateAndCheck** ()

## Signals inherited from [Digikam::DBinaryIface](#)

- void **signalBinaryValid** ()
- void **signalSearchDirectoryAdded** (const QString &dir)

## Static Public Member Functions inherited from [Digikam::DBinaryIface](#)

- static QString **goodBaseName** (const QString &b)

## Protected Member Functions inherited from [Digikam::DBinaryIface](#)

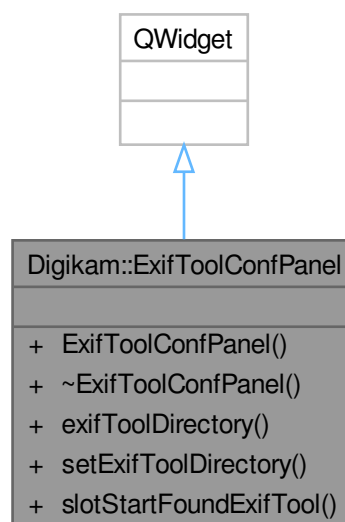
- QString **findHeader** (const QStringList &output, const QString &header) const
- virtual bool **parseHeader** (const QString &output)
- virtual QString **readConfig** ()
- void **setVersion** (QString &version)
- virtual void **writeConfig** ()

## Protected Attributes inherited from [Digikam::DBinaryIface](#)

- const QStringList **m\_binaryArguments**
- const QString **m\_binaryBaseName**
- QLabel \* **m\_binaryLabel** = nullptr
- const bool **m\_checkVersion**
- const QString **m\_configGroup**
- QString **m\_description**
- bool **m\_developmentVersion** = false
- QLabel \* **m\_downloadButton** = nullptr
- bool **m\_hasError** = false
- const int **m\_headerLine**
- const QString **m\_headerStarts**
- bool **m\_isFound** = false
- QLineEdit \* **m\_lineEdit** = nullptr
- const QString **m\_minimalVersion**
- QPushButton \* **m\_pathButton** = nullptr
- QString **m\_pathDir** = QLatin1String("")
- QFrame \* **m\_pathWidget** = nullptr
- const QString **m\_projectName**
- QSet< QString > **m\_searchPaths**
- QLabel \* **m\_statusIcon** = nullptr
- const QUrl **m\_url**
- QString **m\_version** = QLatin1String("")
- QLabel \* **m\_versionLabel** = nullptr

## 9.501 Digikam::ExifToolConfPanel Class Reference

Inheritance diagram for Digikam::ExifToolConfPanel:



### Public Slots

- void **slotStartFoundExifTool** ()

### Signals

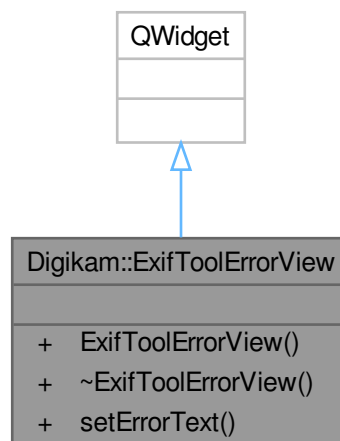
- void **signalExifToolSettingsChanged** (bool available)

### Public Member Functions

- **ExifToolConfPanel** (QWidget \*const parent=nullptr)
- QString **exifToolDirectory** () const
- void **setExifToolDirectory** (const QString &dir)

## 9.502 Digikam::ExifToolErrorView Class Reference

Inheritance diagram for Digikam::ExifToolErrorView:



### Signals

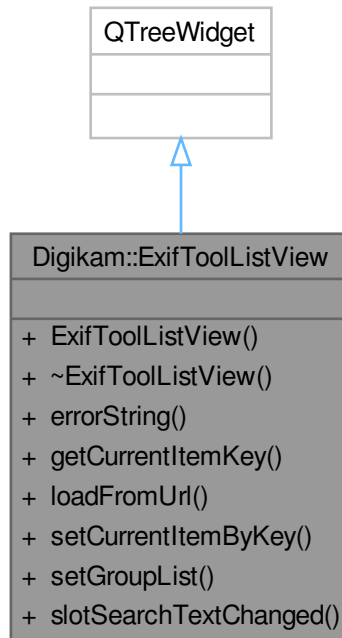
- void **signalSetupExifTool** ()

### Public Member Functions

- **ExifToolErrorView** (QWidget \*const parent)
- void **setErrorText** (const QString &err)

## 9.503 Digikam::ExifToolListView Class Reference

Inheritance diagram for Digikam::ExifToolListView:



### Public Slots

- void **slotSearchTextChanged** (const [SearchTextSettings](#) &)

### Signals

- void **signalLoadingResult** (bool ok)
- void **signalTextFilterMatch** (bool)

### Public Member Functions

- **ExifToolListView** (QWidget \*const parent)
- QString **errorString** () const
- QString **getCurrentItemKey** () const
- void **loadFromUrl** (const QUrl &url)
- void **setCurrentItemByKey** (const QString &itemKey)
- void **setGroupList** (const QStringList &tagsFilter, const QStringList &keysFilter=QStringList())

## 9.503.1 Member Function Documentation

### 9.503.1.1 setGroupList()

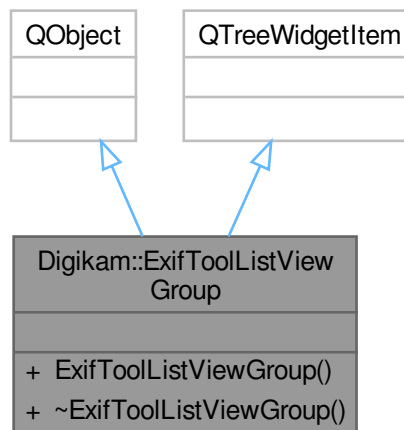
```
void Digikam::ExifToolListView::setGroupList (
    const QStringList & tagsFilter,
    const QStringList & keysFilter = QStringList() )
```

Key is formatted like this:

EXIF.ExifIFD.Image.ExposureCompensation File.File.Other.FileType Composite.Composite.Time.SubSecModify↔  
 Date File.System.Time.FileInodeChangeDate File.System.Other.FileSize EXIF.GPS.Location.GPSLongitude ICC↔  
 \_Profile.ICC-header.Image.ProfileCreator EXIF.IFD1.Image.ThumbnailOffset JFIF.JFIF.Image.YResolution ICC\_↔  
 Profile.ICC\_Profile.Image.GreenMatrixColumn

## 9.504 Digikam::ExifToolListViewGroup Class Reference

Inheritance diagram for Digikam::ExifToolListViewGroup:

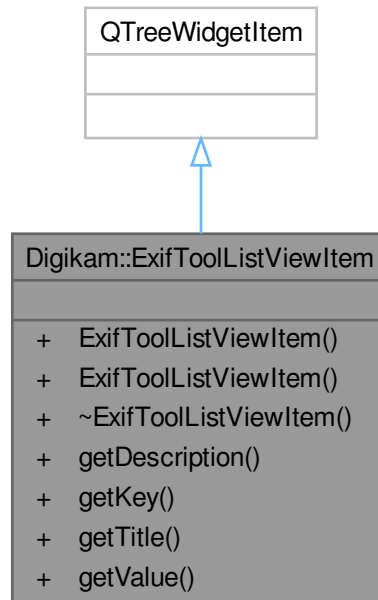


### Public Member Functions

- **ExifToolListViewGroup** (QTreeWidgetItem \*const parent, const QString &group)

## 9.505 Digikam::ExifToolListViewItem Class Reference

Inheritance diagram for Digikam::ExifToolListViewItem:

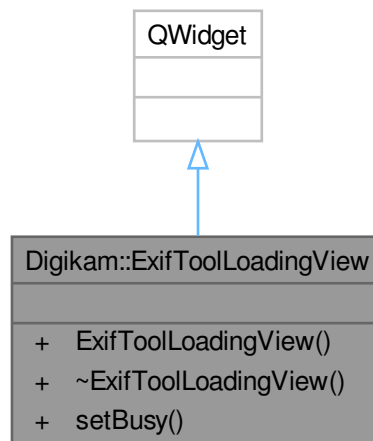


### Public Member Functions

- **ExifToolListViewItem** ([ExifToolListViewGroup](#) \*const parent, const QString &key)
- **ExifToolListViewItem** ([ExifToolListViewGroup](#) \*const parent, const QString &key, const QString &value, const QString &desc)
- QString **getDescription** () const
- QString **getKey** () const
- QString **getTitle** () const
- QString **getValue** () const

## 9.506 Digikam::ExifToolLoadingView Class Reference

Inheritance diagram for Digikam::ExifToolLoadingView:

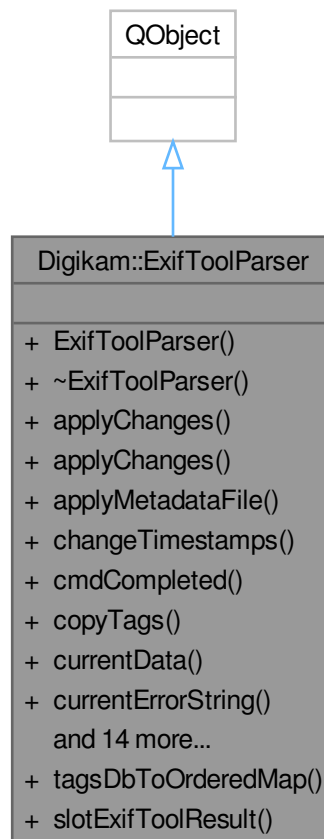


### Public Member Functions

- **ExifToolLoadingView** (`QWidget *const parent`)
- void **setBusy** (`bool b`)

## 9.507 Digikam::ExifToolParser Class Reference

Inheritance diagram for Digikam::ExifToolParser:



### Public Types

- typedef QHash< QString, QVariantList > [ExifToolData](#)

*A map used to store ExifTool data shared with [ExifToolProcess](#) class:*

### Public Slots

- void **slotExifToolResult** (int cmdId)

### Signals

- void **signalExifToolAsyncData** (const [ExifToolParser::ExifToolData](#) &map)
- void **signalExifToolDataAvailable** ()



## Public Member Functions

- **ExifToolParser** (QObject \*const parent, bool async=false)
 

*Constructor, Destructor, and Configuration Accessors. See exiftoolparser.cpp for details.*
- bool **applyChanges** (const QString &path, const **ExifToolData** &newTags)
 

*Apply tag changes to a target file using ExifTool with a list of tag properties.*
- bool **applyChanges** (const QString &path, const QString &exvTempFile, bool hasExif=true, bool hasXmp=true, bool hasCSet=false)
 

*Apply tag changes to a target file using ExifTool with a EXV container.*
- bool **applyMetadataFile** (const QString &path, const QString &meta)
 

*Apply a file with metadata to the target file.*
- bool **changeTimestamps** (const QString &path, const QDateTime &dateTime)
 

*Change all timestamps of the target file using ExifTool.*
- void **cmdCompleted** (const **ExifToolProcess::Result** &result)
 

*ExifTool Output Management Methods. See exiftoolparser\_output.cpp for details.*
- bool **copyTags** (const QString &src, const QString &dst, unsigned char copyOps, unsigned char writeModes=ExifToolProcess::ALL\_MODES)
 

*Copy group of tags from one source file to a destination file, following copy operations defined by 'copyOps'.*
- **ExifToolData** **currentData** () const
- QString **currentErrorString** () const
- QString **currentPath** () const
- void **errorOccurred** (const **ExifToolProcess::Result** &result, QProcess::ProcessError error, const QString &description)
- bool **exifToolAvailable** () const
 

*Check the ExifTool program availability.*
- void **finished** ()
- bool **load** (const QString &path)
 

*ExifTool Command Methods. See exiftoolparser\_command.cpp for details.*
- bool **loadChunk** (const QString &path, bool copyToAll=false)
 

*Load Exif, Iptc, and Xmp chunk as Exiv2 EXV byte-array from a file.*
- bool **readableFormats** ()
 

*Return a list of readable file format extensions.*
- void **setExifToolProgram** (const QString &path)
- void **setOutputStream** (int cmdAction, const QByteArray &cmdOutputChannel, const QByteArray &cmdErrorChannel)
 

*Unit-test method to check ExifTool stream parsing.*
- bool **tagsDatabase** ()
 

*Return a list of all tags from ExifTool database.*
- bool **translateTags** (const QString &path, unsigned char transOps)
 

*Translate group of tags in file.*
- bool **translationsList** ()
 

*Return a list of available translations.*
- bool **version** ()
 

*Return the current version of ExifTool.*
- bool **writableFormats** ()
 

*Return a list of writable file format extensions.*

## Static Public Member Functions

- static **MetaEngine::TagsMap** **tagsDbToOrderedMap** (const **ExifToolData** &tagsDb)
 

*Helper conversion method to translate unordered tags database hash-table to ordered map.*

## 9.507.1 Member Typedef Documentation

### 9.507.1.1 ExifToolData

```
typedef QHash<QString, QVariantList> Digikam::ExifToolParser::ExifToolData
```

With `load()` method, the container is used to get a map of ExifTool tag name as key and tags properties as values: key = ExifTool Tag name (QString - ExifTool Group 0.1.2.4.6) See -G Exiftool option ( [https://exiftool.org/exiftool\\_pod.html#Input-output-text-formatting](https://exiftool.org/exiftool_pod.html#Input-output-text-formatting)). values = ExifTool Tag value (QString). ExifTool Tag type (QString). ExifTool Tag description (QString). ExifTool Tag numerical value (QString) - available if any .

With `loadChunk()` method, the container is used to get a EXV chunk as value: key = "EXV" (QString). value = the Exiv2 metadata container (QByteArray).

With `applyChanges()` method, the container is used as argument to store tuple of ExifTool tag name as key and tag value: key = ExifTool tag name (QString). value = ExifTool Tag value (QString).

With `readableFormats()` method, the container is used to get a list of upper-case file format extensions supported by ExifTool for reading. key = "READ\_FORMAT" (QString). value = list of pairs (ext,desc) (QStringList)

With `writableFormats()` method, the container is used to get a list of upper-case file format extensions supported by ExifTool for writing. key = "WRITE\_FORMAT" (QString). value = list of pairs (ext,desc) (QStringList).

With `translationsList()` method, the container is used to get a list of ExifTool languages available for translations. key = "TRANSLATIONS\_LIST" (QString). value = list of languages as strings (aka fr, en, de, es, etc.) (QStringList).

With `tagsDatabase()` method, the container is used as argument to store tuple of ExifTool tag name as key and tag description: key = ExifTool tag name (QString). values = ExifTool Tag description (QString). ExifTool Tag type (QString). ExifTool Tag writable (QString).

## 9.507.2 Member Function Documentation

### 9.507.2.1 applyChanges() [1/2]

```
bool Digikam::ExifToolParser::applyChanges (
    const QString & path,
    const ExifToolData & newTags )
```

Tags can already exists in target file or new ones can be created. To remove a tag, pass an empty string as value.

#### Parameters

<i>path</i>	is the target files to change.
<i>newTags</i>	is the list of tag properties.

### 9.507.2.2 applyChanges() [2/2]

```
bool Digikam::ExifToolParser::applyChanges (
    const QString & path,
```

```

    const QString & exvTempFile,
    bool hasExif = true,
    bool hasXmp = true,
    bool hasCSet = false )

```

Tags can already exists in target file or new ones can be created.

#### Parameters

<i>path</i>	is the target files to change.
<i>exvTempFile</i>	is the list of changes embedded in EXV container.
<i>hasExif</i>	if the EXV container has Exif metadata restore MarkerNotes.
<i>hasXmp</i>	if the EXV container has Xmp metadata.
<i>hasCSet</i>	if the EXV container has characters set information.

#### 9.507.2.3 applyMetadataFile()

```

bool Digikam::ExifToolParser::applyMetadataFile (
    const QString & path,
    const QString & meta )

```

#### Parameters

<i>path</i>	is the target file to change.
<i>meta</i>	is the metadata file.

#### 9.507.2.4 changeTimestamps()

```

bool Digikam::ExifToolParser::changeTimestamps (
    const QString & path,
    const QDateTime & dateTime )

```

#### Parameters

<i>path</i>	is the target file to change.
<i>dateTime</i>	is the date/time.

#### 9.507.2.5 copyTags()

```

bool Digikam::ExifToolParser::copyTags (
    const QString & src,
    const QString & dst,
    unsigned char copyOps,
    unsigned char writeModes = ExifToolProcess::ALL_MODES )

```

#### Parameters

<i>src</i>	must be a readable file format supported by ExifTool.
------------	---

## Parameters

<i>dst</i>	must be a writable file format supported by ExifTool.
<i>copyOps</i>	is a OR combination of <a href="#">ExifToolProcess::CopyTagsSource</a> values.
<i>writeModes</i>	is a OR combination of <a href="#">ExifToolProcess::WritingTagsMode</a> values.

**9.507.2.6 load()**

```
bool Digikam::ExifToolParser::load (
    const QString & path )
```

Load all metadata with ExifTool from a file. Use `currentData()` to get the ExifTool map.

**9.507.2.7 loadChunk()**

```
bool Digikam::ExifToolParser::loadChunk (
    const QString & path,
    bool copyToAll = false )
```

Use `currentData()` to get the container.

**9.507.2.8 readableFormats()**

```
bool Digikam::ExifToolParser::readableFormats ( )
```

Use `currentData()` to get the container as `QStringList`.

**9.507.2.9 tagsDatabase()**

```
bool Digikam::ExifToolParser::tagsDatabase ( )
```

Use `currentData()` to get the container.

**Warning**

: This method get whole ExifTool database in XML format and take age.

**9.507.2.10 tagsDbToOrderedMap()**

```
MetaEngine::TagsMap Digikam::ExifToolParser::tagsDbToOrderedMap (
    const ExifToolData & tagsDb ) [static]
```

Tag are formatted like this:

```
EXIF.IFD0.Image.XResolution EXIF.IFD0.Image.YCbCrCoefficients EXIF.IFD0.Image.YCbCrPositioning EXIF.IFD0.Image.YCbCrSubSampling EXIF.IFD0.Image.YClipPathUnits EXIF.IFD0.Image.YPosition EXIF.IFD0.Image.YResolution FITS.FITS.Image.Author FITS.FITS.Image.Background FITS.FITS.Image.CreateDate FITS.FITS.Image.Instrument FITS.FITS.Image.Object FITS.FITS.Image.ObservationDate
```

**9.507.2.11 translateTags()**

```
bool Digikam::ExifToolParser::translateTags (
    const QString & path,
    unsigned char transOps )
```

## Parameters

<i>path</i>	must be a readable file format supported by ExifTool.
<i>transOps</i>	is a OR combination of <a href="#">ExifToolProcess::TranslateTagsOps</a> values.

**9.507.2.12 translationsList()**

```
bool Digikam::ExifToolParser::translationsList ( )
```

Use `currentData()` to get the container as `QStringList`.

**9.507.2.13 version()**

```
bool Digikam::ExifToolParser::version ( )
```

Use `currentData()` to get the container as `QString`.

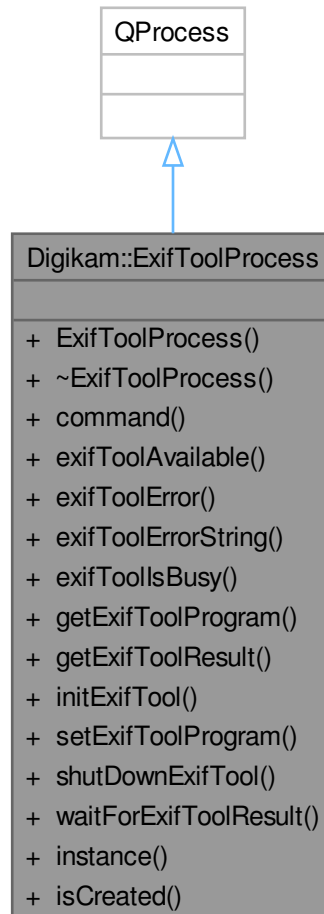
**9.507.2.14 writableFormats()**

```
bool Digikam::ExifToolParser::writableFormats ( )
```

Use `currentData()` to get the container as `QStringList`.

## 9.508 Digikam::ExifToolProcess Class Reference

Inheritance diagram for Digikam::ExifToolProcess:



### Classes

- class [Result](#)

### Public Types

- enum [Action](#) {  
[LOAD\\_METADATA](#) = 0 , [LOAD\\_CHUNKS](#) , [APPLY\\_CHANGES](#) , [APPLY\\_CHANGES\\_EXV](#) ,  
[APPLY\\_METADATA\\_FILE](#) , [CHANGE\\_TIMESTAMPS](#) , [READ\\_FORMATS](#) , [WRITE\\_FORMATS](#) ,  
[TRANSLATIONS\\_LIST](#) , [TAGS\\_DATABASE](#) , [VERSION\\_STRING](#) , [COPY\\_TAGS](#) ,  
[TRANS\\_TAGS](#) , [NO\\_ACTION](#) }

*ExifTool actions to process.*

- enum [CopyTagsSource](#) {  
[COPY\\_EXIF](#) = 0x01 , [COPY\\_MAKERNOTES](#) = 0x02 , [RESTORE\\_PREVIEW](#) = 0x04 , [COPY\\_IPTC](#) = 0x08 ,  
[COPY\\_XMP](#) = 0x10 , [COPY\\_ICC](#) = 0x20 , [COPY\\_ALL](#) = 0x40 , [COPY\\_NONE](#) = 0x80 }  
*Possible copying tags operations to OR combine with COPY\_TAGS action.*
- enum [ResultStatus](#) { [COMMAND\\_RESULT](#) = 0 , [FINISH\\_RESULT](#) , [ERROR\\_RESULT](#) }  
*Command result state.*
- enum [TranslateTagsOps](#) { [TRANS\\_ALL\\_XMP](#) = 0x01 , [TRANS\\_ALL\\_IPTC](#) = 0x02 , [TRANS\\_ALL\\_EXIF](#) = 0x04 }  
*Possible translating tags operations to OR combine with COPY\_TAGS action.*
- enum [WritingTagsMode](#) { [WRITE\\_EXISTING\\_TAGS](#) = 0x01 , [CREATE\\_NEW\\_TAGS](#) = 0x02 , [CREATE\\_NEW\\_GROUPS](#) = 0x04 , [ALL\\_MODES](#) }  
*Possible writing tags mode to OR combine with COPY\_TAGS action.*

## Signals

- void [signalChangeProgram](#) (const QString &etExePath)
- void [signalExecNextCmd](#) ()
- void [signalExifToolResult](#) (int cmdId)

## Public Member Functions

- [ExifToolProcess](#) ()  
*Constructs a [ExifToolProcess](#).*
- [~ExifToolProcess](#) ()  
*Destructs the [ExifToolProcess](#) object, i.e., killing the process.*
- int [command](#) (const QByteArrayList &args, [Action](#) ac)  
*Send a command to exiftool process.*
- bool [exifToolAvailable](#) () const  
*Returns true if [ExifToolProcess](#) is available (process state == Running)*
- [QProcess::ProcessError](#) [exifToolError](#) () const  
*Returns the type of error that occurred last.*
- [QString](#) [exifToolErrorString](#) () const  
*Returns an error message.*
- bool [exifToolsBusy](#) () const  
*Returns true if a command is running.*
- [QString](#) [getExifToolProgram](#) () const
- [ExifToolProcess::Result](#) [getExifToolResult](#) (int cmdId) const  
*Returns the [ExifToolProcess](#) result.*
- void [initExifTool](#) ()  
*Setup connections, apply Settings and start ExifTool process.*
- void [setExifToolProgram](#) (const QString &etExePath)  
*Change the ExifTool path configuration.*
- void [shutDownExifTool](#) ()  
*Attempts to shut down the ExifTool process.*
- [ExifToolProcess::Result](#) [waitForExifToolResult](#) (int cmdId) const  
*WaitCondition for the [ExifToolParser](#) class.*

## Static Public Member Functions

- static [ExifToolProcess](#) \* [instance](#) ()  
*Q\_GLOBAL\_STATIC implementation.*
- static bool [isCreated](#) ()

## 9.508.1 Member Enumeration Documentation

### 9.508.1.1 Action

enum `Digikam::ExifToolProcess::Action`

#### Enumerator

<code>LOAD_METADATA</code>	Load all metadata from a file with ExifTool.
<code>LOAD_CHUNKS</code>	Load Exif, Iptc, and Xmp chunks from a file as byte-array for <a href="#">MetaEngine</a> .
<code>APPLY_CHANGES</code>	Apply tag changes in a file with ExifTool.
<code>APPLY_CHANGES_EXV</code>	Apply tag changes in a file with ExifTool using an EXV container.
<code>APPLY_METADATA_FILE</code>	Apply a metadata file to a file with ExifTool.
<code>CHANGE_TIMESTAMPS</code>	Change all timestamps in a file with ExifTool.
<code>READ_FORMATS</code>	Return the list of readable ExifTool file formats.
<code>WRITE_FORMATS</code>	Return the list of writable ExifTool file formats.
<code>TRANSLATIONS_LIST</code>	List of ExifTool languages available for translations.
<code>TAGS_DATABASE</code>	List of ExifTool tags from database.
<code>VERSION_STRING</code>	Return the ExifTool version as string.
<code>COPY_TAGS</code>	Copy tags from one file to another one. See <code>CopyTagsSource</code> enum for details.
<code>TRANS_TAGS</code>	Translate tags in file. See <code>TranslateTagsOps</code> enum for details.
<code>NO_ACTION</code>	Last value from this list. Do nothing.

### 9.508.1.2 CopyTagsSource

enum `Digikam::ExifToolProcess::CopyTagsSource`

#### Enumerator

<code>COPY_EXIF</code>	Copy all Exif Tags from source file.
<code>COPY_MAKERNOTES</code>	Copy all Makernotes tags from source file.
<code>RESTORE_PREVIEW</code>	Restore preview image from source file.
<code>COPY_IPTC</code>	Copy all Iptc tags from source file.
<code>COPY_XMP</code>	Copy all Xmp tags from source file.
<code>COPY_ICC</code>	Copy ICC profile from source file.
<code>COPY_ALL</code>	Copy all tags from source file.
<code>COPY_NONE</code>	No copy operation.

### 9.508.1.3 TranslateTagsOps

enum `Digikam::ExifToolProcess::TranslateTagsOps`

#### Enumerator

<code>TRANS_ALL_XMP</code>	Translate all existing Tags from source file to Xmp.
<code>TRANS_ALL_IPTC</code>	Translate all existing Tags from source file to Iptc.
<code>TRANS_ALL_EXIF</code>	Translate all existing Tags from source file to Exif.



### 9.508.1.4 WritingTagsMode

```
enum Digikam::ExifToolProcess::WritingTagsMode
```

Enumerator

WRITE_EXISTING_TAGS	Overwrite existing tags.
CREATE_NEW_TAGS	Create new tags.
CREATE_NEW_GROUPS	Create new groups if necessary.

## 9.508.2 Constructor & Destructor Documentation

### 9.508.2.1 ~ExifToolProcess()

```
Digikam::ExifToolProcess::~~ExifToolProcess ( )
```

Note that this function will not return until the process is terminated.

## 9.508.3 Member Function Documentation

### 9.508.3.1 command()

```
int Digikam::ExifToolProcess::command (
    const QByteArrayList & args,
    Action ac )
```

This function can be called from another thread. Return 0: ExifTool not running, write channel is closed or args is empty.

### 9.508.3.2 initExifTool()

```
void Digikam::ExifToolProcess::initExifTool ( )
```

This function cannot be called from another thread.

### 9.508.3.3 setExifToolProgram()

```
void Digikam::ExifToolProcess::setExifToolProgram (
    const QString & etExePath )
```

This function can be called from another thread.

### 9.508.3.4 shutDownExifTool()

```
void Digikam::ExifToolProcess::shutDownExifTool ( )
```

This function cannot be called from another thread.

### 9.508.3.5 waitForExifToolResult()

```
ExifToolProcess::Result Digikam::ExifToolProcess::waitForExifToolResult (
    int cmdId ) const
```

Returns the [ExifToolProcess](#) result.

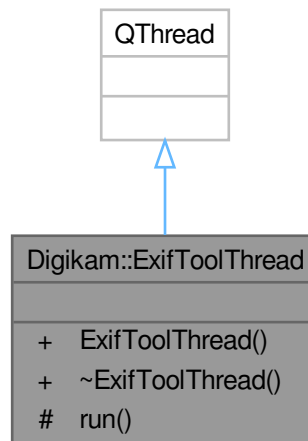
## 9.509 Digikam::ExifToolProcess::Result Class Reference

### Public Attributes

- int **cmdAction** = [ExifToolProcess::NO\\_ACTION](#)
- int **cmdNumber** = 0
- int **cmdStatus** = [ExifToolProcess::COMMAND\\_RESULT](#)
- int **elapsed** = 0
- QByteArray **output**
- bool **waitError** = false

## 9.510 Digikam::ExifToolThread Class Reference

Inheritance diagram for Digikam::ExifToolThread:



### Signals

- void **exifToolProcessStarted** ()

### Public Member Functions

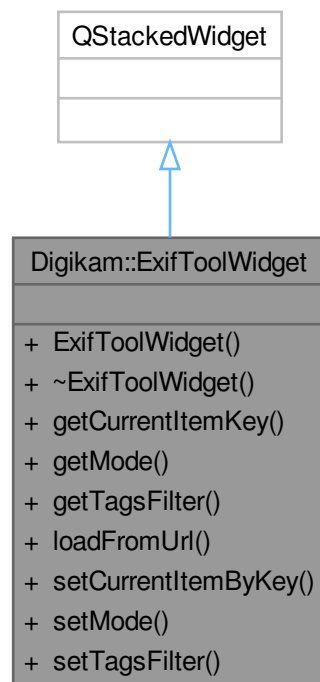
- **ExifToolThread** (QObject \*const parent)

### Protected Member Functions

- void **run** () override  
*Main thread loop.*

## 9.511 Digikam::ExifToolWidget Class Reference

Inheritance diagram for Digikam::ExifToolWidget:



### Public Types

- enum **TagFilters** { `NONE = 0` , `PHOTO` , `CUSTOM` }

### Signals

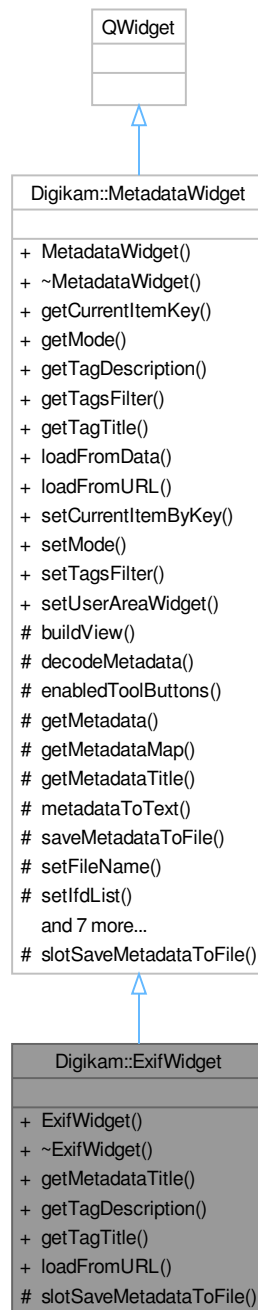
- void **signalSetupExifTool** ()
- void **signalSetupMetadataFilters** ()

**Public Member Functions**

- **ExifToolWidget** (QWidget \*const parent)
- QString **getCurrentItemKey** () const
- int **getMode** () const
- QStringList **getTagsFilter** () const
- void **loadFromUrl** (const QUrl &url)
- void **setCurrentItemByKey** (const QString &itemKey)
- void **setMode** (int mode)
- void **setTagsFilter** (const QStringList &list)

## 9.512 Digikam::ExifWidget Class Reference

Inheritance diagram for Digikam::ExifWidget:



### Public Member Functions

- **ExifWidget** (QWidget \*const parent, const QString &name=QString())
- QString [getMetadataTitle](#) () const override

- QString [getTagDescription](#) (const QString &key) override
- QString [getTagTitle](#) (const QString &key) override
- bool [loadFromURL](#) (const QUrl &url) override

### Public Member Functions inherited from [Digikam::MetadataWidget](#)

- **MetadataWidget** (QWidget \*const parent, const QString &name=QString())
- QString [getCurrentItemKey](#) () const
- int [getMode](#) () const
- QStringList [getTagsFilter](#) () const
- virtual bool [loadFromData](#) (const QString &fileName, const [DMetadadata](#) &data=[DMetadadata](#)())
- void [setCurrentItemByKey](#) (const QString &itemKey)
- void [setMode](#) (int mode)
- void [setTagsFilter](#) (const QStringList &list)
- void [setUserAreaWidget](#) (QWidget \*const w)

### Protected Slots

- void [slotSaveMetadataToFile](#) () override

### Protected Slots inherited from [Digikam::MetadataWidget](#)

- virtual void [slotSaveMetadataToFile](#) ()=0

### Additional Inherited Members

### Public Types inherited from [Digikam::MetadataWidget](#)

- enum [TagFilters](#) { **NONE** = 0 , **PHOTO** , **CUSTOM** }

### Signals inherited from [Digikam::MetadataWidget](#)

- void [signalSetupMetadataFilters](#) ()

### Protected Member Functions inherited from [Digikam::MetadataWidget](#)

- void [enabledToolButtons](#) (bool)
  - [DMetadadata](#) \* [getMetadata](#) () const
  - const [DMetadadata::MetaDatumMap](#) & [getMetadataMap](#) ()
  - QString [metadataToText](#) () const
  - QUrl [saveMetadataToFile](#) (const QString &caption, const QString &fileFilter)
  - void [setFileName](#) (const QString &fileName)
  - void [setIfdList](#) (const [DMetadadata::MetaDatumMap](#) &ifds, const QStringList &keysFilter, const QStringList &tagsFilter)
  - void [setIfdList](#) (const [DMetadadata::MetaDatumMap](#) &ifds, const QStringList &tagsFilter=QStringList())
  - bool [setMetadata](#) (const [DMetadadata](#) &data=[DMetadadata](#)())
  - virtual void [setMetadataEmpty](#) ()
  - void [setMetadataMap](#) (const [DMetadadata::MetaDatumMap](#) &data=[DMetadadata::MetaDatumMap](#)())
  - void [setup](#) ()
- Call this method in children class constructors to init signal/slots connections.*
- bool [storeMetadataToFile](#) (const QUrl &url, const QByteArray &metaData)
  - [MetadataListView](#) \* [view](#) () const

## 9.512.1 Member Function Documentation

### 9.512.1.1 getMetadataTitle()

```
QString Digikam::ExifWidget::getMetadataTitle ( ) const [override], [virtual]
```

Implements [Digikam::MetadataWidget](#).

### 9.512.1.2 getTagDescription()

```
QString Digikam::ExifWidget::getTagDescription (
    const QString & key ) [override], [virtual]
```

Reimplemented from [Digikam::MetadataWidget](#).

### 9.512.1.3 getTagTitle()

```
QString Digikam::ExifWidget::getTagTitle (
    const QString & key ) [override], [virtual]
```

Reimplemented from [Digikam::MetadataWidget](#).

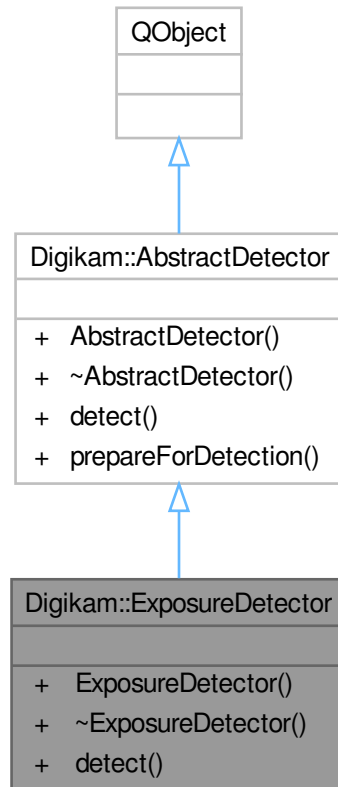
### 9.512.1.4 loadFromURL()

```
bool Digikam::ExifWidget::loadFromURL (
    const QUrl & url ) [override], [virtual]
```

Implements [Digikam::MetadataWidget](#).

## 9.513 Digikam::ExposureDetector Class Reference

Inheritance diagram for Digikam::ExposureDetector:



### Public Member Functions

- float [detect](#) (const cv::Mat &image) const override

### Public Member Functions inherited from [Digikam::AbstractDetector](#)

- **AbstractDetector** (QObject \*const parent=nullptr)

### Additional Inherited Members

### Static Public Member Functions inherited from [Digikam::AbstractDetector](#)

- static cv::Mat **prepareForDetection** (const [DImg](#) &inputImage)

*NOTE: Maybe this function will move to `read_image()` of `imagequalityparser` in case all detectors of IQS use `cv::Mat`.*



## 9.513.1 Member Function Documentation

### 9.513.1.1 detect()

```
float Digikam::ExposureDetector::detect (
    const cv::Mat & image ) const [override], [virtual]
```

Implements [Digikam::AbstractDetector](#).

## 9.514 Digikam::ExposureSettingsContainer Class Reference

### Public Attributes

- bool [exposureIndicatorMode](#) = true  
*If this option is true, over and under exposure indicators will be displayed only when pure white and pure black color matches, as all color components match the condition in the same time.*
- QColor [overExposureColor](#) = Qt::black
- bool [overExposureIndicator](#) = false
- float [overExposurePercent](#) = 1.0
- QColor [underExposureColor](#) = Qt::white
- bool [underExposureIndicator](#) = false
- float [underExposurePercent](#) = 1.0

## 9.514.1 Member Data Documentation

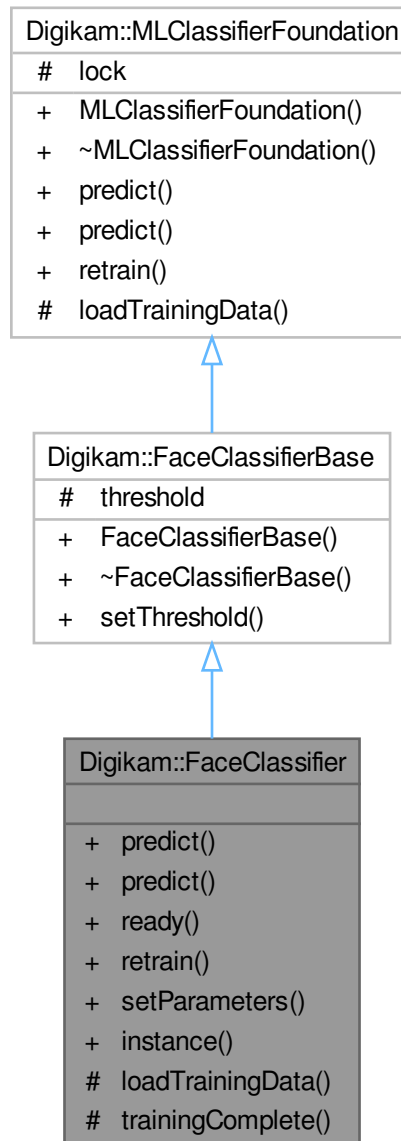
### 9.514.1.1 exposureIndicatorMode

```
bool Digikam::ExposureSettingsContainer::exposureIndicatorMode = true
```

Else indicators are turn on when one of color components match the condition.

## 9.515 Digikam::FaceClassifier Class Reference

Inheritance diagram for Digikam::FaceClassifier:



### Public Member Functions

- int [predict](#) (const cv::Mat &target) const override
- int [predict](#) (const cv::UMat &target) const override
- bool **ready** () const
- bool [retrain](#) () override
- void **setParameters** (const [FaceScanSettings](#) &parameters)

## Public Member Functions inherited from [Digikam::FaceClassifierBase](#)

- void **setThreshold** (float *\_threshold*)

## Static Public Member Functions

- static [FaceClassifier](#) \* **instance** ()

## Protected Member Functions

- bool **loadTrainingData** () override
- void **trainingComplete** ()

## Friends

- class **FaceClassifierCreator**

## Additional Inherited Members

## Protected Attributes inherited from [Digikam::FaceClassifierBase](#)

- float **threshold** = 0.0F

## Protected Attributes inherited from [Digikam::MLClassifierFoundation](#)

- QReadWriteLock **lock**

## 9.515.1 Member Function Documentation

### 9.515.1.1 loadTrainingData()

```
bool Digikam::FaceClassifier::loadTrainingData ( ) [override], [protected], [virtual]
```

Implements [Digikam::MLClassifierFoundation](#).

### 9.515.1.2 predict() [1/2]

```
int Digikam::FaceClassifier::predict (
    const cv::Mat & target ) const [override], [virtual]
```

Implements [Digikam::MLClassifierFoundation](#).

### 9.515.1.3 predict() [2/2]

```
int Digikam::FaceClassifier::predict (
    const cv::UMat & target ) const [override], [virtual]
```

Implements [Digikam::MLClassifierFoundation](#).

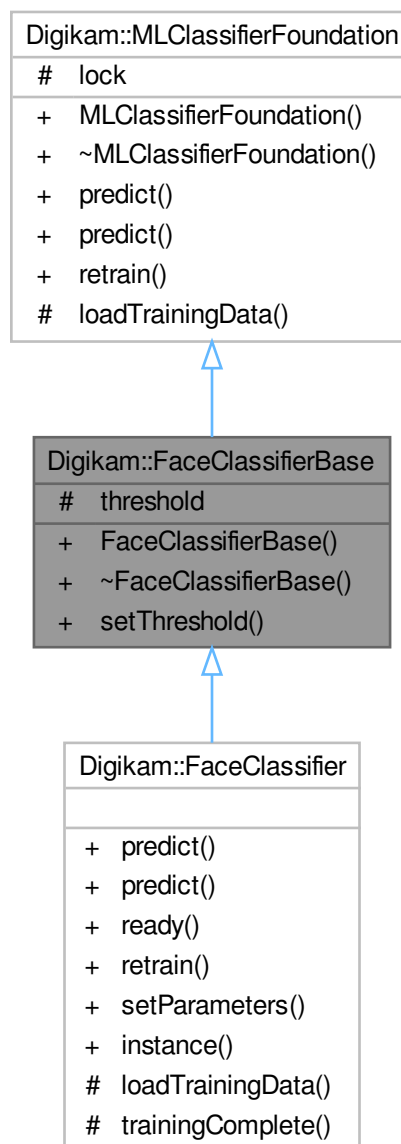
### 9.515.1.4 retrain()

```
bool Digikam::FaceClassifier::retrain ( ) [override], [virtual]
```

Implements [Digikam::MLClassifierFoundation](#).

## 9.516 Digikam::FaceClassifierBase Class Reference

Inheritance diagram for Digikam::FaceClassifierBase:



**Public Member Functions**

- void **setThreshold** (float \_threshold)

**Public Member Functions inherited from [Digikam::MLClassifierFoundation](#)**

- virtual int **predict** (const cv::Mat &target) const =0
- virtual int **predict** (const cv::UMat &target) const =0
- virtual bool **retrain** ()=0

**Protected Attributes**

- float **threshold** = 0.0F

**Protected Attributes inherited from [Digikam::MLClassifierFoundation](#)**

- QReadWriteLock **lock**

**Additional Inherited Members****Protected Member Functions inherited from [Digikam::MLClassifierFoundation](#)**

- virtual bool **loadTrainingData** ()=0

## 9.517 Digikam::FaceDb Class Reference

**Public Member Functions**

- **FaceDb** ([FaceDbBackend](#) \*const db, [FaceScanSettings::FaceRecognitionModel](#) recModel)
- int **addIdentity** () const
- void **clearDNNTraining** ()
  - clearDNNTraining: clear all trained data in the database.*
- void **clearDNNTraining** (const QList< int > &identities)
  - clearDNNTraining: clear*
- void **clearIdentities** ()
- void **deleteIdentity** (const QString &uuid)
- void **deleteIdentity** (int id)
- int **getNumberOfIdentities** () const
- QList< [Identity](#) > **identities** () const
- [Identity](#) **identity** (int id) const
- QList< int > **identityIds** () const
- int **insertFaceVector** (const cv::Mat &faceEmbedding, const int label, const QString &hash) const
  - insertFaceVector: insert a new face embedding to database.*
- bool **integrityCheck** ()
  - Returns true if the integrity of the database is preserved.*
- bool **removeFaceVector** (const int id) const
  - removeFaceVector: remove a face embedding from the database.*
- bool **removeFaceVector** (const QString &hash) const
  - removeFaceVector: remove a face embedding from the database.*
- [BdEngineBackend::QueryState](#) **setSetting** (const QString &keyword, const QString &value)
- QString **setting** (const QString &keyword) const
- cv::Ptr< cv::ml::TrainData > **trainData** () const
  - trainData: extract train data from database.*
- void **updateIdentity** (const [Identity](#) &p)
- void **vacuum** ()
  - Shrinks the database.*

## 9.517.1 Member Function Documentation

### 9.517.1.1 clearDNNTraining()

```
void Digikam::FaceDb::clearDNNTraining (
    const QList< int > & identities )
```

#### Parameters

<i>identities</i>	in the database.
-------------------	------------------

### 9.517.1.2 insertFaceVector()

```
int Digikam::FaceDb::insertFaceVector (
    const cv::Mat & faceEmbedding,
    const int label,
    const QString & hash ) const
```

#### Parameters

<i>faceEmbedding</i>	
<i>label</i>	
<i>hash</i>	

#### Returns

id of newly inserted entry.

### 9.517.1.3 removeFaceVector() [1/2]

```
bool Digikam::FaceDb::removeFaceVector (
    const int id ) const
```

#### Parameters

<i>id</i>	the nodeId (row id) to remove.
-----------	--------------------------------

#### Returns

bool

### 9.517.1.4 removeFaceVector() [2/2]

```
bool Digikam::FaceDb::removeFaceVector (
    const QString & hash ) const
```

## Parameters

<i>hash</i>	the removeHash (removeHash) to remove.
-------------	--

## Returns

bool

## 9.517.1.5 trainData()

```
cv::Ptr< cv::ml::TrainData > Digikam::FaceDb::trainData ( ) const
```

## Returns

the train data instance.

## 9.518 Digikam::FaceDbAccess Class Reference

### Public Member Functions

- [FaceDbAccess](#) ()  
*This class is written in analogy to [CoreDbAccess](#) (some features stripped off).*
- [FaceDbBackend](#) \* **backend** () const
- [FaceDb](#) \* **db** () const
- const QString & **lastError** () const
- void [setLastError](#) (const QString &error)  
*Set the "last error" message.*

### Static Public Member Functions

- static bool **checkReadyForUse** ([InitializationObserver](#) \*const observer)
- static void **cleanUpDatabase** ()
- static void **initDbEngineErrorHandler** ([DbEngineErrorHandler](#) \*const errorHandler)
- static bool **isInitialized** ()
- static [DbEngineParameters](#) **parameters** ()
- static void **setParameters** (const [DbEngineParameters](#) &parameters, [FaceScanSettings::FaceRecognitionModel](#) recognizeModel)

### Friends

- class **FaceDbAccessUnlock**

## 9.518.1 Constructor & Destructor Documentation

### 9.518.1.1 FaceDbAccess()

```
Digikam::FaceDbAccess::FaceDbAccess ( )
```

For documentation, see `coredbaccess.h`

## 9.518.2 Member Function Documentation

### 9.518.2.1 setLastError()

```
void Digikam::FaceDbAccess::setLastError (
    const QString & error )
```

This method is not for public use.

## 9.519 Digikam::FaceDbAccessUnlock Class Reference

### Public Member Functions

- [FaceDbAccessUnlock](#) ()  
*Acquire an object of this class if you want to assure that the [FaceDbAccess](#) is not held during the lifetime of the object.*
- [FaceDbAccessUnlock](#) ([FaceDbAccess](#) \*const access)

## 9.519.1 Constructor & Destructor Documentation

### 9.519.1.1 FaceDbAccessUnlock()

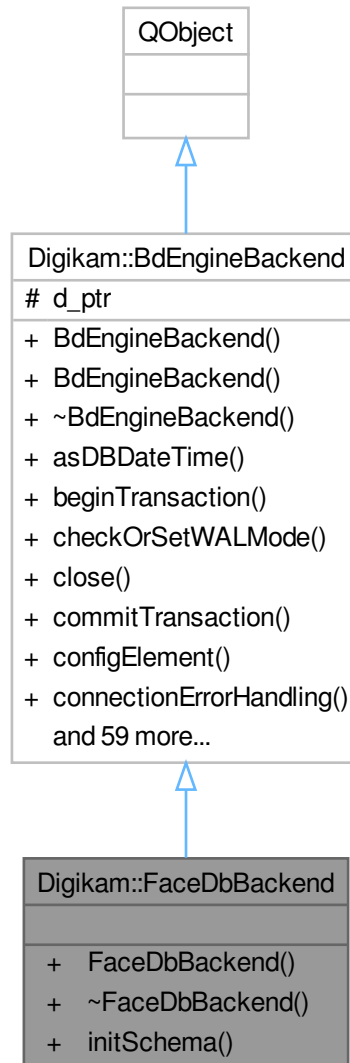
```
Digikam::FaceDbAccessUnlock::FaceDbAccessUnlock ( )
```

At creation, the lock is obtained shortly, then all locks are released. At destruction, all locks are acquired again. If you need to access any locked structures during lifetime, acquire a new [FaceDbAccess](#).



## 9.520 Digikam::FaceDbBackend Class Reference

Inheritance diagram for Digikam::FaceDbBackend:



### Public Member Functions

- **FaceDbBackend** ([DbEngineLocking](#) \*const locking, const QString &backendName=QLatin1String("face← Database-"))
- bool [initSchema](#) ([FaceDbSchemaUpdater](#) \*const updater)

*Initialize the database schema to the current version, carry out upgrades if necessary.*

## Public Member Functions inherited from [Digikam::BdEngineBackend](#)

- [BdEngineBackend](#) (const QString &backendName, [DbEngineLocking](#) \*const locking)
 

*Creates a database backend.*
- **BdEngineBackend** (const QString &backendName, [DbEngineLocking](#) \*const locking, [BdEngineBackend](#)←Private &dd)
- QDateTime [asDBDateTime](#) (const QDateTime &dateTime) const
 

*Depending on the database backend return a local or UTC date format.*
- [BdEngineBackend::QueryState](#) **beginTransaction** ()
 

*Begin a database transaction.*
- bool [checkOrSetWALMode](#) ()
 

*Check or set WAL mode for SQLite database if enabled in settings.*
- void [close](#) ()
 

*Close the database connection.*
- [BdEngineBackend::QueryState](#) **commitTransaction** ()
 

*Commit the current database transaction.*
- [DbEngineConfigSettings](#) **configElement** () const
 

*Return config read from XML, corresponding to this backend's database type.*
- bool [connectionErrorHandling](#) (int retries)
 

*Called when an attempted connection to the database failed.*
- [DbEngineSqlQuery](#) **copyQuery** (const [DbEngineSqlQuery](#) &old)
 

*Creates a faithful copy of the passed query, with the current db connection.*
- DbType **databaseType** () const
 

*Return the database type.*
- bool **exec** ([DbEngineSqlQuery](#) &query)
 

*Calls exec/execBatch on the query, and handles debug output if something went wrong.*
- bool **execBatch** ([DbEngineSqlQuery](#) &query)
- [QueryState](#) **execDBAction** (const [DbEngineAction](#) &action, const QMap< QString, QVariant > &bindingMap, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
 

*Performs the database action on the current database.*
- [QueryState](#) **execDBAction** (const [DbEngineAction](#) &action, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
 

*Performs the database action on the current database.*
- [QueryState](#) **execDBAction** (const QString &action, const QMap< QString, QVariant > &bindingMap, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
 

*Performs the database action on the current database.*
- [QueryState](#) **execDBAction** (const QString &action, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
 

*Performs the database action on the current database.*
- QSqlQuery [execDBActionQuery](#) (const [DbEngineAction](#) &action, const QMap< QString, QVariant > &bindingMap)
 

*Performs the database action on the current database.*
- QSqlQuery **execDBActionQuery** (const QString &action, const QMap< QString, QVariant > &bindingMap)
- [QueryState](#) **execDirectSql** (const QString &query)
 

*Calls exec on the query, and handles debug output if something went wrong.*
- [QueryState](#) **execDirectSqlWithResult** (const QString &query, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
 

*Calls exec on the query, and handles debug output if something went wrong.*
- [DbEngineSqlQuery](#) **execQuery** (const QString &sql)
 

*Executes the statement and returns the query object.*
- [DbEngineSqlQuery](#) **execQuery** (const QString &sql, const QList< QVariant > &boundValues)
- [DbEngineSqlQuery](#) **execQuery** (const QString &sql, const QMap< QString, QVariant > &bindingMap)
 

*Method which accept a hashmap with key, values which are used for named binding.*
- [DbEngineSqlQuery](#) **execQuery** (const QString &sql, const QVariant &boundValue1)

- [DbEngineSqlQuery](#) **execQuery** (const QString &sql, const QVariant &boundValue1, const QVariant &boundValue2)
- [DbEngineSqlQuery](#) **execQuery** (const QString &sql, const QVariant &boundValue1, const QVariant &boundValue2, const QVariant &boundValue3)
- [DbEngineSqlQuery](#) **execQuery** (const QString &sql, const QVariant &boundValue1, const QVariant &boundValue2, const QVariant &boundValue3, const QVariant &boundValue4)
- void **execQuery** ([DbEngineSqlQuery](#) &preparedQuery, const QList< QVariant > &boundValues)
- void **execQuery** ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &boundValue1)
  - Binds the values and executes the prepared query.*
- void **execQuery** ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &boundValue1, const QVariant &boundValue2)
- void **execQuery** ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &boundValue1, const QVariant &boundValue2, const QVariant &boundValue3)
- void **execQuery** ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &boundValue1, const QVariant &boundValue2, const QVariant &boundValue3, const QVariant &boundValue4)
- [QueryState](#) **execSql** (const QString &sql, const QList< QVariant > &boundValues, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** (const QString &sql, const QMap< QString, QVariant > &bindingMap, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
  - Method which accepts a map for named binding.*
- [QueryState](#) **execSql** (const QString &sql, const QVariant &boundValue1, const QVariant &boundValue2, const QVariant &boundValue3, const QVariant &boundValue4, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** (const QString &sql, const QVariant &boundValue1, const QVariant &boundValue2, const QVariant &boundValue3, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** (const QString &sql, const QVariant &boundValue1, const QVariant &boundValue2, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** (const QString &sql, const QVariant &boundValue1, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** (const QString &sql, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
  - Executes the SQL statement, and write the returned data into the values list.*
- [QueryState](#) **execSql** ([DbEngineSqlQuery](#) &preparedQuery, const QList< QVariant > &boundValues, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &boundValue1, const QVariant &boundValue2, const QVariant &boundValue3, const QVariant &boundValue4, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &boundValue1, const QVariant &boundValue2, const QVariant &boundValue3, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &boundValue1, const QVariant &boundValue2, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &boundValue1, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** ([DbEngineSqlQuery](#) &preparedQuery, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execUpsertDBAction** (const [DbEngineAction](#) &action, const QVariant &id, const QStringList &fieldNames, const QList< QVariant > &values)
  - Performs a special DBAction that is usually needed to "INSERT or UPDATE" entries in a table.*
- [QueryState](#) **execUpsertDBAction** (const QString &action, const QVariant &id, const QStringList &fieldNames, const QList< QVariant > &values)
- [DbEngineAction](#) **getDBAction** (const QString &actionName) const
  - Returns a database action with name, specified in actionName, for the current database.*
- [DbEngineSqlQuery](#) **getQuery** ()

- Creates an empty query object waiting for the statement.*

  - [QueryState](#) [handleQueryResult](#) ([DbEngineSqlQuery](#) &query, [QList](#)< [QVariant](#) > \*const values, [QVariant](#) \*const lastInsertId)
- Checks if there was a connection error.*

  - bool [isCompatible](#) (const [DbEngineParameters](#) &parameters)

*Checks if the parameters can be used for this database backend.*
- bool [isInTransaction](#) () const

*Returns if the database is in a different thread in a transaction.*
- bool [isOpen](#) () const
  - bool [isReady](#) () const
  - [QString](#) [lastError](#) ()

*Returns a description of the last error that occurred on this database.*
- [QSqlError](#) [lastSQLError](#) ()

*Returns the last error that occurred on this database.*
- int [maximumBoundValues](#) () const

*Returns the maximum number of bound parameters allowed per query.*
- bool [open](#) (const [DbEngineParameters](#) &parameters)

*Open the database connection.*
- [DbEngineSqlQuery](#) [prepareQuery](#) (const [QString](#) &sql)

*Creates a query object prepared with the statement, waiting for bound values.*
- bool [queryErrorHandling](#) ([DbEngineSqlQuery](#) &query, int retries)

*Called with a failed query.*
- [QList](#)< [QVariant](#) > [readToList](#) ([DbEngineSqlQuery](#) &query)

*Reads data of returned result set into a list which is returned.*
- void [rollbackTransaction](#) ()

*Rollback the current database transaction.*
- void [setDbEngineErrorHandler](#) ([DbEngineErrorHandler](#) \*const handler)

*Add a [DbEngineErrorHandler](#).*
- void [setForeignKeyChecks](#) (bool check)

*Enables or disables FOREIGN\_KEY\_CHECKS for the database.*
- [Status](#) [status](#) () const

*Returns the current status of the database backend.*
- [QStringList](#) [tables](#) ()

*Returns a list with the names of tables in the database.*
- bool [transactionErrorHandling](#) (const [QSqlError](#) &[lastError](#), int retries)

### Additional Inherited Members

### Public Types inherited from [Digikam::BdEngineBackend](#)

- enum [DbType](#) { [SQLite](#) , [MySQL](#) }
- enum [QueryOperationStatus](#) { [ExecuteNormal](#) , [Wait](#) , [AbortQueries](#) }
- enum [QueryStateEnum](#) { [NoErrors](#) , [SQLError](#) , [ConnectionError](#) }
- enum [Status](#) { [Unavailable](#) , [Open](#) , [OpenSchemaChecked](#) }

### Protected Attributes inherited from [Digikam::BdEngineBackend](#)

- [BdEngineBackendPrivate](#) \*const [d\\_ptr](#) = nullptr

## 9.520.1 Member Function Documentation

### 9.520.1.1 initSchema()

```
bool Digikam::FaceDbBackend::initSchema (
    FaceDbSchemaUpdater *const updater )
```

Shall only be called from the thread that called `open()`.

## 9.521 Digikam::FaceDbOperationGroup Class Reference

When you intend to execute a number of write operations to the database, group them while holding a `FaceDbOperationGroup`.

### Public Member Functions

- **FaceDbOperationGroup** ()  
*Retrieve a `FaceDbAccess` object each time when constructing and destructing.*
- **FaceDbOperationGroup** (`FaceDbAccess` \*const dbAccess)  
*Use an existing `FaceDbAccess` object, which must live as long as this object exists.*
- void **allowLift** ()  
*Allows to `lift()`.*
- void **lift** ()  
*This will - if a transaction is held - commit the transaction and acquire a new one.*
- void **resetTime** ()  
*Resets to 0 the time used by `allowLift()`.*
- void **setMaximumTime** (int msec)

### 9.521.1 Detailed Description

For some database systems (SQLite), keeping a transaction across write operations occurring in short time results in enormous speedup (800x). For system that do not need this optimization, this class is a no-op.

## 9.521.2 Member Function Documentation

### 9.521.2.1 allowLift()

```
void Digikam::FaceDbOperationGroup::allowLift ( )
```

The transaction will be lifted if the time set by `setMaximumTime()` has expired.

### 9.521.2.2 lift()

```
void Digikam::FaceDbOperationGroup::lift ( )
```

This may improve concurrent access.

## 9.522 Digikam::FaceDbSchemaUpdater Class Reference

### Public Member Functions

- **FaceDbSchemaUpdater** ([FaceDbAccess](#) \*const dbAccess)
- void **setObserver** ([InitializationObserver](#) \*const observer)
- bool **update** ()

### Static Public Member Functions

- static int **schemaVersion** ()

## 9.523 Digikam::FaceDetector Class Reference

### Public Member Functions

- [FaceDetector](#) ()  
*Provides face detection, that means the process of selecting those regions of a full image which contain face.*
- **FaceDetector** (const [FaceDetector](#) &other)
- QString **backendIdentifier** () const
- QList< QRectF > **detectFaces** (const [DImg](#) &image, const QSize &originalSize=QSize())  
*Scan an image for faces.*
- QList< QRectF > **detectFaces** (const QImage &image, const QSize &originalSize=QSize())  
*Scan an image for faces.*
- QList< QRectF > **detectFaces** (const QString &imagePath)
- [FaceDetector](#) & **operator=** (const [FaceDetector](#) &other)
- QVariantMap **parameters** () const
- int **recommendedImageSize** (const QSize &availableSize=QSize()) const  
*Returns the recommended size if you want to scale images for detection.*
- void **setParameter** (const QString &parameter, const QVariant &value)  
*Tunes backend parameters.*
- void **setParameters** (const QVariantMap &parameters)

### Static Public Member Functions

- static QRect **toAbsoluteRect** (const QRectF &relativeRect, const QSize &size)
- static QList< QRect > **toAbsoluteRects** (const QList< QRectF > &relativeRects, const QSize &size)
- static QRectF **toRelativeRect** (const QRect &absoluteRect, const QSize &size)
- static QList< QRectF > **toRelativeRects** (const QList< QRect > &absoluteRects, const QSize &size)

### 9.523.1 Constructor & Destructor Documentation

#### 9.523.1.1 FaceDetector()

```
Digikam::FaceDetector::FaceDetector ( )
```

This class provides shallow copying The class is fully reentrant (a single object and its copies are not thread-safe). Deferred creation is guaranteed, that means creation of a [FaceDetector](#) object is cheap, the expensive creation of the detection backend is performed when `detectFaces` is called for the first time.

## 9.523.2 Member Function Documentation

### 9.523.2.1 detectFaces() [1/2]

```
QList< QRectF > Digikam::FaceDetector::detectFaces (
    const QImage & image,
    const QSize & originalSize = QSize() )
```

Return a list with regions possibly containing faces. If the image has been downscaled anywhere in the process, provide the original size of the image as this may be of importance in the detection process.

Found faces are returned in relative coordinates.

### 9.523.2.2 detectFaces() [2/2]

```
QList< QRectF > Digikam::FaceDetector::detectFaces (
    const QImage & image,
    const QSize & originalSize = QSize() )
```

Return a list with regions possibly containing faces. If the image has been downscaled anywhere in the process, provide the original size of the image as this may be of importance in the detection process.

Found faces are returned in relative coordinates.

### 9.523.2.3 recommendedImageSize()

```
int Digikam::FaceDetector::recommendedImageSize (
    const QSize & availableSize = QSize() ) const
```

Larger images can be passed, but may be downscaled.

### 9.523.2.4 setParameter()

```
void Digikam::FaceDetector::setParameter (
    const QString & parameter,
    const QVariant & value )
```

Available parameters:

"speed" vs. "accuracy", 0..1, float "sensitivity" vs. "specificity", 0..1, float.

For both pairs: a = 1-b, you can set either. The first pair changes the ROC curve in a trade for computing time. The second pair moves on a given ROC curve towards more false positives, or more missed faces.

## 9.524 Digikam::FaceGroup Class Reference

Inheritance diagram for Digikam::FaceGroup:



### Public Slots

- void `aboutToSetInfo` (const `ItemInfo` &info)  
*Prepares to load a new info.*



- void **addFace** ()  
*Enters a special state where by click + drag a new face can be created.*
- void **markAllAsIgnored** ()  
*Mark all unconfirmed faces as ignored on the current image.*
- void **rejectAll** ()  
*Rejects (clears) all faces on the current image.*
- void **setInfo** (const [ItemInfo](#) &info)  
*Sets the current [ItemInfo](#).*
- void **setVisible** (bool visible)  
*Shows or hides the frames.*
- void **setVisibleItem** ([RegionFrameItem](#) \*item)

### Public Member Functions

- **FaceGroup** ([GraphicsDImgView](#) \*const view)  
*Constructs a new face group, managing [RegionFrameItems](#) for faces of a particular image, displayed on a [GraphicsDImgView](#).*
- bool **acceptsMouseClicked** (const [QPointF](#) &scenePos)
- bool **autoSuggest** () const
- [RegionFrameItem](#) \* **closestItem** (const [QPointF](#) &p, qreal \*const manhattanLength=nullptr) const  
*Returns the item in this group closest to scene position p.*
- void **enterEvent** ([QEvent](#) \*)
- bool **hasUnconfirmed** ()  
*Returns a boolean whether there is at least one unconfirmed face in the group or not.*
- bool **hasVisibleItems** () const
- [ItemInfo](#) **info** () const
- bool **isVisible** () const
- void **itemHoverEnterEvent** ([QGraphicsSceneHoverEvent](#) \*event)
- void **itemHoverLeaveEvent** ([QGraphicsSceneHoverEvent](#) \*event)
- void **itemHoverMoveEvent** ([QGraphicsSceneHoverEvent](#) \*event)
- [QList](#)< [RegionFrameItem](#) \* > **items** () const
- void **leaveEvent** ([QEvent](#) \*)
- void **setAutoSuggest** (bool doAutoSuggest)  
*Auto suggest applies if an image has not been scanned, or an unknown face is registered.*
- void **setShowOnHover** (bool show)  
*Even if visible() is false, show the item under the mouse cursor.*
- bool **showOnHover** () const

### Protected Slots

- void **itemStateChanged** (int)
- void **slotAddItemFinished** (const [QRectF](#) &rect)
- void **slotAddItemMoving** (const [QRectF](#) &rect)
- void **slotAddItemStarted** (const [QPointF](#) &pos)
- void **slotAlbumRenamed** ([Album](#) \*album)
- void **slotAlbumsUpdated** (int type)
- void **slotAssigned** (const [TaggingAction](#) &action, const [ItemInfo](#) &info, const [QVariant](#) &faceIdentifier)
- void **slotCancelAddItem** ()
- void **slotFocusRandomFace** ()
- void **slotIgnored** (const [ItemInfo](#) &info, const [QVariant](#) &faceIdentifier)
- void **slotIgnoredClicked** (const [ItemInfo](#) &info, const [QVariant](#) &faceIdentifier)
- void **slotLabelClicked** (const [ItemInfo](#) &info, const [QVariant](#) &faceIdentifier)
- void **slotRejected** (const [ItemInfo](#) &info, const [QVariant](#) &faceIdentifier)
- void **startAutoSuggest** ()

## Protected Member Functions

- void **applyItemGeometryChanges** ()
- void **clear** ()
- void **load** ()

## Properties

- bool **visible**

## 9.524.1 Member Function Documentation

### 9.524.1.1 aboutToSetInfo

```
void Digikam::FaceGroup::aboutToSetInfo (
    const ItemInfo & info ) [slot]
```

Closes the face group for editing. Pass a null info if about to close.

### 9.524.1.2 closestItem()

```
RegionFrameItem * Digikam::FaceGroup::closestItem (
    const QPointF & p,
    qreal *const manhattanLength = nullptr ) const
```

If given, manhattanLength is set to the manhattan length between p and the closest point of the returned item's bounding rectangle. In particular, if p is inside the item's rectangle, manhattanLength is 0.

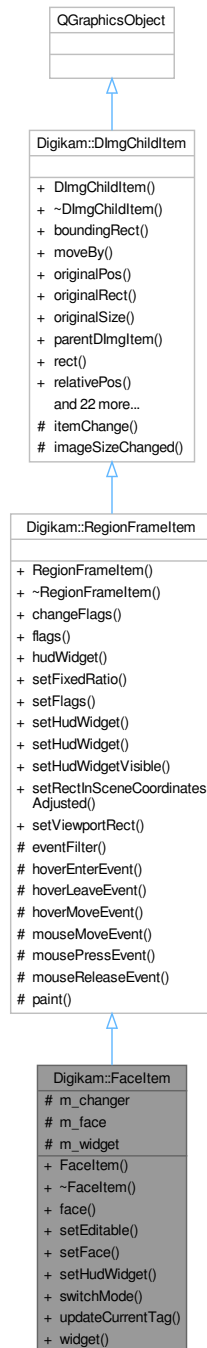
### 9.524.1.3 setAutoSuggest()

```
void Digikam::FaceGroup::setAutoSuggest (
    bool doAutoSuggest )
```

In this case, a new scan will be triggered.

## 9.525 Digikam::Faceltem Class Reference

Inheritance diagram for Digikam::Faceltem:



### Public Member Functions

- **Faceltem** (QGraphicsItem \*const parent)
- [FaceTagsIface](#) **face** () const

- void **setEditable** (bool allowEdit)
- void **setFace** (const [FaceTagsIface](#) &face)
- void **setHudWidget** ([AssignNameWidget](#) \*const widget)
- void **switchMode** ([AssignNameWidget::Mode](#) mode)
- void **updateCurrentTag** ()
- [AssignNameWidget](#) \* **widget** () const

## Public Member Functions inherited from [Digikam::RegionFrameItem](#)

- [RegionFrameItem](#) ([QGraphicsItem](#) \*const parent)
- void **changeFlags** (Flags flags, bool addOrRemove)
- Flags **flags** () const
- [QGraphicsWidget](#) \* **hudWidget** () const
- void **setFixedRatio** (double ratio)
- void **setFlags** (Flags flags)
- void **setHudWidget** ([QGraphicsWidget](#) \*const hudWidget)
 

*Sets a widget item as HUD item.*
- void **setHudWidget** ([QWidget](#) \*const widget, Qt::WindowFlags wFlags=[Qt::WindowFlags](#)())
- void **setHudWidgetVisible** (bool visible)
- void **setRectInSceneCoordinatesAdjusted** (const [QRectF](#) &rect)

## Public Member Functions inherited from [Digikam::DImgChildItem](#)

- [DImgChildItem](#) ([QGraphicsItem](#) \*const parent=nullptr)
 

*This is a base class for items that are positioned on top of a [GraphicsDImgItem](#), positioned in relative coordinates, i.e.*
- [QRectF](#) **boundingRect** () const override
 

*Reimplemented.*
- void **moveBy** (qreal dx, qreal dy)
- [QPoint](#) **originalPos** () const
- [QRect](#) **originalRect** () const
 

*Returns the position and size in coordinates of the original image.*
- [QSize](#) **originalSize** () const
- [GraphicsDImgItem](#) \* **parentDImgItem** () const
 

*If the parent item is a [GraphicsDImgItem](#), return it, if the parent item is null or of a different class, returns 0.*
- [QRectF](#) **rect** () const
 

*Returns position and size of this item, in coordinates of the parent [DImg](#) with the current zoom.*
- [QPointF](#) **relativePos** () const
- [QRectF](#) **relativeRect** () const
 

*Returns the position and size relative to the [DImg](#) displayed in the parent item.*
- [QSizeF](#) **relativeSize** () const
- void **setOriginalPos** (const [QPointF](#) &posInOriginal)
 

*Sets the position and size of this item, in coordinates of the original image.*
- void **setOriginalPos** (qreal x, qreal y)
- void **setOriginalRect** (const [QRectF](#) &rect)
- void **setOriginalRect** (qreal x, qreal y, qreal width, qreal height)
- void **setOriginalSize** (const [QSizeF](#) &sizeInOriginal)
- void **setOriginalSize** (qreal width, qreal height)
- void **setPos** (const [QPointF](#) &zoomedPos)
 

*Sets the position and size of this item, in coordinates of the parent [DImg](#) item.*
- void **setPos** (qreal x, qreal y)
- void **setRect** (const [QRectF](#) &rect)

- void **setRect** (qreal x, qreal y, qreal width, qreal height)
- void **setRectInSceneCoordinates** (const QRectF &rect)
 

*Equivalent to mapping the scene coordinates to the parent item, and calling setRect().*
- void **setRelativePos** (const QPointF &relativePosition)
 

*Sets the position and size of this item, relative to the [DImg](#) displayed in the parent item.*
- void **setRelativePos** (qreal x, qreal y)
- void **setRelativeRect** (const QRectF &rect)
- void **setRelativeRect** (qreal x, qreal y, qreal width, qreal height)
- void **setRelativeSize** (const QSizeF &relativeSize)
- void **setRelativeSize** (qreal width, qreal height)
- void **setSize** (const QSizeF &zoomedSize)
- void **setSize** (qreal width, qreal height)
- QSizeF **size** () const

### Protected Attributes

- [HidingStateChanger](#) \* **m\_changer** = nullptr
- [FaceTagsIface](#) **m\_face**
- [AssignNameWidget](#) \* **m\_widget** = nullptr

### Additional Inherited Members

### Public Types inherited from [Digikam::RegionFrameItem](#)

- enum **Flag** { **NoFlags** = 0 , **ShowResizeHandles** = 1 << 0 , **MoveByDrag** = 1 << 1 , **GeometryEditable** = ShowResizeHandles | MoveByDrag }
- typedef QFlags< Flag > **Flags**

### Public Slots inherited from [Digikam::RegionFrameItem](#)

- void [setViewportRect](#) (const QRectF &rect)
 

*The associated HUD item is dynamically moved to be visible.*

### Signals inherited from [Digikam::RegionFrameItem](#)

- void **geometryEdited** ()

### Signals inherited from [Digikam::DImgChildItem](#)

- void **geometryChanged** ()
- void **geometryOnImageChanged** ()
- void [positionChanged](#) ()
 

*These signals are emitted in any case when the geometry changed: Either after changing the geometry relative to the original image, or when the size of the parent [GraphicsDImgItem](#) changed (zooming).*
- void [positionOnImageChanged](#) ()
 

*These signals are emitted when the geometry, relative to the original image, of this item has changed.*
- void **sizeChanged** ()
- void **sizeOnImageChanged** ()

### Protected Slots inherited from [Digikam::DImgChildItem](#)

- void **imageSizeChanged** (const QSizeF &)

### Protected Member Functions inherited from [Digikam::RegionFrameItem](#)

- bool **eventFilter** (QObject \*watched, QEvent \*event) override
- void **hoverEnterEvent** (QGraphicsSceneHoverEvent \*event) override
- void **hoverLeaveEvent** (QGraphicsSceneHoverEvent \*event) override
- void **hoverMoveEvent** (QGraphicsSceneHoverEvent \*event) override
- void **mouseMoveEvent** (QGraphicsSceneMouseEvent \*) override
- void **mousePressEvent** (QGraphicsSceneMouseEvent \*) override
- void **mouseReleaseEvent** (QGraphicsSceneMouseEvent \*) override
- void **paint** (QPainter \*painter, const QStyleOptionGraphicsItem \*option, QWidget \*widget=nullptr) override

### Protected Member Functions inherited from [Digikam::DImgChildItem](#)

- QVariant **itemChange** (GraphicsItemChange change, const QVariant &value) override

## 9.526 Digikam::FaceltemRetriever Class Reference

### Public Member Functions

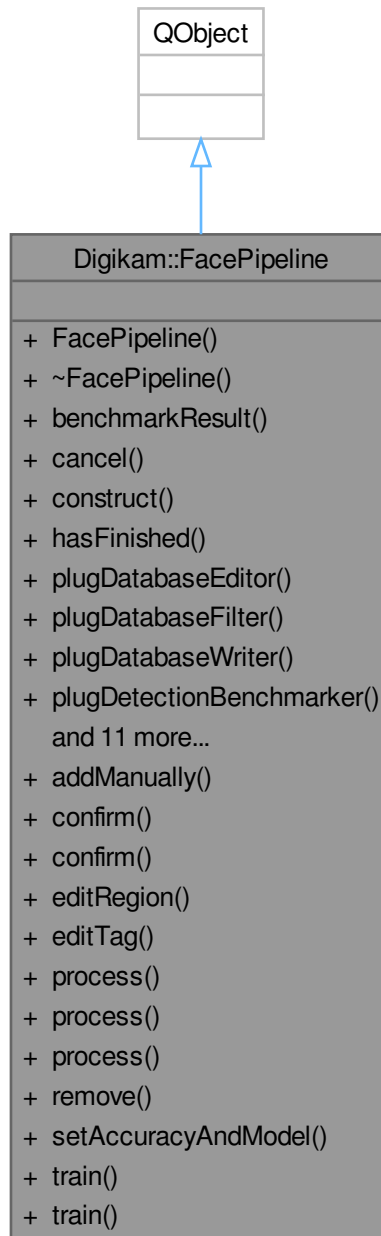
- **FaceltemRetriever** (FacePipeline::Private \*const d)
- void **cancel** ()
- QList< QImage \* > **getDetails** (const [DImg](#) &src, const QList< [FaceTagsIface](#) > &faces) const
- QList< QImage \* > **getDetails** (const [DImg](#) &src, const QList< QRectF > &rects) const
- QList< QImage \* > **getThumbnails** (const QString &filePath, const QList< [FaceTagsIface](#) > &faces) const

### Protected Attributes

- [ThumbnailImageCatcher](#) \* **catcher** = nullptr

## 9.527 Digikam::FacePipeline Class Reference

Inheritance diagram for Digikam::FacePipeline:



### Public Types

- enum `FilterMode` { `ScanAll` , `SkipAlreadyScanned` , `ReadUnconfirmedFaces` , `ReadFacesForTraining` , `ReadConfirmedFaces` }
- enum `WriteMode` { `NormalWrite` , `OverwriteAllFaces` , `OverwriteUnconfirmed` }

## Public Slots

- **FaceTagsIface addManually** (const [ItemInfo](#) &info, const [DImg](#) &image, const [TagRegion](#) &assignedRegion)  
*Add an entry manually.*
- **FaceTagsIface confirm** (const [ItemInfo](#) &info, const [FaceTagsIface](#) &face, const [DImg](#) &image, int assignedTagId=0, const [TagRegion](#) &assignedRegion=[TagRegion](#)())
- **FaceTagsIface confirm** (const [ItemInfo](#) &info, const [FaceTagsIface](#) &face, int assignedTagId=0, const [TagRegion](#) &assignedRegion=[TagRegion](#)())  
*Confirm the face.*
- **FaceTagsIface editRegion** (const [ItemInfo](#) &info, const [DImg](#) &image, const [FaceTagsIface](#) &databaseFace, const [TagRegion](#) &newRegion)  
*Change the given face's region to newRegion.*
- **FaceTagsIface editTag** (const [ItemInfo](#) &info, const [FaceTagsIface](#) &databaseFace, int newTagId)  
*Changes the given face's tagId to newTagId.*
- bool **process** (const [ItemInfo](#) &info)  
*Processes the given image info.*
- bool **process** (const [ItemInfo](#) &info, const [DImg](#) &image)
- void **process** (const QList< [ItemInfo](#) > &infos)  
*Batch processing.*
- void **remove** (const [ItemInfo](#) &info, const [FaceTagsIface](#) &face)  
*Remove the given face.*
- void **setAccuracyAndModel** (int detectAccuracy, [FaceScanSettings::FaceDetectionModel](#) detectModel, [FaceScanSettings::FaceDetectionSize](#) detectSize, int recognizeAccuracy, [FaceScanSettings::FaceRecognitionModel](#) recognizeModel)
- void **train** (const [ItemInfo](#) &info, const QList< [FaceTagsIface](#) > &faces)  
*Train the given faces.*
- void **train** (const [ItemInfo](#) &info, const QList< [FaceTagsIface](#) > &faces, const [DImg](#) &image)

## Signals

- void **finished** ()  
*Emitted when the last package has finished processing.*
- void **processed** (const [FacePipelinePackage](#) &package)  
*Emitted when one package has finished processing.*
- void **processing** (const [FacePipelinePackage](#) &package)  
*Emitted when one package begins processing.*
- void **progressValueChanged** (float progress)
- void **scheduled** ()  
*Emitted when processing is scheduled.*
- void **skipped** (const QList< [ItemInfo](#) > &skippedInfos)  
*Emitted when one or several packages were skipped, usually because they have already been scanned.*
- void **started** (const QString &message)  
*Emitted when processing has started.*



## Public Member Functions

- QString **benchmarkResult** () const
- void **cancel** ()  
 *Cancels all processing.*
- void **construct** ()
- bool **hasFinished** () const
- void **plugDatabaseEditor** ()
- void **plugDatabaseFilter** ([FilterMode](#) mode)  
 *You can plug these four different steps in the working pipeline.*
- void **plugDatabaseWriter** ([WriteMode](#) mode)
- void **plugDetectionBenchmark** ()
- void **plugFaceDetector** ()
- void **plugFacePreviewLoader** ()
- void **plugFaceRecognizer** ()
- void **plugParallelFaceDetectors** ()
- void **plugRecognitionBenchmark** ()
- void **plugRerecognizingDatabaseFilter** ()
- void **plugRetrainingDatabaseFilter** ()
- void **plugTrainer** ()
- QThread::Priority **priority** () const
- void **setPriority** (QThread::Priority priority)  
 *Set the priority of the threads used by this pipeline.*
- void **shutDown** ()  
 *Cancels and waits for the pipeline to finish.*

## Friends

- class **Private**

## 9.527.1 Member Enumeration Documentation

### 9.527.1.1 FilterMode

enum [Digikam::FacePipeline::FilterMode](#)

#### Enumerator

ScanAll	Will read any given image.
SkipAlreadyScanned	Will skip any image that is already marked as scanned.
ReadUnconfirmedFaces	Will read unconfirmed faces for recognition.
ReadFacesForTraining	Will read faces marked for training.
ReadConfirmedFaces	Will read faces which are confirmed.

### 9.527.1.2 WriteMode

enum [Digikam::FacePipeline::WriteMode](#)

## Enumerator

NormalWrite	Write results. Merge with existing entries.
OverwriteAllFaces	Add new results. Previous all results will be cleared.
OverwriteUnconfirmed	Add new results. Previous unconfirmed results will be cleared.

## 9.527.2 Member Function Documentation

### 9.527.2.1 confirm

```
FaceTagsIface Digikam::FacePipeline::confirm (
    const ItemInfo & info,
    const FaceTagsIface & face,
    int assignedTagId = 0,
    const TagRegion & assignedRegion = TagRegion() ) [slot]
```

Pass the original face, and additionally tag id or region if they changed. Returns the confirmed face entry immediately purely for convenience, it is not yet in the database (connect to signal [processed\(\)](#) to react when the processing finished). If a trainer is plugged, the face will be trained.

### 9.527.2.2 editRegion

```
FaceTagsIface Digikam::FacePipeline::editRegion (
    const ItemInfo & info,
    const DImg & image,
    const FaceTagsIface & databaseFace,
    const TagRegion & newRegion ) [slot]
```

Does not care for training atm.

### 9.527.2.3 editTag

```
FaceTagsIface Digikam::FacePipeline::editTag (
    const ItemInfo & info,
    const FaceTagsIface & databaseFace,
    int newTagId ) [slot]
```

Used to Reject Facial Recognition suggestions, since the tag needs to be converted from Unconfirmed to Unknown.

### 9.527.2.4 plugDatabaseFilter()

```
void Digikam::FacePipeline::plugDatabaseFilter (
    FilterMode mode )
```

1) Call any of the four plug...() methods. See below for supported combinations. 2) Call construct() to set up the pipeline.

- Database filter: Prepares database records and/or filters out items. See FilterMode for specification.
- Preview loader: If no preview loader is plugged, you must provide a [DImg](#) for face detection and recognition
- Face Detector: If no recognizer is plugged, all detected face are marked as the unknown person
- Face Recognizer: If no detector is plugged, only already scanned faces marked as unknown will be processed. They are implicitly read from the database.
- [DatabaseWriter](#): Writes the detection and recognition results to the database. The trainer works on a completely different storage and is not affected by the database writer.
- DatabaseEditor: Can confirm or reject faces

PlugParallel: You can call this instead of the simple plugging method. Depending on the number of processor cores of the machine and the memory cost, more than one element may be plugged and process parallelly for this part of the pipeline.

Supported combinations: (Database [Filter](#) ->) (Preview Loader ->) Detector -> Recognizer (-> [DatabaseWriter](#)) (Database [Filter](#) ->) (Preview Loader ->) Detector (-> [DatabaseWriter](#)) (Database [Filter](#) ->) (Preview Loader ->) Recognizer (-> [DatabaseWriter](#)) DatabaseEditor Trainer DatabaseEditor -> Trainer

### 9.527.2.5 process [1/2]

```
bool Digikam::FacePipeline::process (
    const ItemInfo & info ) [slot]
```

If a filter is installed, returns false if the info is skipped, or true if it is processed. If no preview loader is plugged, you must provide a [DImg](#) for detection or recognition. Any of the signals below will only be emitted if true is returned.

### 9.527.2.6 process [2/2]

```
void Digikam::FacePipeline::process (
    const QList< ItemInfo > & infos ) [slot]
```

If a filter is installed, the [skipped\(\)](#) signal will inform about skipped infos. Filtering is done in a thread, returns immediately. Some of the signals below will be emitted in any case.

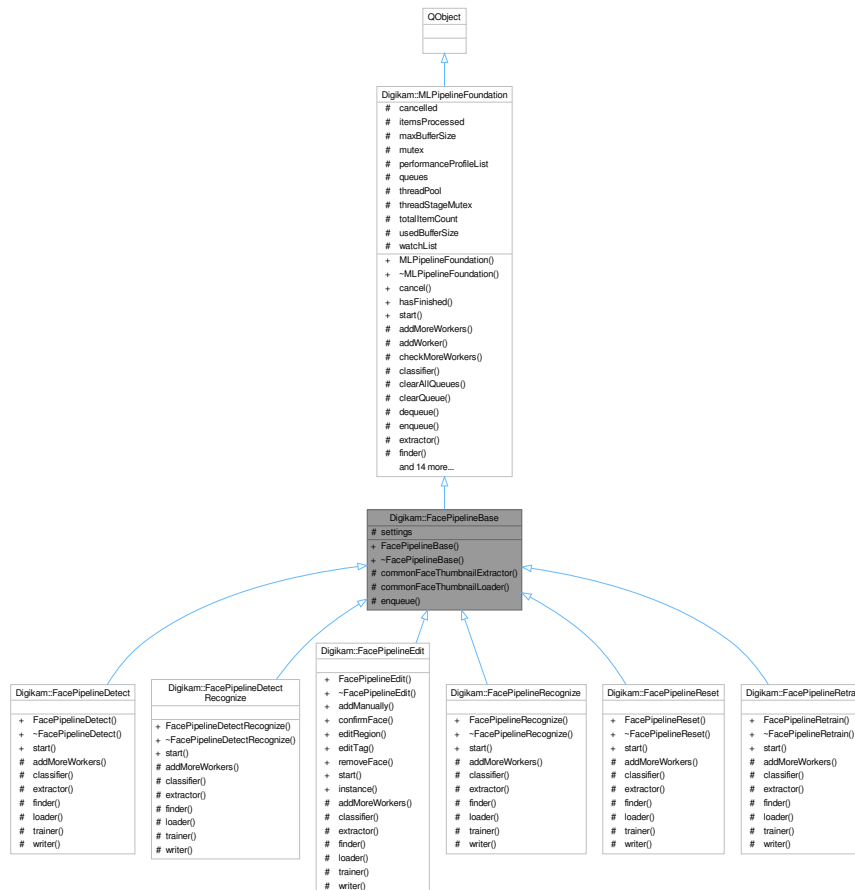
### 9.527.2.7 setPriority()

```
void Digikam::FacePipeline::setPriority (
    QThread::Priority priority )
```

The default setting is QThread::LowPriority.

## 9.528 Digikam::FacePipelineBase Class Reference

Inheritance diagram for Digikam::FacePipelineBase:



### Public Types

- enum [FilterMode](#) { [ScanAll](#) , [ScanNew](#) , [TrainNew](#) , [TrainAll](#) , [TrainRemove](#) , [TrainReset](#) }
- enum [WriteMode](#) { [NormalWrite](#) , [OverwriteAllFaces](#) , [OverwriteUnconfirmed](#) }

### Public Types inherited from [Digikam::MLPipelineFoundation](#)

- enum [MLPipelineNotification](#) { [notifySkipped](#) , [notifyProcessed](#) }
- typedef struct [Digikam::MLPipelineFoundation::\\_MLPipelinePerformanceProfile](#) [MLPipelinePerformanceProfile](#)
- typedef [SharedQueue](#)< [MLPipelinePackageFoundation](#) \* > [MLPipelineQueue](#)
- enum [MLPipelineStage](#) { [Finder](#) , [Loader](#) , [Extractor](#) , [Classifier](#) , [Trainer](#) , [Writer](#) , [None](#) }

## Public Member Functions

- **FacePipelineBase** (const [FaceScanSettings](#) &\_settings)

## Public Member Functions inherited from [Digikam::MLPipelineFoundation](#)

- virtual void [cancel](#) ()
- bool [hasFinished](#) () const
- virtual bool [start](#) ()

## Protected Member Functions

- bool [commonFaceThumbnailExtractor](#) (const QString &pipelineName, [MLPipelineFoundation::MLPipelineStage](#) thisStage, [MLPipelineFoundation::MLPipelineStage](#) nextStage)
- bool [commonFaceThumbnailLoader](#) (const QString &pipelineName, [MLPipelineFoundation::MLPipelineStage](#) thisStage, [MLPipelineFoundation::MLPipelineStage](#) nextStage)
- bool [enqueue](#) ([MLPipelineQueue](#) \*thisQueue, [MLPipelinePackageFoundation](#) \*package) override

## Protected Member Functions inherited from [Digikam::MLPipelineFoundation](#)

- virtual void [addMoreWorkers](#) ()=0
- bool [addWorker](#) (const [MLPipelineStage](#) &stage)
- bool [checkMoreWorkers](#) (int totalItemCount, int currentItemCount, bool useFullCpu)
- virtual bool [classifier](#) ()=0
- void [clearAllQueues](#) ()
- void [clearQueue](#) ([MLPipelineQueue](#) \*thisQueue)
- virtual [MLPipelinePackageFoundation](#) \* [dequeue](#) ([MLPipelineQueue](#) \*thisQueue)
- virtual bool [extractor](#) ()=0
- virtual bool [finder](#) ()=0
- virtual bool [loader](#) ()=0
- void [notify](#) ([MLPipelineNotification](#) notification, const QString &\_name, const QString &\_path, int \_processed, const [DImg](#) &\_thumbnail)
- void [notify](#) ([MLPipelineNotification](#) notification, const QString &\_name, const QString &\_path, int \_processed, const [QIcon](#) &\_thumbnail)
- void [notify](#) ([MLPipelineNotification](#) notification, const QString &\_name, const QString &\_path, int \_processed, const [QImage](#) &\_thumbnail)
- void [pipelinePerformanceEnd](#) (const [MLPipelineStage](#) &stage, int totalItemCount, [QElapsedTimer](#) &timer)
- void [pipelinePerformanceEnd](#) (const [MLPipelineStage](#) &stage, [QElapsedTimer](#) &timer)
- void [pipelinePerformanceStart](#) (const [MLPipelineStage](#) &stage, [QElapsedTimer](#) &timer)
- [MLPipelinePackageFoundation](#) \* [queueEndSignal](#) () const
- void [showPipelinePerformance](#) () const
- void [stageEnd](#) ([MLPipelineStage](#) thisStage, [MLPipelineStage](#) nextStage)
- void [stageStart](#) ([QThread::Priority](#) threadPriority, [MLPipelineStage](#) thisStage, [MLPipelineStage](#) nextStage, [MLPipelineQueue](#) \*&thisQueue, [MLPipelineQueue](#) \*&nextQueue)
- virtual bool [trainer](#) ()=0
- void [waitForStart](#) ()
- virtual bool [writer](#) ()=0

## Protected Attributes

- [FaceScanSettings](#) [settings](#)

## Protected Attributes inherited from [Digikam::MLPipelineFoundation](#)

- bool **cancelled** = false
- QAtomicInteger< int > **itemsProcessed** = 0
- quint64 **maxBufferSize** = 2147483648  
*2 GB default*
- QMutex **mutex**
- QMap< [MLPipelineStage](#), [MLPipelinePerformanceProfile](#) > **performanceProfileList**
- QMap< [MLPipelineStage](#), [MLPipelineQueue](#) \* > **queues**
- QThreadPool \* **threadPool** = nullptr
- QMutex **threadStageMutex**
- QAtomicInteger< int > **totalItemCount** = 0
- quint64 **usedBufferSize** = 0
- QList< QFutureWatcher< bool > \* > **watchList**

## Additional Inherited Members

## Signals inherited from [Digikam::MLPipelineFoundation](#)

- void **finished** ()  
*Emitted when the last package has finished processing.*
- void **processed** (const [MLPipelinePackageNotify::Ptr](#) &package)  
*Emitted when one package has finished processing.*
- void **processing** (const [MLPipelinePackageNotify::Ptr](#) &package)  
*Emitted when one package begins processing.*
- void **progressValueChanged** (float progress)
- void **scheduled** ()  
*Emitted when processing is scheduled.*
- void **signalAddMoreWorkers** ()
- void **signalUpdateItemCount** (const qlonglong itemCount)
- void **skipped** (const [MLPipelinePackageNotify::Ptr](#) &package)  
*Emitted when one or several packages were skipped, usually because they have already been scanned.*
- void **started** (const QString &message)  
*Emitted when processing has started.*

## 9.528.1 Member Enumeration Documentation

### 9.528.1.1 FilterMode

enum [Digikam::FacePipelineBase::FilterMode](#)

#### Enumerator

ScanAll	Will read any given image.
ScanNew	Scan new images, will skip any image that is already marked as scanned.
TrainNew	Adds new face(s) to training.
TrainAll	Retrains the face DB.
TrainRemove	Removes the face(s) from training.
TrainReset	Removes all face training, sets all images to not scanned.

### 9.528.1.2 WriteMode

enum [Digikam::FacePipelineBase::WriteMode](#)

#### Enumerator

NormalWrite	Write results. Merge with existing entries.
OverwriteAllFaces	Add new results. Previous all results will be cleared.
OverwriteUnconfirmed	Add new results. Previous unconfirmed results will be cleared.

## 9.528.2 Member Function Documentation

### 9.528.2.1 enqueue()

```
bool Digikam::FacePipelineBase::enqueue (
    MLPipelineQueue * thisQueue,
    MLPipelinePackageFoundation * package ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::MLPipelineFoundation](#).

## 9.529 Digikam::FacePipelineDetect Class Reference

Inheritance diagram for Digikam::FacePipelineDetect:



### Public Member Functions

- **FacePipelineDetect** (const [FaceScanSettings](#) &\_settings)
- bool [start](#) () override



## Public Member Functions inherited from Digikam::FacePipelineBase

- **FacePipelineBase** (const [FaceScanSettings](#) &\_settings)

## Public Member Functions inherited from Digikam::MLPipelineFoundation

- virtual void [cancel](#) ()
- bool [hasFinished](#) () const

## Protected Member Functions

- void [addMoreWorkers](#) () override
- bool [classifier](#) () override
- bool [extractor](#) () override
- bool [finder](#) () override
- bool [loader](#) () override
- bool [trainer](#) () override
- bool [writer](#) () override

## Protected Member Functions inherited from Digikam::FacePipelineBase

- bool [commonFaceThumbnailExtractor](#) (const QString &pipelineName, [MLPipelineFoundation::MLPipelineStage](#) thisStage, [MLPipelineFoundation::MLPipelineStage](#) nextStage)
- bool [commonFaceThumbnailLoader](#) (const QString &pipelineName, [MLPipelineFoundation::MLPipelineStage](#) thisStage, [MLPipelineFoundation::MLPipelineStage](#) nextStage)
- bool [enqueue](#) ([MLPipelineQueue](#) \*thisQueue, [MLPipelinePackageFoundation](#) \*package) override

## Protected Member Functions inherited from Digikam::MLPipelineFoundation

- bool [addWorker](#) (const [MLPipelineStage](#) &stage)
- bool [checkMoreWorkers](#) (int totalItemCount, int currentItemCount, bool useFullCpu)
- void [clearAllQueues](#) ()
- void [clearQueue](#) ([MLPipelineQueue](#) \*thisQueue)
- virtual [MLPipelinePackageFoundation](#) \* [dequeue](#) ([MLPipelineQueue](#) \*thisQueue)
- void [notify](#) ([MLPipelineNotification](#) notification, const QString &\_name, const QString &\_path, int \_processed, const [DImg](#) &\_thumbnail)
- void [notify](#) ([MLPipelineNotification](#) notification, const QString &\_name, const QString &\_path, int \_processed, const [QIcon](#) &\_thumbnail)
- void [notify](#) ([MLPipelineNotification](#) notification, const QString &\_name, const QString &\_path, int \_processed, const [QImage](#) &\_thumbnail)
- void [pipelinePerformanceEnd](#) (const [MLPipelineStage](#) &stage, int totalItemCount, [QElapsedTimer](#) &timer)
- void [pipelinePerformanceEnd](#) (const [MLPipelineStage](#) &stage, [QElapsedTimer](#) &timer)
- void [pipelinePerformanceStart](#) (const [MLPipelineStage](#) &stage, [QElapsedTimer](#) &timer)
- [MLPipelinePackageFoundation](#) \* [queueEndSignal](#) () const
- void [showPipelinePerformance](#) () const
- void [stageEnd](#) ([MLPipelineStage](#) thisStage, [MLPipelineStage](#) nextStage)
- void [stageStart](#) ([QThread::Priority](#) threadPriority, [MLPipelineStage](#) thisStage, [MLPipelineStage](#) nextStage, [MLPipelineQueue](#) \*&thisQueue, [MLPipelineQueue](#) \*&nextQueue)
- void [waitForStart](#) ()

### Additional Inherited Members

#### Public Types inherited from [Digikam::FacePipelineBase](#)

- enum [FilterMode](#) { [ScanAll](#) , [ScanNew](#) , [TrainNew](#) , [TrainAll](#) , [TrainRemove](#) , [TrainReset](#) }
- enum [WriteMode](#) { [NormalWrite](#) , [OverwriteAllFaces](#) , [OverwriteUnconfirmed](#) }

#### Public Types inherited from [Digikam::MLPipelineFoundation](#)

- enum [MLPipelineNotification](#) { [notifySkipped](#) , [notifyProcessed](#) }
- typedef struct [Digikam::MLPipelineFoundation::\\_MLPipelinePerformanceProfile](#) [MLPipelinePerformanceProfile](#)
- typedef [SharedQueue](#)< [MLPipelinePackageFoundation](#) \* > [MLPipelineQueue](#)
- enum [MLPipelineStage](#) { [Finder](#) , [Loader](#) , [Extractor](#) , [Classifier](#) , [Trainer](#) , [Writer](#) , [None](#) }

#### Signals inherited from [Digikam::MLPipelineFoundation](#)

- void **finished** ()  
*Emitted when the last package has finished processing.*
- void **processed** (const [MLPipelinePackageNotify::Ptr](#) &package)  
*Emitted when one package has finished processing.*
- void **processing** (const [MLPipelinePackageNotify::Ptr](#) &package)  
*Emitted when one package begins processing.*
- void **progressValueChanged** (float progress)
- void **scheduled** ()  
*Emitted when processing is scheduled.*
- void **signalAddMoreWorkers** ()
- void **signalUpdateItemCount** (const qlonglong itemCount)
- void **skipped** (const [MLPipelinePackageNotify::Ptr](#) &package)  
*Emitted when one or several packages were skipped, usually because they have already been scanned.*
- void **started** (const [QString](#) &message)  
*Emitted when processing has started.*

#### Protected Attributes inherited from [Digikam::FacePipelineBase](#)

- [FaceScanSettings](#) **settings**

#### Protected Attributes inherited from [Digikam::MLPipelineFoundation](#)

- bool **cancelled** = false
- [QAtomicInteger](#)< int > **itemsProcessed** = 0
- quint64 **maxBufferSize** = 2147483648  
*2 GB default*
- [QMutex](#) **mutex**
- [QMap](#)< [MLPipelineStage](#), [MLPipelinePerformanceProfile](#) > **performanceProfileList**
- [QMap](#)< [MLPipelineStage](#), [MLPipelineQueue](#) \* > **queues**
- [QThreadPool](#) \* **threadPool** = nullptr
- [QMutex](#) **threadStageMutex**
- [QAtomicInteger](#)< int > **totalItemCount** = 0
- quint64 **usedBufferSize** = 0
- [QList](#)< [QFutureWatcher](#)< bool > \* > **watchList**

## 9.529.1 Member Function Documentation

### 9.529.1.1 addMoreWorkers()

```
void Digikam::FacePipelineDetect::addMoreWorkers ( ) [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

### 9.529.1.2 classifier()

```
bool Digikam::FacePipelineDetect::classifier ( ) [inline], [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

### 9.529.1.3 extractor()

```
bool Digikam::FacePipelineDetect::extractor ( ) [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

### 9.529.1.4 finder()

```
bool Digikam::FacePipelineDetect::finder ( ) [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

### 9.529.1.5 loader()

```
bool Digikam::FacePipelineDetect::loader ( ) [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

### 9.529.1.6 start()

```
bool Digikam::FacePipelineDetect::start ( ) [override], [virtual]
```

Reimplemented from [Digikam::MLPipelineFoundation](#).

### 9.529.1.7 trainer()

```
bool Digikam::FacePipelineDetect::trainer ( ) [inline], [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

### 9.529.1.8 writer()

```
bool Digikam::FacePipelineDetect::writer ( ) [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

## 9.530 Digikam::FacePipelineDetectRecognize Class Reference

Inheritance diagram for Digikam::FacePipelineDetectRecognize:



**Public Member Functions**

- **FacePipelineDetectRecognize** (const [FaceScanSettings](#) &\_settings)
- bool **start** () override

**Public Member Functions inherited from [Digikam::FacePipelineBase](#)**

- **FacePipelineBase** (const [FaceScanSettings](#) &\_settings)

**Public Member Functions inherited from [Digikam::MLPipelineFoundation](#)**

- virtual void **cancel** ()
- bool **hasFinished** () const

**Protected Member Functions**

- void **addMoreWorkers** () override
- bool **classifier** () override
- bool **extractor** () override
- bool **finder** () override
- bool **loader** () override
- bool **trainer** () override
- bool **writer** () override

**Protected Member Functions inherited from [Digikam::FacePipelineBase](#)**

- bool **commonFaceThumbnailExtractor** (const QString &pipelineName, [MLPipelineFoundation::MLPipelineStage](#) thisStage, [MLPipelineFoundation::MLPipelineStage](#) nextStage)
- bool **commonFaceThumbnailLoader** (const QString &pipelineName, [MLPipelineFoundation::MLPipelineStage](#) thisStage, [MLPipelineFoundation::MLPipelineStage](#) nextStage)
- bool **enqueue** ([MLPipelineQueue](#) \*thisQueue, [MLPipelinePackageFoundation](#) \*package) override

**Protected Member Functions inherited from [Digikam::MLPipelineFoundation](#)**

- bool **addWorker** (const [MLPipelineStage](#) &stage)
- bool **checkMoreWorkers** (int totalItemCount, int currentItemCount, bool useFullCpu)
- void **clearAllQueues** ()
- void **clearQueue** ([MLPipelineQueue](#) \*thisQueue)
- virtual [MLPipelinePackageFoundation](#) \* **dequeue** ([MLPipelineQueue](#) \*thisQueue)
- void **notify** ([MLPipelineNotification](#) notification, const QString &\_name, const QString &\_path, int \_processed, const [DImg](#) &\_thumbnail)
- void **notify** ([MLPipelineNotification](#) notification, const QString &\_name, const QString &\_path, int \_processed, const [QIcon](#) &\_thumbnail)
- void **notify** ([MLPipelineNotification](#) notification, const QString &\_name, const QString &\_path, int \_processed, const [QImage](#) &\_thumbnail)
- void **pipelinePerformanceEnd** (const [MLPipelineStage](#) &stage, int totalItemCount, [QElapsedTimer](#) &timer)
- void **pipelinePerformanceEnd** (const [MLPipelineStage](#) &stage, [QElapsedTimer](#) &timer)
- void **pipelinePerformanceStart** (const [MLPipelineStage](#) &stage, [QElapsedTimer](#) &timer)
- [MLPipelinePackageFoundation](#) \* **queueEndSignal** () const
- void **showPipelinePerformance** () const
- void **stageEnd** ([MLPipelineStage](#) thisStage, [MLPipelineStage](#) nextStage)
- void **stageStart** ([QThread::Priority](#) threadPriority, [MLPipelineStage](#) thisStage, [MLPipelineStage](#) nextStage, [MLPipelineQueue](#) \*&thisQueue, [MLPipelineQueue](#) \*&nextQueue)
- void **waitForStart** ()

## Additional Inherited Members

### Public Types inherited from [Digikam::FacePipelineBase](#)

- enum [FilterMode](#) { [ScanAll](#) , [ScanNew](#) , [TrainNew](#) , [TrainAll](#) , [TrainRemove](#) , [TrainReset](#) }
- enum [WriteMode](#) { [NormalWrite](#) , [OverwriteAllFaces](#) , [OverwriteUnconfirmed](#) }

### Public Types inherited from [Digikam::MLPipelineFoundation](#)

- enum [MLPipelineNotification](#) { [notifySkipped](#) , [notifyProcessed](#) }
- typedef struct [Digikam::MLPipelineFoundation::\\_MLPipelinePerformanceProfile](#) [MLPipelinePerformanceProfile](#)
- typedef [SharedQueue< MLPipelinePackageFoundation \\* >](#) [MLPipelineQueue](#)
- enum [MLPipelineStage](#) { [Finder](#) , [Loader](#) , [Extractor](#) , [Classifier](#) , [Trainer](#) , [Writer](#) , [None](#) }

### Signals inherited from [Digikam::MLPipelineFoundation](#)

- void [finished](#) ()  
*Emitted when the last package has finished processing.*
- void [processed](#) (const [MLPipelinePackageNotify::Ptr](#) &package)  
*Emitted when one package has finished processing.*
- void [processing](#) (const [MLPipelinePackageNotify::Ptr](#) &package)  
*Emitted when one package begins processing.*
- void [progressValueChanged](#) (float progress)
- void [scheduled](#) ()  
*Emitted when processing is scheduled.*
- void [signalAddMoreWorkers](#) ()
- void [signalUpdateItemCount](#) (const qlonglong itemCount)
- void [skipped](#) (const [MLPipelinePackageNotify::Ptr](#) &package)  
*Emitted when one or several packages were skipped, usually because they have already been scanned.*
- void [started](#) (const QString &message)  
*Emitted when processing has started.*

### Protected Attributes inherited from [Digikam::FacePipelineBase](#)

- [FaceScanSettings](#) [settings](#)

### Protected Attributes inherited from [Digikam::MLPipelineFoundation](#)

- bool [cancelled](#) = false
- [QAtomicInteger< int >](#) [itemsProcessed](#) = 0
- [quint64](#) [maxBufferSize](#) = 2147483648  
*2 GB default*
- [QMutex](#) [mutex](#)
- [QMap< MLPipelineStage, MLPipelinePerformanceProfile >](#) [performanceProfileList](#)
- [QMap< MLPipelineStage, MLPipelineQueue \\* >](#) [queues](#)
- [QThreadPool](#) \* [threadPool](#) = nullptr
- [QMutex](#) [threadStageMutex](#)
- [QAtomicInteger< int >](#) [totalItemCount](#) = 0
- [quint64](#) [usedBufferSize](#) = 0
- [QList< QFutureWatcher< bool > \\* >](#) [watchList](#)

## 9.530.1 Member Function Documentation

### 9.530.1.1 addMoreWorkers()

```
void Digikam::FacePipelineDetectRecognize::addMoreWorkers ( ) [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

### 9.530.1.2 classifier()

```
bool Digikam::FacePipelineDetectRecognize::classifier ( ) [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

### 9.530.1.3 extractor()

```
bool Digikam::FacePipelineDetectRecognize::extractor ( ) [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

### 9.530.1.4 finder()

```
bool Digikam::FacePipelineDetectRecognize::finder ( ) [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

### 9.530.1.5 loader()

```
bool Digikam::FacePipelineDetectRecognize::loader ( ) [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

### 9.530.1.6 start()

```
bool Digikam::FacePipelineDetectRecognize::start ( ) [override], [virtual]
```

Reimplemented from [Digikam::MLPipelineFoundation](#).

### 9.530.1.7 trainer()

```
bool Digikam::FacePipelineDetectRecognize::trainer ( ) [inline], [override], [protected],  
[virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

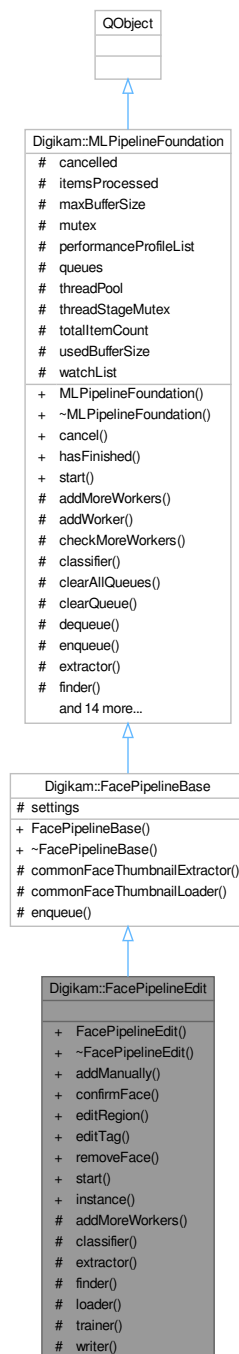
### 9.530.1.8 writer()

```
bool Digikam::FacePipelineDetectRecognize::writer ( ) [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

## 9.531 Digikam::FacePipelineEdit Class Reference

Inheritance diagram for Digikam::FacePipelineEdit:





### Public Member Functions

- [FaceTagsIface](#) **addManually** (const [ItemInfo](#) &info, const [DImg](#) &image, const [TagRegion](#) &region, bool retrain=true)
- [FaceTagsIface](#) **confirmFace** (const [ItemInfo](#) &info, const [FaceTagsIface](#) &face, int tagId, bool retrain=true)
- [FaceTagsIface](#) **editRegion** (const [ItemInfo](#) &info, const [FaceTagsIface](#) &face, const [TagRegion](#) &region, const [DImg](#) &image, bool retrain=true)
- [FaceTagsIface](#) **editTag** (const [ItemInfo](#) &info, const [FaceTagsIface](#) &face, int newTagId)
- void **removeFace** (const [ItemInfo](#) &info, const [FaceTagsIface](#) &face)
- bool **start** () override

### Public Member Functions inherited from [Digikam::FacePipelineBase](#)

- [FacePipelineBase](#) (const [FaceScanSettings](#) &\_settings)

### Public Member Functions inherited from [Digikam::MLPipelineFoundation](#)

- virtual void **cancel** ()
- bool **hasFinished** () const

### Static Public Member Functions

- static [FacePipelineEdit](#) \* **instance** ()

### Protected Member Functions

- void **addMoreWorkers** () override
- bool **classifier** () override
- bool **extractor** () override
- bool **finder** () override
- bool **loader** () override
- bool **trainer** () override
- bool **writer** () override

### Protected Member Functions inherited from [Digikam::FacePipelineBase](#)

- bool **commonFaceThumbnailExtractor** (const QString &pipelineName, [MLPipelineFoundation::MLPipelineStage](#) thisStage, [MLPipelineFoundation::MLPipelineStage](#) nextStage)
- bool **commonFaceThumbnailLoader** (const QString &pipelineName, [MLPipelineFoundation::MLPipelineStage](#) thisStage, [MLPipelineFoundation::MLPipelineStage](#) nextStage)
- bool **enqueue** ([MLPipelineQueue](#) \*thisQueue, [MLPipelinePackageFoundation](#) \*package) override

## Protected Member Functions inherited from [Digikam::MLPipelineFoundation](#)

- bool **addWorker** (const [MLPipelineStage](#) &stage)
- bool **checkMoreWorkers** (int totalItemCount, int currentItemCount, bool useFullCpu)
- void **clearAllQueues** ()
- void **clearQueue** ([MLPipelineQueue](#) \*thisQueue)
- virtual [MLPipelinePackageFoundation](#) \* **dequeue** ([MLPipelineQueue](#) \*thisQueue)
- void **notify** (MLPipelineNotification notification, const QString &\_name, const QString &\_path, int \_processed, const [DImg](#) &\_thumbnail)
- void **notify** (MLPipelineNotification notification, const QString &\_name, const QString &\_path, int \_processed, const [QIcon](#) &\_thumbnail)
- void **notify** (MLPipelineNotification notification, const QString &\_name, const QString &\_path, int \_processed, const [QImage](#) &\_thumbnail)
- void **pipelinePerformanceEnd** (const [MLPipelineStage](#) &stage, int totalItemCount, [QElapsedTimer](#) &timer)
- void **pipelinePerformanceEnd** (const [MLPipelineStage](#) &stage, [QElapsedTimer](#) &timer)
- void **pipelinePerformanceStart** (const [MLPipelineStage](#) &stage, [QElapsedTimer](#) &timer)
- [MLPipelinePackageFoundation](#) \* **queueEndSignal** () const
- void **showPipelinePerformance** () const
- void **stageEnd** ([MLPipelineStage](#) thisStage, [MLPipelineStage](#) nextStage)
- void **stageStart** ([QThread::Priority](#) threadPriority, [MLPipelineStage](#) thisStage, [MLPipelineStage](#) nextStage, [MLPipelineQueue](#) \*&thisQueue, [MLPipelineQueue](#) \*&nextQueue)
- void **waitForStart** ()

## Additional Inherited Members

## Public Types inherited from [Digikam::FacePipelineBase](#)

- enum [FilterMode](#) {  
[ScanAll](#) , [ScanNew](#) , [TrainNew](#) , [TrainAll](#) ,  
[TrainRemove](#) , [TrainReset](#) }
- enum [WriteMode](#) { [NormalWrite](#) , [OverwriteAllFaces](#) , [OverwriteUnconfirmed](#) }

## Public Types inherited from [Digikam::MLPipelineFoundation](#)

- enum [MLPipelineNotification](#) { [notifySkipped](#) , [notifyProcessed](#) }
- typedef struct [Digikam::MLPipelineFoundation::\\_MLPipelinePerformanceProfile](#) [MLPipelinePerformanceProfile](#)
- typedef [SharedQueue](#)< [MLPipelinePackageFoundation](#) \* > [MLPipelineQueue](#)
- enum [MLPipelineStage](#) {  
[Finder](#) , [Loader](#) , [Extractor](#) , [Classifier](#) ,  
[Trainer](#) , [Writer](#) , [None](#) }

## Signals inherited from [Digikam::MLPipelineFoundation](#)

- void **finished** ()  
*Emitted when the last package has finished processing.*
- void **processed** (const [MLPipelinePackageNotify::Ptr](#) &package)  
*Emitted when one package has finished processing.*
- void **processing** (const [MLPipelinePackageNotify::Ptr](#) &package)  
*Emitted when one package begins processing.*
- void **progressValueChanged** (float progress)

- void **scheduled** ()  
*Emitted when processing is scheduled.*
- void **signalAddMoreWorkers** ()
- void **signalUpdateItemCount** (const qlonglong itemCount)
- void **skipped** (const MLPipelinePackageNotify::Ptr &package)  
*Emitted when one or several packages were skipped, usually because they have already been scanned.*
- void **started** (const QString &message)  
*Emitted when processing has started.*

## Protected Attributes inherited from [Digikam::FacePipelineBase](#)

- [FaceScanSettings](#) **settings**

## Protected Attributes inherited from [Digikam::MLPipelineFoundation](#)

- bool **cancelled** = false
- QAtomicInteger< int > **itemsProcessed** = 0
- quint64 **maxBufferSize** = 2147483648  
*2 GB default*
- QMutex **mutex**
- QMap< [MLPipelineStage](#), [MLPipelinePerformanceProfile](#) > **performanceProfileList**
- QMap< [MLPipelineStage](#), [MLPipelineQueue](#) \* > **queues**
- QThreadPool \* **threadPool** = nullptr
- QMutex **threadStageMutex**
- QAtomicInteger< int > **totalItemCount** = 0
- quint64 **usedBufferSize** = 0
- QList< QFutureWatcher< bool > \* > **watchList**

### 9.531.1 Member Function Documentation

#### 9.531.1.1 addMoreWorkers()

```
void Digikam::FacePipelineEdit::addMoreWorkers ( ) [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

#### 9.531.1.2 classifier()

```
bool Digikam::FacePipelineEdit::classifier ( ) [inline], [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

#### 9.531.1.3 extractor()

```
bool Digikam::FacePipelineEdit::extractor ( ) [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

**9.531.1.4 finder()**

```
bool Digikam::FacePipelineEdit::finder ( ) [inline], [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

**9.531.1.5 loader()**

```
bool Digikam::FacePipelineEdit::loader ( ) [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

**9.531.1.6 start()**

```
bool Digikam::FacePipelineEdit::start ( ) [override], [virtual]
```

Reimplemented from [Digikam::MLPipelineFoundation](#).

**9.531.1.7 trainer()**

```
bool Digikam::FacePipelineEdit::trainer ( ) [inline], [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

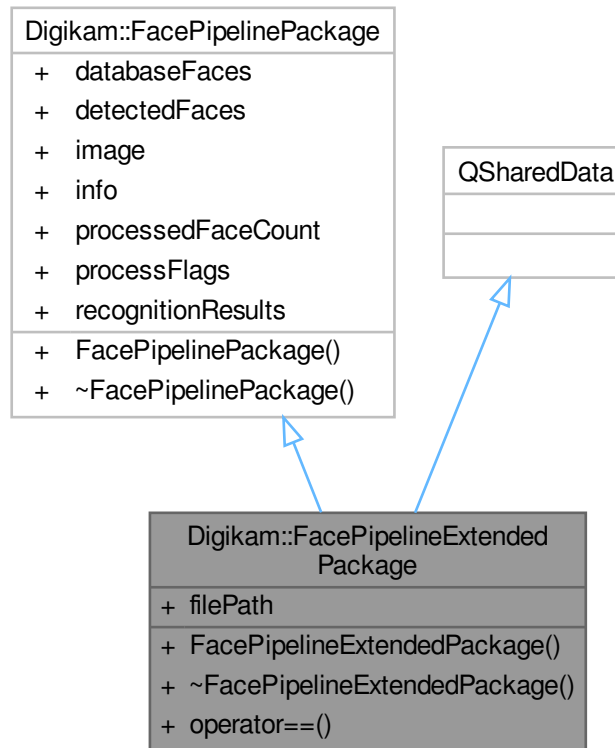
**9.531.1.8 writer()**

```
bool Digikam::FacePipelineEdit::writer ( ) [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

## 9.532 Digikam::FacePipelineExtendedPackage Class Reference

Inheritance diagram for Digikam::FacePipelineExtendedPackage:



### Public Types

- typedef `QExplicitlySharedDataPointer< FacePipelineExtendedPackage >` **Ptr**

### Public Types inherited from [Digikam::FacePipelinePackage](#)

- enum **ProcessFlag** {  
**NotProcessed** = 0 , **PreviewImageLoaded** = 1 << 0 , **ProcessedByDetector** = 1 << 1 , **ProcessedByRecognizer** = 1 << 2 ,  
**WrittenToDatabase** = 1 << 3 , **ProcessedByTrainer** = 1 << 4 }
- typedef `QFlags< ProcessFlag >` **ProcessFlags**

### Public Member Functions

- bool **operator==** (const [LoadingDescription](#) &description) const

**Public Attributes**

- QString **filePath**

**Public Attributes inherited from [Digikam::FacePipelinePackage](#)**

- [FacePipelineFaceTagsIfaceList](#) **databaseFaces**
- QList< QRectF > **detectedFaces**
- [DImg](#) **image**
- [ItemInfo](#) **info**
- int **processedFaceCount** = 0
- ProcessFlags **processFlags** = NotProcessed
- QList< [Identity](#) > **recognitionResults**

## 9.533 Digikam::FacePipelineFaceTagsIface Class Reference

Inheritance diagram for Digikam::FacePipelineFaceTagsIface:



### Public Types

- enum [Role](#) {
  - NoRole** = 0 , **GivenAsArgument** = 1 << 0 , **ReadFromDatabase** = 1 << 1 , **DetectedFromImage** = 1 <<

- 2 ,
- [ForRecognition](#) = 1 << 10 , [ForConfirmation](#) = 1 << 11 , [ForTraining](#) = 1 << 12 , [ForEditing](#) = 1 << 13
- ,
- [Confirmed](#) = 1 << 20 , [Trained](#) = 1 << 21 , [Edited](#) = 1 << 22 }
- typedef QFlags< [Role](#) > [Roles](#)

## Public Types inherited from [Digikam::FaceTagsIface](#)

- enum [Type](#) {
- [InvalidFace](#) = 0 , [UnknownName](#) = 1 << 0 , [UnconfirmedName](#) = 1 << 1 , [IgnoredName](#) = 1 << 2 ,
- [ConfirmedName](#) = 1 << 3 , [FaceForTraining](#) = 1 << 4 , [UnconfirmedTypes](#) = UnknownName |
- UnconfirmedName , [NormalFaces](#) = UnknownName | UnconfirmedName | IgnoredName | ConfirmedName
- ,
- [AllTypes](#) = UnknownName | UnconfirmedName | IgnoredName | ConfirmedName | FaceForTraining , [Type↔](#)
- [First](#) = UnknownName , [TypeLast](#) = FaceForTraining }
- typedef QFlags< [Type](#) > [TypeFlags](#)

## Public Member Functions

- [FacePipelineFaceTagsIface](#) (const [FaceTagsIface](#) &face)
- [FacePipelineFaceTagsIface](#) & [operator=](#) (const [FacePipelineFaceTagsIface](#) &other)

## Public Member Functions inherited from [Digikam::FaceTagsIface](#)

- [FaceTagsIface](#) (const [FaceTagsIface](#) &other)
- [FaceTagsIface](#) (const QString &attribute, qlonglong imageld, int tagId, const [TagRegion](#) &region)
- [FaceTagsIface](#) ([Type](#) type, qlonglong imageld, int tagId, const [TagRegion](#) &region)
- QString [getAutodetectedPersonString](#) () const
- Returns the string tagId + ',' + unconfirmedFace + ',' + regionXml.*
- const QString [hash](#) () const
- Generate a hash based on the imageld, tagId, and rect to uniquely identify this entry in the face training DB.*
- qlonglong [imageld](#) () const
- bool [isConfirmedName](#) () const
- bool [isForTraining](#) () const
- bool [isIgnoredName](#) () const
- bool [isInvalidFace](#) () const
- bool [isNull](#) () const
- bool [isUnconfirmedName](#) () const
- bool [isUnconfirmedType](#) () const
- bool [isUnknownName](#) () const
- [FaceTagsIface](#) & [operator=](#) (const [FaceTagsIface](#) &other)
- bool [operator==](#) (const [FaceTagsIface](#) &other) const
- [TagRegion](#) [region](#) () const
- void [removeFaceTraining](#) () const
- Remove the face from face training based on the current imageld, tagId, and rect hash.*
- void [setRegion](#) (const [TagRegion](#) &region)
- void [setTagId](#) (int tagId)
- void [setType](#) ([Type](#) type)
- int [tagId](#) () const
- QVariant [toVariant](#) () const
- [Type](#) [type](#) () const



## Public Attributes

- [TagRegion](#) **assignedRegion**
- int **assignedTagId** = 0
- Roles **roles** = NoRole

## Additional Inherited Members

## Static Public Member Functions inherited from [Digikam::FaceTagsIface](#)

- static QString **attributeForType** (Type type)  
*Return the corresponding image tag property for the given type.*
- static QStringList **attributesForFlags** (TypeFlags flags)  
*Returns a list of all image tag properties for which flags are set.*
- static [FaceTagsIface](#) **fromListing** (qulonglong imageid, const QList< QVariant > &values)  
*Create a [FaceTagsIface](#) from the extraValues returned from [ItemLister](#).*
- static [FaceTagsIface](#) **fromVariant** (const QVariant &var)  
*Writes the contents of this face - in a compact way - in the QVariant.*
- static Type **typeForAttribute** (const QString &attribute, int tagId=0)  
*Return the Type for the given attribute.*
- static Type **typeForId** (int tagId)  
*Returns the Face Type corresponding to the given TagId.*

## Protected Attributes inherited from [Digikam::FaceTagsIface](#)

- qulonglong **m\_imageid** = 0
- [TagRegion](#) **m\_region**
- int **m\_tagId** = 0
- Type **m\_type** = InvalidFace

## 9.533.1 Member Enumeration Documentation

### 9.533.1.1 Role

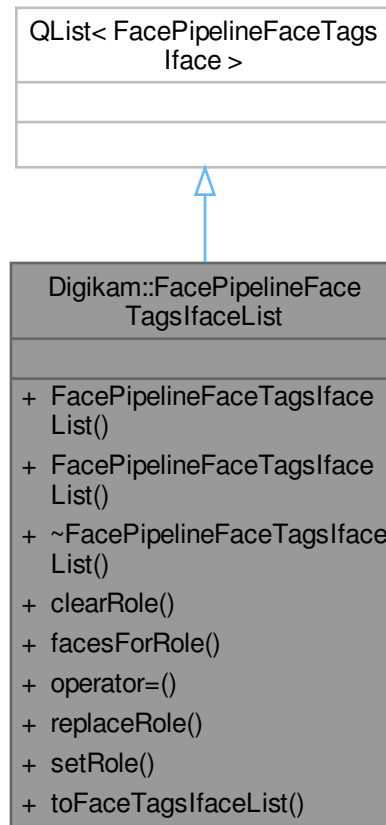
```
enum Digikam::FacePipelineFaceTagsIface::Role
```

#### Enumerator

GivenAsArgument	Source.
ForRecognition	Task.
Confirmed	Executed action (task is cleared).

## 9.534 Digikam::FacePipelineFaceTagsIfaceList Class Reference

Inheritance diagram for Digikam::FacePipelineFaceTagsIfaceList:

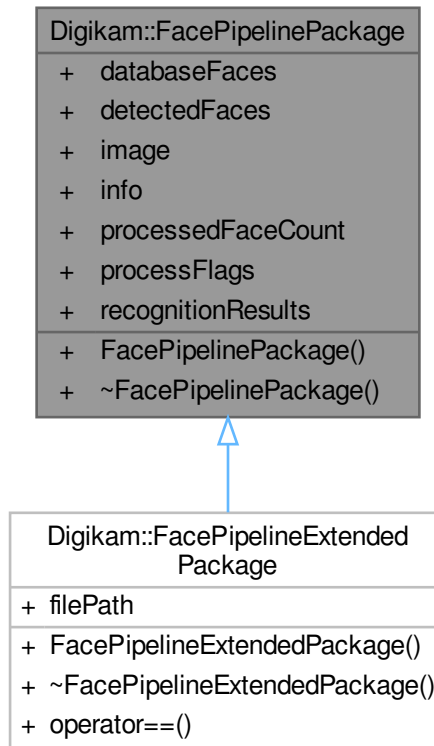


### Public Member Functions

- **FacePipelineFaceTagsIfaceList** (const QList< [FaceTagsIface](#) > &faces)
- void **clearRole** (FacePipelineFaceTagsIface::Roles role)
- [FacePipelineFaceTagsIfaceList](#) **facesForRole** (FacePipelineFaceTagsIface::Roles role) const
- [FacePipelineFaceTagsIfaceList](#) & **operator=** (const QList< [FaceTagsIface](#) > &faces)
- void **replaceRole** (FacePipelineFaceTagsIface::Roles remove, FacePipelineFaceTagsIface::Roles add)
- void **setRole** (FacePipelineFaceTagsIface::Roles role)
- QList< [FaceTagsIface](#) > **toFaceTagsIfaceList** () const

## 9.535 Digikam::FacePipelinePackage Class Reference

Inheritance diagram for Digikam::FacePipelinePackage:



### Public Types

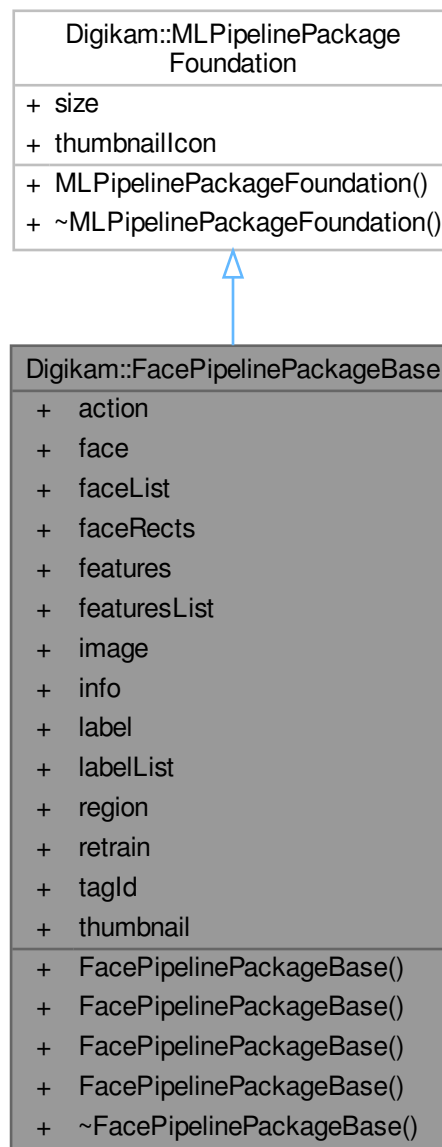
- enum `ProcessFlag` {  
`NotProcessed = 0` , `PreviewImageLoaded = 1 << 0` , `ProcessedByDetector = 1 << 1` , `ProcessedByRecognizer = 1 << 2` ,  
`WrittenToDatabase = 1 << 3` , `ProcessedByTrainer = 1 << 4` }
- typedef `QFlags< ProcessFlag >` `ProcessFlags`

### Public Attributes

- [FacePipelineFaceTagsList](#) `databaseFaces`
- `QList< QRectF >` `detectedFaces`
- [DImg](#) `image`
- [ItemInfo](#) `info`
- int `processedFaceCount = 0`
- `ProcessFlags` `processFlags = NotProcessed`
- `QList< Identity >` `recognitionResults`

## 9.536 Digikam::FacePipelinePackageBase Class Reference

Inheritance diagram for Digikam::FacePipelinePackageBase:



### Public Types

- enum **EditPipelineAction** { **Confirm** , **Remove** , **EditTag** , **EditRegion** , **AddManually** }

### Public Member Functions

- **FacePipelinePackageBase** (const [ItemInfo](#) &\_info, const [FaceTagsIface](#) &\_face, int \_tagId, const [TagRegion](#) &\_region, const [DImg](#) &\_image, [EditPipelineAction](#) \_action, bool \_retrain)
- **FacePipelinePackageBase** (qulonglong \_imageId)
- **FacePipelinePackageBase** (qulonglong \_imageId, const [FaceTagsIface](#) &\_face)

### Public Attributes

- [EditPipelineAction](#) **action** = [EditPipelineAction::Confirm](#)
- [FaceTagsIface](#) **face**
- [QList](#)< [FaceTagsIface](#) > **faceList**
- [QList](#)< [QRectF](#) > **faceRects**
- [cv::Mat](#) **features**
- [QList](#)< [cv::Mat](#) > **featuresList**
- [DImg](#) **image**
- [ItemInfo](#) **info**
- int **label** = -1
- [QList](#)< int > **labelList**
- [TagRegion](#) **region**
- bool **retrain** = false
- int **tagId** = -1
- [QImage](#) **thumbnail**

### Public Attributes inherited from [Digikam::MLPipelinePackageFoundation](#)

- quint64 **size** = 0
- [QIcon](#) **thumbnailIcon**

## 9.537 Digikam::FacePipelineRecognize Class Reference

Inheritance diagram for Digikam::FacePipelineRecognize:



### Public Member Functions

- **FacePipelineRecognize** (const [FaceScanSettings](#) &\_settings)
- bool [start](#) () override

## Public Member Functions inherited from Digikam::FacePipelineBase

- **FacePipelineBase** (const [FaceScanSettings](#) &\_settings)

## Public Member Functions inherited from Digikam::MLPipelineFoundation

- virtual void [cancel](#) ()
- bool [hasFinished](#) () const

## Protected Member Functions

- void [addMoreWorkers](#) () override
- bool [classifier](#) () override
- bool [extractor](#) () override
- bool [finder](#) () override
- bool [loader](#) () override
- bool [trainer](#) () override
- bool [writer](#) () override

## Protected Member Functions inherited from Digikam::FacePipelineBase

- bool [commonFaceThumbnailExtractor](#) (const QString &pipelineName, [MLPipelineFoundation::MLPipelineStage](#) thisStage, [MLPipelineFoundation::MLPipelineStage](#) nextStage)
- bool [commonFaceThumbnailLoader](#) (const QString &pipelineName, [MLPipelineFoundation::MLPipelineStage](#) thisStage, [MLPipelineFoundation::MLPipelineStage](#) nextStage)
- bool [enqueue](#) ([MLPipelineQueue](#) \*thisQueue, [MLPipelinePackageFoundation](#) \*package) override

## Protected Member Functions inherited from Digikam::MLPipelineFoundation

- bool [addWorker](#) (const [MLPipelineStage](#) &stage)
- bool [checkMoreWorkers](#) (int totalItemCount, int currentItemCount, bool useFullCpu)
- void [clearAllQueues](#) ()
- void [clearQueue](#) ([MLPipelineQueue](#) \*thisQueue)
- virtual [MLPipelinePackageFoundation](#) \* [dequeue](#) ([MLPipelineQueue](#) \*thisQueue)
- void [notify](#) ([MLPipelineNotification](#) notification, const QString &\_name, const QString &\_path, int \_processed, const [DImg](#) &\_thumbnail)
- void [notify](#) ([MLPipelineNotification](#) notification, const QString &\_name, const QString &\_path, int \_processed, const [QIcon](#) &\_thumbnail)
- void [notify](#) ([MLPipelineNotification](#) notification, const QString &\_name, const QString &\_path, int \_processed, const [QImage](#) &\_thumbnail)
- void [pipelinePerformanceEnd](#) (const [MLPipelineStage](#) &stage, int totalItemCount, [QElapsedTimer](#) &timer)
- void [pipelinePerformanceEnd](#) (const [MLPipelineStage](#) &stage, [QElapsedTimer](#) &timer)
- void [pipelinePerformanceStart](#) (const [MLPipelineStage](#) &stage, [QElapsedTimer](#) &timer)
- [MLPipelinePackageFoundation](#) \* [queueEndSignal](#) () const
- void [showPipelinePerformance](#) () const
- void [stageEnd](#) ([MLPipelineStage](#) thisStage, [MLPipelineStage](#) nextStage)
- void [stageStart](#) ([QThread::Priority](#) threadPriority, [MLPipelineStage](#) thisStage, [MLPipelineStage](#) nextStage, [MLPipelineQueue](#) \*&thisQueue, [MLPipelineQueue](#) \*&nextQueue)
- void [waitForStart](#) ()

### Additional Inherited Members

#### Public Types inherited from [Digikam::FacePipelineBase](#)

- enum [FilterMode](#) { [ScanAll](#) , [ScanNew](#) , [TrainNew](#) , [TrainAll](#) , [TrainRemove](#) , [TrainReset](#) }
- enum [WriteMode](#) { [NormalWrite](#) , [OverwriteAllFaces](#) , [OverwriteUnconfirmed](#) }

#### Public Types inherited from [Digikam::MLPipelineFoundation](#)

- enum [MLPipelineNotification](#) { [notifySkipped](#) , [notifyProcessed](#) }
- typedef struct [Digikam::MLPipelineFoundation::\\_MLPipelinePerformanceProfile](#) [MLPipelinePerformanceProfile](#)
- typedef [SharedQueue](#)< [MLPipelinePackageFoundation](#) \* > [MLPipelineQueue](#)
- enum [MLPipelineStage](#) { [Finder](#) , [Loader](#) , [Extractor](#) , [Classifier](#) , [Trainer](#) , [Writer](#) , [None](#) }

#### Signals inherited from [Digikam::MLPipelineFoundation](#)

- void **finished** ()  
*Emitted when the last package has finished processing.*
- void **processed** (const [MLPipelinePackageNotify::Ptr](#) &package)  
*Emitted when one package has finished processing.*
- void **processing** (const [MLPipelinePackageNotify::Ptr](#) &package)  
*Emitted when one package begins processing.*
- void **progressValueChanged** (float progress)
- void **scheduled** ()  
*Emitted when processing is scheduled.*
- void **signalAddMoreWorkers** ()
- void **signalUpdateItemCount** (const qlonglong itemCount)
- void **skipped** (const [MLPipelinePackageNotify::Ptr](#) &package)  
*Emitted when one or several packages were skipped, usually because they have already been scanned.*
- void **started** (const [QString](#) &message)  
*Emitted when processing has started.*

#### Protected Attributes inherited from [Digikam::FacePipelineBase](#)

- [FaceScanSettings](#) **settings**

#### Protected Attributes inherited from [Digikam::MLPipelineFoundation](#)

- bool **cancelled** = false
- [QAtomicInteger](#)< int > **itemsProcessed** = 0
- quint64 **maxBufferSize** = 2147483648  
*2 GB default*
- [QMutex](#) **mutex**
- [QMap](#)< [MLPipelineStage](#), [MLPipelinePerformanceProfile](#) > **performanceProfileList**
- [QMap](#)< [MLPipelineStage](#), [MLPipelineQueue](#) \* > **queues**
- [QThreadPool](#) \* **threadPool** = nullptr
- [QMutex](#) **threadStageMutex**
- [QAtomicInteger](#)< int > **totalItemCount** = 0
- quint64 **usedBufferSize** = 0
- [QList](#)< [QFutureWatcher](#)< bool > \* > **watchList**



## 9.537.1 Member Function Documentation

### 9.537.1.1 addMoreWorkers()

```
void Digikam::FacePipelineRecognize::addMoreWorkers ( ) [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

### 9.537.1.2 classifier()

```
bool Digikam::FacePipelineRecognize::classifier ( ) [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

### 9.537.1.3 extractor()

```
bool Digikam::FacePipelineRecognize::extractor ( ) [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

### 9.537.1.4 finder()

```
bool Digikam::FacePipelineRecognize::finder ( ) [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

### 9.537.1.5 loader()

```
bool Digikam::FacePipelineRecognize::loader ( ) [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

### 9.537.1.6 start()

```
bool Digikam::FacePipelineRecognize::start ( ) [override], [virtual]
```

Reimplemented from [Digikam::MLPipelineFoundation](#).

### 9.537.1.7 trainer()

```
bool Digikam::FacePipelineRecognize::trainer ( ) [inline], [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

### 9.537.1.8 writer()

```
bool Digikam::FacePipelineRecognize::writer ( ) [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

## 9.538 Digikam::FacePipelineReset Class Reference

Inheritance diagram for Digikam::FacePipelineReset:



**Public Member Functions**

- **FacePipelineReset** (const [FaceScanSettings](#) &\_settings)
- bool **start** () override

**Public Member Functions inherited from [Digikam::FacePipelineBase](#)**

- **FacePipelineBase** (const [FaceScanSettings](#) &\_settings)

**Public Member Functions inherited from [Digikam::MLPipelineFoundation](#)**

- virtual void **cancel** ()
- bool **hasFinished** () const

**Protected Member Functions**

- void **addMoreWorkers** () override
- bool **classifier** () override
- bool **extractor** () override
- bool **finder** () override
- bool **loader** () override
- bool **trainer** () override
- bool **writer** () override

**Protected Member Functions inherited from [Digikam::FacePipelineBase](#)**

- bool **commonFaceThumbnailExtractor** (const QString &pipelineName, [MLPipelineFoundation::MLPipelineStage](#) thisStage, [MLPipelineFoundation::MLPipelineStage](#) nextStage)
- bool **commonFaceThumbnailLoader** (const QString &pipelineName, [MLPipelineFoundation::MLPipelineStage](#) thisStage, [MLPipelineFoundation::MLPipelineStage](#) nextStage)
- bool **enqueue** ([MLPipelineQueue](#) \*thisQueue, [MLPipelinePackageFoundation](#) \*package) override

**Protected Member Functions inherited from [Digikam::MLPipelineFoundation](#)**

- bool **addWorker** (const [MLPipelineStage](#) &stage)
- bool **checkMoreWorkers** (int totalItemCount, int currentItemCount, bool useFullCpu)
- void **clearAllQueues** ()
- void **clearQueue** ([MLPipelineQueue](#) \*thisQueue)
- virtual [MLPipelinePackageFoundation](#) \* **dequeue** ([MLPipelineQueue](#) \*thisQueue)
- void **notify** ([MLPipelineNotification](#) notification, const QString &\_name, const QString &\_path, int \_processed, const [DImg](#) &\_thumbnail)
- void **notify** ([MLPipelineNotification](#) notification, const QString &\_name, const QString &\_path, int \_processed, const [QIcon](#) &\_thumbnail)
- void **notify** ([MLPipelineNotification](#) notification, const QString &\_name, const QString &\_path, int \_processed, const [QImage](#) &\_thumbnail)
- void **pipelinePerformanceEnd** (const [MLPipelineStage](#) &stage, int totalItemCount, [QElapsedTimer](#) &timer)
- void **pipelinePerformanceEnd** (const [MLPipelineStage](#) &stage, [QElapsedTimer](#) &timer)
- void **pipelinePerformanceStart** (const [MLPipelineStage](#) &stage, [QElapsedTimer](#) &timer)
- [MLPipelinePackageFoundation](#) \* **queueEndSignal** () const
- void **showPipelinePerformance** () const
- void **stageEnd** ([MLPipelineStage](#) thisStage, [MLPipelineStage](#) nextStage)
- void **stageStart** ([QThread::Priority](#) threadPriority, [MLPipelineStage](#) thisStage, [MLPipelineStage](#) nextStage, [MLPipelineQueue](#) \*&thisQueue, [MLPipelineQueue](#) \*&nextQueue)
- void **waitForStart** ()

### Additional Inherited Members

#### Public Types inherited from [Digikam::FacePipelineBase](#)

- enum [FilterMode](#) { [ScanAll](#) , [ScanNew](#) , [TrainNew](#) , [TrainAll](#) , [TrainRemove](#) , [TrainReset](#) }
- enum [WriteMode](#) { [NormalWrite](#) , [OverwriteAllFaces](#) , [OverwriteUnconfirmed](#) }

#### Public Types inherited from [Digikam::MLPipelineFoundation](#)

- enum [MLPipelineNotification](#) { [notifySkipped](#) , [notifyProcessed](#) }
- typedef struct [Digikam::MLPipelineFoundation::\\_MLPipelinePerformanceProfile](#) [MLPipelinePerformanceProfile](#)
- typedef [SharedQueue< MLPipelinePackageFoundation \\* >](#) [MLPipelineQueue](#)
- enum [MLPipelineStage](#) { [Finder](#) , [Loader](#) , [Extractor](#) , [Classifier](#) , [Trainer](#) , [Writer](#) , [None](#) }

#### Signals inherited from [Digikam::MLPipelineFoundation](#)

- void [finished](#) ()  
*Emitted when the last package has finished processing.*
- void [processed](#) (const [MLPipelinePackageNotify::Ptr](#) &package)  
*Emitted when one package has finished processing.*
- void [processing](#) (const [MLPipelinePackageNotify::Ptr](#) &package)  
*Emitted when one package begins processing.*
- void [progressValueChanged](#) (float progress)
- void [scheduled](#) ()  
*Emitted when processing is scheduled.*
- void [signalAddMoreWorkers](#) ()
- void [signalUpdateItemCount](#) (const qlonglong itemCount)
- void [skipped](#) (const [MLPipelinePackageNotify::Ptr](#) &package)  
*Emitted when one or several packages were skipped, usually because they have already been scanned.*
- void [started](#) (const QString &message)  
*Emitted when processing has started.*

#### Protected Attributes inherited from [Digikam::FacePipelineBase](#)

- [FaceScanSettings](#) [settings](#)

#### Protected Attributes inherited from [Digikam::MLPipelineFoundation](#)

- bool [cancelled](#) = false
- [QAtomicInteger< int >](#) [itemsProcessed](#) = 0
- [quint64](#) [maxBufferSize](#) = 2147483648  
*2 GB default*
- [QMutex](#) [mutex](#)
- [QMap< MLPipelineStage, MLPipelinePerformanceProfile >](#) [performanceProfileList](#)
- [QMap< MLPipelineStage, MLPipelineQueue \\* >](#) [queues](#)
- [QThreadPool](#) \* [threadPool](#) = nullptr
- [QMutex](#) [threadStageMutex](#)
- [QAtomicInteger< int >](#) [totalItemCount](#) = 0
- [quint64](#) [usedBufferSize](#) = 0
- [QList< QFutureWatcher< bool > \\* >](#) [watchList](#)

## 9.538.1 Member Function Documentation

### 9.538.1.1 addMoreWorkers()

```
void Digikam::FacePipelineReset::addMoreWorkers ( ) [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

### 9.538.1.2 classifier()

```
bool Digikam::FacePipelineReset::classifier ( ) [inline], [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

### 9.538.1.3 extractor()

```
bool Digikam::FacePipelineReset::extractor ( ) [inline], [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

### 9.538.1.4 finder()

```
bool Digikam::FacePipelineReset::finder ( ) [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

### 9.538.1.5 loader()

```
bool Digikam::FacePipelineReset::loader ( ) [inline], [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

### 9.538.1.6 start()

```
bool Digikam::FacePipelineReset::start ( ) [override], [virtual]
```

Reimplemented from [Digikam::MLPipelineFoundation](#).

### 9.538.1.7 trainer()

```
bool Digikam::FacePipelineReset::trainer ( ) [inline], [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

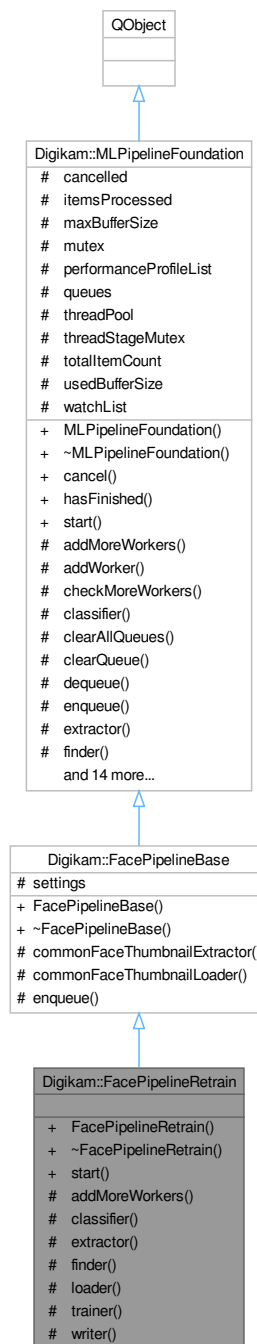
### 9.538.1.8 writer()

```
bool Digikam::FacePipelineReset::writer ( ) [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

## 9.539 Digikam::FacePipelineRetrain Class Reference

Inheritance diagram for Digikam::FacePipelineRetrain:



**Public Member Functions**

- **FacePipelineRetrain** (const [FaceScanSettings](#) &\_settings)
- bool **start** () override

**Public Member Functions inherited from [Digikam::FacePipelineBase](#)**

- **FacePipelineBase** (const [FaceScanSettings](#) &\_settings)

**Public Member Functions inherited from [Digikam::MLPipelineFoundation](#)**

- virtual void **cancel** ()
- bool **hasFinished** () const

**Protected Member Functions**

- void **addMoreWorkers** () override
- bool **classifier** () override
- bool **extractor** () override
- bool **finder** () override
- bool **loader** () override
- bool **trainer** () override
- bool **writer** () override

**Protected Member Functions inherited from [Digikam::FacePipelineBase](#)**

- bool **commonFaceThumbnailExtractor** (const QString &pipelineName, [MLPipelineFoundation::MLPipelineStage](#) thisStage, [MLPipelineFoundation::MLPipelineStage](#) nextStage)
- bool **commonFaceThumbnailLoader** (const QString &pipelineName, [MLPipelineFoundation::MLPipelineStage](#) thisStage, [MLPipelineFoundation::MLPipelineStage](#) nextStage)
- bool **enqueue** ([MLPipelineQueue](#) \*thisQueue, [MLPipelinePackageFoundation](#) \*package) override

**Protected Member Functions inherited from [Digikam::MLPipelineFoundation](#)**

- bool **addWorker** (const [MLPipelineStage](#) &stage)
- bool **checkMoreWorkers** (int totalItemCount, int currentItemCount, bool useFullCpu)
- void **clearAllQueues** ()
- void **clearQueue** ([MLPipelineQueue](#) \*thisQueue)
- virtual [MLPipelinePackageFoundation](#) \* **dequeue** ([MLPipelineQueue](#) \*thisQueue)
- void **notify** ([MLPipelineNotification](#) notification, const QString &\_name, const QString &\_path, int \_processed, const [DImg](#) &\_thumbnail)
- void **notify** ([MLPipelineNotification](#) notification, const QString &\_name, const QString &\_path, int \_processed, const [QIcon](#) &\_thumbnail)
- void **notify** ([MLPipelineNotification](#) notification, const QString &\_name, const QString &\_path, int \_processed, const [QImage](#) &\_thumbnail)
- void **pipelinePerformanceEnd** (const [MLPipelineStage](#) &stage, int totalItemCount, [QElapsedTimer](#) &timer)
- void **pipelinePerformanceEnd** (const [MLPipelineStage](#) &stage, [QElapsedTimer](#) &timer)
- void **pipelinePerformanceStart** (const [MLPipelineStage](#) &stage, [QElapsedTimer](#) &timer)
- [MLPipelinePackageFoundation](#) \* **queueEndSignal** () const
- void **showPipelinePerformance** () const
- void **stageEnd** ([MLPipelineStage](#) thisStage, [MLPipelineStage](#) nextStage)
- void **stageStart** ([QThread::Priority](#) threadPriority, [MLPipelineStage](#) thisStage, [MLPipelineStage](#) nextStage, [MLPipelineQueue](#) \*&thisQueue, [MLPipelineQueue](#) \*&nextQueue)
- void **waitForStart** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::FacePipelineBase](#)

- enum [FilterMode](#) { [ScanAll](#) , [ScanNew](#) , [TrainNew](#) , [TrainAll](#) , [TrainRemove](#) , [TrainReset](#) }
- enum [WriteMode](#) { [NormalWrite](#) , [OverwriteAllFaces](#) , [OverwriteUnconfirmed](#) }

### Public Types inherited from [Digikam::MLPipelineFoundation](#)

- enum [MLPipelineNotification](#) { [notifySkipped](#) , [notifyProcessed](#) }
- typedef struct [Digikam::MLPipelineFoundation::\\_MLPipelinePerformanceProfile](#) [MLPipelinePerformanceProfile](#)
- typedef [SharedQueue< MLPipelinePackageFoundation \\* >](#) [MLPipelineQueue](#)
- enum [MLPipelineStage](#) { [Finder](#) , [Loader](#) , [Extractor](#) , [Classifier](#) , [Trainer](#) , [Writer](#) , [None](#) }

### Signals inherited from [Digikam::MLPipelineFoundation](#)

- void [finished](#) ()  
*Emitted when the last package has finished processing.*
- void [processed](#) (const [MLPipelinePackageNotify::Ptr](#) &package)  
*Emitted when one package has finished processing.*
- void [processing](#) (const [MLPipelinePackageNotify::Ptr](#) &package)  
*Emitted when one package begins processing.*
- void [progressValueChanged](#) (float progress)
- void [scheduled](#) ()  
*Emitted when processing is scheduled.*
- void [signalAddMoreWorkers](#) ()
- void [signalUpdateItemCount](#) (const qlonglong itemCount)
- void [skipped](#) (const [MLPipelinePackageNotify::Ptr](#) &package)  
*Emitted when one or several packages were skipped, usually because they have already been scanned.*
- void [started](#) (const QString &message)  
*Emitted when processing has started.*

### Protected Attributes inherited from [Digikam::FacePipelineBase](#)

- [FaceScanSettings](#) [settings](#)

### Protected Attributes inherited from [Digikam::MLPipelineFoundation](#)

- bool [cancelled](#) = false
- [QAtomicInteger< int >](#) [itemsProcessed](#) = 0
- [quint64](#) [maxBufferSize](#) = 2147483648  
*2 GB default*
- [QMutex](#) [mutex](#)
- [QMap< MLPipelineStage, MLPipelinePerformanceProfile >](#) [performanceProfileList](#)
- [QMap< MLPipelineStage, MLPipelineQueue \\* >](#) [queues](#)
- [QThreadPool](#) \* [threadPool](#) = nullptr
- [QMutex](#) [threadStageMutex](#)
- [QAtomicInteger< int >](#) [totalItemCount](#) = 0
- [quint64](#) [usedBufferSize](#) = 0
- [QList< QFutureWatcher< bool > \\* >](#) [watchList](#)



## 9.539.1 Member Function Documentation

### 9.539.1.1 addMoreWorkers()

```
void Digikam::FacePipelineRetrain::addMoreWorkers ( ) [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

### 9.539.1.2 classifier()

```
bool Digikam::FacePipelineRetrain::classifier ( ) [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

### 9.539.1.3 extractor()

```
bool Digikam::FacePipelineRetrain::extractor ( ) [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

### 9.539.1.4 finder()

```
bool Digikam::FacePipelineRetrain::finder ( ) [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

### 9.539.1.5 loader()

```
bool Digikam::FacePipelineRetrain::loader ( ) [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

### 9.539.1.6 start()

```
bool Digikam::FacePipelineRetrain::start ( ) [override], [virtual]
```

Reimplemented from [Digikam::MLPipelineFoundation](#).

### 9.539.1.7 trainer()

```
bool Digikam::FacePipelineRetrain::trainer ( ) [inline], [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

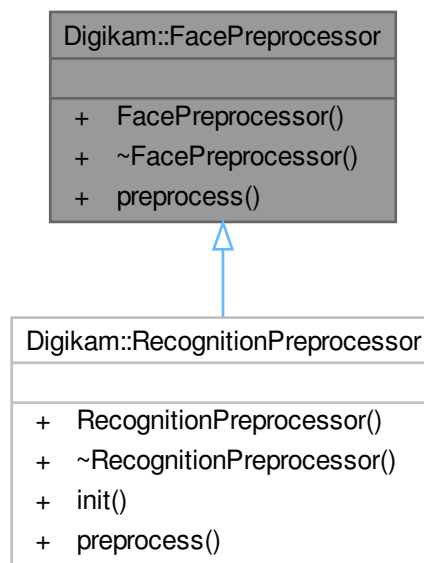
### 9.539.1.8 writer()

```
bool Digikam::FacePipelineRetrain::writer ( ) [override], [protected], [virtual]
```

Implements [Digikam::MLPipelineFoundation](#).

## 9.540 Digikam::FacePreprocessor Class Reference

Inheritance diagram for Digikam::FacePreprocessor:

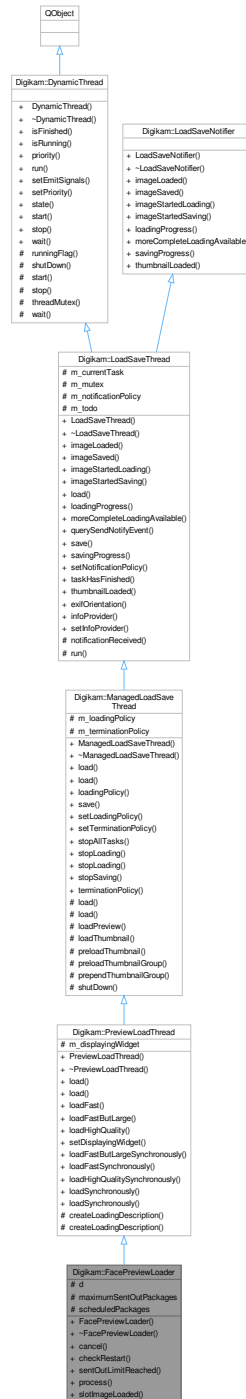


### Public Member Functions

- virtual cv::Mat **preprocess** (const cv::Mat &image) const =0

## 9.541 Digikam::FacePreviewLoader Class Reference

Inheritance diagram for Digikam::FacePreviewLoader:



### Public Slots

- void **process** (const FacePipelineExtendedPackage::Ptr &package)
- void **slotImageLoaded** (const LoadingDescription &loadingDescription, const DImg &img)

## Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()
  - Stop computation, sets the running flag to false.*
- void **wait** ()
  - Waits until the thread finishes.*

## Signals

- void **processed** (const [FacePipelineExtendedPackage::Ptr](#) &package)

## Signals inherited from [Digikam::LoadSaveThread](#)

- void **signalImageLoaded** (const [LoadingDescription](#) &loadingDescription, const [DImg](#) &img)
  - This signal is emitted when the loading process has finished.*
- void **signalImageSaved** (const [QString](#) &filePath, bool success)
- void **signalImageStartedLoading** (const [LoadingDescription](#) &loadingDescription)
  - All signals are delivered to the thread from where the [LoadSaveThread](#) object has been created.*
- void **signalImageStartedSaving** (const [QString](#) &filePath)
- void **signalLoadingProgress** (const [LoadingDescription](#) &loadingDescription, float progress)
  - This signal is emitted whenever new progress info is available and the notification policy allows emitting the signal.*
- void **signalMoreCompleteLoadingAvailable** (const [LoadingDescription](#) &oldLoadingDescription, const [LoadingDescription](#) &newLoadingDescription)
  - This signal is emitted if.*
- void **signalSavingProgress** (const [QString](#) &filePath, float progress)
- void **signalThumbnailLoaded** (const [LoadingDescription](#) &loadingDescription, const [QImage](#) &img)

## Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()
  - Emitted if `emitSignals` is enabled.*

## Public Member Functions

- **FacePreviewLoader** ([FacePipeline::Private](#) \*const dd)
- void **cancel** ()
- void **checkRestart** ()
- bool **sentOutLimitReached** () const

## Public Member Functions inherited from Digikam::PreviewLoadThread

- [PreviewLoadThread](#) (QObject \*const parent=nullptr)  
*Creates a preview load thread.*
- void [load](#) (const [LoadingDescription](#) &description)  
*Load a preview.*
- void [load](#) (const QString &filePath, const [PreviewSettings](#) &settings, int size=0)  
*Load a preview.*
- void [loadFast](#) (const QString &filePath, int size)  
*Load a preview that is optimized for fast loading.*
- void [loadFastButLarge](#) (const QString &filePath, int minimumSize)  
*Load a preview that is as large as possible without sacrificing speed for performance.*
- void [loadHighQuality](#) (const QString &filePath, [PreviewSettings::RawLoading](#) rawLoadingMode=[PreviewSettings::RawPreviewAutomatic](#))  
*Load a preview with higher resolution, trading more quality for less speed.*
- void [setDisplayingWidget](#) (QWidget \*const widget)  
*Optionally, set the displaying widget for color management.*

## Public Member Functions inherited from Digikam::ManagedLoadSaveThread

- [ManagedLoadSaveThread](#) (QObject \*const parent=nullptr)  
*Termination is controlled by setting the TerminationPolicy Default is TerminationPolicyTerminateLoading.*
- void [load](#) (const [LoadingDescription](#) &description)  
*Append a task to load the given file to the task list.*
- void [load](#) (const [LoadingDescription](#) &description, [LoadingPolicy](#) policy)
- [LoadingPolicy](#) [loadingPolicy](#) () const
- void [save](#) (const [DImg](#) &image, const QString &filePath, const QString &format)  
*Append a task to save the image to the task list.*
- void [setLoadingPolicy](#) ([LoadingPolicy](#) policy)  
*Set the loading policy.*
- void [setTerminationPolicy](#) ([TerminationPolicy](#) terminationPolicy)
- void [stopAllTasks](#) ()
- void [stopLoading](#) (const [LoadingDescription](#) &desc, [LoadingTaskFilter](#) filter=[LoadingTaskFilterAll](#))  
*Same than previous method, but Stop and remove tasks filtered by LoadingDescription.*
- void [stopLoading](#) (const QString &filePath=QString(), [LoadingTaskFilter](#) filter=[LoadingTaskFilterAll](#))  
*Stop and remove tasks filtered by filePath and policy.*
- void [stopSaving](#) (const QString &filePath=QString())  
*Stop and remove saving tasks filtered by filePath.*
- [TerminationPolicy](#) [terminationPolicy](#) () const

## Public Member Functions inherited from Digikam::LoadSaveThread

- [LoadSaveThread](#) (QObject \*const parent=nullptr)
- [~LoadSaveThread](#) () override  
*Destructor: The thread will execute all pending tasks and wait for this upon destruction.*
- void [imageLoaded](#) (const [LoadingDescription](#) &loadingDescription, const [DImg](#) &img) override
- void [imageSaved](#) (const QString &filePath, bool success) override
- void [imageStartedLoading](#) (const [LoadingDescription](#) &loadingDescription) override
- void [imageStartedSaving](#) (const QString &filePath) override
- void [load](#) (const [LoadingDescription](#) &description)

*Append a task to load the given file to the task list.*

- void [loadingProgress](#) (const [LoadingDescription](#) &loadingDescription, float progress) override
- void [moreCompleteLoadingAvailable](#) (const [LoadingDescription](#) &oldLoadingDescription, const [LoadingDescription](#) &newLoadingDescription) override
- virtual bool **querySendNotifyEvent** () const
- void **save** (const [DImg](#) &image, const QString &filePath, const QString &format)

*Append a task to save the image to the task list.*

- void [savingProgress](#) (const QString &filePath, float progress) override
- void **setNotificationPolicy** ([NotificationPolicy](#) notificationPolicy)
- virtual void **taskHasFinished** ()
- void [thumbnailLoaded](#) (const [LoadingDescription](#) &loadingDescription, const QImage &img) override

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread](#) (QObject \*const parent=nullptr)

*This class extends [QRunnable](#), so you have to reimplement virtual void [run\(\)](#).*

- [~DynamicThread](#) () override

*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*

- bool **isFinished** () const
- bool **isRunning** () const
- QThread::Priority **priority** () const
- void **setEmitSignals** (bool emitThem)
- void [setPriority](#) (QThread::Priority priority)

*Sets the priority for this dynamic thread.*

- State **state** () const

## Protected Attributes

- [FacePipeline::Private](#) \*const **d** = nullptr
  - int **maximumSentOutPackages** = qMin(QThread::idealThreadCount(), 4)
- Upper limit for memory cost.*
- [PackageLoadingDescriptionList](#) **scheduledPackages**

## Protected Attributes inherited from [Digikam::PreviewLoadThread](#)

- QWidget \* **m\_displayingWidget** = nullptr

## Protected Attributes inherited from [Digikam::ManagedLoadSaveThread](#)

- [LoadingPolicy](#) **m\_loadingPolicy** = [LoadingPolicyAppend](#)
- [TerminationPolicy](#) **m\_terminationPolicy** = [TerminationPolicyTerminateLoading](#)

## Protected Attributes inherited from [Digikam::LoadSaveThread](#)

- [LoadSaveTask](#) \* **m\_currentTask** = nullptr
- QMutex **m\_mutex**
- [NotificationPolicy](#) **m\_notificationPolicy** = [NotificationPolicyTimeLimited](#)
- QList< [LoadSaveTask](#) \* > **m\_todo**

## Additional Inherited Members

### Public Types inherited from [Digikam::ManagedLoadSaveThread](#)

- enum [LoadingMode](#) { [LoadingModeNormal](#) , [LoadingModeShared](#) }  
*used by [SharedLoadSaveThread](#) only*
- enum [LoadingPolicy](#) { [LoadingPolicyFirstRemovePrevious](#) , [LoadingPolicyPrepend](#) , [LoadingPolicySimplePrepend](#) , [LoadingPolicyAppend](#) , [LoadingPolicySimpleAppend](#) , [LoadingPolicyPreload](#) }
- enum [LoadingTaskFilter](#) { [LoadingTaskFilterAll](#) , [LoadingTaskFilterPreloading](#) }
- enum [TerminationPolicy](#) { [TerminationPolicyTerminateLoading](#) , [TerminationPolicyTerminatePreloading](#) , [TerminationPolicyWait](#) , [TerminationPolicyTerminateAll](#) }

### Public Types inherited from [Digikam::LoadSaveThread](#)

- enum [AccessMode](#) { [AccessModeRead](#) , [AccessModeReadWrite](#) }  
*used by [SharedLoadSaveThread](#) only*
- enum [NotificationPolicy](#) { [NotificationPolicyDirect](#) , [NotificationPolicyTimeLimited](#) }

### Public Types inherited from [Digikam::DynamicThread](#)

- enum [State](#) { [Inactive](#) , [Scheduled](#) , [Running](#) , [Deactivating](#) }

### Static Public Member Functions inherited from [Digikam::PreviewLoadThread](#)

- static [DImg](#) [loadFastButLargeSynchronously](#) (const [QString](#) &filePath, int minimumSize, const [IccProfile](#) &profile=[IccProfile](#)())
- static [DImg](#) [loadFastSynchronously](#) (const [QString](#) &filePath, int size, const [IccProfile](#) &profile=[IccProfile](#)())  
*Synchronous versions of the above methods.*
- static [DImg](#) [loadHighQualitySynchronously](#) (const [QString](#) &filePath, [PreviewSettings::RawLoading](#) raw↔  
[LoadingMode](#)=[PreviewSettings::RawPreviewAutomatic](#), const [IccProfile](#) &profile=[IccProfile](#)())
- static [DImg](#) [loadSynchronously](#) (const [LoadingDescription](#) &description)
- static [DImg](#) [loadSynchronously](#) (const [QString](#) &filePath, const [PreviewSettings](#) &previewSettings, int size, const [IccProfile](#) &profile=[IccProfile](#)())

### Static Public Member Functions inherited from [Digikam::LoadSaveThread](#)

- static int [exifOrientation](#) (const [QString](#) &filePath, const [DMetadata](#) &metadata, bool isRaw, bool fromRaw↔  
[EmbeddedPreview](#))  
*Retrieves the Exif orientation, either from the info provider if available, or from the metadata.*
- static [LoadSaveFileInfoProvider](#) \* [infoProvider](#) ()
- static void [setInfoProvider](#) ([LoadSaveFileInfoProvider](#) \*const [infoProvider](#))

### Protected Member Functions inherited from [Digikam::PreviewLoadThread](#)

- [LoadingDescription](#) [createLoadingDescription](#) (const [QString](#) &filePath, const [PreviewSettings](#) &settings, int size)

## Protected Member Functions inherited from [Digikam::ManagedLoadSaveThread](#)

- void **load** (const [LoadingDescription](#) &description, [LoadingMode](#) loadingMode, [AccessMode](#) mode=[AccessModeReadWrite](#))
- void **load** (const [LoadingDescription](#) &description, [LoadingMode](#) loadingMode, [LoadingPolicy](#) policy, [AccessMode](#) mode=[AccessModeReadWrite](#))
- void **loadPreview** (const [LoadingDescription](#) &description, [LoadingPolicy](#) policy)
- void **loadThumbnail** (const [LoadingDescription](#) &description)
- void **preloadThumbnail** (const [LoadingDescription](#) &description)
- void **preloadThumbnailGroup** (const QList< [LoadingDescription](#) > &descriptions)
- void **prependThumbnailGroup** (const QList< [LoadingDescription](#) > &descriptions)
- void **shutDown** ()

## Protected Member Functions inherited from [Digikam::LoadSaveThread](#)

- void **notificationReceived** ()
- void **run** () override

*Implement this pure virtual function in your subclass.*

## Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool **runningFlag** () const volatile  
*In your [run\(\)](#) method, you shall regularly check for [runningFlag\(\)](#) and cleanup and return if false.*
- void **shutDown** ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call [stop\(\)](#) and [wait\(\)](#), knowing that nothing will call [start\(\)](#) anymore after this 3) Be sure the thread will never be running at destruction.*
- void **start** (QMutexLocker< QMutex > &locker)  
*Doing the same as [start\(\)](#), [stop\(\)](#) and [wait](#) above, provide it with a locked QMutexLocker on mutex().*
- void **stop** (const QMutexLocker< QMutex > &locker)
- QMutex \* **threadMutex** () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void **wait** (QMutexLocker< QMutex > &locker)

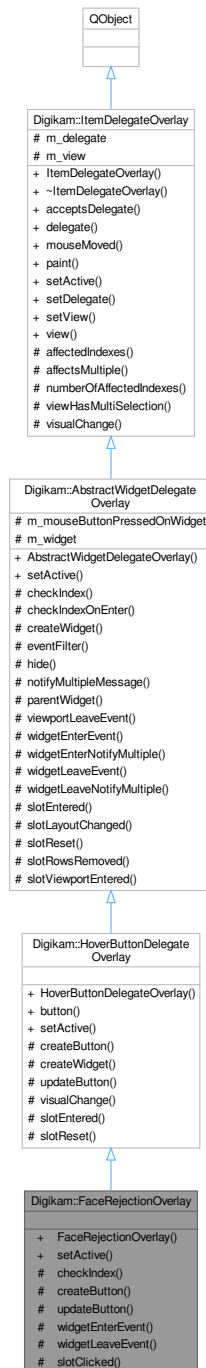
## Static Protected Member Functions inherited from [Digikam::PreviewLoadThread](#)

- static [LoadingDescription](#) **createLoadingDescription** (const QString &filePath, const [PreviewSettings](#) &settings, int size, const [lccProfile](#) &profile)



## 9.542 Digikam::FaceRejectionOverlay Class Reference

Inheritance diagram for Digikam::FaceRejectionOverlay:



### Signals

- void **rejectFaces** (const QList< QModelIndex > &indexes)

## Signals inherited from [Digikam::ItemDelegateOverlay](#)

- void **hideNotification** ()
- void **requestNotification** (const QModelIndex &index, const QString &message)
- void **update** (const QModelIndex &index)

## Public Member Functions

- **FaceRejectionOverlay** (QObject \*const parent)
- void **setActive** (bool active) override  
*If active is true, this will call [createWidget\(\)](#), initialize the widget for use, and setup connections for the virtual slots.*

## Public Member Functions inherited from [Digikam::HoverButtonDelegateOverlay](#)

- **HoverButtonDelegateOverlay** (QObject \*const parent)
- **ItemViewHoverButton** \* **button** () const
- void **setActive** (bool active) override  
*Will call [createButton\(\)](#).*

## Public Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- **AbstractWidgetDelegateOverlay** (QObject \*const parent)  
*This class provides functionality for using a widget in an overlay.*

## Public Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- **ItemDelegateOverlay** (QObject \*const parent=nullptr)
- virtual bool **acceptsDelegate** (QAbstractItemDelegate \*) const
- QAbstractItemDelegate \* **delegate** () const
- virtual void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)  
*Only these two methods are implemented as virtual methods.*
- virtual void **paint** (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index)
- void **setDelegate** (QAbstractItemDelegate \*delegate)
- void **setView** (QAbstractItemView \*view)
- QAbstractItemView \* **view** () const

## Protected Slots

- void **slotClicked** ()

## Protected Slots inherited from [Digikam::HoverButtonDelegateOverlay](#)

- void **slotEntered** (const QModelIndex &index) override
- void **slotReset** () override

## Protected Slots inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- virtual void [slotEntered](#) (const QModelIndex &index)  
*Default implementation shows the widget iff the index is valid and checkIndex returns true.*
- virtual void [slotLayoutChanged](#) ()
- virtual void [slotReset](#) ()  
*Default implementations of these three slots call [hide\(\)](#)*
- virtual void [slotRowsRemoved](#) (const QModelIndex &parent, int start, int end)
- virtual void [slotViewportEntered](#) ()

## Protected Slots inherited from [Digikam::ItemDelegateOverlay](#)

### Protected Member Functions

- bool [checkIndex](#) (const QModelIndex &index) const override  
*Return true here if you want to show the overlay for the given index.*
- [ItemViewHoverButton](#) \* [createButton](#) () override  
*Create your widget here.*
- void [updateButton](#) (const QModelIndex &index) override  
*Called when a new index is entered.*
- void [widgetEnterEvent](#) () override  
*Called when a QEvent::Enter resp.*
- void [widgetLeaveEvent](#) () override

## Protected Member Functions inherited from [Digikam::HoverButtonDelegateOverlay](#)

- QWidget \* [createWidget](#) () override  
*Create your widget here.*
- void [visualChange](#) () override  
*Called when any change from the delegate occurs - when the overlay is installed, when size hints, styles or fonts change.*

## Protected Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- bool [checkIndexOnEnter](#) (const QModelIndex &index) const  
*Utility method called from slotEntered.*
- bool [eventFilter](#) (QObject \*obj, QEvent \*event) override
- virtual void [hide](#) ()  
*Called when the widget shall be hidden (mouse cursor left index, viewport, uninstalled etc.).*
- virtual QString [notifyMultipleMessage](#) (const QModelIndex &, int number)
- QWidget \* [parentWidget](#) () const  
*Returns the widget to be used as parent for your widget created in [createWidget\(\)](#)*
- virtual void [viewportLeaveEvent](#) (QObject \*obj, QEvent \*event)  
*Called when a QEvent::Leave of the viewport is received.*
- void [widgetEnterNotifyMultiple](#) (const QModelIndex &index)  
*A sample implementation for above methods.*
- void [widgetLeaveNotifyMultiple](#) ()

## Protected Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- `QList< QModelIndex > affectedIndexes` (const QModelIndex &index) const
- `bool affectsMultiple` (const QModelIndex &index) const  
*For the context that an overlay can affect multiple items: Assuming the currently overlaid index is given.*
- `int numberOfAffectedIndexes` (const QModelIndex &index) const
- `bool viewHasMultiSelection` () const  
*Utility method.*

## Additional Inherited Members

## Protected Attributes inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- `bool m_mouseButtonPressedOnWidget` = false
- `QWidget * m_widget` = nullptr

## Protected Attributes inherited from [Digikam::ItemDelegateOverlay](#)

- `QAbstractItemDelegate * m_delegate` = nullptr
- `QAbstractItemView * m_view` = nullptr

## 9.542.1 Member Function Documentation

### 9.542.1.1 checkIndex()

```
bool Digikam::FaceRejectionOverlay::checkIndex (
    const QModelIndex & index ) const [override], [protected], [virtual]
```

The default implementation returns true.

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.542.1.2 createButton()

```
ItemViewHoverButton * Digikam::FaceRejectionOverlay::createButton ( ) [override], [protected],
[virtual]
```

Pass view() as parent.

Implements [Digikam::HoverButtonDelegateOverlay](#).

### 9.542.1.3 setActive()

```
void Digikam::FaceRejectionOverlay::setActive (
    bool active ) [override], [virtual]
```

If active is false, this will delete the widget and disconnect all signal from model and view to this object (!)

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

#### 9.542.1.4 updateButton()

```
void Digikam::FaceRejectionOverlay::updateButton (
    const QModelIndex & index ) [override], [protected], [virtual]
```

Reposition your button here, adjust and store state.

Implements [Digikam::HoverButtonDelegateOverlay](#).

#### 9.542.1.5 widgetEnterEvent()

```
void Digikam::FaceRejectionOverlay::widgetEnterEvent ( ) [override], [protected], [virtual]
```

QEvent::Leave event for the widget is received. The default implementation does nothing.

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

#### 9.542.1.6 widgetLeaveEvent()

```
void Digikam::FaceRejectionOverlay::widgetLeaveEvent ( ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

## 9.543 Digikam::FaceRejectionOverlayButton Class Reference

Inheritance diagram for Digikam::FaceRejectionOverlayButton:



### Public Member Functions

- **FaceRejectionOverlayButton** (QAbstractItemView \*const parentView)
- QSize [sizeHint](#) () const override

*Reimplement to match the size of your icon.*

## Public Member Functions inherited from [Digikam::ItemViewHoverButton](#)

- **ItemViewHoverButton** (QAbstractItemView \*const parentView)
- QModelIndex **index** () const
- void **initIcon** ()
- void **reset** ()
- void **setIndex** (const QModelIndex &index)
- void **setVisible** (bool visible) override

## Protected Member Functions

- QIcon **icon** () override  
*Return your icon here.*
- void **updateToolTip** () override  
*Optionally update tooltip here.*

## Protected Member Functions inherited from [Digikam::ItemViewHoverButton](#)

- void **enterEvent** (QEnterEvent \*event)
- void **leaveEvent** (QEvent \*event)
- void **paintEvent** (QPaintEvent \*event)
- void **setup** ()  
*to call in children class constructors to init signal/slot connections.*

## Additional Inherited Members

## Protected Slots inherited from [Digikam::ItemViewHoverButton](#)

- void **refreshIcon** ()
- void **setFadingValue** (int value)
- void **startFading** ()
- void **stopFading** ()

## Protected Attributes inherited from [Digikam::ItemViewHoverButton](#)

- QTimerLine \* **m\_fadingTimeLine** = nullptr
- int **m\_fadingValue** = 0
- QIcon **m\_icon**
- QPersistentModelIndex **m\_index**
- bool **m\_isHovered** = false

## 9.543.1 Member Function Documentation

### 9.543.1.1 icon()

```
QIcon Digikam::FaceRejectionOverlayButton::icon ( ) [override], [protected], [virtual]
```

Will be queried again on toggle.

Implements [Digikam::ItemViewHoverButton](#).

### 9.543.1.2 sizeHint()

QSize Digikam::FaceRejectionOverlayButton::sizeHint ( ) const [override], [virtual]

Implements [Digikam::ItemViewHoverButton](#).

### 9.543.1.3 updateToolTip()

void Digikam::FaceRejectionOverlayButton::updateToolTip ( ) [override], [protected], [virtual]

Will be called again on state change.

Reimplemented from [Digikam::ItemViewHoverButton](#).

## 9.544 Digikam::FaceScanSettings Class Reference

### Public Types

- enum [AlreadyScannedHandling](#) { [Skip](#) , [Rescan](#) , [ClearAll](#) , [RecognizeOnly](#) }  
*To detect and recognize.*
- enum [FaceDetectionModel](#) { [SSDMOBILENET](#) , [YOLOv3](#) , [YuNet](#) }  
*Face detection AI models.*
- enum [FaceDetectionSize](#) { [ExtraSmall](#) , [Small](#) , [Medium](#) , [Large](#) , [ExtraLarge](#) }  
*Face detection size.*
- enum [FaceRecognitionModel](#) { [OpenFace](#) , [SFace](#) }  
*Face recognition AI models.*
- enum [ScanTask](#) { [DetectAndRecognize](#) , [RecognizeMarkedFaces](#) , [RetrainAll](#) , [Reset](#) }  
*Different possible tasks processed while scanning operation.*

### Public Attributes

- AlbumList **albums**  
*Albums to scan.*
- [AlreadyScannedHandling](#) **alreadyScannedHandling** = [Skip](#)
- int [detectAccuracy](#) = DNN\_MODEL\_THRESHOLD\_NOT\_SET  
*Detection accuracy.*
- [FaceDetectionModel](#) **detectModel** = [FaceDetectionModel::YuNet](#)  
*Detection Model.*
- [FaceDetectionSize](#) **detectSize** = [FaceDetectionSize::Large](#)  
*Detection Model.*
- [ItemInfoList](#) **infos**  
*Image infos to scan.*
- int [recognizeAccuracy](#) = DNN\_MODEL\_THRESHOLD\_NOT\_SET  
*Detection accuracy.*
- [FaceRecognitionModel](#) **recognizeModel** = [FaceRecognitionModel::OpenFace](#)  
*Detection Model.*
- [ScanTask](#) **task** = [DetectAndRecognize](#)
- bool **useFullCpu** = false  
*Processing power.*
- bool **wholeAlbums** = false  
*Whole albums checked.*



## 9.544.1 Member Enumeration Documentation

### 9.544.1.1 AlreadyScannedHandling

enum `Digikam::FaceScanSettings::AlreadyScannedHandling`

#### Enumerator

Skip	Skip faces from images already scanned.
Rescan	Rescan faces from images already scanned.
ClearAll	Clear all faces data from images already scanned. Clear identities and training data from FacesDb.
RecognizeOnly	Recognize faces from images already scanned.

### 9.544.1.2 FaceDetectionModel

enum `Digikam::FaceScanSettings::FaceDetectionModel`

#### Enumerator

SSDMOBILENET	SSD MobileNet neural network inference [ <a href="https://github.com/arunponnusamy/cvlib">https://github.com/arunponnusamy/cvlib</a> ].
YOLOv3	YOLO neural network inference [ <a href="https://github.com/sthanhng/yoloface">https://github.com/sthanhng/yoloface</a> ].
YuNet	YuNet neural network inference [ <a href="https://github.com/opencv/opencv_zoo/tree/main">https://github.com/opencv/opencv_zoo/tree/main</a> ].

### 9.544.1.3 FaceRecognitionModel

enum `Digikam::FaceScanSettings::FaceRecognitionModel`

#### Enumerator

OpenFace	OpenFace pre-trained neural network model [ <a href="https://github.com/sahilshah/openface/tree/master">https://github.com/sahilshah/openface/tree/master</a> ].
SFace	SFace pre-trained neural network model [ <a href="https://github.com/opencv/opencv_zoo/blob/main/models/face_recognition_sface/">https://github.com/opencv/opencv_zoo/blob/main/models/face_recognition_sface/</a> ].

### 9.544.1.4 ScanTask

enum `Digikam::FaceScanSettings::ScanTask`

#### Enumerator

DetectAndRecognize	Detect and recognize faces only.
RecognizeMarkedFaces	Recognize already marked faces only.
RetrainAll	Retrain faces only.

## 9.544.2 Member Data Documentation

### 9.544.2.1 detectAccuracy

```
int Digikam::FaceScanSettings::detectAccuracy = DNN_MODEL_THRESHOLD_NOT_SET
```

use default value from dnnmodels.conf

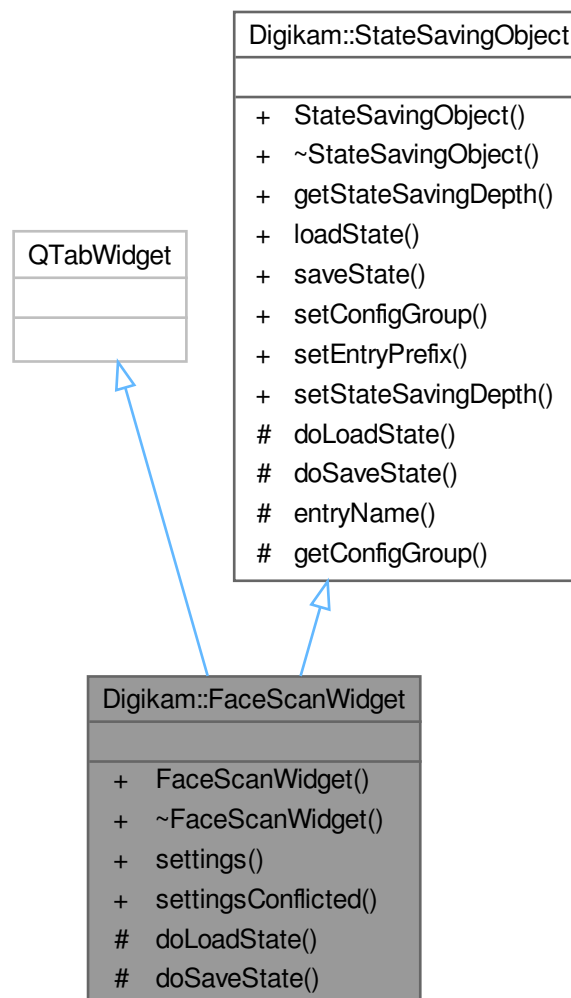
### 9.544.2.2 recognizeAccuracy

```
int Digikam::FaceScanSettings::recognizeAccuracy = DNN_MODEL_THRESHOLD_NOT_SET
```

use default value from dnnmodels.conf

## 9.545 Digikam::FaceScanWidget Class Reference

Inheritance diagram for Digikam::FaceScanWidget:



## Public Member Functions

- **FaceScanWidget** (QWidget \*const parent=nullptr)
- **FaceScanSettings settings** () const
- bool **settingsConflicted** () const

## Public Member Functions inherited from Digikam::StateSavingObject

- **StateSavingObject** (QObject \*const host)  
*Constructor.*
- virtual **~StateSavingObject** ()  
*Destructor.*
- **StateSavingDepth getStateSavingDepth** () const  
*Returns the depth used for state saving or loading.*
- void **loadState** ()  
*Invokes loading the class' state.*
- void **saveState** ()  
*Invokes saving the class' state.*
- virtual void **setConfigGroup** (const KConfigGroup &group)  
*Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void **setEntryPrefix** (const QString &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void **setStateSavingDepth** (const **StateSavingDepth** depth)  
*Sets the depth used for state saving or loading.*

## Protected Member Functions

- void **doLoadState** () override  
*Implement this hook method for state loading.*
- void **doSaveState** () override  
*Implement this hook method for state saving.*

## Protected Member Functions inherited from Digikam::StateSavingObject

- QString **entryName** (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup **getConfigGroup** () const  
*Returns the config group that must be used for state saving and loading.*

## Additional Inherited Members

## Public Types inherited from Digikam::StateSavingObject

- enum **StateSavingDepth** { **INSTANCE** , **DIRECT\_CHILDREN** , **RECURSIVE** }  
*This enum defines the "depth" of the `StateSavingObject::loadState()` and `StateSavingObject::saveState()` methods.*

## 9.545.1 Member Function Documentation

### 9.545.1.1 doLoadState()

```
void Digikam::FaceScanWidget::doLoadState ( ) [override], [protected], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation. ClearAll isn't a valid value anymore so set it Rescan. ClearAll is only used by ResetFacesDb in maintenance.

Implements [Digikam::StateSavingObject](#).

### 9.545.1.2 doSaveState()

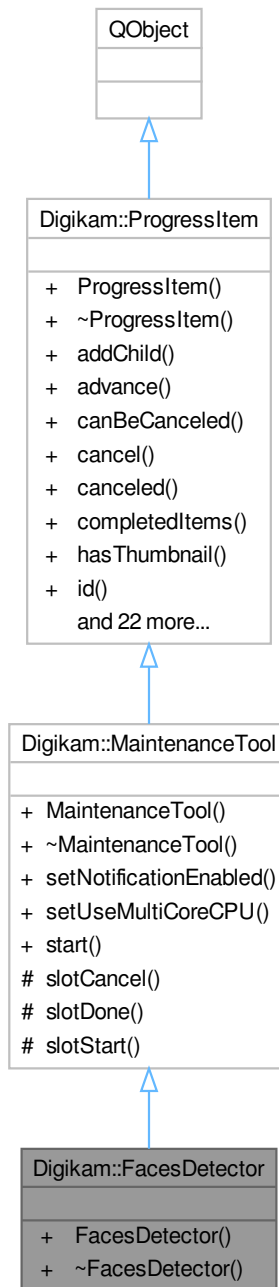
```
void Digikam::FaceScanWidget::doSaveState ( ) [override], [protected], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

## 9.546 Digikam::FacesDetector Class Reference

Inheritance diagram for Digikam::FacesDetector:



### Public Types

- enum **InputSource** { **Albums** = 0 , **Infos** , **Ids** }

## Signals

- void **signalScanNotification** (const QString &msg, int type)

## Signals inherited from [Digikam::MaintenanceTool](#)

- void **signalCanceled** ()  
*Emit when process is canceled.*
- void **signalComplete** ()  
*Emit when process is done (not canceled).*

## Signals inherited from [Digikam::ProgressItem](#)

- void [progressItemAdded](#) ([ProgressItem](#) \*item)  
*Emitted when a new [ProgressItem](#) is added.*
- void [progressItemCanceled](#) ([ProgressItem](#) \*item)  
*Emitted when an item was canceled.*
- void **progressItemCanceledById** (const QString &id)
- void [progressItemCompleted](#) ([ProgressItem](#) \*item)  
*Emitted when a progress item was completed.*
- void [progressItemLabel](#) ([ProgressItem](#) \*item, const QString &label)  
*Emitted when the label of an item changed.*
- void [progressItemProgress](#) ([ProgressItem](#) \*item, unsigned int v)  
*Emitted when the progress value of an item changes.*
- void [progressItemStatus](#) ([ProgressItem](#) \*item, const QString &mess)  
*Emitted when the status message of an item changed.*
- void [progressItemThumbnail](#) ([ProgressItem](#) \*item, const QPixmap &thumb)  
*Emitted when the thumbnail data must be set in item.*
- void [progressItemUsesBusyIndicator](#) ([ProgressItem](#) \*item, bool value)  
*Emitted when the busy indicator state of an item changes.*

## Public Member Functions

- **FacesDetector** (const [FaceScanSettings](#) &settings, [ProgressItem](#) \*const parent=nullptr)

## Public Member Functions inherited from [Digikam::MaintenanceTool](#)

- **MaintenanceTool** (const QString &id, [ProgressItem](#) \*const parent=nullptr)
- void **setNotificationEnabled** (bool b)  
*If true, show a notification message on desktop notification manager with time elapsed to run process.*
- virtual void [setUseMultiCoreCPU](#) (bool)  
*Re-implement this method if your tool is able to use multi-core CPU to process item in parallel.*

## Public Member Functions inherited from Digikam::ProgressItem

- **ProgressItem** ([ProgressItem](#) \*const [parent](#), const QString &[id](#), const QString &[label](#), const QString &[status](#), bool [canBeCanceled](#), bool hasThumb)
- void **addChild** ([ProgressItem](#) \*const kiddo)
- bool [advance](#) (unsigned int v)
  - Advance total items processed by n values and update percentage in progressbar.*
- bool [canBeCanceled](#) () const
- void **cancel** ()
- bool **canceled** () const
- unsigned int **completedItems** () const
- bool [hasThumbnail](#) () const
- const QString & [id](#) () const
- bool **incCompletedItems** (unsigned int v=1)
- void **incTotalItems** (unsigned int v=1)
- const QString & [label](#) () const
- [ProgressItem](#) \* [parent](#) () const
- unsigned int [progress](#) () const
- void **removeChild** ([ProgressItem](#) \*const kiddo)
- void **reset** ()
  - Reset the progress value of this item to 0 and the status string to the empty string.*
- void [setComplete](#) ()
  - Tell the item it has finished.*
- bool **setCompletedItems** (unsigned int v)
- void [setLabel](#) (const QString &v)
- void [setProgress](#) (unsigned int v)
  - Set the progress (percentage of completion) value of this item.*
- void [setShowAtStart](#) (bool [showAtStart](#))
  - Set the property to pop-up item when it's added in progress manager.*
- void [setStatus](#) (const QString &v)
  - Set the string to be used for showing this item's current status.*
- void [setThumbnail](#) (const QIcon &icon)
  - Sets whether this item has a thumbnail.*
- void **setTotalItems** (unsigned int v)
- void [setUsesBusyIndicator](#) (bool useBusyIndicator)
  - Sets whether this item uses a busy indicator instead of real progress for its progress bar.*
- bool [showAtStart](#) () const
- const QString & [status](#) () const
- bool **totalCompleted** () const
- unsigned int **totalItems** () const
- void **updateProgress** ()
  - Recalculate progress according to total/completed items and update.*
- bool [usesBusyIndicator](#) () const

## Additional Inherited Members

## Public Slots inherited from Digikam::MaintenanceTool

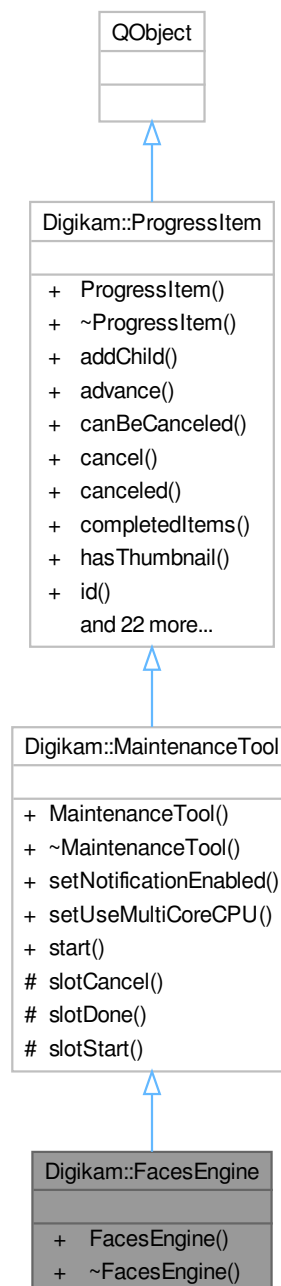
- void **start** ()

## Protected Slots inherited from [Digikam::MaintenanceTool](#)

- virtual void `slotCancel()`
- virtual void `slotDone()`
- virtual void `slotStart()`

## 9.547 Digikam::FacesEngine Class Reference

Inheritance diagram for Digikam::FacesEngine:





## Public Types

- enum **InputSource** { **Albums** = 0 , **Infos** , **Ids** }

## Signals

- void **signalScanNotification** (const QString &msg, int type)

## Signals inherited from [Digikam::MaintenanceTool](#)

- void **signalCanceled** ()  
*Emit when process is canceled.*
- void **signalComplete** ()  
*Emit when process is done (not canceled).*

## Signals inherited from [Digikam::ProgressItem](#)

- void [progressItemAdded](#) ([ProgressItem](#) \*item)  
*Emitted when a new [ProgressItem](#) is added.*
- void [progressItemCanceled](#) ([ProgressItem](#) \*item)  
*Emitted when an item was canceled.*
- void **progressItemCanceledById** (const QString &id)
- void [progressItemCompleted](#) ([ProgressItem](#) \*item)  
*Emitted when a progress item was completed.*
- void [progressItemLabel](#) ([ProgressItem](#) \*item, const QString &label)  
*Emitted when the label of an item changed.*
- void [progressItemProgress](#) ([ProgressItem](#) \*item, unsigned int v)  
*Emitted when the progress value of an item changes.*
- void [progressItemStatus](#) ([ProgressItem](#) \*item, const QString &mess)  
*Emitted when the status message of an item changed.*
- void [progressItemThumbnail](#) ([ProgressItem](#) \*item, const QPixmap &thumb)  
*Emitted when the thumbnail data must be set in item.*
- void [progressItemUsesBusyIndicator](#) ([ProgressItem](#) \*item, bool value)  
*Emitted when the busy indicator state of an item changes.*

## Public Member Functions

- **FacesEngine** (const [FaceScanSettings](#) &settings, [ProgressItem](#) \*const parent=nullptr)

## Public Member Functions inherited from [Digikam::MaintenanceTool](#)

- **MaintenanceTool** (const QString &id, [ProgressItem](#) \*const parent=nullptr)
- void **setNotificationEnabled** (bool b)  
*If true, show a notification message on desktop notification manager with time elapsed to run process.*
- virtual void **setUseMultiCoreCPU** (bool)  
*Re-implement this method if your tool is able to use multi-core CPU to process item in parallel.*

## Public Member Functions inherited from [Digikam::ProgressItem](#)

- **ProgressItem** ([ProgressItem](#) \*const [parent](#), const QString &[id](#), const QString &[label](#), const QString &[status](#), bool [canBeCanceled](#), bool hasThumb)
- void **addChild** ([ProgressItem](#) \*const kiddo)
- bool [advance](#) (unsigned int v)
 

*Advance total items processed by n values and update percentage in progressbar.*
- bool [canBeCanceled](#) () const
- void **cancel** ()
- bool **canceled** () const
- unsigned int **completedItems** () const
- bool [hasThumbnail](#) () const
- const QString & [id](#) () const
- bool **incCompletedItems** (unsigned int v=1)
- void **incTotalItems** (unsigned int v=1)
- const QString & [label](#) () const
- [ProgressItem](#) \* [parent](#) () const
- unsigned int [progress](#) () const
- void **removeChild** ([ProgressItem](#) \*const kiddo)
- void **reset** ()
 

*Reset the progress value of this item to 0 and the status string to the empty string.*
- void [setComplete](#) ()
 

*Tell the item it has finished.*
- bool **setCompletedItems** (unsigned int v)
- void [setLabel](#) (const QString &v)
- void [setProgress](#) (unsigned int v)
 

*Set the progress (percentage of completion) value of this item.*
- void [setShowAtStart](#) (bool [showAtStart](#))
 

*Set the property to pop-up item when it's added in progress manager.*
- void [setStatus](#) (const QString &v)
 

*Set the string to be used for showing this item's current status.*
- void [setThumbnail](#) (const QIcon &icon)
 

*Sets whether this item has a thumbnail.*
- void **setTotalItems** (unsigned int v)
- void [setUsesBusyIndicator](#) (bool useBusyIndicator)
 

*Sets whether this item uses a busy indicator instead of real progress for its progress bar.*
- bool [showAtStart](#) () const
- const QString & [status](#) () const
- bool **totalCompleted** () const
- unsigned int **totalItems** () const
- void **updateProgress** ()
 

*Recalculate progress according to total/completed items and update.*
- bool [usesBusyIndicator](#) () const

## Additional Inherited Members

## Public Slots inherited from [Digikam::MaintenanceTool](#)

- void **start** ()

## Protected Slots inherited from [Digikam::MaintenanceTool](#)

- virtual void **slotCancel** ()
- virtual void **slotDone** ()
- virtual void **slotStart** ()

## 9.548 Digikam::FaceTags Class Reference

### Static Public Member Functions

- static QList< QString > **allPersonNames** ()  
*A method to return a list of all person tag names in the DB.*
- static QList< QString > **allPersonPaths** ()  
*A method to return a list of all person tag paths in the DB.*
- static QList< int > **allPersonTags** ()  
*A method to return a list of all person tags in the DB.*
- static void **applyTagIdentityMapping** (int tagId, const QMap< QString, QString > &attributes)  
*Map an existing tag to a [FacesEngine Identity](#).*
- static void **ensureIsPerson** (int tagId, const QString &fullName=QString())  
*Ensure that the given tag is a person tag.*
- static QString **faceNameForTag** (int tagId)  
*Return a person's name for a tag.*
- static QString **getNameForRect** (qulonglong imageid, const QRect &faceRect)
- static int **getOrCreateTagForIdentity** (const QMap< QString, QString > &attributes)  
*Use attributes as used by [FacesEngine](#) to identify or create a person tag; From the database, produce the identity attributes identifying the corresponding identity.*
- static int **getOrCreateTagForPerson** (const QString &name, int parentId=-1, const QString &fullName=QString())  
*First, looks for the given person name in person tags, and returns an ID.*
- static QMap< QString, QString > **identityAttributes** (int tagId)
- static int **ignoredPersonTagId** ()
- static bool **isPerson** (int tagId)  
*Returns a boolean value indicating whether the given tagId represents a person.*
- static bool **isSystemPersonTagId** (int tagId)
- static bool **isTheIgnoredPerson** (int tagId)
- static bool **isTheUnconfirmedPerson** (int tagId)
- static bool **isTheUnknownPerson** (int tagId)
- static int **personParentTag** ()  
*The suggested parent tag for persons.*
- static int **scannedForFacesTagId** ()
- static int **tagForPerson** (const QString &name, int parentId=-1, const QString &fullName=QString())  
*Looks for the given person name under the People tags tree, and returns an ID.*
- static int **unconfirmedPersonTagId** ()
- static int **unknownPersonTagId** ()

### 9.548.1 Member Function Documentation

#### 9.548.1.1 applyTagIdentityMapping()

```
void Digikam::FaceTags::applyTagIdentityMapping (
    int tagId,
    const QMap< QString, QString > & attributes ) [static]
```

Subsequently, the [Identity](#) can be retrieved via the `identityAttributes()`.

### 9.548.1.2 ensureIsPerson()

```
void Digikam::FaceTags::ensureIsPerson (
    int tagId,
    const QString & fullName = QString() ) [static]
```

If not, it will be converted. Optionally, pass the full name. (tag name is not changed).

### 9.548.1.3 getOrCreateTagForPerson()

```
int Digikam::FaceTags::getOrCreateTagForPerson (
    const QString & name,
    int parentId = -1,
    const QString & fullName = QString() ) [static]
```

If not, creates a new tag. Per default, fullName is the same as name.

### 9.548.1.4 tagForPerson()

```
int Digikam::FaceTags::tagForPerson (
    const QString & name,
    int parentId = -1,
    const QString & fullName = QString() ) [static]
```

Returns 0 if no name found. Per default, fullName is the same as name. As parentId of -1 signals to look for any tag, a valid parentId will limit the search to direct children of this tag. parentId of 0 means top-level tag.

## 9.549 Digikam::FaceTagsEditor Class Reference

Inheritance diagram for Digikam::FaceTagsEditor:



### Public Member Functions

- void `add` (const `FaceTagsIface` &face, bool trainFace=true)  
*Adds a new entry to the database.*

- [FaceTagsIface](#) **add** (qulonglong imageid, int tagId, const [TagRegion](#) &region, bool trainFace=true)
- [FaceTagsIface](#) **addManually** (const [FaceTagsIface](#) &face)
- [FaceTagsIface](#) **changeRegion** (const [FaceTagsIface](#) &face, const [TagRegion](#) &newRegion)
 

*Changes the region of the given entry.*
- [FaceTagsIface](#) **changeSuggestedName** (const [FaceTagsIface](#) &previousEntry, int unconfirmedNameTagId)
 

*Switches an unknownPersonEntry or unconfirmedEntry to an unconfirmedEntry (with a different suggested name).*
- [FaceTagsIface](#) **changeTag** (const [FaceTagsIface](#) &face, int newTagId)
 

*Changes the tag of the given entry.*
- [QList](#)< [FaceTagsIface](#) > **confirmedFaceTagsIfaces** (qulonglong imageid) const
- [FaceTagsIface](#) **confirmName** (const [FaceTagsIface](#) &face, int tagId=-1, const [TagRegion](#) &confirmedRegion=[TagRegion](#)())
 

*Assign the name tag for given face entry.*
- [QList](#)< [FaceTagsIface](#) > **databaseFaces** (qulonglong imageid) const
 

*Reads the FaceTagsIfaces for the given image id from the database.*
- [QList](#)< [FaceTagsIface](#) > **databaseFaces** (qulonglong imageid, [FaceTagsIface::TypeFlags](#) flags) const
- [QList](#)< [FaceTagsIface](#) > **databaseFacesForTraining** (qulonglong imageid) const
- int **faceCountForPersonInImage** (qulonglong imageid, int tagId) const
 

*Returns the number of faces a particular person has in the specified image.*
- [QList](#)< [ItemTagPair](#) > **faceItemTagPairs** (qulonglong imageid, [FaceTagsIface::TypeFlags](#) flags) const
- [QMap](#)< [QString](#), [QString](#) > **getSuggestedNames** (qulonglong id) const
 

*Returns a Map of Tag Regions (in XML format) to Suggested Name (from Face Recognition) for the given image.*
- [QList](#)< [QRect](#) > **getTagRects** (qulonglong imageid) const
 

*Returns a list of all tag rectangles for the image.*
- [QList](#)< [FaceTagsIface](#) > **ignoredFaceTagsIfaces** (qulonglong imageid) const
- int **numberOfFaces** (qulonglong imageid) const
 

*Returns the number of faces present in an image.*
- void **removeAllFaces** (qulonglong imageid)
 

*Unassigns all face tags from the image and sets it's scanned property to false.*
- void **removeFace** (const [FaceTagsIface](#) &face, bool touchTags=true)
 

*Remove the given face.*
- void **removeFace** (qulonglong imageid, const [QRect](#) &rect)
 

*Remove a face or the face for a certain rect from an image.*
- void **removeFaces** (const [QList](#)< [FaceTagsIface](#) > &faces)
- bool **rotateFaces** (qulonglong imageid, const [QSize](#) &size, int oldOrientation, int newOrientation)
 

*Rotate face tags.*
- [QList](#)< [FaceTagsIface](#) > **unconfirmedFaceTagsIfaces** (qulonglong imageid) const
 

*Returns list of Unconfirmed and Unknown faces in the Image.*
- [QList](#)< [FaceTagsIface](#) > **unconfirmedNameFaceTagsIfaces** (qulonglong imageid) const
 

*Returns a list of UnconfirmedFaces in the Image.*

### Static Public Member Functions

- static [FaceTagsIface](#) **confirmedEntry** (const [FaceTagsIface](#) &face, int tagId=-1, const [TagRegion](#) &confirmedRegion=[TagRegion](#)())
 

*Returns the entry that would be added if the given face is confirmed.*
- static [FaceTagsIface](#) **unconfirmedEntry** (qulonglong imageid, int tagId, const [TagRegion](#) &region)
 

*Returns the entry that would be added if the given face is autodetected.*
- static [FaceTagsIface](#) **unknownPersonEntry** (qulonglong imageid, const [TagRegion](#) &region)

## Protected Member Functions

- void **addFaceAndTag** ([ItemTagPair](#) &pair, const [FaceTagsIface](#) &face, const QStringList &properties, bool addTag)
- virtual void **addNormalTag** (qulonglong imageId, int tagId)
- void **removeFaceAndTag** ([ItemTagPair](#) &pair, const [FaceTagsIface](#) &face, bool touchTags)
- virtual void **removeNormalTag** (qulonglong imageId, int tagId)
- virtual void **removeNormalTags** (qulonglong imageId, const QList< int > &tagIds)

## 9.549.1 Member Function Documentation

### 9.549.1.1 add()

```
void Digikam::FaceTagsEditor::add (
    const FaceTagsIface & face,
    bool trainFace = true )
```

The convenience wrapper will return the newly created entry. If trainFace is true, the face will also be listed in the db as needing training. The tag of the face will, if necessary, be converted to a person tag.

### 9.549.1.2 addNormalTag()

```
void Digikam::FaceTagsEditor::addNormalTag (
    qulonglong imageId,
    int tagId ) [protected], [virtual]
```

Reimplemented in [Digikam::FaceUtils](#).

### 9.549.1.3 changeRegion()

```
FaceTagsIface Digikam::FaceTagsEditor::changeRegion (
    const FaceTagsIface & face,
    const TagRegion & newRegion )
```

Returns the face with the new region set.

### 9.549.1.4 changeTag()

```
FaceTagsIface Digikam::FaceTagsEditor::changeTag (
    const FaceTagsIface & face,
    int newTagId )
```

Returns the face with the new Tag. Since a new Tag is going to be assigned to the Face, it's important to remove the association between the face and the old tagId.

If the face is being ignored and it was an unconfirmed or unknown face don't remove a possible tag. See bug 449142.

We store metadata of [FaceTags](#), if it's a confirmed person.

### 9.549.1.5 confirmName()

```
FaceTagsIface Digikam::FaceTagsEditor::confirmName (
    const FaceTagsIface & face,
    int tagId = -1,
    const TagRegion & confirmedRegion = TagRegion() )
```

Pass the tagId if it changed or was newly assigned (UnknownName). Pass the new, corrected region if it changed. If the default values are passed, tag id or region are taken from the given face. The given face should be an unchanged entry read from the database. The confirmed tag will, if necessary, be converted to a person tag. Returns the newly inserted entry.

### 9.549.1.6 getSuggestedNames()

```
QMap< QString, QString > Digikam::FaceTagsEditor::getSuggestedNames (
    qlonglong id ) const
```

This function makes read operations to the database, and hence can be inefficient when called repeatedly. A cached version is provided in [ItemInfo](#), and should be preferred for intensive operations such as sorting, categorizing etc. For Unconfirmed Results, the value is stored as a tuple of (SuggestedId, Property, Region). Look at the digikam.db file for more details.

### 9.549.1.7 getTagRects()

```
QList< QRect > Digikam::FaceTagsEditor::getTagRects (
    qlonglong imageid ) const
```

Unlike findAndTagFaces, this does not take a [DImg](#), because it returns only a QRect, not a Face, so no need of cropping a face rectangle.

### 9.549.1.8 removeFace()

```
void Digikam::FaceTagsEditor::removeFace (
    const FaceTagsIface & face,
    bool touchTags = true )
```

If appropriate, the tag is also removed.

### 9.549.1.9 removeNormalTag()

```
void Digikam::FaceTagsEditor::removeNormalTag (
    qlonglong imageId,
    int tagId ) [protected], [virtual]
```

Reimplemented in [Digikam::FaceUtils](#).



#### 9.549.1.10 unconfirmedEntry()

```
FaceTagsIface Digikam::FaceTagsEditor::unconfirmedEntry (
    qlonglong imageId,
    int tagId,
    const TagRegion & region ) [static]
```

If tagId is -1, the unknown person will be taken.

#### 9.549.1.11 unconfirmedFaceTagsIfaces()

```
QList< FaceTagsIface > Digikam::FaceTagsEditor::unconfirmedFaceTagsIfaces (
    qlonglong imageid ) const
```

If you want just Unconfirmed Faces,

See also

[unconfirmedNameFaceTagsIfaces](#).

#### 9.549.1.12 unconfirmedNameFaceTagsIfaces()

```
QList< FaceTagsIface > Digikam::FaceTagsEditor::unconfirmedNameFaceTagsIfaces (
    qlonglong imageid ) const
```

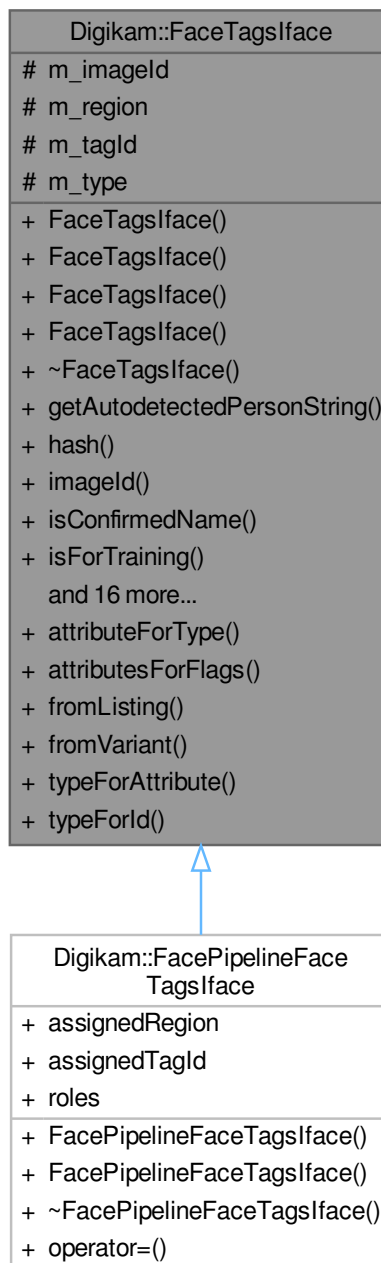
Different from

See also

[unconfirmedFaceTagsIfaces.](#)

## 9.550 Digikam::FaceTagsIface Class Reference

Inheritance diagram for Digikam::FaceTagsIface:



## Public Types

- enum **Type** {  
**InvalidFace** = 0 , **UnknownName** = 1 << 0 , **UnconfirmedName** = 1 << 1 , **IgnoredName** = 1 << 2 ,  
**ConfirmedName** = 1 << 3 , **FaceForTraining** = 1 << 4 , **UnconfirmedTypes** = UnknownName |  
UnconfirmedName , **NormalFaces** = UnknownName | UnconfirmedName | IgnoredName | ConfirmedName  
, **AllTypes** = UnknownName | UnconfirmedName | IgnoredName | ConfirmedName | FaceForTraining , **Type↔  
First** = UnknownName , **TypeLast** = FaceForTraining }
- typedef QFlags< Type > **TypeFlags**

## Public Member Functions

- **FaceTagsface** (const [FaceTagsface](#) &other)
- **FaceTagsface** (const QString &attribute, qulonglong imageld, int tagld, const [TagRegion](#) &region)
- **FaceTagsface** (Type type, qulonglong imageld, int tagld, const [TagRegion](#) &region)
- QString **getAutodetectedPersonString** () const  
*Returns the string tagld + ',' + unconfirmedFace + ',' + regionXml.*
- const QString **hash** () const  
*Generate a hash based on the imageld, tagld, and rect to uniquely identify this entry in the face training DB.*
- qulonglong **imageld** () const
- bool **isConfirmedName** () const
- bool **isForTraining** () const
- bool **isIgnoredName** () const
- bool **isInvalidFace** () const
- bool **isNull** () const
- bool **isUnconfirmedName** () const
- bool **isUnconfirmedType** () const
- bool **isUnknownName** () const
- [FaceTagsface](#) & **operator=** (const [FaceTagsface](#) &other)
- bool **operator==** (const [FaceTagsface](#) &other) const
- [TagRegion](#) **region** () const
- void **removeFaceTraining** () const  
*Remove the face from face training based on the current imageld, tagld, and rect hash.*
- void **setRegion** (const [TagRegion](#) &region)
- void **setTagld** (int tagld)
- void **setType** (Type type)
- int **tagld** () const
- QVariant **toVariant** () const
- Type **type** () const

## Static Public Member Functions

- static QString **attributeForType** (Type type)  
*Return the corresponding image tag property for the given type.*
- static QStringList **attributesForFlags** (TypeFlags flags)  
*Returns a list of all image tag properties for which flags are set.*
- static [FaceTagsface](#) **fromListing** (qulonglong imageid, const QList< QVariant > &values)  
*Create a [FaceTagsface](#) from the extraValues returned from [ItemLISTER](#).*
- static [FaceTagsface](#) **fromVariant** (const QVariant &var)  
*Writes the contents of this face - in a compact way - in the QVariant.*
- static Type **typeForAttribute** (const QString &attribute, int tagld=0)  
*Return the Type for the given attribute.*
- static Type **typeForId** (int tagld)  
*Returns the Face Type corresponding to the given Tagld.*

## Protected Attributes

- `qulonglong m_imageld = 0`
- `TagRegion m_region`
- `int m_tagId = 0`
- Type `m_type = InvalidFace`

## 9.550.1 Member Function Documentation

### 9.550.1.1 fromVariant()

```
FaceTagsIface Digikam::FaceTagsIface::fromVariant (
    const QVariant & var ) [static]
```

Only native QVariant types are used, that is, the QVariant will not have a custom type, thus it can be compared by value by operator==.

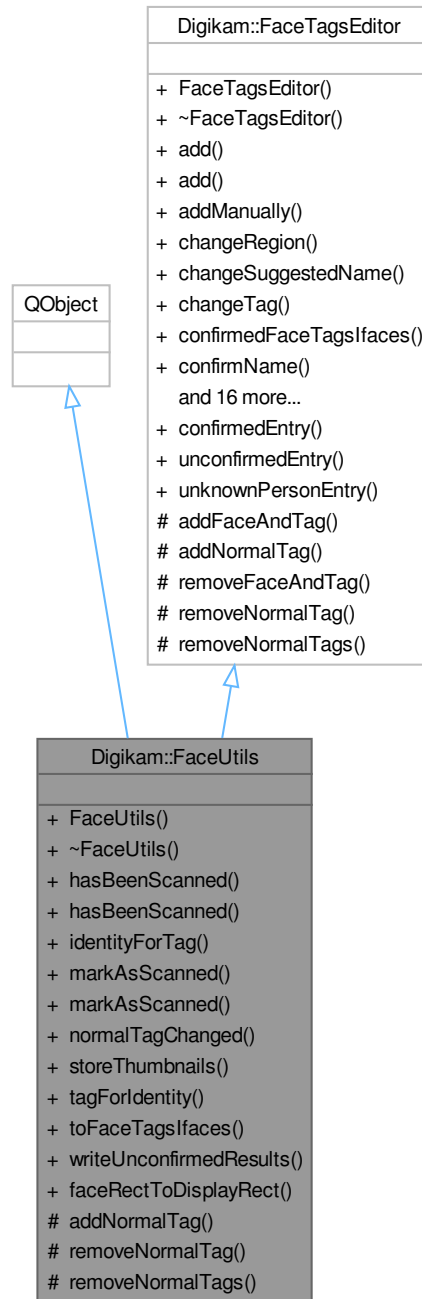
### 9.550.1.2 typeForAttribute()

```
FaceTagsIface::Type Digikam::FaceTagsIface::typeForAttribute (
    const QString & attribute,
    int tagId = 0 ) [static]
```

To distinguish between UnknownName and UnconfirmedName, the tagId must be given.

## 9.551 Digikam::FaceUtils Class Reference

Inheritance diagram for Digikam::FaceUtils:



### Public Types

- enum **FaceRecognitionSteps** { **DetectFaceRegions** , **DetectAndRecognize** }

## Public Member Functions

- **FaceUtils** (QObject \*const parent=nullptr)
- bool **hasBeenScanned** (const [ItemInfo](#) &info) const  
*Tells if the image has been scanned for faces or not.*
- bool **hasBeenScanned** (qulonglong imageid) const
- [Identity](#) **identityForTag** (int tagId) const
- void **markAsScanned** (const [ItemInfo](#) &info, bool [hasBeenScanned](#)=true) const
- void **markAsScanned** (qulonglong imageid, bool [hasBeenScanned](#)=true) const  
*Marks the image as scanned for faces.*
- bool **normalTagChanged** () const
- void **storeThumbnails** ([ThumbnailLoadThread](#) \*const thread, const QString &filePath, const QList<[FaceTagsIface](#)> &databaseFaces, const [DImg](#) &image)  
*This uses a thumbnail load thread to load the image detail.*
- int **tagForIdentity** (const [Identity](#) &identity) const
- QList<[FaceTagsIface](#)> **toFaceTagsIfaces** (qulonglong imageid, const QList< [QRectF](#) > &detectedFaces, const QList< [Identity](#) > &recognitionResults, const QSize &fullSize) const  
*Conversion.*
- QList<[FaceTagsIface](#)> **writeUnconfirmedResults** (qulonglong imageid, const QList< [QRectF](#) > &detectedFaces, const QList< [Identity](#) > &recognitionResults, const QSize &fullSize)  
*The given face list is a result of automatic detection and possibly recognition.*

## Public Member Functions inherited from [Digikam::FaceTagsEditor](#)

- void **add** (const [FaceTagsIface](#) &face, bool trainFace=true)  
*Adds a new entry to the database.*
- [FaceTagsIface](#) **add** (qulonglong imageid, int tagId, const [TagRegion](#) &region, bool trainFace=true)
- [FaceTagsIface](#) **addManually** (const [FaceTagsIface](#) &face)
- [FaceTagsIface](#) **changeRegion** (const [FaceTagsIface](#) &face, const [TagRegion](#) &newRegion)  
*Changes the region of the given entry.*
- [FaceTagsIface](#) **changeSuggestedName** (const [FaceTagsIface](#) &previousEntry, int unconfirmedNameTagId)  
*Switches an unknownPersonEntry or unconfirmedEntry to an unconfirmedEntry (with a different suggested name).*
- [FaceTagsIface](#) **changeTag** (const [FaceTagsIface](#) &face, int newTagId)  
*Changes the tag of the given entry.*
- QList<[FaceTagsIface](#)> **confirmedFaceTagsIfaces** (qulonglong imageid) const
- [FaceTagsIface](#) **confirmName** (const [FaceTagsIface](#) &face, int tagId=-1, const [TagRegion](#) &confirmedRegion=[TagRegion](#)())  
*Assign the name tag for given face entry.*
- QList<[FaceTagsIface](#)> **databaseFaces** (qulonglong imageid) const  
*Reads the FaceTagsIfaces for the given image id from the database.*
- QList<[FaceTagsIface](#)> **databaseFaces** (qulonglong imageid, [FaceTagsIface::TypeFlags](#) flags) const
- QList<[FaceTagsIface](#)> **databaseFacesForTraining** (qulonglong imageid) const
- int **faceCountForPersonInImage** (qulonglong imageid, int tagId) const  
*Returns the number of faces a particular person has in the specified image.*
- QList<[ItemTagPair](#)> **faceItemTagPairs** (qulonglong imageid, [FaceTagsIface::TypeFlags](#) flags) const
- QMap< QString, QString > **getSuggestedNames** (qulonglong id) const  
*Returns a Map of Tag Regions (in XML format) to Suggested Name (from Face Recognition) for the given image.*
- QList< [QRect](#) > **getTagRects** (qulonglong imageid) const  
*Returns a list of all tag rectangles for the image.*
- QList<[FaceTagsIface](#)> **ignoredFaceTagsIfaces** (qulonglong imageid) const
- int **numberOfFaces** (qulonglong imageid) const  
*Returns the number of faces present in an image.*

- void **removeAllFaces** (qulonglong imageid)
 

*Unassigns all face tags from the image and sets it's scanned property to false.*
- void **removeFace** (const [FaceTagsIface](#) &face, bool touchTags=true)
 

*Remove the given face.*
- void **removeFace** (qulonglong imageid, const QRect &rect)
 

*Remove a face or the face for a certain rect from an image.*
- void **removeFaces** (const QList< [FaceTagsIface](#) > &faces)
- bool **rotateFaces** (qulonglong imageid, const QSize &size, int oldOrientation, int newOrientation)
 

*Rotate face tags.*
- QList< [FaceTagsIface](#) > **unconfirmedFaceTagsIfaces** (qulonglong imageid) const
 

*Returns list of Unconfirmed and Unknown faces in the Image.*
- QList< [FaceTagsIface](#) > **unconfirmedNameFaceTagsIfaces** (qulonglong imageid) const
 

*Returns a list of UnconfirmedFaces in the Image.*

### Static Public Member Functions

- static QRect **faceRectToDisplayRect** (const QRect &rect)
 

*For display, it may be desirable to display a slightly larger region than the strict face rectangle.*

### Static Public Member Functions inherited from [Digikam::FaceTagsEditor](#)

- static [FaceTagsIface](#) **confirmedEntry** (const [FaceTagsIface](#) &face, int tagId=-1, const [TagRegion](#) &confirmedRegion=[TagRegion](#)())
 

*Returns the entry that would be added if the given face is confirmed.*
- static [FaceTagsIface](#) **unconfirmedEntry** (qulonglong imageid, int tagId, const [TagRegion](#) &region)
 

*Returns the entry that would be added if the given face is autodetected.*
- static [FaceTagsIface](#) **unknownPersonEntry** (qulonglong imageid, const [TagRegion](#) &region)

### Protected Member Functions

- void **addNormalTag** (qulonglong imageid, int tagId) override
 

*Reimplemented from parent class.*
- void **removeNormalTag** (qulonglong imageid, int tagId) override
- void **removeNormalTags** (qulonglong imageid, const QList< int > &tagIds) override

### Protected Member Functions inherited from [Digikam::FaceTagsEditor](#)

- void **addFaceAndTag** ([ItemTagPair](#) &pair, const [FaceTagsIface](#) &face, const QStringList &properties, bool addTag)
- void **removeFaceAndTag** ([ItemTagPair](#) &pair, const [FaceTagsIface](#) &face, bool touchTags)

## 9.551.1 Member Function Documentation

### 9.551.1.1 addNormalTag()

```
void Digikam::FaceUtils::addNormalTag (
    qulonglong imageId,
    int tagId ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::FaceTagsEditor](#).

### 9.551.1.2 faceRectToDisplayRect()

```
QRect Digikam::FaceUtils::faceRectToDisplayRect (
    const QRect & rect ) [static]
```

This returns a pixel margin commonly used to increase the rectangle size in all four directions.

### 9.551.1.3 removeNormalTag()

```
void Digikam::FaceUtils::removeNormalTag (
    qulonglong imageId,
    int tagId ) [override], [protected], [virtual]
```

If the face just removed was the final face associated with that Tag, reset Tag Icon.

Reimplemented from [Digikam::FaceTagsEditor](#).

### 9.551.1.4 removeNormalTags()

```
void Digikam::FaceUtils::removeNormalTags (
    qulonglong imageId,
    const QList< int > & tagIds ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::FaceTagsEditor](#).

### 9.551.1.5 storeThumbnails()

```
void Digikam::FaceUtils::storeThumbnails (
    ThumbnailLoadThread *const thread,
    const QString & filePath,
    const QList< FaceTagsIface > & databaseFaces,
    const DImg & image )
```

Images in faces and thumbnails.

If requested, the faces will be scaled to the given (fixed) size. Store the needed thumbnails for the given faces. This can be a huge optimization when the has already been loaded anyway.

### 9.551.1.6 toFaceTagsIfaces()

```
QList< FaceTagsIface > Digikam::FaceUtils::toFaceTagsIfaces (
    qulonglong imageid,
    const QList< QRectF > & detectedFaces,
    const QList< Identity > & recognitionResults,
    const QSize & fullSize ) const
```

Convert between [FacesEngine](#) results and [FaceTagsIface](#).



### 9.551.1.7 writeUnconfirmedResults()

```
QList< FaceTagsIface > Digikam::FaceUtils::writeUnconfirmedResults (
    qlonglong imageid,
    const QList< QRectF > & detectedFaces,
    const QList< Identity > & recognitionResults,
    const QSize & fullSize )
```

The results are written to the database and merged with existing entries. The returned list contains the faces written to the database and has the same size as the given list. If a face was skipped (because of an existing entry), a null [FaceTagsIface](#) will be at this place.

## 9.552 Digikam::FacialRecognitionWrapper Class Reference

### Public Member Functions

- **FacialRecognitionWrapper** (const [FacialRecognitionWrapper](#) &)
- **Identity addIdentity** (const QMap< QString, QString > &attributes)
  - Adds a new identity with the specified attributes.*
- void **addIdentityAttribute** (int id, const QString &attribute, const QString &value)
- void **addIdentityAttributes** (int id, const QMap< QString, QString > &attributes)
  - Adds or sets, resp., the attributes of an identity.*
- **Identity addIdentityDebug** (const QMap< QString, QString > &attributes)
  - This is the debug version of addIdentity, so the identity is only added to identityCache, but not into the recognition database.*
- QList< [Identity](#) > **allIdentities** () const
  - Returns all identities known to the database.*
- void **clearAllTraining** ()
  - Deletes the training data for all identities, leaving the identities as such in the database.*
- void **clearTraining** (const QList< [Identity](#) > &identitiesToClean)
  - Deletes the training data for the given identity, leaving the identity as such in the database.*
- void **clearTraining** (const QString &hash)
  - Deletes the training image for the given identity, leaving the identity as such in the database.*
- void **deleteIdentities** (QList< [Identity](#) > identitiesToBeDeleted)
  - Deletes a list of identities from the database.*
- void **deleteIdentity** (const [Identity](#) &identityToBeDeleted)
  - Deletes an identity from the database.*
- **Identity findIdentity** (const QMap< QString, QString > &attributes) const
  - Finds the identity matching the given attributes.*
- **Identity findIdentity** (const QString &attribute, const QString &value) const
  - Finds the first identity with matching attribute - value.*
- **Identity identity** (int id) const
- bool **integrityCheck** ()
  - Checks the integrity and returns true if everything is fine.*
- QVariantMap **parameters** () const
- **Identity recognizeFace** (QImage \*const image)
- QList< [Identity](#) > **recognizeFaces** (const QList< QImage \* > &images)
- QList< [Identity](#) > **recognizeFaces** ([ImageListProvider](#) \*const images)
  - Returns the recommended size if you want to scale face images for recognition.*
- void **setIdentityAttributes** (int id, const QMap< QString, QString > &attributes)
- void **setParameter** (const QString &parameter, const QVariant &value)

*Tunes backend parameters.*

- void **setParameters** (const [FaceScanSettings](#) &parameters)
- void **setParameters** (const QVariantMap &parameters)
- void **train** (const [Identity](#) &identityToBeTrained, const QList< QPair< QImage \*, QString > > &images)
- void **train** (const [Identity](#) &identityToBeTrained, const QPair< QImage \*, QString > &image)

*Performs training by using image data directly.*

- void **train** (const [Identity](#) &identityToBeTrained, [TrainingDataProvider](#) \*const data)
- void **train** (const QList< [Identity](#) > &identitiesToBeTrained, [TrainingDataProvider](#) \*const data)

*Performs training.*

- void **vacuum** ()

*Shrinks the database.*

## 9.552.1 Member Function Documentation

### 9.552.1.1 addIdentity()

```
Identity Digikam::FacialRecognitionWrapper::addIdentity (
    const QMap< QString, QString > & attributes )
```

Please note that a UUID is automatically generated.

### 9.552.1.2 allIdentities()

```
QList< Identity > Digikam::FacialRecognitionWrapper::allIdentities ( ) const
```

#### Note

For the documentation of standard attributes, see [identity.h](#)

### 9.552.1.3 findIdentity() [1/2]

```
Identity Digikam::FacialRecognitionWrapper::findIdentity (
    const QMap< QString, QString > & attributes ) const
```

Attributes are first checked with knowledge of their meaning. Secondly, all unknown attributes are used. Returns a null [Identity](#) if no match is possible or the map is empty.

### 9.552.1.4 findIdentity() [2/2]

```
Identity Digikam::FacialRecognitionWrapper::findIdentity (
    const QString & attribute,
    const QString & value ) const
```

Returns a null identity if no match is found or attribute is empty.

### 9.552.1.5 recognizeFaces()

```
QList< Identity > Digikam::FacialRecognitionWrapper::recognizeFaces (
    ImageListProvider *const images )
```

Larger images can be passed, but may be downscaled. Performs recognition. The face details to be recognized are passed by the provider. For each entry in the provider, in 1-to-1 mapping, a recognized identity or the null identity is returned.

### 9.552.1.6 setParameter()

```
void Digikam::FacialRecognitionWrapper::setParameter (
    const QString & parameter,
    const QVariant & value )
```

Available parameters: "accuracy", synonymous: "threshold", range: 0-1, type: float Determines recognition threshold, 0->accept very insecure recognitions, 1-> be very sure about a recognition.

"k-nearest" : limit the number of nearest neighbors for KNN "recognizeModel" : sets the recognizer model used to instantiate the correct recognizer

### 9.552.1.7 train() [1/2]

```
void Digikam::FacialRecognitionWrapper::train (
    const Identity & identityToBeTrained,
    const QPair< QImage *, QString > & image )
```

These are convenience functions for simple setups. If you want good performance and/or a more versatile implementation, be sure to implement your own [TrainingDataProvider](#) and use one of the above functions.

### 9.552.1.8 train() [2/2]

```
void Digikam::FacialRecognitionWrapper::train (
    const QList< Identity > & identitiesToBeTrained,
    TrainingDataProvider *const data )
```

The identities which have new images to be trained are given. An empty list means that all identities are checked.

All needed data will be queried from the provider.

An identifier for the current training context is given, which can identify the application or group of collections. (It is assumed that training from different contexts is based on non-overlapping collections of images. Keep it always constant for your app.)

## 9.553 Digikam::FFmpegBinary Class Reference

Inheritance diagram for Digikam::FFmpegBinary:



### Public Member Functions

- **FFmpegBinary** (QObject \*const parent=nullptr)

## Public Member Functions inherited from [Digikam::DBinaryIface](#)

- **DBinaryIface** (const QString &binaryName, const QString &minimalVersion, const QString &header, const int headerLine, const QString &projectName, const QString &url, const QString &pluginName, const QStringList &args=QStringList(), const QString &desc=QString())
- **DBinaryIface** (const QString &binaryName, const QString &projectName, const QString &url, const QString &pluginName, const QStringList &args=QStringList(), const QString &desc=QString())
- virtual QString **baseName** () const
- virtual bool **checkDir** ()
- virtual bool **checkDirForPath** (const QString &path)
- const QString & **description** () const
- bool **developmentVersion** () const
- virtual QString **directory** () const
- bool **hasError** () const
- bool **isFound** () const
- bool **isValid** () const
- virtual QString **minimalVersion** () const
- virtual QString **path** () const
- virtual QString **path** (const QString &dir) const
- virtual QString **projectName** () const
- virtual bool **recheckDirectories** ()
- virtual void **setup** (const QString &prev=QString())
- virtual QUrl **url** () const
- const QString & **version** () const
- bool **versionsRight** () const
- bool **versionsRight** (const float) const

## Static Public Member Functions

- static QString **ffmpegToolBin** ()

## Static Public Member Functions inherited from [Digikam::DBinaryIface](#)

- static QString **goodBaseName** (const QString &b)

## Additional Inherited Members

## Public Slots inherited from [Digikam::DBinaryIface](#)

- virtual void **slotAddPossibleSearchDirectory** (const QString &dir)
- virtual void **slotAddSearchDirectory** (const QString &dir)
- virtual void **slotNavigateAndCheck** ()

## Signals inherited from [Digikam::DBinaryIface](#)

- void **signalBinaryValid** ()
- void **signalSearchDirectoryAdded** (const QString &dir)

## Protected Member Functions inherited from [Digikam::DBinaryIface](#)

- `QString findHeader` (const QStringList &output, const QString &header) const
- virtual `bool parseHeader` (const QString &output)
- virtual `QString readConfig` ()
- void `setVersion` (QString &version)
- virtual void `writeConfig` ()

## Protected Attributes inherited from [Digikam::DBinaryIface](#)

- const QStringList `m_binaryArguments`
- const QString `m_binaryBaseName`
- QLabel \* `m_binaryLabel` = nullptr
- const bool `m_checkVersion`
- const QString `m_configGroup`
- QString `m_description`
- bool `m_developmentVersion` = false
- QLabel \* `m_downloadButton` = nullptr
- bool `m_hasError` = false
- const int `m_headerLine`
- const QString `m_headerStarts`
- bool `m_isFound` = false
- QLineEdit \* `m_lineEdit` = nullptr
- const QString `m_minimalVersion`
- QPushButton \* `m_pathButton` = nullptr
- QString `m_pathDir` = QLatin1String("")
- QFrame \* `m_pathWidget` = nullptr
- const QString `m_projectName`
- QSet< QString > `m_searchPaths`
- QLabel \* `m_statusIcon` = nullptr
- const QUrl `m_url`
- QString `m_version` = QLatin1String("")
- QLabel \* `m_versionLabel` = nullptr

## 9.554 Digikam::FFMpegConfigHelper Class Reference

### Static Public Member Functions

- static FFMpegProperties [getAudioCodecsProperties](#) ()  
*Return a map of Audio Codec Name with a list of properties:*
- static FFMpegProperties [getExtensionsProperties](#) ()  
*Return a map, of File extensions supported with a list of properties:*
- static FFMpegProperties [getVideoCodecsProperties](#) ()  
*Return a map of Video Codec Name with a list of properties:*

## 9.554.1 Member Function Documentation

### 9.554.1.1 getAudioCodecsProperties()

FFmpegProperties Digikam::FFmpegConfigHelper::getAudioCodecsProperties ( ) [static]

- Codecs description.
- Read support.
- Write support.

### 9.554.1.2 getExtensionsProperties()

FFmpegProperties Digikam::FFmpegConfigHelper::getExtensionsProperties ( ) [static]

- Format description.

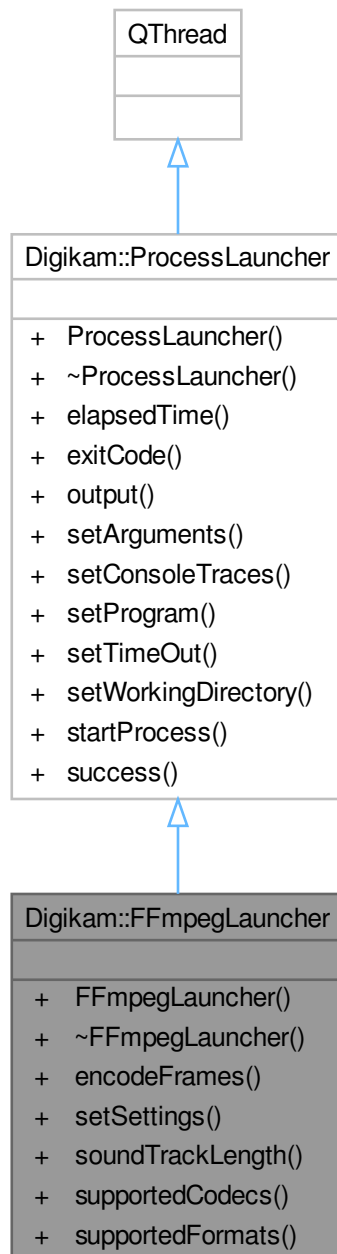
### 9.554.1.3 getVideoCodecsProperties()

FFmpegProperties Digikam::FFmpegConfigHelper::getVideoCodecsProperties ( ) [static]

- Codecs description.
- Read support.
- Write support.

## 9.555 Digikam::FFmpegLauncher Class Reference

Inheritance diagram for Digikam::FFmpegLauncher:



### Public Member Functions

- **FFmpegLauncher** (`QObject *const parent=nullptr`)
- void **encodeFrames** ()



- Encode frames in a separated thread.*

  - void **setSettings** ([VidSlideSettings](#) \*const settings)
  - Set encoding frames settings.*
  - QTime **soundTrackLength** (const QString &audioPath)
  - Return the length of an audio file.*
  - QMap< QString, QString > **supportedCodecs** ()
  - Get the map of supported codecs with features.*
  - QMap< QString, QString > **supportedFormats** ()
  - Get the map of supported formats with features.*

## Public Member Functions inherited from [Digikam::ProcessLauncher](#)

- **ProcessLauncher** (QObject \*const parent=nullptr)
- qint64 **elapsedTime** () const
- Return the elapsed time in ms to run the process.*
- int **exitCode** () const
- Return the exit code from the process.*
- QString **output** () const
- Return the process output as string.*
- void **setArguments** (const QStringList &args)
- void **setConsoleTraces** (bool b)
- If turned on, all traces from the process are printed on the console.*
- void **setProgram** (const QString &prog)
- void **setTimeout** (int msec)
- void **setWorkingDirectory** (const QString &dir)
- void **startProcess** ()
- Start the process.*
- bool **success** () const
- Return true if the process is started and completed without error.*

## Additional Inherited Members

## Signals inherited from [Digikam::ProcessLauncher](#)

- void **signalComplete** (bool [success](#), int [exitCode](#))

## 9.555.1 Member Function Documentation

### 9.555.1.1 soundTrackLength()

```
QTime Digikam::FFmpegLauncher::soundTrackLength (
    const QString & audioPath )
```

If duration cannot be decoded, it returns a null QTime.

## 9.556 Digikam::FieldQueryBuilder Class Reference

### Public Member Functions

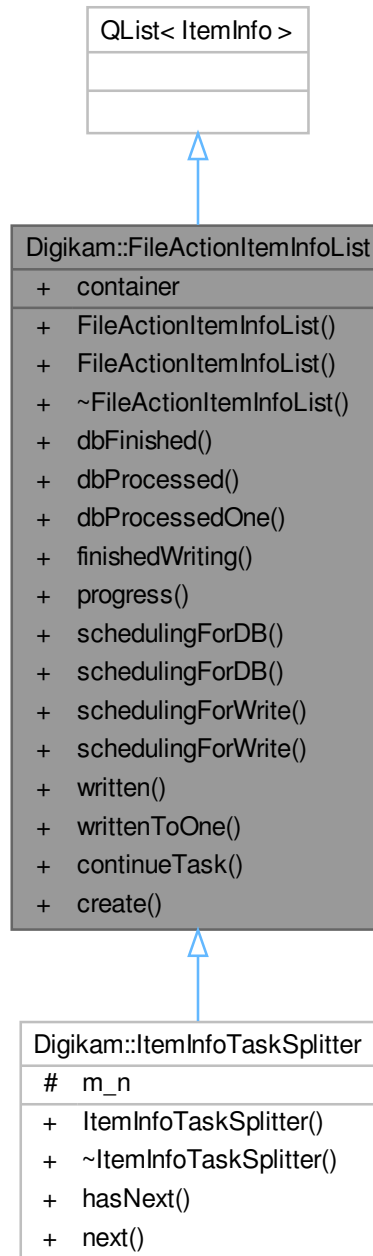
- **FieldQueryBuilder** (QString &sql, [SearchXmlCachingReader](#) &reader, QList< QVariant > \*boundValues, [ItemQueryPostHooks](#) \*const hooks, SearchXml::Relation relation)
- void **addChoiceIntField** (const QString &name)
- void **addChoiceStringField** (const QString &name)
- void **addDateField** (const QString &name)
- void **addDoubleField** (const QString &name)
- void **addIntBitmaskField** (const QString &name)
- void **addIntField** (const QString &name)
- void **addLongField** (const QString &name)
- void **addLongListField** (const QString &name)
- void **addPosition** ()
- void **addRectanglePositionSearch** (double lon1, double lat1, double lon2, double lat2) const
- void **addStringField** (const QString &name)
- QString **prepareForLike** (const QString &str) const

### Public Attributes

- QList< QVariant > \* **boundValues**
- [ItemQueryPostHooks](#) \* **hooks** = nullptr
- [SearchXmlCachingReader](#) & **reader**
- SearchXml::Relation **relation** = SearchXml::Equal
- QString & **sql**

## 9.557 Digikam::FileActionItemInfoList Class Reference

Inheritance diagram for Digikam::FileActionItemInfoList:



### Public Member Functions

- **FileActionItemInfoList** (const [FileActionItemInfoList](#) &copy)
- void **dbFinished** () const

- void **dbProcessed** (int numberOfInfos) const
- void **dbProcessedOne** () const  
*db worker progress info*
- void **finishedWriting** () const
- [FileActionProgressItemContainer](#) \* **progress** () const
- void **schedulingForDB** (const QString &action, [FileActionProgressItemCreator](#) \*const creator)
- void **schedulingForDB** (int numberOfInfos, const QString &action, [FileActionProgressItemCreator](#) \*const creator)  
*before sending to db worker*
- void **schedulingForWrite** (const QString &action, [FileActionProgressItemCreator](#) \*const creator) const
- void **schedulingForWrite** (int numberOfInfos, const QString &action, [FileActionProgressItemCreator](#) \*const creator) const  
*db worker calls this before sending to file worker*
- void **written** (int numberOfInfos) const
- void **writtenToOne** () const  
*file worker calls this when finished*

### Static Public Member Functions

- static [FileActionItemInfoList](#) **continueTask** (const QList< [ItemInfo](#) > &list, [FileActionProgressItemContainer](#) \*const container)
- static [FileActionItemInfoList](#) **create** (const QList< [ItemInfo](#) > &list)

### Public Attributes

- QExplicitlySharedDataPointer< [FileActionProgressItemContainer](#) > **container**

## 9.558 Digikam::FileActionMngr Class Reference

Inheritance diagram for Digikam::FileActionMngr:



### Public Types

- enum **GroupAction** { **AddToGroup** , **RemoveFromGroup** , **SplitGroup** }

### Public Slots

- void **addToGroup** (const [ItemInfo](#) &pick, const QList< [ItemInfo](#) > &infos)
- void **applyMetadata** (const QList< [ItemInfo](#) > &infos, const [DisjointMetadata](#) &hub)
- void **applyMetadata** (const QList< [ItemInfo](#) > &infos, [DisjointMetadata](#) \*hub)
- void **assignColorLabel** (const [ItemInfo](#) &info, int colorId)
- void **assignColorLabel** (const QList< [ItemInfo](#) > &infos, int colorId)
- void **assignPickLabel** (const [ItemInfo](#) &info, int pickId)
- void **assignPickLabel** (const QList< [ItemInfo](#) > &infos, int pickId)

- void **assignRating** (const [ItemInfo](#) &info, int rating)
- void **assignRating** (const QList< [ItemInfo](#) > &infos, int rating)
- void **assignTag** (const [ItemInfo](#) &info, int tagID)
- void **assignTag** (const QList< [ItemInfo](#) > &infos, int tagID)
- void **assignTags** (const [ItemInfo](#) &info, const QList< int > &tagIDs)
- void **assignTags** (const QList< [ItemInfo](#) > &infos, const QList< int > &tagIDs)
- void **assignTags** (const QList< qlonglong > &imageIDs, const QList< int > &tagIDs)
- void **copyAttributes** (const [ItemInfo](#) &source, const QString &derivedPath)
- void **copyAttributes** (const [ItemInfo](#) &source, const QStringList &derivedPaths)
- void **removeFromGroup** (const [ItemInfo](#) &info)
- void **removeFromGroup** (const QList< [ItemInfo](#) > &infos)
- void **removeTag** (const [ItemInfo](#) &info, int tagID)
- void **removeTag** (const QList< [ItemInfo](#) > &infos, int tagID)
- void **removeTags** (const [ItemInfo](#) &info, const QList< int > &tagIDs)
- void **removeTags** (const QList< [ItemInfo](#) > &infos, const QList< int > &tagIDs)
- void **setExifOrientation** (const QList< [ItemInfo](#) > &infos, int orientation)
- void **transform** (const QList< [ItemInfo](#) > &infos, [MetaEngineRotation::TransformationAction](#) action)  
*Flip or rotate.*
- void **ungroup** (const [ItemInfo](#) &info)
- void **ungroup** (const QList< [ItemInfo](#) > &infos)

## Signals

- void **signalImageChangeFailed** (const QString &message, const QStringList &fileNames)

## Public Member Functions

- bool **isActive** ()
- bool **requestShutDown** ()
- void **shutDown** ()

## Static Public Member Functions

- static [FileActionMgr](#) \* **instance** ()

## Friends

- class **FileActionMgrCreator**

## 9.558.1 Member Function Documentation

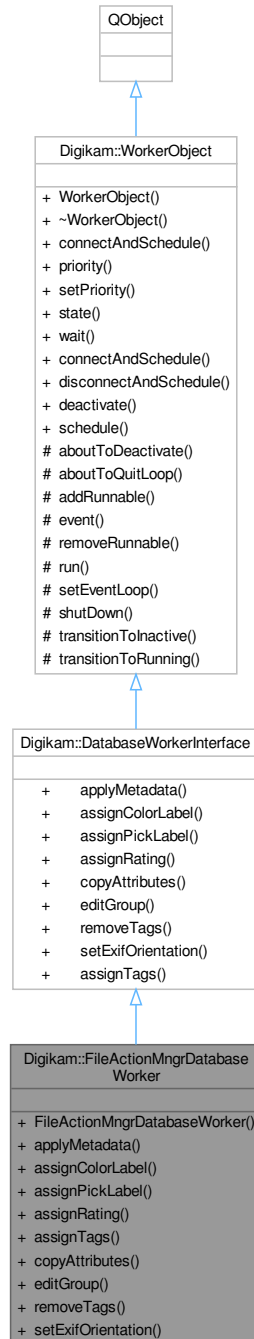
### 9.558.1.1 transform

```
void Digikam::FileActionMgr::transform (
    const QList< ItemInfo > & infos,
    MetaEngineRotation::TransformationAction action ) [slot]
```

Note: The NoTransformation action is interpreted as Exif auto-rotate

## 9.559 Digikam::FileActionMngrDatabaseWorker Class Reference

Inheritance diagram for Digikam::FileActionMngrDatabaseWorker:



### Public Member Functions

- **FileActionMngrDatabaseWorker** (FileActionMngr::Private \*const dd)
- void [applyMetadata](#) (const [FileActionItemInfoList](#) &infos, [DisjointMetadata](#) \*hub) override

- void [assignColorLabel](#) (const [FileActionItemInfoList](#) &infos, int colorId) override
- void [assignPickLabel](#) (const [FileActionItemInfoList](#) &infos, int pickId) override
- void [assignRating](#) (const [FileActionItemInfoList](#) &infos, int rating) override
- void [assignTags](#) (const [FileActionItemInfoList](#) &infos, const QList< int > &tagIDs) override
- void [copyAttributes](#) (const [FileActionItemInfoList](#) &infos, const QStringList &derivedPaths) override
- void [editGroup](#) (int groupAction, const [ItemInfo](#) &pick, const [FileActionItemInfoList](#) &infos) override
- void [removeTags](#) (const [FileActionItemInfoList](#) &infos, const QList< int > &tagIDs) override
- void [setExifOrientation](#) (const [FileActionItemInfoList](#) &infos, int orientation) override

## Public Member Functions inherited from [Digikam::WorkerObject](#)

- [WorkerObject](#) ()  
*Deriving from a worker object allows you to execute your slots in a thread.*
- bool [connectAndSchedule](#) (const QObject \*sender, const char \*signal, const char \*method, Qt::ConnectionType type=Qt::AutoConnection) const  
*You must normally call [schedule\(\)](#) to ensure that the object is active when you send a signal with work data.*
- QThread::Priority **priority** () const
- void [setPriority](#) (QThread::Priority priority)  
*Sets the priority for this dynamic thread.*
- State **state** () const
- void **wait** ()

## Additional Inherited Members

## Public Types inherited from [Digikam::WorkerObject](#)

- enum [DeactivatingMode](#) { [FlushSignals](#) , [KeepSignals](#) , [PhaseOut](#) }
- enum **State** { [Inactive](#) , [Scheduled](#) , [Running](#) , [Deactivating](#) }

## Public Slots inherited from [Digikam::DatabaseWorkerInterface](#)

## Public Slots inherited from [Digikam::WorkerObject](#)

- void [deactivate](#) ([DeactivatingMode](#) mode=[FlushSignals](#))  
*Quits execution of this worker object.*
- void **schedule** ()  
*Starts execution of this worker object: The object is moved to a thread and an event loop started, so that queued signals will be received.*

## Signals inherited from [Digikam::DatabaseWorkerInterface](#)

- void **writeMetadata** ([FileActionItemInfoList](#) infos, int flag)
- void **writeMetadataToFiles** ([FileActionItemInfoList](#) infos)
- void **writeOrientationToFiles** ([FileActionItemInfoList](#) infos, int orientation)

## Signals inherited from [Digikam::WorkerObject](#)

- void **finished** ()
- void **started** ()



## Static Public Member Functions inherited from [Digikam::WorkerObject](#)

- static bool **connectAndSchedule** (const QObject \*sender, const char \*signal, const [WorkerObject](#) \*receiver, const char \*method, Qt::ConnectionType type=Qt::AutoConnection)
- static bool **disconnectAndSchedule** (const QObject \*sender, const char \*signal, const [WorkerObject](#) \*receiver, const char \*method)

## Protected Member Functions inherited from [Digikam::WorkerObject](#)

- virtual void [aboutToDeactivate](#) ()  
*Called from [deactivate\(\)](#), typically from a different thread than the worker thread, possibly the UI thread.*
- virtual void [aboutToQuitLoop](#) ()  
*Called from within thread's event loop to quit processing.*
- void **addRunnable** (WorkerObjectRunnable \*loop)
- bool **event** (QEvent \*e) override
- void **removeRunnable** (WorkerObjectRunnable \*loop)
- void **run** ()
- void **setEventLoop** (QEventLoop \*loop)
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void **transitionToInactive** ()
- bool **transitionToRunning** ()

## 9.559.1 Member Function Documentation

### 9.559.1.1 [applyMetadata\(\)](#)

```
void Digikam::FileActionMngrDatabaseWorker::applyMetadata (
    const FileActionItemInfoList & infos,
    DisjointMetadata * hub ) [override], [virtual]
```

Reimplemented from [Digikam::DatabaseWorkerInterface](#).

### 9.559.1.2 [assignColorLabel\(\)](#)

```
void Digikam::FileActionMngrDatabaseWorker::assignColorLabel (
    const FileActionItemInfoList & infos,
    int colorId ) [override], [virtual]
```

Reimplemented from [Digikam::DatabaseWorkerInterface](#).

### 9.559.1.3 [assignPickLabel\(\)](#)

```
void Digikam::FileActionMngrDatabaseWorker::assignPickLabel (
    const FileActionItemInfoList & infos,
    int pickId ) [override], [virtual]
```

Reimplemented from [Digikam::DatabaseWorkerInterface](#).

#### 9.559.1.4 assignRating()

```
void Digikam::FileActionMngrDatabaseWorker::assignRating (
    const FileActionItemInfoList & infos,
    int rating ) [override], [virtual]
```

Reimplemented from [Digikam::DatabaseWorkerInterface](#).

#### 9.559.1.5 assignTags()

```
void Digikam::FileActionMngrDatabaseWorker::assignTags (
    const FileActionItemInfoList & infos,
    const QList< int > & tagIDs ) [override], [virtual]
```

Reimplemented from [Digikam::DatabaseWorkerInterface](#).

#### 9.559.1.6 copyAttributes()

```
void Digikam::FileActionMngrDatabaseWorker::copyAttributes (
    const FileActionItemInfoList & infos,
    const QStringList & derivedPaths ) [override], [virtual]
```

Reimplemented from [Digikam::DatabaseWorkerInterface](#).

#### 9.559.1.7 editGroup()

```
void Digikam::FileActionMngrDatabaseWorker::editGroup (
    int groupAction,
    const ItemInfo & pick,
    const FileActionItemInfoList & infos ) [override], [virtual]
```

Reimplemented from [Digikam::DatabaseWorkerInterface](#).

#### 9.559.1.8 removeTags()

```
void Digikam::FileActionMngrDatabaseWorker::removeTags (
    const FileActionItemInfoList & infos,
    const QList< int > & tagIDs ) [override], [virtual]
```

Reimplemented from [Digikam::DatabaseWorkerInterface](#).

#### 9.559.1.9 setExifOrientation()

```
void Digikam::FileActionMngrDatabaseWorker::setExifOrientation (
    const FileActionItemInfoList & infos,
    int orientation ) [override], [virtual]
```

Reimplemented from [Digikam::DatabaseWorkerInterface](#).

## 9.560 Digikam::FileActionMngrFileWorker Class Reference

Inheritance diagram for Digikam::FileActionMngrFileWorker:



### Public Member Functions

- **FileActionMngrFileWorker** (`FileActionMngr::Private *const dd`)
- void **transform** (`const FileActionItemInfoList &infos, int orientation`) override

- void [writeMetadata](#) (const [FileActionItemInfoList](#) &infos, int flags) override
- void [writeMetadataToFiles](#) (const [FileActionItemInfoList](#) &infos) override
- void [writeOrientationToFiles](#) (const [FileActionItemInfoList](#) &infos, int orientation) override

## Public Member Functions inherited from [Digikam::WorkerObject](#)

- [WorkerObject](#) ()  
*Deriving from a worker object allows you to execute your slots in a thread.*
- bool [connectAndSchedule](#) (const [QObject](#) \*sender, const char \*signal, const char \*method, [Qt::](#)↔[ConnectionType](#) type=[Qt::AutoConnection](#)) const  
*You must normally call [schedule\(\)](#) to ensure that the object is active when you send a signal with work data.*
- [QThread::Priority](#) **priority** () const
- void [setPriority](#) ([QThread::Priority](#) priority)  
*Sets the priority for this dynamic thread.*
- State **state** () const
- void **wait** ()

## Additional Inherited Members

## Public Types inherited from [Digikam::WorkerObject](#)

- enum [DeactivatingMode](#) { [FlushSignals](#) , [KeepSignals](#) , [PhaseOut](#) }
- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

## Public Slots inherited from [Digikam::FileWorkerInterface](#)

## Public Slots inherited from [Digikam::WorkerObject](#)

- void [deactivate](#) ([DeactivatingMode](#) mode=[FlushSignals](#))  
*Quits execution of this worker object.*
- void **schedule** ()  
*Starts execution of this worker object: The object is moved to a thread and an event loop started, so that queued signals will be received.*

## Signals inherited from [Digikam::FileWorkerInterface](#)

- void **imageChangeFailed** (const [QString](#) &message, const [QStringList](#) &fileNames)
- void **imageDataChanged** (const [QString](#) &path, bool removeThumbnails, bool notifyCache)

## Signals inherited from [Digikam::WorkerObject](#)

- void **finished** ()
- void **started** ()

## Static Public Member Functions inherited from [Digikam::WorkerObject](#)

- static bool **connectAndSchedule** (const QObject \*sender, const char \*signal, const [WorkerObject](#) \*receiver, const char \*method, Qt::ConnectionType type=Qt::AutoConnection)
- static bool **disconnectAndSchedule** (const QObject \*sender, const char \*signal, const [WorkerObject](#) \*receiver, const char \*method)

## Protected Member Functions inherited from [Digikam::WorkerObject](#)

- virtual void [aboutToDeactivate](#) ()  
*Called from [deactivate\(\)](#), typically from a different thread than the worker thread, possibly the UI thread.*
- virtual void [aboutToQuitLoop](#) ()  
*Called from within thread's event loop to quit processing.*
- void **addRunnable** (WorkerObjectRunnable \*loop)
- bool **event** (QEvent \*e) override
- void **removeRunnable** (WorkerObjectRunnable \*loop)
- void **run** ()
- void **setEventLoop** (QEventLoop \*loop)
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void **transitionToInactive** ()
- bool **transitionToRunning** ()

## 9.560.1 Member Function Documentation

### 9.560.1.1 transform()

```
void Digikam::FileActionMngrFileWorker::transform (
    const FileActionItemInfoList & infos,
    int orientation ) [override], [virtual]
```

Reimplemented from [Digikam::FileWorkerInterface](#).

### 9.560.1.2 writeMetadata()

```
void Digikam::FileActionMngrFileWorker::writeMetadata (
    const FileActionItemInfoList & infos,
    int flags ) [override], [virtual]
```

Reimplemented from [Digikam::FileWorkerInterface](#).

### 9.560.1.3 writeMetadataToFiles()

```
void Digikam::FileActionMngrFileWorker::writeMetadataToFiles (
    const FileActionItemInfoList & infos ) [override], [virtual]
```

Reimplemented from [Digikam::FileWorkerInterface](#).

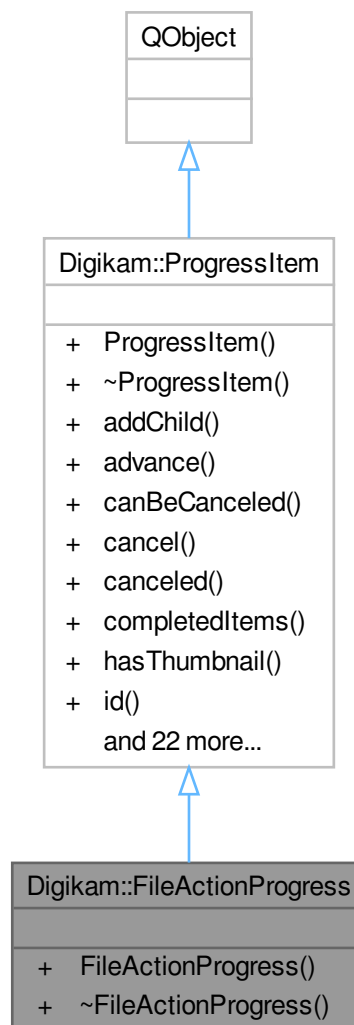
### 9.560.1.4 writeOrientationToFiles()

```
void Digikam::FileActionMngrFileWorker::writeOrientationToFiles (
    const FileActionItemInfoList & infos,
    int orientation ) [override], [virtual]
```

Reimplemented from [Digikam::FileWorkerInterface](#).

## 9.561 Digikam::FileActionProgress Class Reference

Inheritance diagram for Digikam::FileActionProgress:



### Signals

- void **signalComplete** ()

## Signals inherited from [Digikam::ProgressItem](#)

- void [progressItemAdded](#) ([ProgressItem](#) \*item)  
*Emitted when a new [ProgressItem](#) is added.*
- void [progressItemCanceled](#) ([ProgressItem](#) \*item)  
*Emitted when an item was canceled.*
- void [progressItemCanceledById](#) (const QString &id)
- void [progressItemCompleted](#) ([ProgressItem](#) \*item)  
*Emitted when a progress item was completed.*
- void [progressItemLabel](#) ([ProgressItem](#) \*item, const QString &label)  
*Emitted when the label of an item changed.*
- void [progressItemProgress](#) ([ProgressItem](#) \*item, unsigned int v)  
*Emitted when the progress value of an item changes.*
- void [progressItemStatus](#) ([ProgressItem](#) \*item, const QString &mess)  
*Emitted when the status message of an item changed.*
- void [progressItemThumbnail](#) ([ProgressItem](#) \*item, const QPixmap &thumb)  
*Emitted when the thumbnail data must be set in item.*
- void [progressItemUsesBusyIndicator](#) ([ProgressItem](#) \*item, bool value)  
*Emitted when the busy indicator state of an item changes.*

## Public Member Functions

- [FileActionProgress](#) (const QString &name)

## Public Member Functions inherited from [Digikam::ProgressItem](#)

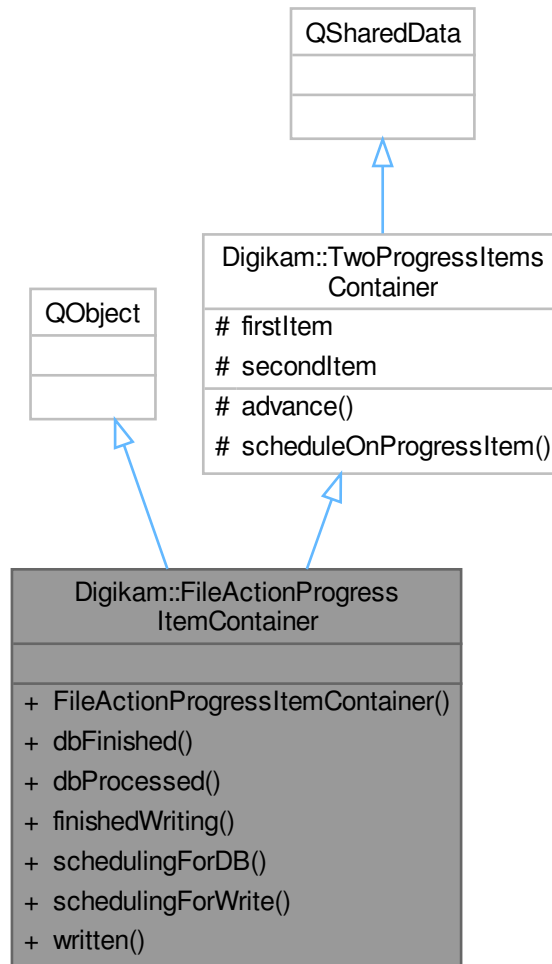
- [ProgressItem](#) ([ProgressItem](#) \*const parent, const QString &id, const QString &label, const QString &status, bool canBeCanceled, bool hasThumb)
- void [addChild](#) ([ProgressItem](#) \*const kiddo)
- bool [advance](#) (unsigned int v)  
*Advance total items processed by n values and update percentage in progressbar.*
- bool [canBeCanceled](#) () const
- void [cancel](#) ()
- bool [canceled](#) () const
- unsigned int [completedItems](#) () const
- bool [hasThumbnail](#) () const
- const QString & [id](#) () const
- bool [incCompletedItems](#) (unsigned int v=1)
- void [incTotalItems](#) (unsigned int v=1)
- const QString & [label](#) () const
- [ProgressItem](#) \* [parent](#) () const
- unsigned int [progress](#) () const
- void [removeChild](#) ([ProgressItem](#) \*const kiddo)
- void [reset](#) ()  
*Reset the progress value of this item to 0 and the status string to the empty string.*
- void [setComplete](#) ()  
*Tell the item it has finished.*
- bool [setCompletedItems](#) (unsigned int v)
- void [setLabel](#) (const QString &v)
- void [setProgress](#) (unsigned int v)  
*Set the progress (percentage of completion) value of this item.*

- void [setShowAtStart](#) (bool [showAtStart](#))  
*Set the property to pop-up item when it's added in progress manager.*
- void [setStatus](#) (const QString &v)  
*Set the string to be used for showing this item's current status.*
- void [setThumbnail](#) (const QIcon &icon)  
*Sets whether this item has a thumbnail.*
- void **setTotalItems** (unsigned int v)
- void [setUsesBusyIndicator](#) (bool useBusyIndicator)  
*Sets whether this item uses a busy indicator instead of real progress for its progress bar.*
- bool [showAtStart](#) () const
- const QString & [status](#) () const
- bool **totalCompleted** () const
- unsigned int **totalItems** () const
- void **updateProgress** ()  
*Recalculate progress according to total/completed items and update.*
- bool [usesBusyIndicator](#) () const



## 9.562 Digikam::FileActionProgressItemContainer Class Reference

Inheritance diagram for Digikam::FileActionProgressItemContainer:



### Signals

- void **signalWritingDone** ()

### Public Member Functions

- void **dbFinished** ()
- void **dbProcessed** (int numberOfInfos)
- void **finishedWriting** ()
- void **schedulingForDB** (int numberOfInfos, const QString &action, [FileActionProgressItemCreator](#) \*const creator)
- void **schedulingForWrite** (int numberOfInfos, const QString &action, [FileActionProgressItemCreator](#) \*const creator)
- void **written** (int numberOfInfos)

### Additional Inherited Members

#### Protected Member Functions inherited from [Digikam::TwoProgressItemsContainer](#)

- void **advance** (QAtomicPointer< [ProgressItem](#) > &ptr, int n)
- void **scheduleOnProgressItem** (QAtomicPointer< [ProgressItem](#) > &ptr, int total, const QString &action, [FileActionProgressItemCreator](#) \*const creator)

#### Protected Attributes inherited from [Digikam::TwoProgressItemsContainer](#)

- QAtomicPointer< [ProgressItem](#) > **firstItem**
- QAtomicPointer< [ProgressItem](#) > **secondItem**

## 9.563 [Digikam::FileActionProgressItemCreator](#) Class Reference

### Public Member Functions

- virtual void **addProgressItem** ([ProgressItem](#) \*const item)=0
- virtual [ProgressItem](#) \* **createProgressItem** (const QString &action) const =0

## 9.564 Digikam::FilePropertiesOption Class Reference

Inheritance diagram for Digikam::FilePropertiesOption:



### Protected Member Functions

- QString [parseOperation](#) ([ParseSettings](#) &settings, const QRegularExpressionMatch &match) override  
*TODO: describe me.*

## Protected Member Functions inherited from [Digikam::Rule](#)

- bool [addToken](#) (const QString &id, const QString &description, const QString &actionName=QString())  
*add a token to the parser, every parser should at least assign one token object*
- void [setDescription](#) (const QString &desc)
- void [setIcon](#) (const QString &pixmap)
- void [setRegExp](#) (const QRegularExpression &regExp)
- void [setUseTokenMenu](#) (bool value)  
*If multiple tokens have been assigned to a rule, a menu will be created.*

## Additional Inherited Members

## Public Types inherited from [Digikam::Rule](#)

- enum [IconType](#) { [Action](#) = 0 , [Dialog](#) }

## Signals inherited from [Digikam::Rule](#)

- void [signalTokenTriggered](#) (const QString &)

## Public Member Functions inherited from [Digikam::Option](#)

- [Option](#) (const QString &name, const QString &description)
- [Option](#) (const QString &name, const QString &description, const QString &icon)

## Public Member Functions inherited from [Digikam::Rule](#)

- [Rule](#) (const QString &name)
- [Rule](#) (const QString &name, const QString &icon)
- QString [description](#) () const
- QPixmap [icon](#) (Rule::IconType type=Rule::Action) const
- bool [isValid](#) () const  
*Checks the validity of the parse object.*
- [ParseResults](#) [parse](#) ([ParseSettings](#) &settings)
- QRegularExpression & [regExp](#) () const  
*TODO: This is probably not needed anymore.*
- QPushButton \* [registerButton](#) (QWidget \*parent)  
*Register a button in the parent object.*
- QAction \* [registerMenu](#) (QMenu \*parent)  
*Register a menu action in the parent object.*
- virtual void [reset](#) ()  
*Resets the parser to its initial state.*
- TokenList & [tokens](#) () const
- bool [useTokenMenu](#) () const  
*Returns true if a token menu is used.*

## Static Public Member Functions inherited from [Digikam::Rule](#)

- static QString [escapeToken](#) (const QString &token)  
*Escape the token characters to make them work in regular expressions.*

## Protected Slots inherited from [Digikam::Rule](#)

- virtual void [slotTokenTriggered](#) (const QString &)

### 9.564.1 Member Function Documentation

#### 9.564.1.1 [parseOperation\(\)](#)

```
QString Digikam::FilePropertiesOption::parseOperation (
    ParseSettings & settings,
    const QRegularExpressionMatch & match ) [override], [protected], [virtual]
```

##### Parameters

<i>settings</i>	contains settings
<i>match</i>	result of the regular expression match done in <a href="#">Option::parse()</a>

##### Returns

Implements [Digikam::Option](#).

## 9.565 Digikam::FileReadLocker Class Reference

### Public Member Functions

- [FileReadLocker](#) (const QString &filePath)

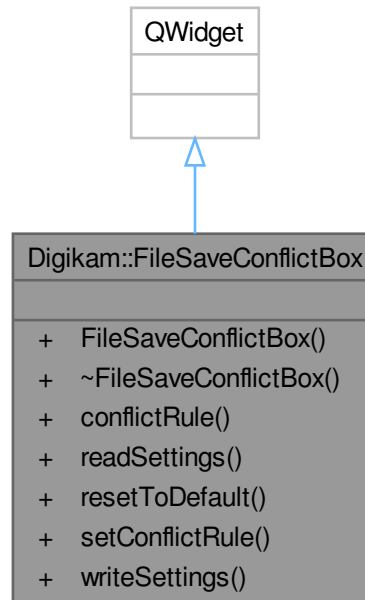
## 9.566 Digikam::FileReadWriteLockKey Class Reference

### Public Member Functions

- [FileReadWriteLockKey](#) (const QString &filePath)
- void [lockForRead](#) ()
- void [lockForWrite](#) ()
- bool [tryLockForRead](#) ()
- bool [tryLockForRead](#) (int timeout)
- bool [tryLockForWrite](#) ()
- bool [tryLockForWrite](#) (int timeout)
- void [unlock](#) ()

## 9.567 Digikam::FileSaveConflictBox Class Reference

Inheritance diagram for Digikam::FileSaveConflictBox:



### Public Types

- enum **ConflictRule** { **OVERWRITE** = 0 , **DIFFNAME** , **SKIPFILE** }

### Signals

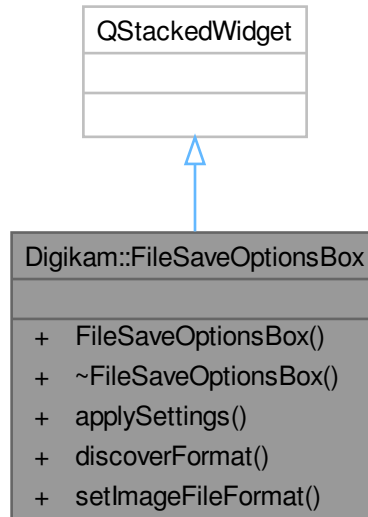
- void **signalConflictButtonChanged** (int)

### Public Member Functions

- **FileSaveConflictBox** (QWidget \*const parent, bool addSkip=false)
- ConflictRule **conflictRule** () const
- void **readSettings** (const KConfigGroup &group)
- void **resetToDefault** ()
- void **setConflictRule** (ConflictRule r)
- void **writeSettings** (KConfigGroup &group)

## 9.568 Digikam::FileSaveOptionsBox Class Reference

Inheritance diagram for Digikam::FileSaveOptionsBox:



### Public Types

- enum `FORMAT` {  
`NONE = 0`, `JPEG`, `PNG`, `TIFF`,  
`JP2K`, `PGF`, `HEIF`, `JXL`,  
`WEBP`, `AVIF` }

### Public Member Functions

- `FileSaveOptionsBox` (`QWidget *const parent=nullptr`)  
*Constructor.*
- `~FileSaveOptionsBox` () override  
*Destructor.*
- void `applySettings` ()
- `FORMAT` `discoverFormat` (`const QString &filename`, `FORMAT` `fallback=NONE`)  
*Tries to discover a file format that has options to change based on a filename.*
- void `setImageFileFormat` (`const QString &`)

## 9.568.1 Member Enumeration Documentation

### 9.568.1.1 FORMAT

```
enum Digikam::FileSaveOptionsBox::FORMAT
```

## Enumerator

NONE	<p>Warning</p> <p>Order is important here. See filesaveoptionbox.cpp which use these values to fill a stack of widgets.</p>
------	---

## 9.568.2 Constructor & Destructor Documentation

### 9.568.2.1 FileSaveOptionsBox()

```
Digikam::FileSaveOptionsBox::FileSaveOptionsBox (
    QWidget *const parent = nullptr ) [explicit]
```

Don't forget to call setDialog after creation of the dialog.

## Parameters

<i>parent</i>	the parent for Qt's parent child mechanism
---------------	--

## 9.568.3 Member Function Documentation

### 9.568.3.1 discoverFormat()

```
FileSaveOptionsBox::FORMAT Digikam::FileSaveOptionsBox::discoverFormat (
    const QString & filename,
    FileSaveOptionsBox::FORMAT fallback = NONE )
```

## Parameters

<i>filename</i>	file name to discover the desired format from
<i>fallback</i>	the fallback format to return if no format could be discovered based on the filename

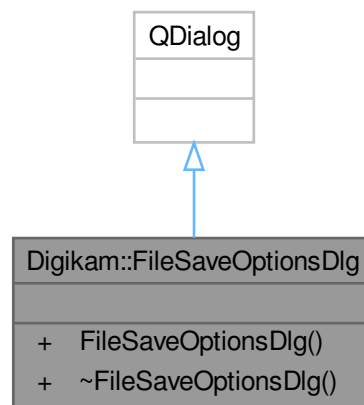


**Returns**

file format guessed from the file name or the given fallback format if no format could be guessed based on the file name

## 9.569 Digikam::FileSaveOptionsDlg Class Reference

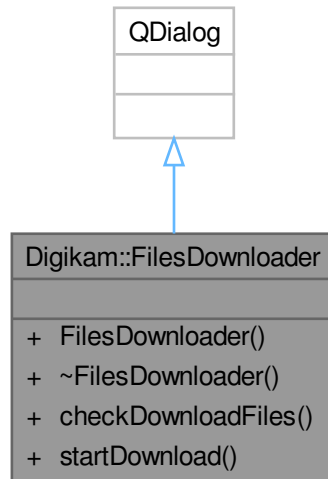
Inheritance diagram for Digikam::FileSaveOptionsDlg:

**Public Member Functions**

- **FileSaveOptionsDlg** (`QWidget *const parent`, [FileSaveOptionsBox](#) `*const options`)

## 9.570 Digikam::FilesDownloader Class Reference

Inheritance diagram for Digikam::FilesDownloader:



### Public Member Functions

- **FilesDownloader** (`QWidget *const parent=nullptr`)
- bool **checkDownloadFiles** () const
- void **startDownload** ()

## 9.571 Digikam::FileWorkerInterface Class Reference

Inheritance diagram for Digikam::FileWorkerInterface:



### Public Slots

- virtual void **writeOrientationToFiles** (const [FileActionItemInfoList](#) &, int)

## Public Slots inherited from [Digikam::WorkerObject](#)

- void **deactivate** ([DeactivatingMode](#) mode=[FlushSignals](#))  
*Quits execution of this worker object.*
- void **schedule** ()  
*Starts execution of this worker object: The object is moved to a thread and an event loop started, so that queued signals will be received.*

## Signals

- void **imageChangeFailed** (const [QString](#) &message, const [QStringList](#) &fileNames)
- void **imageDataChanged** (const [QString](#) &path, bool removeThumbnails, bool notifyCache)

## Signals inherited from [Digikam::WorkerObject](#)

- void **finished** ()
- void **started** ()

## Public Member Functions

- virtual void **transform** (const [FileActionItemInfoList](#) &, int)
- virtual void **writeMetadata** (const [FileActionItemInfoList](#) &, int)
- virtual void **writeMetadataToFiles** (const [FileActionItemInfoList](#) &)

## Public Member Functions inherited from [Digikam::WorkerObject](#)

- [WorkerObject](#) ()  
*Deriving from a worker object allows you to execute your slots in a thread.*
- bool **connectAndSchedule** (const [QObject](#) \*sender, const char \*signal, const char \*method, [Qt::](#)↔[ConnectionType](#) type=[Qt::AutoConnection](#)) const  
*You must normally call [schedule\(\)](#) to ensure that the object is active when you send a signal with work data.*
- [QThread::Priority](#) **priority** () const
- void **setPriority** ([QThread::Priority](#) priority)  
*Sets the priority for this dynamic thread.*
- State **state** () const
- void **wait** ()

## Additional Inherited Members

## Public Types inherited from [Digikam::WorkerObject](#)

- enum [DeactivatingMode](#) { [FlushSignals](#) , [KeepSignals](#) , [PhaseOut](#) }
- enum **State** { [Inactive](#) , [Scheduled](#) , [Running](#) , [Deactivating](#) }

## Static Public Member Functions inherited from [Digikam::WorkerObject](#)

- static bool **connectAndSchedule** (const [QObject](#) \*sender, const char \*signal, const [WorkerObject](#) \*receiver, const char \*method, [Qt::](#)[ConnectionType](#) type=[Qt::AutoConnection](#))
- static bool **disconnectAndSchedule** (const [QObject](#) \*sender, const char \*signal, const [WorkerObject](#) \*receiver, const char \*method)

## Protected Member Functions inherited from [Digikam::WorkerObject](#)

- virtual void [aboutToDeactivate](#) ()
  - Called from [deactivate\(\)](#), typically from a different thread than the worker thread, possibly the UI thread.*
- virtual void [aboutToQuitLoop](#) ()
  - Called from within thread's event loop to quit processing.*
- void **addRunnable** (WorkerObjectRunnable \*loop)
- bool **event** (QEvent \*e) override
- void **removeRunnable** (WorkerObjectRunnable \*loop)
- void **run** ()
- void **setEventLoop** (QEventLoop \*loop)
- void [shutDown](#) ()
  - If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void **transitionToInactive** ()
- bool **transitionToRunning** ()

## 9.572 Digikam::FileWriteLocker Class Reference

### Public Member Functions

- **FileWriteLocker** (const QString &filePath)

## 9.573 Digikam::FilmContainer Class Reference

### Classes

- class [ListItem](#)

### Public Types

- enum **CNFilmProfile** {
  - CNNeutral** = 0 , **CNKodakGold100** , **CNKodakGold200** , **CNKodakEktar100** ,
  - CNKodakProfessionalPortra160NC** , **CNKodakProfessionalPortra160VC** , **CNKodakProfessional↔**
  - Portra400NC** , **CNKodakProfessionalPortra400VC** ,
  - CNKodakProfessionalPortra800Box** , **CNKodakProfessionalPortra800P1** , **CNKodakProfessional↔**
  - Portra800P2** , **CNKodakProfessionalNewPortra160** ,
  - CNKodakProfessionalNewPortra400** , **CNKodakFarbwelt100** , **CNKodakFarbwelt200** , **CNKodak↔**
  - Farbwelt400** ,
  - CNKodakRoyalGold400** , **CNAgfaphotoVistaPlus200** , **CNAgfaphotoVistaPlus400** , **CNFujicolor↔**
  - Pro160S** ,
  - CNFujicolorPro160C** , **CNFujicolorNPL160** , **CNFujicolorPro400H** , **CNFujicolorPro800Z** ,
  - CNFujicolorSuperiaReala** , **CNFujicolorSuperia100** , **CNFujicolorSuperia200** , **CNFujicolorSuperia↔**
  - Xtra400** ,
  - CNFujicolorSuperiaXtra800** , **CNFujicolorTrueDefinition400** , **CNFujicolorSuperia1600** }

### Public Member Functions

- **FilmContainer** (CNFilmProfile profile, double gamma, bool sixteenBit)
- bool **applyBalance** () const
- CNFilmProfile **cnType** () const
- double **exposure** () const
- double **gamma** () const
- void **setApplyBalance** (bool val)
- void **setCNType** (CNFilmProfile profile)
- void **setExposure** (double strength)
- void **setGamma** (double val)
- void **setSixteenBit** (bool val)
- void **setWhitePoint** (const [DColor](#) &wp)
- [CBContainer](#) **toCB** () const
- [LevelsContainer](#) **toLevels** () const
- [DColor](#) **whitePoint** () const

### Static Public Member Functions

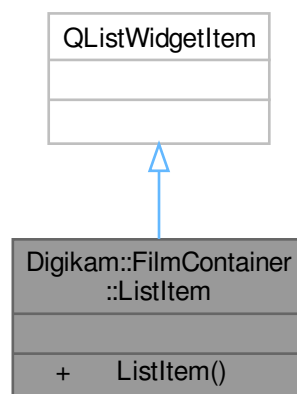
- static QList< [ListItem](#) \* > **profileItemList** (QListWidget \*const view)

### Static Public Attributes

- static const QMap< int, QString > **profileMap** = FilmContainer::profileMapInitializer()

## 9.574 Digikam::FilmContainer::ListItem Class Reference

Inheritance diagram for Digikam::FilmContainer::ListItem:

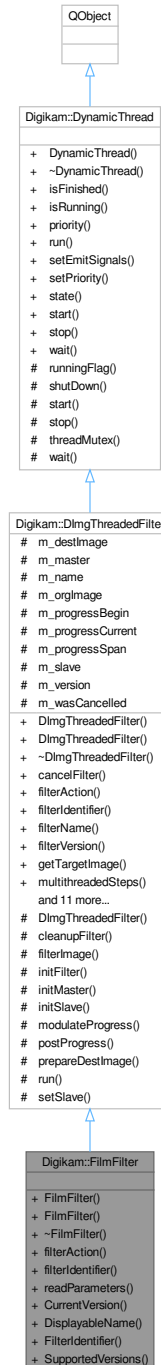


### Public Member Functions

- **ListItem** (const QString &text, QListWidget \*const parent, CNFilmProfile type)

## 9.575 Digikam::FilmFilter Class Reference

Inheritance diagram for Digikam::FilmFilter:



### Public Member Functions

- **FilmFilter** (`Dlmg *const orgImage`, `QObject *const parent=nullptr`, `const FilmContainer &settings=FilmContainer()`)
- **FilmFilter** (`QObject *const parent=nullptr`)

- [FilterAction filterAction \(\)](#) override  
*Returns the action description corresponding to currently set options.*
- [QString filterIdentifier \(\)](#) const override  
*Return the identifier for this filter in the image history.*
- void [readParameters \(const FilterAction &action\)](#) override

## Public Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter \(DImg \\*const orgImage, QObject \\*const parent, const QString &name=QString\(\)\)](#)  
*Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter \(QObject \\*const parent=nullptr, const QString &name=QString\(\)\)](#)  
*Constructs a filter without argument.*
- virtual void [cancelFilter \(\)](#)  
*Cancel the threaded computation.*
- const [QString &filterName \(\)](#)
- int [filterVersion \(\)](#) const
- [DImg getTargetImage \(\)](#)
- [QList< int > multithreadedSteps \(int stop, int start=0\)](#) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool [parametersSuccessfullyRead \(\)](#) const  
*Optional: error handling for readParameters.*
- virtual [QString readParametersError \(const FilterAction &actionThatFailed\)](#) const
- void [setFilterName \(const QString &name\)](#)
- void [setFilterVersion \(int version\)](#)  
*Replaying a filter action: Set the filter version.*
- void [setOriginalImage \(const DImg &orgImage\)](#)
- void [setupAndStartDirectly \(const DImg &orgImage, DImgThreadedFilter \\*const master, int progress←Begin=0, int progressEnd=100\)](#)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void [setupFilter \(const DImg &orgImage\)](#)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void [startFilter \(\)](#)  
*Start the threaded computation.*
- virtual void [startFilterDirectly \(\)](#)  
*Start computation of this filter, directly in this thread.*
- virtual [QList< int > supportedVersions \(\)](#) const

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread \(QObject \\*const parent=nullptr\)](#)  
*This class extends QRunnable, so you have to reimplement virtual void [run\(\)](#).*
- [~DynamicThread \(\)](#) override  
*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool [isFinished \(\)](#) const
- bool [isRunning \(\)](#) const
- [QThread::Priority priority \(\)](#) const
- void [setEmitSignals \(bool emitThem\)](#)
- void [setPriority \(QThread::Priority priority\)](#)  
*Sets the priority for this dynamic thread.*
- State [state \(\)](#) const



### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from [Digikam::DImgThreadedFilter](#)

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.575.1 Member Function Documentation

### 9.575.1.1 filterAction()

`FilterAction` Digikam::FilmFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.575.1.2 filterIdentifier()

`QString` Digikam::FilmFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.575.1.3 readParameters()

```
void Digikam::FilmFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

## 9.576 Digikam::FilmGrainContainer Class Reference

### Public Member Functions

- `bool isDirty () const`

### Public Attributes

- `bool addChrominanceBlueNoise = false`
- `bool addChrominanceRedNoise = false`
- `bool addLuminanceNoise = true`
- `int chromaBlueHighlights = -100`
- `int chromaBlueIntensity = 25`
- `int chromaBlueMidtones = 0`
- `int chromaBlueShadows = -100`
- `int chromaRedHighlights = -100`
- `int chromaRedIntensity = 25`
- `int chromaRedMidtones = 0`
- `int chromaRedShadows = -100`
- `int grainSize = 1`
- `int lumaHighlights = -100`
- `int lumaIntensity = 25`
- `int lumaMidtones = 0`
- `int lumaShadows = -100`
- `bool photoDistribution = false`

## 9.577 Digikam::FilmGrainFilter Class Reference

Inheritance diagram for Digikam::FilmGrainFilter:



### Public Member Functions

- **FilmGrainFilter** (`DImg *const orgImage`, `QObject *const parent=nullptr`, `const FilmGrainContainer &settings=FilmGrainContainer()`)

- **FilmGrainFilter** ([DImgThreadedFilter](#) \*const parentFilter, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const [FilmGrainContainer](#) &settings=[FilmGrainContainer](#)())  
*Constructor for slave mode: execute immediately in current thread with specified master filter.*
- **FilmGrainFilter** (QObject \*const parent=nullptr)
- [FilterAction](#) filterAction () override  
*Returns the action description corresponding to currently set options.*
- QString filterIdentifier () const override  
*Return the identifier for this filter in the image history.*
- void readParameters (const [FilterAction](#) &action) override

## Public Member Functions inherited from Digikam::DImgThreadedFilter

- [DImgThreadedFilter](#) ([DImg](#) \*const orgImage, QObject \*const parent, const QString &name=QString())  
*Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter](#) (QObject \*const parent=nullptr, const QString &name=QString())  
*Constructs a filter without argument.*
- virtual void cancelFilter ()  
*Cancel the threaded computation.*
- const QString & filterName ()
- int filterVersion () const
- [DImg](#) getTargetImage ()
- QList< int > multithreadedSteps (int stop, int start=0) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool parametersSuccessfullyRead () const  
*Optional: error handling for readParameters.*
- virtual QString readParametersError (const [FilterAction](#) &actionThatFailed) const
- void setFilterName (const QString &name)
- void setFilterVersion (int version)  
*Replaying a filter action: Set the filter version.*
- void setOriginalImage (const [DImg](#) &orgImage)
- void setupAndStartDirectly (const [DImg](#) &orgImage, [DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void setupFilter (const [DImg](#) &orgImage)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void startFilter ()  
*Start the threaded computation.*
- virtual void startFilterDirectly ()  
*Start computation of this filter, directly in this thread.*
- virtual QList< int > supportedVersions () const

## Public Member Functions inherited from Digikam::DynamicThread

- [DynamicThread](#) (QObject \*const parent=nullptr)  
*This class extends QRunnable, so you have to reimplement virtual void run().*
- ~[DynamicThread](#) () override  
*The destructor calls stop() and wait(), but if you, in your destructor, delete any data that is accessed by your run() method, you must call stop() and wait() before yourself.*
- bool isFinished () const
- bool isRunning () const
- QThread::Priority priority () const
- void setEmitSignals (bool emitThem)
- void setPriority (QThread::Priority priority)  
*Sets the priority for this dynamic thread.*
- State state () const

### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from Digikam::DImgThreadedFilter

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int [progress](#))  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int [progress](#))  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from Digikam::DynamicThread

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from Digikam::DImgThreadedFilter

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.577.1 Member Function Documentation

### 9.577.1.1 filterAction()

`FilterAction` `Digikam::FilmGrainFilter::filterAction ( )` [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.577.1.2 filterIdentifier()

`QString` `Digikam::FilmGrainFilter::filterIdentifier ( )` const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

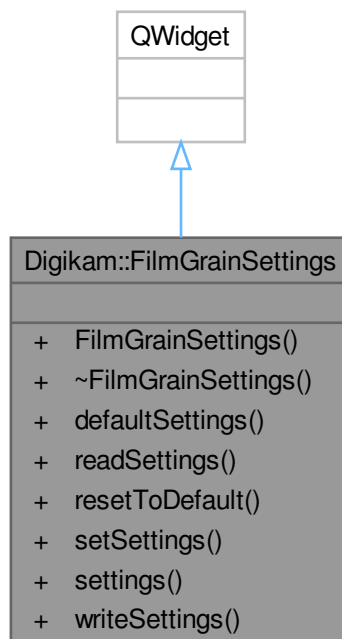
### 9.577.1.3 readParameters()

`void` `Digikam::FilmGrainFilter::readParameters (`  
     const `FilterAction` & `action` ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

## 9.578 Digikam::FilmGrainSettings Class Reference

Inheritance diagram for `Digikam::FilmGrainSettings`:





## Signals

- void **signalSettingsChanged** ()

## Public Member Functions

- **FilmGrainSettings** (QWidget \*const parent)
- **FilmGrainContainer defaultSettings** () const
- void **readSettings** (const KConfigGroup &group)
- void **resetToDefault** ()
- void **setSettings** (const **FilmGrainContainer** &settings)
- **FilmGrainContainer settings** () const
- void **writeSettings** (KConfigGroup &group)

## 9.579 Digikam::Filter Class Reference

### Public Member Functions

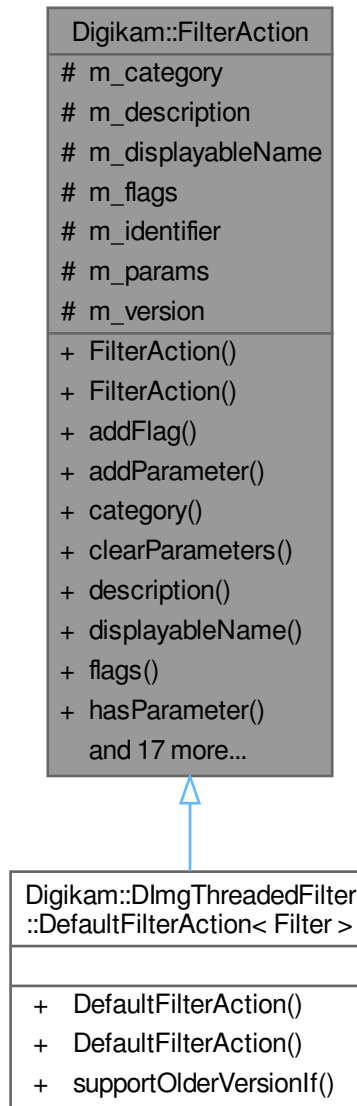
- void **fromString** (const QString &filter)
- bool **match** (const QStringList &wildcards, const QString &name)
- bool **matchesCurrentFilter** (const **CamItemInfo** &item)
- const QStringList & **mimeWildcards** (const QString &mime)
- const QRegularExpression & **regexp** (const QString &wildcard)
- QString **toString** ()

### Public Attributes

- QStringList **fileFilter**
- QHash< QString, QRegularExpression > **filterHash**
- QStringList **ignoreExtensions**
- QStringList **ignoreNames**
- QString **mimeFilter**
- QHash< QString, QStringList > **mimeHash**
- QString **name**
- bool **onlyNew** = false
- QStringList **pathFilter**

## 9.580 Digikam::FilterAction Class Reference

Inheritance diagram for Digikam::FilterAction:



### Public Types

- enum [Category](#) { [ReproducibleFilter](#) = 0 , [ComplexFilter](#) = 1 , [DocumentedHistory](#) = 2 , **CategoryFirst** = [ReproducibleFilter](#) , **CategoryLast** = [DocumentedHistory](#) }
- enum [Flag](#) { [ExplicitBranch](#) = 1 << 0 }
- typedef QFlags< [Flag](#) > **Flags**

## Public Member Functions

- **FilterAction** (const QString &identifier, int version, Category category=ReproducibleFilter)
- void **addFlag** (Flags flags)
- void **addParameter** (const QString &key, const QVariant &value)
  - Sets parameter, removing all other values for the same key.*
- Category **category** () const
- void **clearParameters** ()
  - Clear all parameters.*
- QString **description** () const
  - Returns a description / comment for this action.*
- QString **displayName** () const
- Flags **flags** () const
- bool **hasParameter** (const QString &key) const
- bool **hasParameters** () const
  - Access parameters.*
- QString **identifier** () const
  - Returns a technical identifier for the filter used to produce this action.*
- bool **isNull** () const
- bool **operator==** (const FilterAction &other) const
- QVariant & **parameter** (const QString &key)
- const QVariant **parameter** (const QString &key) const
- template<typename T >
  - T **parameter** (const QString &key) const
    - Returns parameter converted from QVariant to given type.*
- template<typename T >
  - T **parameter** (const QString &key, const T &defaultValue) const
    - Read parameter with a default value: If there is a parameter for the given key, return it converted from QVariant to the template type.*
- QHash< QString, QVariant > & **parameters** ()
- const QHash< QString, QVariant > & **parameters** () const
- void **removeFlag** (Flags flags)
- void **removeParameters** (const QString &key)
  - Removes all parameters for key.*
- void **setDescription** (const QString &description)
- void **setDisplayName** (const QString &displayName)
- void **setFlags** (Flags flags)
- void **setParameters** (const QHash< QString, QVariant > &params)
  - Replaces parameters.*
- int **version** () const
  - Returns the version (>= 1) of the filter used to produce this action.*

## Protected Attributes

- Category **m\_category** = ReproducibleFilter
- QString **m\_description**
- QString **m\_displayableName**
- Flags **m\_flags**
- QString **m\_identifier**
- QHash< QString, QVariant > **m\_params**
- int **m\_version** = 0

## 9.580.1 Member Enumeration Documentation

### 9.580.1.1 Category

enum `Digikam::FilterAction::Category`

## Enumerator

ReproducibleFilter	When given the set of stored parameters and the original data, an identical result will be produced.  <b>Note</b>  Do not change existing values, they are written to XML in files!
ComplexFilter	The operation is documented and a number of parameters may be known, but the identical result cannot be reproduced. It may be possible to produce a sufficiently similar result.
DocumentedHistory	The source images are known, a textual description may be added, but there is no way to automatically replay.

**9.580.1.2 Flag**

```
enum Digikam::FilterAction::Flag
```

## Enumerator

ExplicitBranch	The editing step of this filter action explicitly branches from the parent. This is an optional hint that the result is meant as a new version.
----------------	---

**9.580.2 Member Function Documentation****9.580.2.1 description()**

```
QString Digikam::FilterAction::description ( ) const
```

In the case of DocumentedHistory, this may be the most useful value.

**9.580.2.2 hasParameters()**

```
bool Digikam::FilterAction::hasParameters ( ) const
```

A parameters is a key -> value pair. Keys need to be unique.

**9.580.2.3 identifier()**

```
QString Digikam::FilterAction::identifier ( ) const
```

Can include a namespace. Example: digikam:charcoal

#### 9.580.2.4 parameter()

```
template<typename T >
T Digikam::FilterAction::parameter (
    const QString & key,
    const T & defaultValue ) const [inline]
```

If there is no parameter, return the given default value.

#### 9.580.2.5 version()

```
int Digikam::FilterAction::version ( ) const
```

When a filter / tool is found by the identifier, it can decide by the version if it supports this action and which parameters it expects.

### 9.580.3 Member Data Documentation

#### 9.580.3.1 m\_category

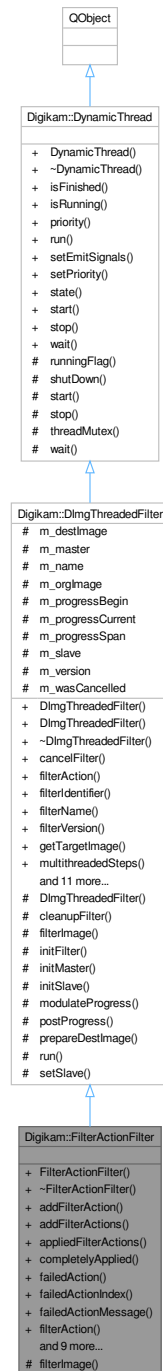
```
Category Digikam::FilterAction::m_category = ReproducibleFilter [protected]
```

## Note

Value class, do not create a d-pointer

## 9.581 Digikam::FilterActionFilter Class Reference

Inheritance diagram for Digikam::FilterActionFilter:



## Public Member Functions

- **FilterActionFilter** (QObject \*const parent=nullptr)
  - A meta-filter applying other filter according to a list of FilterActions.*
- void **addFilterAction** (const [FilterAction](#) &action)
- void **addFilterActions** (const QList< [FilterAction](#) > &actions)
- QList< [FilterAction](#) > **appliedFilterActions** () const
  - Returns the list of applied filter actions.*
- bool **completelyApplied** () const
  - After the thread was run, you can find out if application was successful.*
- [FilterAction](#) **failedAction** () const
- int **failedActionIndex** () const
- QString **failedActionMessage** () const
- [FilterAction](#) **filterAction** () override
  - These methods do not make sense here.*
- QList< [FilterAction](#) > **filterActions** () const
- QString **filterIdentifier** () const override
  - Return the identifier for this filter in the image history.*
- bool **isComplexAction** () const
  - Returns true if all FilterActions are reproducible or are ComplexFilters.*
- bool **isReproducible** () const
  - Returns true if all FilterActions are reproducible.*
- bool **isSupported** () const
  - Returns true if all actions are supported.*
- void **readParameters** (const [FilterAction](#) &) override
- void **setContinueOnError** (bool cont)
  - Per default, the filter will stop when it encounters an unsupported action.*
- void **setFilterAction** (const [FilterAction](#) &action)
- void **setFilterActions** (const QList< [FilterAction](#) > &actions)
  - Set - or add to existing list - the given filter actions.*

## Public Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImg](#) \*const orgImage, QObject \*const parent, const QString &name=QString())
  - Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter](#) (QObject \*const parent=nullptr, const QString &name=QString())
  - Constructs a filter without argument.*
- virtual void **cancelFilter** ()
  - Cancel the threaded computation.*
- const QString & **filterName** ()
- int **filterVersion** () const
- [DImg](#) **getTargetImage** ()
- QList< int > **multithreadedSteps** (int stop, int start=0) const
  - This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool **parametersSuccessfullyRead** () const
  - Optional: error handling for readParameters.*
- virtual QString **readParametersError** (const [FilterAction](#) &actionThatFailed) const
- void **setFilterName** (const QString &name)
- void **setFilterVersion** (int version)
  - Replaying a filter action: Set the filter version.*
- void **setOriginalImage** (const [DImg](#) &orgImage)



- void **setupAndStartDirectly** (const [DImg](#) &orgImage, [DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void **setupFilter** (const [DImg](#) &orgImage)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void **startFilter** ()  
*Start the threaded computation.*
- virtual void **startFilterDirectly** ()  
*Start computation of this filter, directly in this thread.*
- virtual [QList< int >](#) **supportedVersions** () const

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread](#) ([QObject](#) \*const parent=nullptr)  
*This class extends [QRunnable](#), so you have to reimplement virtual void [run\(\)](#).*
- [~DynamicThread](#) () override  
*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool **isFinished** () const
- bool **isRunning** () const
- [QThread::Priority](#) **priority** () const
- void **setEmitSignals** (bool emitThem)
- void **setPriority** ([QThread::Priority](#) priority)  
*Sets the priority for this dynamic thread.*
- State **state** () const

## Protected Member Functions

- void **filterImage** () override  
*Main image filter method.*

## Protected Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const [QString](#) &name=[QString](#)())  
*Support for chaining two filters as master and thread.*
- virtual void **cleanupFilter** ()  
*Clean up filter data if necessary, called by [stopComputation\(\)](#) method.*
- virtual void **initFilter** ()  
*Start filter operation before threaded method.*
- void **initMaster** ()
- void **initSlave** ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int **modulateProgress** (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void **postProgress** (int progress)  
*Emit progress info.*
- virtual void **prepareDestImage** ()
- void **run** () override  
*List of threaded operations by filter.*
- void **setSlave** ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool **runningFlag** () const volatile  
*In you [run\(\)](#) method, you shall regularly check for [runningFlag\(\)](#) and cleanup and return if false.*
- void **shutDown** ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call [stop\(\)](#) and [wait\(\)](#), knowing that nothing will call [start\(\)](#) anymore after this 3) Be sure the thread will never be running at destruction.*
- void **start** (QMutexLocker< QMutex > &locker)  
*Doing the same as [start\(\)](#), [stop\(\)](#) and [wait](#) above, provide it with a locked QMutexLocker on mutex().*
- void **stop** (const QMutexLocker< QMutex > &locker)
- QMutex \* **threadMutex** () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void **wait** (QMutexLocker< QMutex > &locker)

## Additional Inherited Members

## Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

## Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

## Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

## Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if [emitSignals](#) is enabled.*

## Protected Attributes inherited from [Digikam::DImgThreadedFilter](#)

- [DImg m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter \\* m\\_master](#) = nullptr  
*The master of this slave filter.*
- [QString m\\_name](#)  
*Filter name.*
- [DImg m\\_orgImage](#)  
*Copy of original Image data.*
- [int m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- [int m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in [postProgress\(\)](#).*
- [int m\\_progressSpan](#) = 0
- [DImgThreadedFilter \\* m\\_slave](#) = nullptr  
*The current slave.*
- [int m\\_version](#) = 1
- [bool m\\_wasCancelled](#) = false

### 9.581.1 Member Function Documentation

#### 9.581.1.1 [appliedFilterActions\(\)](#)

```
QList< FilterAction > Digikam::FilterActionFilter::appliedFilterActions ( ) const
```

This is probably identical to [filterActions](#), but it can differ in some situations:

- if [completelyApplied\(\)](#) is false, it will contain only the successful actions
- the list is regenerated by the filters. If [filterActions](#) contains actions with an older version, still supported by the filter, the filter will now possibly return the newer, current version

#### 9.581.1.2 [completelyApplied\(\)](#)

```
bool Digikam::FilterActionFilter::completelyApplied ( ) const
```

A precondition is that at least [isComplexAction\(\)](#) and [isSupported\(\)](#) returns true. If all filters applied cleanly, [completelyApplied\(\)](#) returns true. [appliedActions\(\)](#) returns all applied actions, if [completelyApplied\(\)](#), the same as [filterActions\(\)](#). If not [completelyApplied](#), [failedAction\(\)](#) returns the action that failed, [failedActionIndex](#) its index in [filterActions\(\)](#), and [failedActionMessage](#) an optional error message. Note that [finished\(true\)](#) does not mean that [completelyApplied\(\)](#) is also true.

#### 9.581.1.3 [filterAction\(\)](#)

```
FilterAction Digikam::FilterActionFilter::filterAction ( ) [inline], [override], [virtual]
```

Use [filterActions](#).

Implements [Digikam::DImgThreadedFilter](#).

#### 9.581.1.4 filterIdentifier()

```
QString Digikam::FilterActionFilter::filterIdentifier ( ) const [inline], [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

#### 9.581.1.5 filterImage()

```
void Digikam::FilterActionFilter::filterImage ( ) [override], [protected], [virtual]
```

Override in subclass.

Implements [Digikam::DImgThreadedFilter](#).

#### 9.581.1.6 isComplexAction()

```
bool Digikam::FilterActionFilter::isComplexAction ( ) const
```

That means the identical result may not be reproducible, but a sufficiently similar result may be available and apply will probably complete.

#### 9.581.1.7 readParameters()

```
void Digikam::FilterActionFilter::readParameters (
    const FilterAction & ) [inline], [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

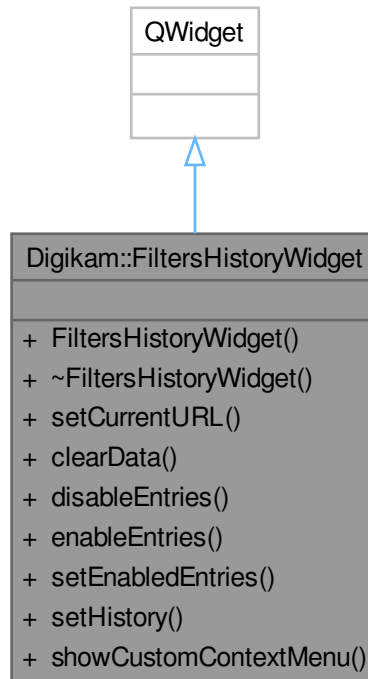
#### 9.581.1.8 setContinueOnError()

```
void Digikam::FilterActionFilter::setContinueOnError (
    bool cont )
```

If you want it to continue, set this to true. Only the last occurred error will then be reported.

## 9.582 Digikam::FiltersHistoryWidget Class Reference

Inheritance diagram for Digikam::FiltersHistoryWidget:



### Public Slots

- void **clearData** ()
- void **disableEntries** (int count)
- void **enableEntries** (int count)
- void **setEnabledEntries** (int count)
- void **setHistory** (const [DImageHistory](#) &history)
- void **showCustomContextMenu** (const QPoint &position)

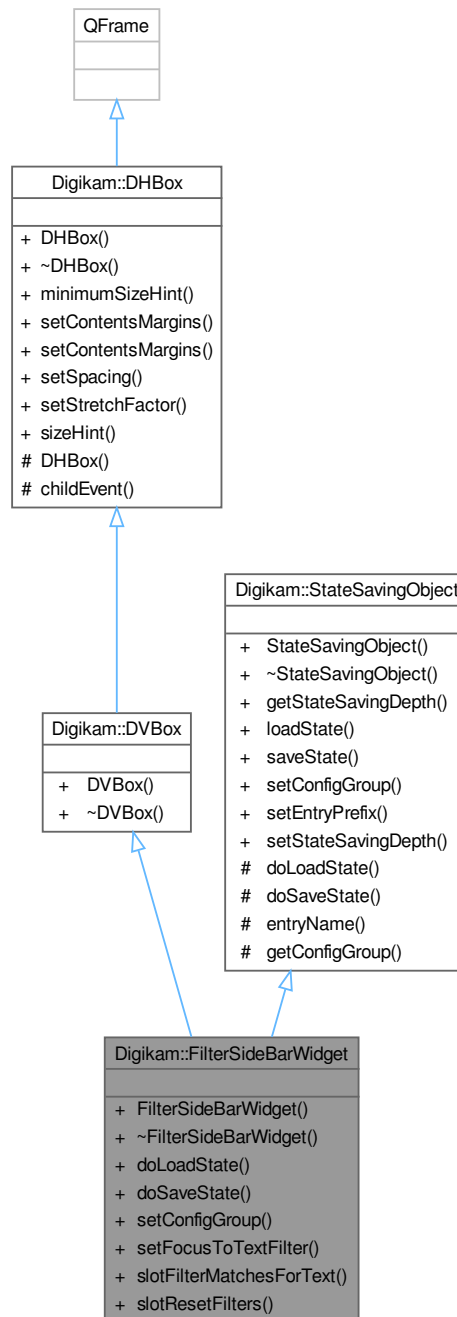
### Public Member Functions

- **FiltersHistoryWidget** (QWidget \*const parent)
- void **setCurrentURL** (const QUrl &url=QUrl())

## 9.583 Digikam::FilterSideBarWidget Class Reference

[Sidebar](#) widget containing the all filter widgets.

Inheritance diagram for Digikam::FilterSideBarWidget:



### Public Slots

- void **slotFilterMatchesForText** (bool)
- void **slotResetFilters** ()

*Resets all selected filters.*

## Signals

- void **signalGeolocationFilterChanged** ([ItemFilterSettings::GeolocationCondition](#))
- void **signalMimeTypeFilterChanged** (int)
- void **signalRatingFilterChanged** (int, [ItemFilterSettings::RatingCondition](#), bool)
- void **signalSearchTextFilterChanged** (const [SearchTextFilterSettings](#) &)
- void **signalTagFilterChanged** (const QList< int > &includedTags, const QList< int > &excludedTags, [ItemFilterSettings::MatchingCondition](#) matchingCond, bool showUnTagged, const QList< int > &clTagIds, const QList< int > &plTagIds)

*Emitted if the selected filter has changed.*

## Public Member Functions

- [FilterSideBarWidget](#) (QWidget \*const parent, [TagModel](#) \*const tagFilterModel)  
*Constructor.*
- [~FilterSideBarWidget](#) () override  
*Destructor.*
- void [doLoadState](#) () override  
*Implement this hook method for state loading.*
- void [doSaveState](#) () override  
*Implement this hook method for state saving.*
- void [setConfigGroup](#) (const KConfigGroup &group) override  
*Sets a dedicated config group that will be used to store and reload the state from.*
- void [setFocusToTextFilter](#) ()

## Public Member Functions inherited from [Digikam::DVBox](#)

- [DVBox](#) (QWidget \*const parent=nullptr)

## Public Member Functions inherited from [Digikam::DHBox](#)

- [DHBox](#) (QWidget \*const parent=nullptr)
- QSize [minimumSizeHint](#) () const override
- void [setContentsMargins](#) (const QMargins &margins)
- void [setContentsMargins](#) (int left, int top, int right, int bottom)
- void [setSpacing](#) (int space)
- void [setStretchFactor](#) (QWidget \*const widget, int stretch)
- QSize [sizeHint](#) () const override

## Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (QObject \*const host)  
*Constructor.*
- virtual [~StateSavingObject](#) ()  
*Destructor.*
- [StateSavingDepth](#) [getStateSavingDepth](#) () const  
*Returns the depth used for state saving or loading.*
- void [loadState](#) ()  
*Invokes loading the class' state.*
- void [saveState](#) ()  
*Invokes saving the class' state.*
- virtual void [setEntryPrefix](#) (const QString &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void [setStateSavingDepth](#) (const [StateSavingDepth](#) depth)  
*Sets the depth used for state saving or loading.*

## Additional Inherited Members

### Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }

*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

### Protected Member Functions inherited from [Digikam::DHBox](#)

- [DHBox](#) (bool vertical, QWidget \*const parent)
- void [childEvent](#) (QChildEvent \*e) override

### Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- QString [entryName](#) (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup [getConfigGroup](#) () const

*Returns the config group that must be used for state saving and loading.*

## 9.583.1 Detailed Description

### Author

jwienke

## 9.583.2 Constructor & Destructor Documentation

### 9.583.2.1 [FilterSideBarWidget\(\)](#)

```
Digikam::FilterSideBarWidget::FilterSideBarWidget (
    QWidget *const parent,
    TagModel *const tagFilterModel ) [explicit]
```

#### Parameters

<i>parent</i>	the parent for qt parent child mechanism
<i>tagFilterModel</i>	tag model to work on

## 9.583.3 Member Function Documentation

### 9.583.3.1 [doLoadState\(\)](#)

```
void Digikam::FilterSideBarWidget::doLoadState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).



**9.583.3.2 doSaveState()**

```
void Digikam::FilterSideBarWidget::doSaveState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

**9.583.3.3 setConfigGroup()**

```
void Digikam::FilterSideBarWidget::setConfigGroup (
    const KConfigGroup & group ) [override], [virtual]
```

If this method is not called, a group based on the object name is used.

You can re-implement this method to pass the group set here to child objects. Don't forget to call this method in your implementation.

**Parameters**

<i>group</i>	config group to use for state saving and restoring
--------------	--

Reimplemented from [Digikam::StateSavingObject](#).

**9.583.3.4 signalTagFilterChanged**

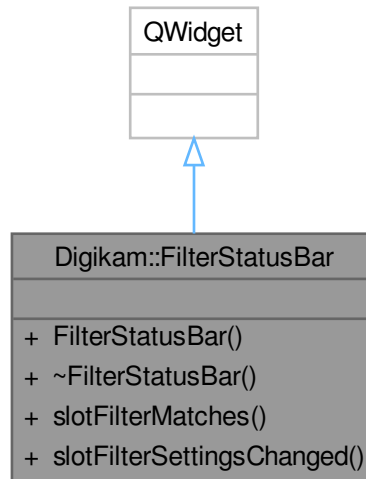
```
void Digikam::FilterSideBarWidget::signalTagFilterChanged (
    const QList< int > & includedTags,
    const QList< int > & excludedTags,
    ItemFilterSettings::MatchingCondition matchingCond,
    bool showUnTagged,
    const QList< int > & clTagIds,
    const QList< int > & plTagIds ) [signal]
```

**Parameters**

<i>includedTags</i>	a list of included tag ids
<i>excludedTags</i>	a list of excluded tag ids
<i>matchingCond</i>	condition to join the selected tags
<i>showUnTagged</i>	if this is true, only photos without a tag shall be shown
<i>clTagIds</i>	a list of color label tag ids
<i>plTagIds</i>	a list of pick label tag ids

## 9.584 Digikam::FilterStatusBar Class Reference

Inheritance diagram for Digikam::FilterStatusBar:



### Public Slots

- void **slotFilterMatches** (bool)
- void **slotFilterSettingsChanged** (const [ItemFilterSettings](#) &settings)

### Signals

- void **signalPopupFiltersView** ()
- void **signalResetFilters** ()

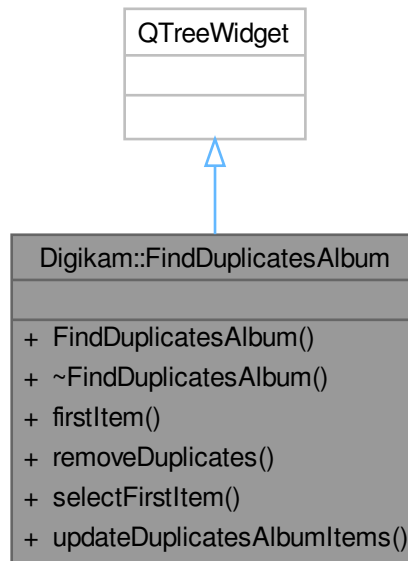
### Public Member Functions

- **FilterStatusBar** (QWidget \*const parent)

## 9.585 Digikam::FindDuplicatesAlbum Class Reference

The [FindDuplicatesAlbum](#) class Widgets used to show all reference images.

Inheritance diagram for Digikam::FindDuplicatesAlbum:

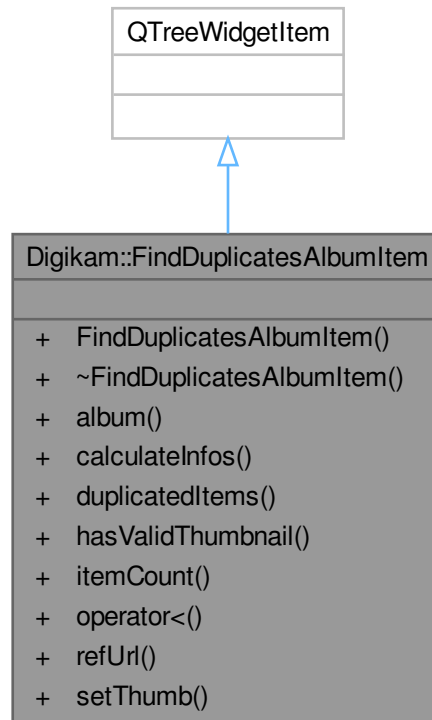


### Public Member Functions

- **FindDuplicatesAlbum** (`QWidget *const parent=nullptr`)
- `QTreeWidgetItem * firstItem ()`
- void **removeDuplicates** ()
- void **selectFirstItem** ()
- void **updateDuplicatesAlbumItems** (`const QList< SAlbum * > &sAlbumsToRebuild, const QList< qlong-long > &deletedImages`)

## 9.586 Digikam::FindDuplicatesAlbumItem Class Reference

Inheritance diagram for Digikam::FindDuplicatesAlbumItem:



### Public Types

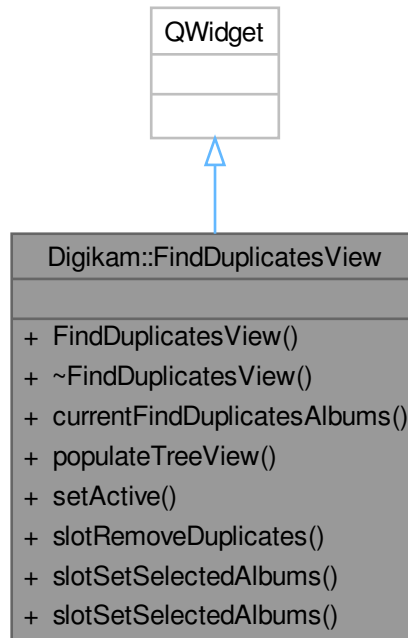
- enum `Column` {  
**REFERENCE\_IMAGE** = 0 , **REFERENCE\_DATE** = 1 , **REFERENCE\_ALBUM** = 2 , **RESULT\_COUNT** = 3 ,  
**AVG\_SIMILARITY** = 4 }

### Public Member Functions

- `FindDuplicatesAlbumItem` (`QTreeWidgetItem *const parent`, `SAlbum *const album`)
- `SAlbum * album` () const
- void `calculateInfos` (const `QList< qlonglong >` &deletedImages=`QList< qlonglong >()`)  
*Calculates the duplicates count and average similarity.*
- `QList< ItemInfo > duplicatedItems` ()
- bool `hasValidThumbnail` () const
- int `itemCount` () const  
*Returns the item count.*
- bool `operator<` (const `QTreeWidgetItem &other`) const override
- `QUrl refUrl` () const
- void `setThumb` (const `QPixmap &pix`, bool hasThumb=true)

## 9.587 Digikam::FindDuplicatesView Class Reference

Inheritance diagram for Digikam::FindDuplicatesView:



### Public Slots

- void **slotRemoveDuplicates** ()
- void **slotSetSelectedAlbums** (const QList< [PAlbum](#) \* > &albums)
- void **slotSetSelectedAlbums** (const QList< [TAlbum](#) \* > &albums)

### Signals

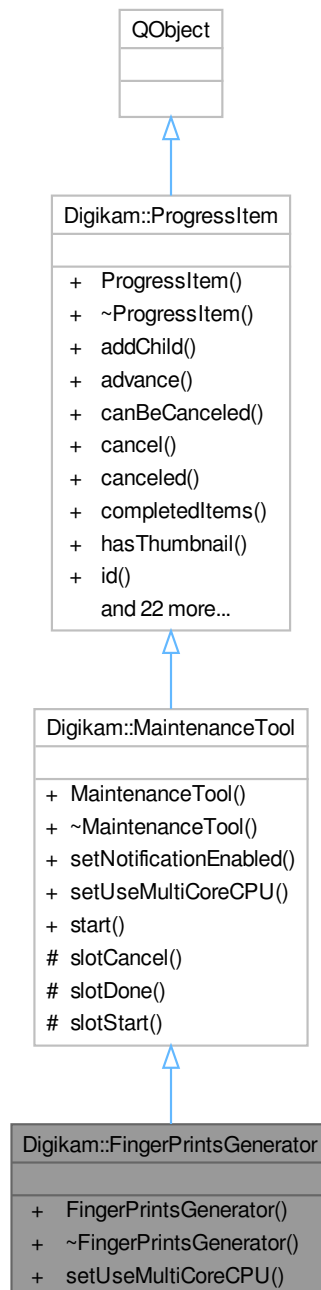
- void **signalScanNotification** (const QString &msg, int type)

### Public Member Functions

- **FindDuplicatesView** (QWidget \*const parent=nullptr)
- QList< [SAlbum](#) \* > **currentFindDuplicatesAlbums** () const
- void **populateTreeView** ()
- void **setActive** (bool val)

## 9.588 Digikam::FingerPrintsGenerator Class Reference

Inheritance diagram for Digikam::FingerPrintsGenerator:



### Signals

- void **signalScanNotification** (const QString &msg, int type)

## Signals inherited from [Digikam::MaintenanceTool](#)

- void **signalCanceled** ()  
*Emit when process is canceled.*
- void **signalComplete** ()  
*Emit when process is done (not canceled).*

## Signals inherited from [Digikam::ProgressItem](#)

- void [progressItemAdded](#) ([ProgressItem](#) \*item)  
*Emitted when a new [ProgressItem](#) is added.*
- void [progressItemCanceled](#) ([ProgressItem](#) \*item)  
*Emitted when an item was canceled.*
- void **progressItemCanceledById** (const QString &id)
- void [progressItemCompleted](#) ([ProgressItem](#) \*item)  
*Emitted when a progress item was completed.*
- void [progressItemLabel](#) ([ProgressItem](#) \*item, const QString &label)  
*Emitted when the label of an item changed.*
- void [progressItemProgress](#) ([ProgressItem](#) \*item, unsigned int v)  
*Emitted when the progress value of an item changes.*
- void [progressItemStatus](#) ([ProgressItem](#) \*item, const QString &mess)  
*Emitted when the status message of an item changed.*
- void [progressItemThumbnail](#) ([ProgressItem](#) \*item, const QPixmap &thumb)  
*Emitted when the thumbnail data must be set in item.*
- void [progressItemUsesBusyIndicator](#) ([ProgressItem](#) \*item, bool value)  
*Emitted when the busy indicator state of an item changes.*

## Public Member Functions

- [FingerPrintsGenerator](#) (const bool rebuildAll, const AlbumList &list=AlbumList(), [ProgressItem](#) \*const parent=nullptr)  
*Constructor using AlbumList as argument.*
- void [setUseMultiCoreCPU](#) (bool b) override  
*Re-implement this method if your tool is able to use multi-core CPU to process item in parallel.*

## Public Member Functions inherited from [Digikam::MaintenanceTool](#)

- **MaintenanceTool** (const QString &id, [ProgressItem](#) \*const parent=nullptr)
- void **setNotificationEnabled** (bool b)  
*If true, show a notification message on desktop notification manager with time elapsed to run process.*

## Public Member Functions inherited from [Digikam::ProgressItem](#)

- **ProgressItem** ([ProgressItem](#) \*const [parent](#), const QString &[id](#), const QString &[label](#), const QString &[status](#), bool [canBeCanceled](#), bool hasThumb)
- void **addChild** ([ProgressItem](#) \*const kiddo)
- bool [advance](#) (unsigned int v)
  - Advance total items processed by n values and update percentage in progressbar.*
- bool [canBeCanceled](#) () const
- void **cancel** ()
- bool **canceled** () const
- unsigned int **completedItems** () const
- bool [hasThumbnail](#) () const
- const QString & [id](#) () const
- bool **incCompletedItems** (unsigned int v=1)
- void **incTotalItems** (unsigned int v=1)
- const QString & [label](#) () const
- [ProgressItem](#) \* [parent](#) () const
- unsigned int [progress](#) () const
- void **removeChild** ([ProgressItem](#) \*const kiddo)
- void **reset** ()
  - Reset the progress value of this item to 0 and the status string to the empty string.*
- void [setComplete](#) ()
  - Tell the item it has finished.*
- bool **setCompletedItems** (unsigned int v)
- void [setLabel](#) (const QString &v)
- void [setProgress](#) (unsigned int v)
  - Set the progress (percentage of completion) value of this item.*
- void [setShowAtStart](#) (bool [showAtStart](#))
  - Set the property to pop-up item when it's added in progress manager.*
- void [setStatus](#) (const QString &v)
  - Set the string to be used for showing this item's current status.*
- void [setThumbnail](#) (const QIcon &icon)
  - Sets whether this item has a thumbnail.*
- void **setTotalItems** (unsigned int v)
- void [setUsesBusyIndicator](#) (bool useBusyIndicator)
  - Sets whether this item uses a busy indicator instead of real progress for its progress bar.*
- bool [showAtStart](#) () const
- const QString & [status](#) () const
- bool **totalCompleted** () const
- unsigned int **totalItems** () const
- void **updateProgress** ()
  - Recalculate progress according to total/completed items and update.*
- bool [usesBusyIndicator](#) () const

## Additional Inherited Members

## Public Slots inherited from [Digikam::MaintenanceTool](#)

- void **start** ()



## Protected Slots inherited from [Digikam::MaintenanceTool](#)

- virtual void **slotCancel** ()
- virtual void **slotDone** ()
- virtual void **slotStart** ()

### 9.588.1 Constructor & Destructor Documentation

#### 9.588.1.1 FingerPrintsGenerator()

```
Digikam::FingerPrintsGenerator::FingerPrintsGenerator (
    const bool rebuildAll,
    const AlbumList & list = AlbumList(),
    ProgressItem *const parent = nullptr ) [explicit]
```

If list is empty, whole Albums collection is processed.

### 9.588.2 Member Function Documentation

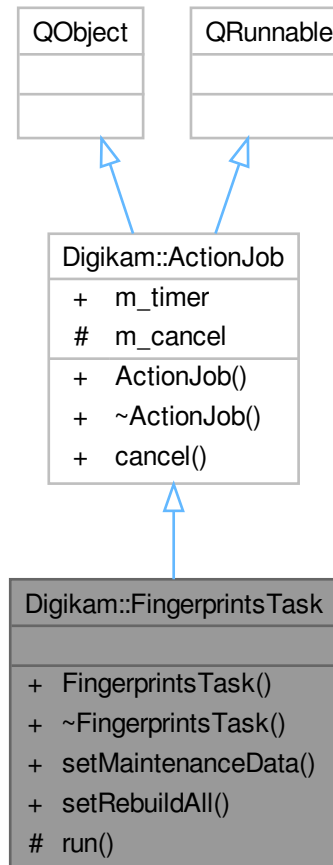
#### 9.588.2.1 setUseMultiCoreCPU()

```
void Digikam::FingerPrintsGenerator::setUseMultiCoreCPU (
    bool ) [override], [virtual]
```

Reimplemented from [Digikam::MaintenanceTool](#).

## 9.589 Digikam::FingerprintsTask Class Reference

Inheritance diagram for Digikam::FingerprintsTask:



### Signals

- void **signalFinished** (const [ItemInfo](#) &, const QImage &)

### Signals inherited from [Digikam::ActionJob](#)

- void **signalDone** ()
  - Use this signal in your implementation to inform [ActionThreadBase](#) manager the job is done.*
- void **signalProgress** (int)
  - Use this signal in your implementation to inform [ActionThreadBase](#) manager the job progress.*
- void **signalStarted** ()
  - Use this signal in your implementation to inform [ActionThreadBase](#) manager that job is started.*

### Public Member Functions

- void **setMaintenanceData** ([MaintenanceData](#) \*const data=nullptr)
- void **setRebuildAll** (bool b)

### Public Member Functions inherited from [Digikam::ActionJob](#)

- **ActionJob** (QObject \*const parent=nullptr)  
*Constructor which delegate deletion of QRunnable instance to [ActionThreadBase](#), not QThreadPool.*
- **~ActionJob** () override  
*Re-implement destructor in you implementation.*

### Protected Member Functions

- void **run** () override

### Additional Inherited Members

### Public Slots inherited from [Digikam::ActionJob](#)

- void **cancel** ()  
*Call this method to cancel job.*

### Public Attributes inherited from [Digikam::ActionJob](#)

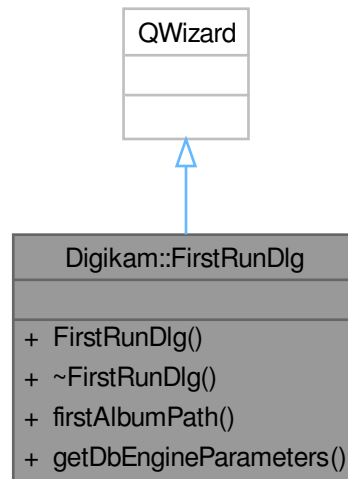
- QElapsedTimer **m\_timer**  
*Timer to determine the running time of the job.*

### Protected Attributes inherited from [Digikam::ActionJob](#)

- bool **m\_cancel** = false  
*You can use this boolean in your implementation to know if job must be canceled.*

## 9.590 Digikam::FirstRunDlg Class Reference

Inheritance diagram for Digikam::FirstRunDlg:



### Public Member Functions

- **FirstRunDlg** (`QWidget *const parent=nullptr`)
- `QString` **firstAlbumPath** () const
- `DbEngineParameters` **getDbEngineParameters** () const

## 9.591 Digikam::FocusPoint Class Reference

### Public Types

- enum `TypePoint` { `Inactive` = 0 , `InFocus` = 1 , `Selected` = 2 , `SelectedInFocus` = 3 }

### Public Member Functions

- **FocusPoint** (const `FocusPoint` &other)
- **FocusPoint** (const `QRectF` &rectF)
- **FocusPoint** (float x\_position, float y\_position, float width, float height)
- `FocusPoint` (float x\_position, float y\_position, float width, float height, `TypePoint` type)
  - Focus point container constructors.*
- `QPointF` **getCenterPosition** () const
- `QRectF` **getRect** () const
- `QRect` **getRectBySize** (const `QSize` &size) const
  - Return the real aera properties in image coordinates depending of the size.*
- `QSizeF` **getSize** () const

- [TypePoint](#) `getType ()` const
- `QString` `getTypeDescription ()` const
- [FocusPoint](#) & `operator=` (const [FocusPoint](#) &other)  
*Equivalent to the copy constructor.*
- void `setCenterPosition` (float `x_position`, float `y_position`)  
*Accessors to relative properties of focus point area.*
- void `setRect` (const `QRectF` &rectF)
- void `setSize` (float `width`, float `height`)
- void `setType` ([TypePoint](#) `type`)  
*Focus point type properties accessor.*

## 9.591.1 Member Enumeration Documentation

### 9.591.1.1 TypePoint

```
enum Digikam::FocusPoint::TypePoint
```

Enumerator

Inactive	The AF-point is not active.
InFocus	The AF-point is in focus.
Selected	The AF-point is selected but not in focus.
SelectedInFocus	The AF-point is selected and in focus.

## 9.591.2 Constructor & Destructor Documentation

### 9.591.2.1 FocusPoint()

```
Digikam::FocusPoint::FocusPoint (
    float x_position,
    float y_position,
    float width,
    float height,
    TypePoint type )
```

Position and size are in float and a relative to the original image size. Typically, the area is define as percents of values depending of image size used to extract information from metadata. Like this, focus area can be drawn easily over a resized version of image.

## 9.591.3 Member Function Documentation

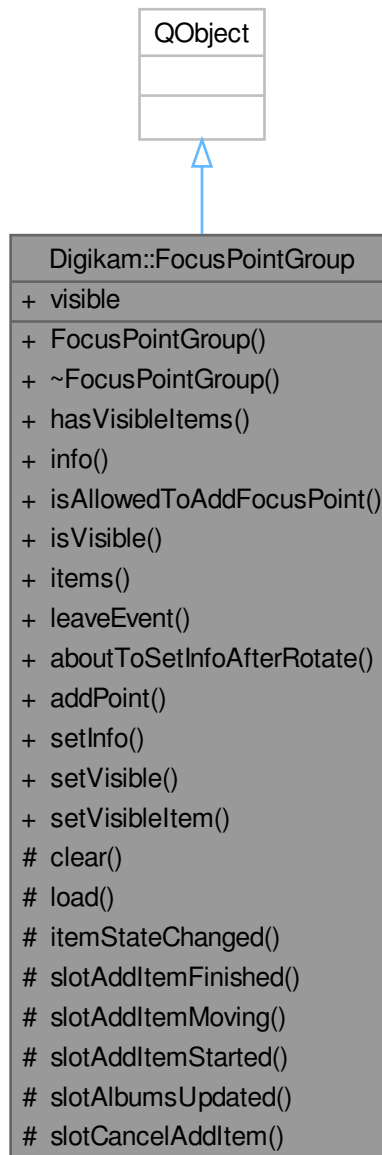
### 9.591.3.1 setType()

```
void Digikam::FocusPoint::setType (
    TypePoint type )
```

See [TypePoint](#) enum definition for details.

## 9.592 Digikam::FocusPointGroup Class Reference

Inheritance diagram for Digikam::FocusPointGroup:



### Public Slots

- void **aboutToSetInfoAfterRotate** (const [ItemInfo](#) &info)
- void **addPoint** ()
- void **setInfo** (const [ItemInfo](#) &info)
  - Sets the current [ItemInfo](#).
- void **setVisible** (bool visible)
  - Shows or hides the frames.
- void **setVisibleItem** ([RegionFrameItem](#) \*const item)

## Public Member Functions

- **FocusPointGroup** ([GraphicsDImgView](#) \*const view)
- bool **hasVisibleItems** () const
- [ItemInfo](#) **info** () const
- bool **isAllowedToAddFocusPoint** () const
- bool **isVisible** () const
- QList< [RegionFrameItem](#) \* > **items** () const
- void **leaveEvent** (QEvent \*)

## Protected Slots

- void **itemStateChanged** (int)
- void **slotAddItemFinished** (const QRectF &rect)
- void **slotAddItemMoving** (const QRectF &rect)
- void **slotAddItemStarted** (const QPointF &pos)
- void **slotAlbumsUpdated** (int type)
- void **slotCancelAddItem** ()

## Protected Member Functions

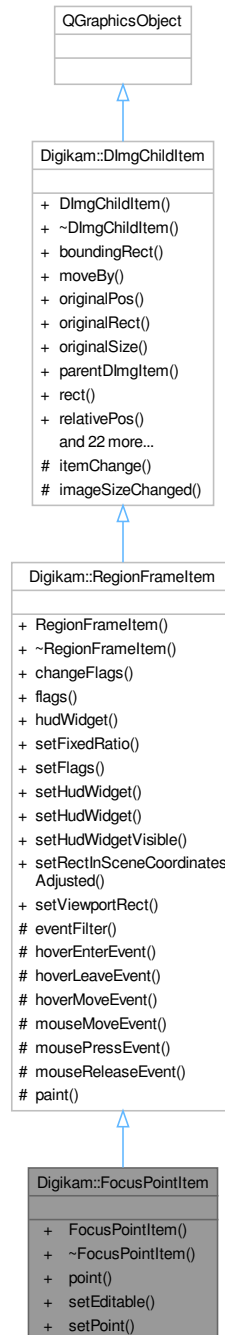
- void **clear** ()
- void **load** ()

## Properties

- bool **visible**

## 9.593 Digikam::FocusPointItem Class Reference

Inheritance diagram for Digikam::FocusPointItem:



### Public Member Functions

- **FocusPointItem** (`QGraphicsItem *const parent`)
- **FocusPoint point** () const
- void **setEditable** (bool allowEdit)
- void **setPoint** (const [FocusPoint](#) &point)



## Public Member Functions inherited from Digikam::RegionFrameItem

- **RegionFrameItem** (QGraphicsItem \*const parent)
- void **changeFlags** (Flags flags, bool addOrRemove)
- Flags **flags** () const
- QGraphicsWidget \* **hudWidget** () const
- void **setFixedRatio** (double ratio)
- void **setFlags** (Flags flags)
- void **setHudWidget** (QGraphicsWidget \*const hudWidget)
  - Sets a widget item as HUD item.*
- void **setHudWidget** (QWidget \*const widget, Qt::WindowFlags wFlags=Qt::WindowFlags())
- void **setHudWidgetVisible** (bool visible)
- void **setRectInSceneCoordinatesAdjusted** (const QRectF &rect)

## Public Member Functions inherited from Digikam::DImgChildItem

- **DImgChildItem** (QGraphicsItem \*const parent=nullptr)
  - This is a base class for items that are positioned on top of a [GraphicsDImgItem](#), positioned in relative coordinates, i.e.*
- QRectF **boundingRect** () const override
  - Reimplemented.*
- void **moveBy** (qreal dx, qreal dy)
- QPoint **originalPos** () const
- QRect **originalRect** () const
  - Returns the position and size in coordinates of the original image.*
- QSize **originalSize** () const
- [GraphicsDImgItem](#) \* **parentDImgItem** () const
  - If the parent item is a [GraphicsDImgItem](#), return it, if the parent item is null or of a different class, returns 0.*
- QRectF **rect** () const
  - Returns position and size of this item, in coordinates of the parent [DImg](#) with the current zoom.*
- QPointF **relativePos** () const
- QRectF **relativeRect** () const
  - Returns the position and size relative to the [DImg](#) displayed in the parent item.*
- QSizeF **relativeSize** () const
- void **setOriginalPos** (const QPointF &posInOriginal)
  - Sets the position and size of this item, in coordinates of the original image.*
- void **setOriginalPos** (qreal x, qreal y)
- void **setOriginalRect** (const QRectF &rect)
- void **setOriginalRect** (qreal x, qreal y, qreal width, qreal height)
- void **setOriginalSize** (const QSizeF &sizeInOriginal)
- void **setOriginalSize** (qreal width, qreal height)
- void **setPos** (const QPointF &zoomedPos)
  - Sets the position and size of this item, in coordinates of the parent [DImg](#) item.*
- void **setPos** (qreal x, qreal y)
- void **setRect** (const QRectF &rect)
- void **setRect** (qreal x, qreal y, qreal width, qreal height)
- void **setRectInSceneCoordinates** (const QRectF &rect)
  - Equivalent to mapping the scene coordinates to the parent item, and calling [setRect\(\)](#).*
- void **setRelativePos** (const QPointF &relativePosition)
  - Sets the position and size of this item, relative to the [DImg](#) displayed in the parent item.*
- void **setRelativePos** (qreal x, qreal y)
- void **setRelativeRect** (const QRectF &rect)
- void **setRelativeRect** (qreal x, qreal y, qreal width, qreal height)
- void **setRelativeSize** (const QSizeF &relativeSize)
- void **setRelativeSize** (qreal width, qreal height)
- void **setSize** (const QSizeF &zoomedSize)
- void **setSize** (qreal width, qreal height)
- QSizeF **size** () const

## Additional Inherited Members

### Public Types inherited from [Digikam::RegionFrameItem](#)

- enum **Flag** { **NoFlags** = 0 , **ShowResizeHandles** = 1 << 0 , **MoveByDrag** = 1 << 1 , **GeometryEditable** = ShowResizeHandles | MoveByDrag }
- typedef QFlags< Flag > **Flags**

### Public Slots inherited from [Digikam::RegionFrameItem](#)

- void [setViewportRect](#) (const QRectF &rect)  
*The associated HUD item is dynamically moved to be visible.*

### Signals inherited from [Digikam::RegionFrameItem](#)

- void **geometryEdited** ()

### Signals inherited from [Digikam::DImgChildItem](#)

- void **geometryChanged** ()
- void **geometryOnImageChanged** ()
- void [positionChanged](#) ()  
*These signals are emitted in any case when the geometry changed: Either after changing the geometry relative to the original image, or when the size of the parent [GraphicsDImgItem](#) changed (zooming).*
- void [positionOnImageChanged](#) ()  
*These signals are emitted when the geometry, relative to the original image, of this item has changed.*
- void **sizeChanged** ()
- void **sizeOnImageChanged** ()

### Protected Slots inherited from [Digikam::DImgChildItem](#)

- void **imageSizeChanged** (const QSizeF &)

### Protected Member Functions inherited from [Digikam::RegionFrameItem](#)

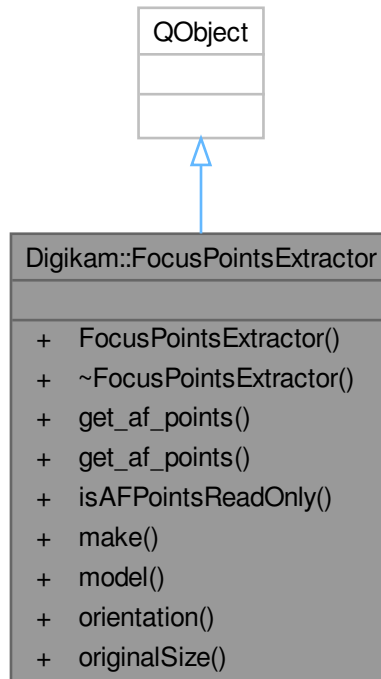
- bool **eventFilter** (QObject \*watched, QEvent \*event) override
- void **hoverEnterEvent** (QGraphicsSceneHoverEvent \*event) override
- void **hoverLeaveEvent** (QGraphicsSceneHoverEvent \*event) override
- void **hoverMoveEvent** (QGraphicsSceneHoverEvent \*event) override
- void **mouseMoveEvent** (QGraphicsSceneMouseEvent \*) override
- void **mousePressEvent** (QGraphicsSceneMouseEvent \*) override
- void **mouseReleaseEvent** (QGraphicsSceneMouseEvent \*) override
- void **paint** (QPainter \*painter, const QStyleOptionGraphicsItem \*option, QWidget \*widget=nullptr) override

### Protected Member Functions inherited from [Digikam::DImgChildItem](#)

- QVariant **itemChange** (GraphicsItemChange change, const QVariant &value) override

## 9.594 Digikam::FocusPointsExtractor Class Reference

Inheritance diagram for Digikam::FocusPointsExtractor:



### Public Types

- using `ListAFPoints` = `QList< FocusPoint >`  
*A list used to store focus points of a image extracted from meta data.*

### Public Member Functions

- `FocusPointsExtractor` (`QObject *const parent`, `const QString &path`)
- `ListAFPoints get_af_points` ()
- `ListAFPoints get_af_points` (`FocusPoint::TypePoint` type)
- `bool isAFPointsReadOnly` () const
- `QString make` () const
- `QString model` () const
- `MetaEngine::ImageOrientation orientation` () const
- `QSize originalSize` () const

## 9.594.1 Member Typedef Documentation

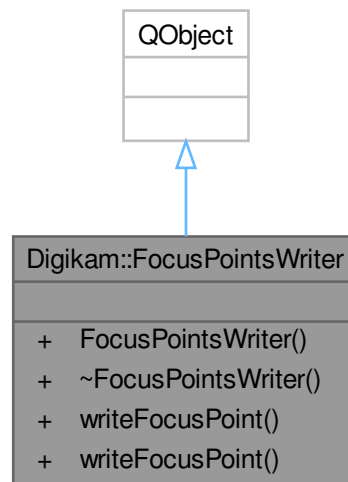
### 9.594.1.1 ListAFPoints

```
using Digikam::FocusPointsExtractor::ListAFPoints = QList<FocusPoint>
```

With extract() function, an exiftool parser is used to read data from metadata and lists all focus points. Each focus point is defined by their relative centers coordinate and relative size. Each point has own type (Inactive, Infocus, Selected, SelectedInFocus)

## 9.595 Digikam::FocusPointsWriter Class Reference

Inheritance diagram for Digikam::FocusPointsWriter:



### Public Member Functions

- **FocusPointsWriter** (QObject \*const parent, const QString &path)
- void **writeFocusPoint** (const FocusPoint &point)
- void **writeFocusPoint** (const QRectF &rectF)

## 9.596 Digikam::FrameOsd Class Reference

### Public Member Functions

- void **insertMessageOsdToFrame** (QImage &frame, const QSize &JPEGsize, const QString &mess)  
*Insert message OSD on broken frame or end frame.*

- void **insertOsdToFrame** (QImage &frame, const QUrl &url, const [FrameOsdSettings](#) &settings, const [DInfoInterface](#) \*const info)  
*Insert OSD on frame.*
- void **populateOSD** (const QUrl &url, const [FrameOsdSettings](#) &settings, const [DInfoInterface](#) \*const info)  
*Populate OSD items properties base on Url.*
- void **printComments** (const QString &comments)  
*print comments*
- void **printTags** (QStringList &tags)  
*print tags*

### Public Attributes

- QString **m\_desc** = QLatin1String("")
- Qt::Alignment **m\_descAlign** = Qt::AlignLeft
- QColor **m\_descBg** = Qt::darkGray
- QFont **m\_descFnt** = QFont(QLatin1String("Monospace"))
- QPoint **m\_descPos** = QPoint(10, 10)

## 9.597 Digikam::FrameOsdSettings Class Reference

### Public Member Functions

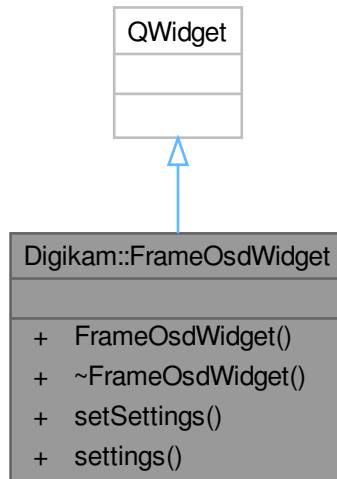
- void **readSettings** (const KConfigGroup &group)  
*Read and write settings in config file between sessions.*
- void **writeSettings** (KConfigGroup &group)

### Public Attributes

- QFont **osdFont** = QFontDatabase::systemFont(QFontDatabase::GeneralFont)  
*Font for the display of osd text.*
- bool **printApertureFocal** = false  
*Print camera Aperture and Focal while streaming.*
- bool **printCapIfNoTitle** = false  
*Print image captions if no title available while streaming.*
- bool **printComment** = false  
*Print picture comment while streaming.*
- bool **printDate** = true  
*Print picture creation date while streaming.*
- bool **printExpoSensitivity** = false  
*Print camera Exposure and Sensitivity while streaming.*
- bool **printLensModel** = false  
*Print camera Lens model while streaming.*
- bool **printMakeModel** = false  
*Print camera Make and Model while streaming.*
- bool **printName** = true  
*Print picture file name while streaming.*
- bool **printRating** = false  
*Print rating while streaming.*
- bool **printTags** = false  
*Print tags title while streaming.*
- bool **printTitle** = false  
*Print image title while streaming.*

## 9.598 Digikam::FrameOsdWidget Class Reference

Inheritance diagram for Digikam::FrameOsdWidget:



### Signals

- void **signalSettingsChanged** ()

### Public Member Functions

- **FrameOsdWidget** (`QWidget *const parent`)
- void **setSettings** (`const FrameOsdSettings &settings`)
- **FrameOsdSettings settings** () const

## 9.599 Digikam::FrameUtils Class Reference

### Static Public Member Functions

- static QImage **makeFramedImage** (`const QString &file, const QSize &outSize`)
- static QImage **makeScaledImage** (`QImage &img, const QSize &outSize`)

## 9.600 Digikam::FreeRotationContainer Class Reference

### Public Types

- enum **AutoCropTypes** { `NoAutoCrop = 0` , `WidestArea` , `LargestArea` }

### Public Attributes

- double **angle** = 0.0
- bool **antiAlias** = true
- int **autoCrop** = NoAutoCrop
- QColor **backgroundColor** = Qt::black
- QSize **newSize**
- int **orgH** = 0
- int **orgW** = 0

## 9.601 Digikam::FreeRotationFilter Class Reference

Inheritance diagram for Digikam::FreeRotationFilter:



### Public Member Functions

- **FreeRotationFilter** (`Dimg *const orgImage`, `QObject *const parent=nullptr`, `const FreeRotationContainer &settings=FreeRotationContainer()`)



- **FreeRotationFilter** (QObject \*const parent=nullptr)
- **FilterAction filterAction** () override  
*Returns the action description corresponding to currently set options.*
- **QString filterIdentifier** () const override  
*Return the identifier for this filter in the image history.*
- **QSize getNewSize** () const
- void **readParameters** (const **FilterAction** &action) override

## Public Member Functions inherited from Digikam::DImgThreadedFilter

- **DImgThreadedFilter** (DImg \*const orgImage, QObject \*const parent, const QString &name=QString())  
*Constructs a filter with all arguments (ready to use).*
- **DImgThreadedFilter** (QObject \*const parent=nullptr, const QString &name=QString())  
*Constructs a filter without argument.*
- virtual void **cancelFilter** ()  
*Cancel the threaded computation.*
- const QString & **filterName** ()
- int **filterVersion** () const
- **DImg getTargetImage** ()
- QList< int > **multithreadedSteps** (int stop, int start=0) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool **parametersSuccessfullyRead** () const  
*Optional: error handling for readParameters.*
- virtual QString **readParametersError** (const **FilterAction** &actionThatFailed) const
- void **setFilterName** (const QString &name)
- void **setFilterVersion** (int version)  
*Replaying a filter action: Set the filter version.*
- void **setOriginalImage** (const DImg &orgImage)
- void **setupAndStartDirectly** (const DImg &orgImage, DImgThreadedFilter \*const master, int progress←Begin=0, int progressEnd=100)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void **setupFilter** (const DImg &orgImage)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void **startFilter** ()  
*Start the threaded computation.*
- virtual void **startFilterDirectly** ()  
*Start computation of this filter, directly in this thread.*
- virtual QList< int > **supportedVersions** () const

## Public Member Functions inherited from Digikam::DynamicThread

- **DynamicThread** (QObject \*const parent=nullptr)  
*This class extends QRunnable, so you have to reimplement virtual void run().*
- **~DynamicThread** () override  
*The destructor calls stop() and wait(), but if you, in your destructor, delete any data that is accessed by your run() method, you must call stop() and wait() before yourself.*
- bool **isFinished** () const
- bool **isRunning** () const
- **QThread::Priority priority** () const
- void **setEmitSignals** (bool emitThem)
- void **setPriority** (QThread::Priority priority)  
*Sets the priority for this dynamic thread.*
- State **state** () const

### Static Public Member Functions

- static double **calculateAngle** (const QPoint &p1, const QPoint &p2)
- static double **calculateAngle** (int x1, int y1, int x2, int y2)
- static int **CurrentVersion** ()
- static QString **DisplayableName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from Digikam::DImgThreadedFilter

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from Digikam::DynamicThread

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from Digikam::DImgThreadedFilter

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.601.1 Member Function Documentation

### 9.601.1.1 filterAction()

`FilterAction` `Digikam::FreeRotationFilter::filterAction ( )` [override], [virtual]

Implements `Digikam::DImgThreadedFilter`.

### 9.601.1.2 filterIdentifier()

`QString` `Digikam::FreeRotationFilter::filterIdentifier ( ) const` [inline], [override], [virtual]

Implements `Digikam::DImgThreadedFilter`.

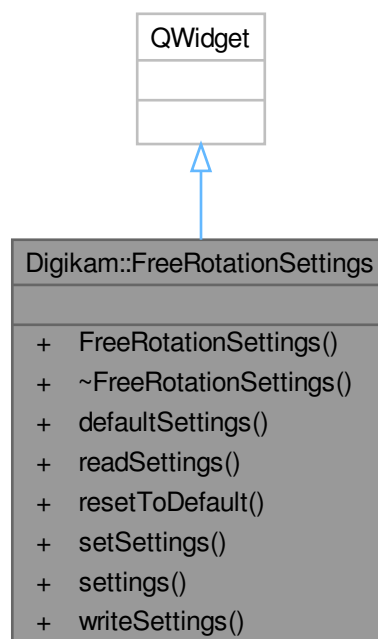
### 9.601.1.3 readParameters()

`void` `Digikam::FreeRotationFilter::readParameters (`  
     `const FilterAction & action )` [override], [virtual]

Implements `Digikam::DImgThreadedFilter`.

## 9.602 Digikam::FreeRotationSettings Class Reference

Inheritance diagram for `Digikam::FreeRotationSettings`:



## Signals

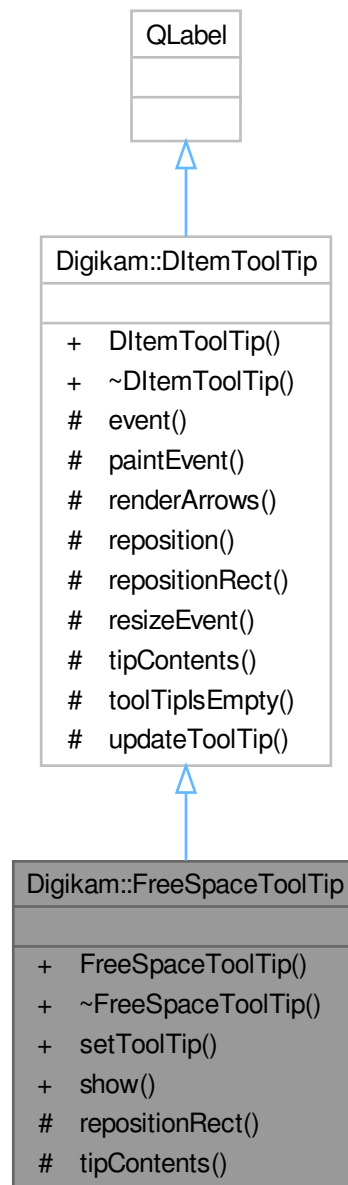
- void **signalSettingsChanged** ()

## Public Member Functions

- **FreeRotationSettings** (QWidget \*const parent)
- [FreeRotationContainer](#) **defaultSettings** () const
- void **readSettings** (const KConfigGroup &group)
- void **resetToDefault** ()
- void **setSettings** (const [FreeRotationContainer](#) &settings)
- [FreeRotationContainer](#) **settings** () const
- void **writeSettings** (KConfigGroup &group)

## 9.603 Digikam::FreeSpaceToolTip Class Reference

Inheritance diagram for Digikam::FreeSpaceToolTip:



### Public Member Functions

- **FreeSpaceToolTip** (QWidget \*const parent)
- void **setToolTip** (const QString &tip)
- void **show** ()

## Public Member Functions inherited from [Digikam::DItemToolTip](#)

- **DItemToolTip** (QWidget \*const parent=nullptr)

## Protected Member Functions

- QRect [repositionRect](#) () override
- QString [tipContents](#) () override

## Protected Member Functions inherited from [Digikam::DItemToolTip](#)

- bool **event** (QEvent \*) override
- void **paintEvent** (QPaintEvent \*) override
- void **renderArrows** ()
- void **reposition** ()
- void **resizeEvent** (QResizeEvent \*) override
- bool **toolTipsEmpty** () const
- void **updateToolTip** ()

## 9.603.1 Member Function Documentation

### 9.603.1.1 [repositionRect\(\)](#)

QRect Digikam::FreeSpaceToolTip::repositionRect ( ) [override], [protected], [virtual]

Implements [Digikam::DItemToolTip](#).

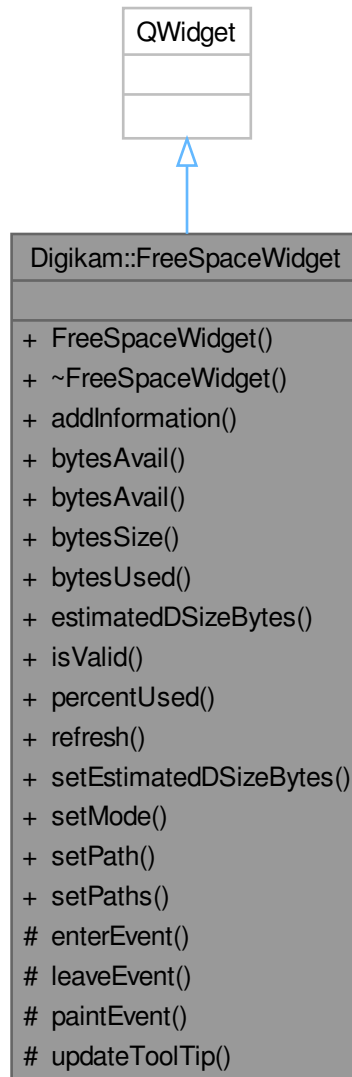
### 9.603.1.2 [tipContents\(\)](#)

QString Digikam::FreeSpaceToolTip::tipContents ( ) [override], [protected], [virtual]

Implements [Digikam::DItemToolTip](#).

## 9.604 Digikam::FreeSpaceWidget Class Reference

Inheritance diagram for Digikam::FreeSpaceWidget:



### Public Types

- enum **FreeSpaceMode** { **AlbumLibrary** = 0 , **UMSCamera** , **GPhotoCamera** }

### Public Member Functions

- **FreeSpaceWidget** (QWidget \*const parent, int width)
- void **addInformation** (qint64 bytesSize, qint64 bytesUsed, qint64 bytesAvail, const QString &mountPoint)
- qint64 **bytesAvail** () const



- qint64 **bytesAvail** (const QString &path) const
- qint64 **bytesSize** () const
- qint64 **bytesUsed** () const
- qint64 **estimatedDSizeBytes** () const
- bool **isValid** () const
- int **percentUsed** () const
- void **refresh** ()
- void **setEstimatedDSizeBytes** (qint64 dSize)
- void **setMode** (FreeSpaceMode mode)
- void **setPath** (const QString &path)
- void **setPaths** (const QStringList &paths)

### Protected Member Functions

- void **enterEvent** (QEnterEvent \*) override
- void **leaveEvent** (QEvent \*) override
- void **paintEvent** (QPaintEvent \*) override
- void **updateToolTip** ()

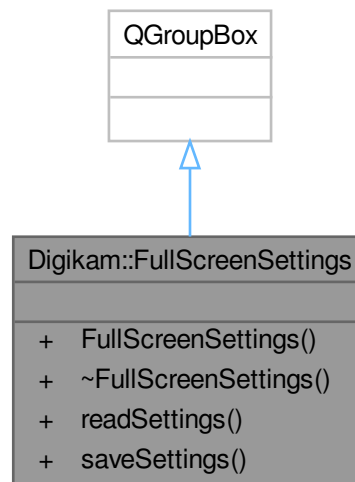
## 9.605 Digikam::FullObjectDetection Class Reference

### Public Member Functions

- **FullObjectDetection** (const cv::Rect &rect\_)
- **FullObjectDetection** (const cv::Rect &rect\_, const std::vector< std::vector< float > > &parts\_)
- cv::Rect & **get\_rect** ()
- const cv::Rect & **get\_rect** () const
- unsigned long **num\_parts** () const
- std::vector< float > & **part** (unsigned long idx)
- const std::vector< float > & **part** (unsigned long idx) const

## 9.606 Digikam::FullScreenSettings Class Reference

Inheritance diagram for Digikam::FullScreenSettings:

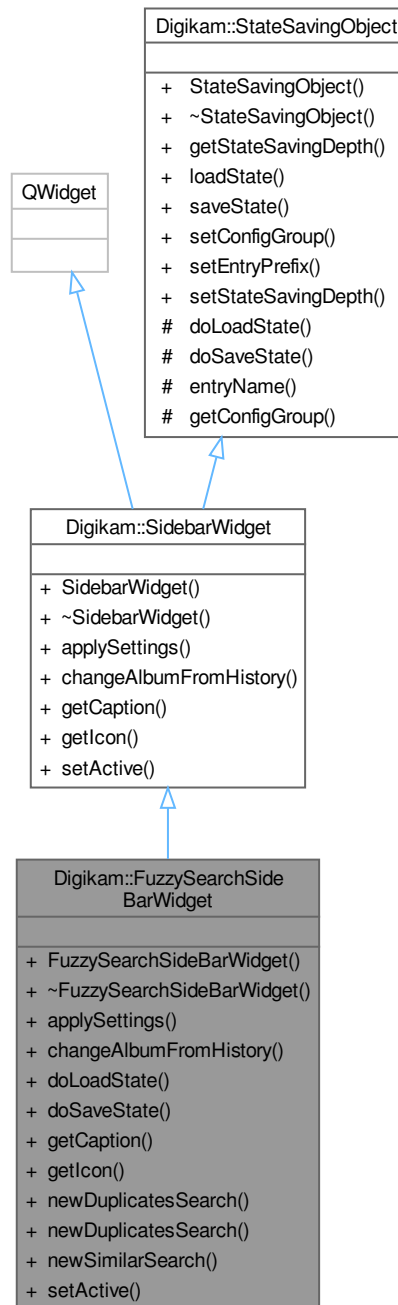


### Public Member Functions

- **FullScreenSettings** (int options, QWidget \*const parent)
- void **readSettings** (const KConfigGroup &group)
- void **saveSettings** (KConfigGroup &group)

## 9.607 Digikam::FuzzySearchSideBarWidget Class Reference

Inheritance diagram for Digikam::FuzzySearchSideBarWidget:



### Signals

- void **signalActive** (bool)

## Signals inherited from [Digikam::SidebarWidget](#)

- void **requestActiveTab** ([SidebarWidget](#) \*)  
*This signal can be emitted if this sidebar widget wants to be the one that is active.*
- void **signalNotificationError** (const QString &message, int type)  
*To dispatch error message to temporized pop-up notification widget hosted with icon-view.*

## Public Member Functions

- **FuzzySearchSideBarWidget** (QWidget \*const parent, [searchModel](#) \*const searchModel, [SearchModificationHelper](#) \*const searchModificationHelper)
- void **applySettings** () override  
*This method is invoked when the application settings should be (re-) applied to this widget.*
- void **changeAlbumFromHistory** (const QList< [Album](#) \* > &album) override  
*This is called on this widget when the history requires to move back to the specified album.*
- void **doLoadState** () override  
*Implement this hook method for state loading.*
- void **doSaveState** () override  
*Implement this hook method for state saving.*
- const QString **getCaption** () override  
*Must be implemented to return the title of this sidebar's tab.*
- const QIcon **getIcon** () override  
*Must be implemented and return the icon that shall be visible for this sidebar widget.*
- void **newDuplicatesSearch** (const QList< [PAAlbum](#) \* > &albums)
- void **newDuplicatesSearch** (const QList< [TAAlbum](#) \* > &albums)
- void **newSimilarSearch** (const [ItemInfo](#) &imageInfo)
- void **setActive** (bool active) override  
*This method is called if the visible sidebar widget is changed.*

## Public Member Functions inherited from [Digikam::SidebarWidget](#)

- [SidebarWidget](#) (QWidget \*const parent)  
*Constructor.*
- **~SidebarWidget** () override=default  
*Destructor.*

## Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (QObject \*const host)  
*Constructor.*
- virtual **~StateSavingObject** ()  
*Destructor.*
- [StateSavingDepth](#) **getStateSavingDepth** () const  
*Returns the depth used for state saving or loading.*
- void **loadState** ()  
*Invokes loading the class' state.*
- void **saveState** ()  
*Invokes saving the class' state.*
- virtual void **setConfigGroup** (const KConfigGroup &group)  
*Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void **setEntryPrefix** (const QString &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void **setStateSavingDepth** (const [StateSavingDepth](#) depth)  
*Sets the depth used for state saving or loading.*

## Additional Inherited Members

## Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }

*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

## Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- QString [entryName](#) (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup [getConfigGroup](#) () const  
*Returns the config group that must be used for state saving and loading.*

## 9.607.1 Member Function Documentation

### 9.607.1.1 [applySettings\(\)](#)

```
void Digikam::FuzzySearchSideBarWidget::applySettings ( ) [override], [virtual]
```

Implements [Digikam::SidebarWidget](#).

### 9.607.1.2 [changeAlbumFromHistory\(\)](#)

```
void Digikam::FuzzySearchSideBarWidget::changeAlbumFromHistory (
    const QList< Album * > & album ) [override], [virtual]
```

Implements [Digikam::SidebarWidget](#).

### 9.607.1.3 [doLoadState\(\)](#)

```
void Digikam::FuzzySearchSideBarWidget::doLoadState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

### 9.607.1.4 [doSaveState\(\)](#)

```
void Digikam::FuzzySearchSideBarWidget::doSaveState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

### 9.607.1.5 `getCaption()`

```
const QString Digikam::FuzzySearchSideBarWidget::getCaption ( ) [override], [virtual]
```

#### Returns

localized title string

Implements [Digikam::SidebarWidget](#).

### 9.607.1.6 `getIcon()`

```
const QIcon Digikam::FuzzySearchSideBarWidget::getIcon ( ) [override], [virtual]
```

#### Returns

pixmap icon

Implements [Digikam::SidebarWidget](#).

### 9.607.1.7 `setActive()`

```
void Digikam::FuzzySearchSideBarWidget::setActive (
    bool active ) [override], [virtual]
```

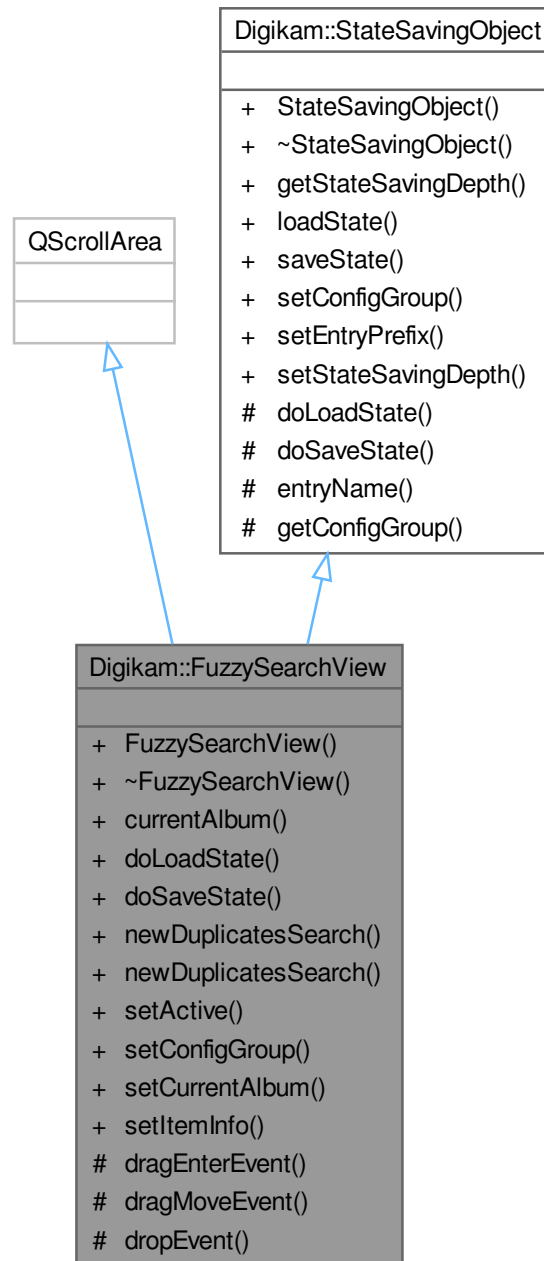
#### Parameters

<i>active</i>	if true, this widget is the new active widget, if false another widget is active
---------------	--

Implements [Digikam::SidebarWidget](#).

## 9.608 Digikam::FuzzySearchView Class Reference

Inheritance diagram for Digikam::FuzzySearchView:



### Signals

- void **signalNotificationError** (const QString &message, int type)

## Public Member Functions

- **FuzzySearchView** ([searchModel](#) \*const searchModel, [searchModificationHelper](#) \*const searchModificationHelper, [QWidget](#) \*const parent=nullptr)
- [SAAlbum](#) \* **currentAlbum** () const
- void **doLoadState** () override  
*Implement this hook method for state loading.*
- void **doSaveState** () override  
*Implement this hook method for state saving.*
- void **newDuplicatesSearch** (const QList< [PAAlbum](#) \* > &albums)
- void **newDuplicatesSearch** (const QList< [TAAlbum](#) \* > &albums)
- void **setActive** (bool val)
- void **setConfigGroup** (const [KConfigGroup](#) &group) override  
*Sets a dedicated config group that will be used to store and reload the state from.*
- void **setCurrentAlbum** ([SAAlbum](#) \*const album)
- void **setItemInfo** (const [ItemInfo](#) &info)

## Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) ([QObject](#) \*const host)  
*Constructor.*
- virtual **~StateSavingObject** ()  
*Destructor.*
- [StateSavingDepth](#) **getStateSavingDepth** () const  
*Returns the depth used for state saving or loading.*
- void **loadState** ()  
*Invokes loading the class' state.*
- void **saveState** ()  
*Invokes saving the class' state.*
- virtual void **setEntryPrefix** (const [QString](#) &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void **setStateSavingDepth** (const [StateSavingDepth](#) depth)  
*Sets the depth used for state saving or loading.*

## Protected Member Functions

- void **dragEnterEvent** ([QDragEnterEvent](#) \*e) override
- void **dragMoveEvent** ([QDragMoveEvent](#) \*e) override
- void **dropEvent** ([QDropEvent](#) \*e) override

## Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- [QString](#) **entryName** (const [QString](#) &base) const  
*Always use this method to create config group entry names.*
- [KConfigGroup](#) **getConfigGroup** () const  
*Returns the config group that must be used for state saving and loading.*



## Additional Inherited Members

## Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }

*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

## 9.608.1 Member Function Documentation

### 9.608.1.1 doLoadState()

```
void Digikam::FuzzySearchView::doLoadState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

### 9.608.1.2 doSaveState()

```
void Digikam::FuzzySearchView::doSaveState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

### 9.608.1.3 setConfigGroup()

```
void Digikam::FuzzySearchView::setConfigGroup (
    const KConfigGroup & group ) [override], [virtual]
```

If this method is not called, a group based on the object name is used.

You can re-implement this method to pass the group set here to child objects. Don't forget to call this method in your implementation.

#### Parameters

<i>group</i>	config group to use for state saving and restoring
--------------	--

Reimplemented from [Digikam::StateSavingObject](#).

## 9.609 Digikam::GeoCoordinates Class Reference

### Public Types

- enum **HasFlag** {  
    **HasNothing** = 0 , **HasLatitude** = 1 , **HasLongitude** = 2 , **HasCoordinates** = 3 ,  
    **HasAltitude** = 4 }

- typedef QFlags< HasFlag > **HasFlags**
- typedef QList< [GeoCoordinates](#) > **List**
- typedef QPair< [GeoCoordinates](#), [GeoCoordinates](#) > **Pair**
- typedef QList< [GeoCoordinates::Pair](#) > **PairList**

## Public Member Functions

- **GeoCoordinates** (const double inLat, const double inLon)
- **GeoCoordinates** (const double inLat, const double inLon, const double inAlt)
- double **alt** () const
- QString **altString** () const
- void **clear** ()
- void **clearAlt** ()
- QString **geoUrl** () const
- bool **hasAltitude** () const
- bool **hasCoordinates** () const
- HasFlags **hasFlags** () const
- bool **hasLatitude** () const
- bool **hasLongitude** () const
- double **lat** () const
- QString **latString** () const
- double **lon** () const
- QString **lonString** () const
- bool **operator==** (const [GeoCoordinates](#) &other) const
- bool **sameLonLatAs** (const [GeoCoordinates](#) &other) const
- void **setAlt** (const double inAlt)
- void **setLatLon** (const double inLat, const double inLon)
- Marble::GeoDataCoordinates **toMarbleCoordinates** () const

## Static Public Member Functions

- static [GeoCoordinates](#) **fromGeoUrl** (const QString &url, bool \*const parsedOkay=nullptr)
- static [GeoCoordinates](#) **fromMarbleCoordinates** (const Marble::GeoDataCoordinates &marbleCoordinates)
- static Pair **makePair** (const qreal lat1, const qreal lon1, const qreal lat2, const qreal lon2)

## 9.609.1 Member Function Documentation

### 9.609.1.1 fromMarbleCoordinates()

```
GeoCoordinates Digikam::GeoCoordinates::fromMarbleCoordinates (
    const Marble::GeoDataCoordinates & marbleCoordinates ) [static]
```

## 9.610 Digikam::GeodeticCalculator Class Reference

### Public Member Functions

- [GeodeticCalculator](#) (const [Ellipsoid](#) &e=[Ellipsoid::WGS84](#)())  
*Performs geodetic calculations on an ellipsoid.*
- double [azimuth](#) ()  
*Returns the azimuth.*
- bool [checkOrthodromicDistance](#) ()  
*Computes the orthodromic distance using the algorithm implemented in the Geotools's ellipsoid class (if available), and check if the error is smaller than some tolerance error.*
- bool [computeDestinationPoint](#) ()  
*Computes the destination point from the starting point, the azimuth and the orthodromic distance.*
- bool [computeDirection](#) ()  
*Computes the azimuth and orthodromic distance from the startingGeographicPoint starting point and the destinationGeographicPoint destination point.*
- QPointF [destinationGeographicPoint](#) ()
- bool [destinationGeographicPoint](#) (double \*longitude, double \*latitude)  
*Returns the destination point.*
- [Ellipsoid ellipsoid](#) () const  
*Returns the referenced ellipsoid.*
- double [meridianArcLength](#) (double latitude1, double latitude2)  
*Calculates the meridian arc length between two points in the same meridian in the referenced ellipsoid.*
- double [meridianArcLengthRadians](#) (double P1, double P2)  
*Calculates the meridian arc length between two points in the same meridian in the referenced ellipsoid.*
- double [orthodromicDistance](#) ()  
*Returns the orthodromic distance.*
- void [setDestinationGeographicPoint](#) (double longitude, double latitude)  
*Set the destination point in geographic coordinates.*
- void [setDirection](#) (double [azimuth](#), double distance)  
*Set the azimuth and the distance from the startingGeographicPoint starting point.*
- void [setStartingGeographicPoint](#) (double longitude, double latitude)  
*Set the starting point in geographic coordinates.*

### Protected Member Functions

- double [castToAngleRange](#) (const double alpha)
- bool [checkAzimuth](#) (double \*azimuth)  
*Checks the azimuth validity.*
- bool [checkLatitude](#) (double \*latitude)  
*Checks the latitude validity.*
- bool [checkLongitude](#) (double \*longitude)  
*Checks the longitude validity.*
- bool [checkOrthodromicDistance](#) (const double distance)  
*Checks the orthodromic distance validity.*

## Protected Attributes

- double **a01** = 0.0

*Parameters computed from the ellipsoid.*

- double **a02** = 0.0
- double **a03** = 0.0
- double **a21** = 0.0
- double **a22** = 0.0
- double **a23** = 0.0
- double **a42** = 0.0
- double **a43** = 0.0
- double **a63** = 0.0
- double **f** = 0.0
- double **f2** = 0.0
- double **f3** = 0.0
- double **f4** = 0.0
- double **fo** = 0.0

*GPNHRI parameters computed from the ellipsoid.*

- double **m\_A** = 0.0

*GPNARC parameters computed from the ellipsoid.*

- double **m\_azimuth** = 0.0
- double **m\_B** = 0.0
- double **m\_C** = 0.0
- double **m\_D** = 0.0
- bool **m\_destinationValid** = false

*Tell if the destination point is valid.*

- bool **m\_directionValid** = false

*Tell if the azimuth and the distance are valids.*

- double **m\_distance** = 0.0

*The distance and azimuth (in radians) from the starting point (long1, lat1) to the destination point (long2, lat2).*

- double **m\_E** = 0.0
- double **m\_eccentricitySquared** = 0.0

*The eccentricity squared of the referenced ellipsoid.*

- **Ellipsoid m\_ellipsoid**

*The encapsulated ellipsoid.*

- double **m\_F** = 0.0
- double **m\_lat1** = 0.0

*The (latitude, longitude) coordinate of the first point in radians.*

- double **m\_lat2** = 0.0

*The (latitude, longitude) coordinate of the destination point in radians.*

- double **m\_long1** = 0.0
- double **m\_long2** = 0.0
- double **m\_maxOrthodromicDistance** = 0.0

*The maximum orthodromic distance that could be calculated onto the referenced ellipsoid.*

- double **m\_semiMajorAxis** = 0.0

*The semi major axis of the referenced ellipsoid.*

- double **m\_semiMinorAxis** = 0.0

*The semi minor axis of the referenced ellipsoid.*

- double **m\_TOLERANCE\_0** = 5.0e-15

*Tolerance factors from the strictest (TOLERANCE\_0) to the most relax one (TOLERANCE\_3).*

- double **m\_TOLERANCE\_1** = 5.0e-14
- double **m\_TOLERANCE\_2** = 5.0e-13

- double `m_TOLERANCE_3` = 7.0e-3
- double `m_TOLERANCE_CHECK` = 1E-8  
*Tolerance factor for assertions.*
- double `T1` = 1.0  
*Parameters computed from the ellipsoid.*
- double `T2` = 0.0
- double `T4` = 0.0
- double `T6` = 0.0

## 9.610.1 Constructor & Destructor Documentation

### 9.610.1.1 GeodeticCalculator()

```
Digikam::GeodeticCalculator::GeodeticCalculator (
    const Ellipsoid & e = Ellipsoid::WGS84() ) [explicit]
```

This class encapsulates a generic ellipsoid and calculates the following properties:

Distance and azimuth between two points. Point located at a given distance and azimuth from an other point.

The calculation use the following information:

The starting position (`setStartingPosition`), which is always considered valid. It is initially set at (0,0) and can only be changed to another legitimate value. Only one of the following:

The destination position (`setDestinationPosition`), or  
An azimuth and distance (`setDirection`).

The latest one set overrides the other and determines what will be calculated.

## 9.610.2 Member Function Documentation

### 9.610.2.1 azimuth()

```
double Digikam::GeodeticCalculator::azimuth ( )
```

This method returns the value set by the last call to `setDirection(double, double)` `setDirection(azimuth, distance)`, except if `setDestinationGeographicPoint(double, double)` `setDestinationGeographicPoint(...)` has been invoked after. In this later case, the azimuth will be computed from the startingGeographicPoint starting point to the destination point.

#### Returns

The azimuth, in decimal degrees from -180° to +180°.

### 9.610.2.2 checkAzimuth()

```
bool Digikam::GeodeticCalculator::checkAzimuth (
    double * azimuth ) [protected]
```

The argument `azimuth` should be greater or equal than -180 degrees and lower or equals than +180 degrees. As a convenience, this method converts the azimuth to radians.

**Parameters**

<i>azimuth</i>	The azimuth value in decimal degrees.
----------------	---------------------------------------

**9.610.2.3 checkLatitude()**

```
bool Digikam::GeodeticCalculator::checkLatitude (
    double * latitude ) [protected]
```

The argument *latitude* should be greater or equal than -90 degrees and lower or equals than +90 degrees. As a convenience, this method converts the latitude to radians.

**Parameters**

<i>latitude</i>	The latitude value in decimal degrees.
-----------------	--

**9.610.2.4 checkLongitude()**

```
bool Digikam::GeodeticCalculator::checkLongitude (
    double * longitude ) [protected]
```

The argument *longitude* should be greater or equal than -180 degrees and lower or equals than +180 degrees. As a convenience, this method converts the longitude to radians.

**Parameters**

<i>longitude</i>	The longitude value in decimal degrees.
------------------	---

**9.610.2.5 checkOrthodromicDistance()**

```
bool Digikam::GeodeticCalculator::checkOrthodromicDistance (
    const double distance ) [protected]
```

Arguments *orthodromicDistance* should be greater or equal than 0 and lower or equals than the maximum orthodromic distance.

**Parameters**

<i>distance</i>	The orthodromic distance value.
-----------------	---------------------------------

**9.610.2.6 computeDirection()**

```
bool Digikam::GeodeticCalculator::computeDirection ( )
```

Computes the azimuth and orthodromic distance from the startingGeographicPoint() and the destinationGeographicPoint().

**9.610.2.7 destinationGeographicPoint()**

```
bool Digikam::GeodeticCalculator::destinationGeographicPoint (
    double * longitude,
    double * latitude )
```

This method returns the point set by the last call to a `setDestinationGeographicPoint(...)` method, except if `setDirection(...)` has been invoked after. In this later case, the destination point will be computed from the starting point to the azimuth and distance specified. Coordinates positive North and East.

**Returns**

The destination point. The x and y coordinates are the longitude and latitude in decimal degrees, respectively.

**9.610.2.8 meridianArcLength()**

```
double Digikam::GeodeticCalculator::meridianArcLength (
    double latitude1,
    double latitude2 )
```

**Parameters**

<i>latitude1</i>	The latitude of the first point (in decimal degrees).
<i>latitude2</i>	The latitude of the second point (in decimal degrees).

**Returns**

Returned the meridian arc length between `latitude1` and `latitude2`

**9.610.2.9 meridianArcLengthRadians()**

```
double Digikam::GeodeticCalculator::meridianArcLengthRadians (
    double P1,
    double P2 )
```

**Parameters**

<i>P1</i>	The latitude of the first point (in radians).
<i>P2</i>	The latitude of the second point (in radians).

**Returns**

Returned the meridian arc length between `P1` and `P2`

**9.610.2.10 orthodromicDistance()**

```
double Digikam::GeodeticCalculator::orthodromicDistance ( )
```

This method returns the value set by the last call to `setDirection(double, double)` `setDirection(azimuth, distance)`, except if `setDestinationGeographicPoint(double, double)` `setDestinationGeographicPoint(...)` has been invoked after. In this later case, the distance will be computed from the startingGeographicPoint starting point to the destination point.

#### Returns

The orthodromic distance, in the same units as the `getEllipsoid` ellipsoid axis.

#### 9.610.2.11 setDestinationGeographicPoint()

```
void Digikam::GeodeticCalculator::setDestinationGeographicPoint (
    double longitude,
    double latitude )
```

The azimuth and distance values will be updated as a side effect of this call. They will be recomputed the next time `getAzimuth()` or `getOrthodromicDistance()` are invoked. Coordinates positive North and East.

#### Parameters

<i>longitude</i>	The longitude in decimal degrees between -180 and +180°
<i>latitude</i>	The latitude in decimal degrees between -90 and +90°

#### 9.610.2.12 setDirection()

```
void Digikam::GeodeticCalculator::setDirection (
    double azimuth,
    double distance )
```

The destination point will be updated as a side effect of this call. It will be recomputed the next time `destinationGeographicPoint()` is invoked. Azimuth 0° North.

#### Parameters

<i>azimuth</i>	The azimuth in decimal degrees from -180° to 180°.
<i>distance</i>	The orthodromic distance in the same units as the ellipsoid axis.

#### 9.610.2.13 setStartingGeographicPoint()

```
void Digikam::GeodeticCalculator::setStartingGeographicPoint (
    double longitude,
    double latitude )
```

The azimuth, the orthodromic distance and the destination point are discarded. They will need to be specified again. Coordinates positive North and East.

#### Parameters

<i>longitude</i>	The longitude in decimal degrees between -180 and +180°
<i>latitude</i>	The latitude in decimal degrees between -90 and +90°



### 9.610.3 Member Data Documentation

#### 9.610.3.1 fo

```
double Digikam::GeodeticCalculator::fo = 0.0 [protected]
```

f if the flattening of the referenced ellipsoid. f2, f3 and f4 are  $f^2$ ,  $f^3$  and  $f^4$  respectively.

#### 9.610.3.2 m\_destinationValid

```
bool Digikam::GeodeticCalculator::m_destinationValid = false [protected]
```

false if long2 and lat2 need to be computed.

#### 9.610.3.3 m\_directionValid

```
bool Digikam::GeodeticCalculator::m_directionValid = false [protected]
```

false if distance and azimuth need to be computed.

#### 9.610.3.4 m\_lat1

```
double Digikam::GeodeticCalculator::m_lat1 = 0.0 [protected]
```

This point is set by setStartingGeographicPoint.

#### 9.610.3.5 m\_lat2

```
double Digikam::GeodeticCalculator::m_lat2 = 0.0 [protected]
```

This point is set by setDestinationGeographicPoint.

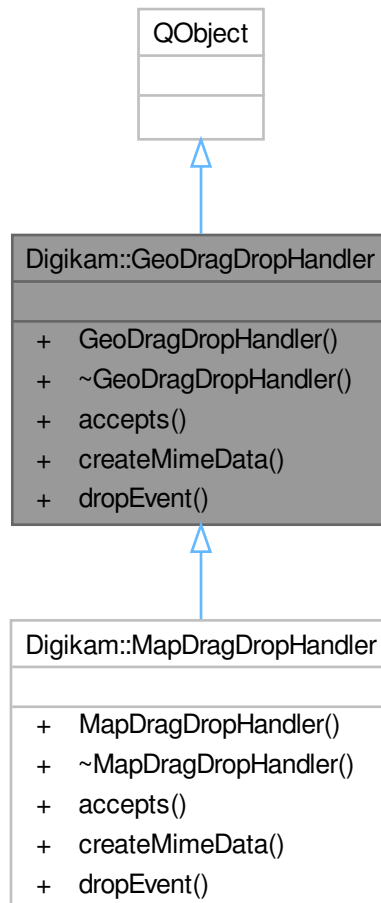
#### 9.610.3.6 m\_TOLERANCE\_CHECK

```
double Digikam::GeodeticCalculator::m_TOLERANCE_CHECK = 1E-8 [protected]
```

It has no impact on computed values.

## 9.611 Digikam::GeoDragDropHandler Class Reference

Inheritance diagram for Digikam::GeoDragDropHandler:



### Public Member Functions

- **GeoDragDropHandler** (`QObject *const parent=nullptr`)
- virtual `Qt::DropAction` **accepts** (`const QDropEvent *e`)=0
- virtual `QMimeData *` **createMimeData** (`const QList< QPersistentModelIndex > &modelIndices`)=0
- virtual `bool` **dropEvent** (`const QDropEvent *e`, `const GeoCoordinates &dropCoordinates`)=0

## 9.612 Digikam::GeofaceCluster Class Reference

### Public Types

- typedef `QList< GeofaceCluster >` **List**
- enum **PixmapType** { `PixmapMarker` , `PixmapCircle` , `PixmapImage` }

### Public Attributes

- [GeoCoordinates](#) **coordinates**
- GeoGroupState **groupState**
- int **markerCount**
- int **markerSelectedCount**
- QPoint **pixelPos**
- QPoint **pixmapOffset**  
*anchor point of the image, measured from bottom-left*
- QSize **pixmapSize**
- enum Digikam::GeolfaceCluster::PixmapType **pixmapType**
- QMap< int, QVariant > **representativeMarkers**
- QList< [TileIndex](#) > **tileIndicesList**

## 9.613 Digikam::GeolfaceGlobalObject Class Reference

Global object for geolocation interface to hold items common to all geolocation interface Widget instances.

Inheritance diagram for Digikam::GeolfaceGlobalObject:



## Public Member Functions

### Shared pixmaps

- QPixmap **getMarkerPixmap** (const QString &pixmapId)
- QPixmap **getStandardMarkerPixmap** ()
- QUrl **locateDataFile** (const QString &filename)

## Static Public Member Functions

- static [GeolfaceGlobalObject](#) \* **instance** ()

### Shared internal map widgets

- class **GeolfaceGlobalObjectCreator**
- void **removeMyInternalWidgetFromPool** (const [MapBackend](#) \*const mapBackend)
- bool **getInternalWidgetFromPool** (const [MapBackend](#) \*const mapBackend, [GeolfaceInternalWidgetInfo](#) \*const targetInfo)
- void **addMyInternalWidgetToPool** (const [GeolfaceInternalWidgetInfo](#) &info)
- void **updatePooledWidgetState** (const QWidget \*const widget, const [GeolfaceInternalWidgetInfo](#)::[InternalWidgetState](#) newState)
- void **clearWidgetPool** ()

## 9.614 Digikam::GeolfaceInternalWidgetInfo Class Reference

Class to hold information about map widgets stored in the [GeolfaceGlobalObject](#).

### Public Types

- typedef void(\* **DeleteFunction**) ([GeolfaceInternalWidgetInfo](#) \*const info)
- enum **InternalWidgetState** { **InternalWidgetReleased** = 1 , **InternalWidgetUndocked** = 2 , **InternalWidgetStillDocked** = 4 }
- typedef QFlags< InternalWidgetState > **InternalWidgetStates**

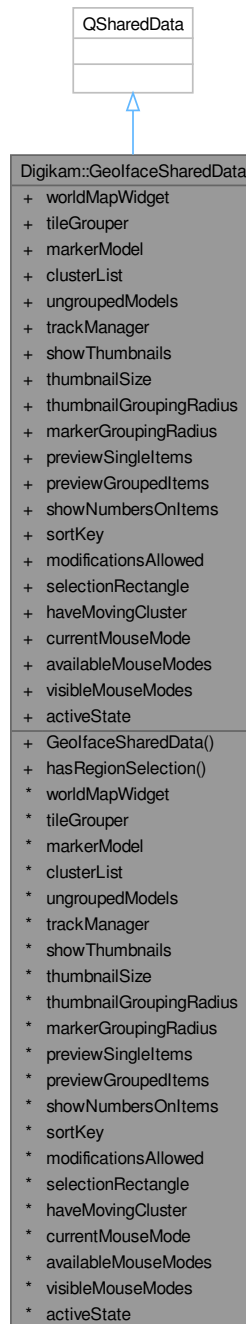
### Public Attributes

- QVariant **backendData**
- QString **backendName**
- QPointer< QObject > **currentOwner**
- DeleteFunction **deleteFunction**
- InternalWidgetStates **state**
- QPointer< QWidget > **widget**

### 9.614.1 Detailed Description

## 9.615 Digikam::GeolfaceSharedData Class Reference

Inheritance diagram for Digikam::GeolfaceSharedData:



### Public Member Functions

- bool `hasRegionSelection ()` const

## Public Attributes

### Objects

- [MapWidget](#) \* **worldMapWidget**
- [TileGrouper](#) \* **tileGrouper**
- [AbstractMarkerTiler](#) \* **markerModel**
- [GeofaceCluster::List](#) **clusterList**
- [QList< GeoModelHelper \\* >](#) **ungroupedModels**
- [TrackManager](#) \* **trackManager**

### Display options

- bool **showThumbnails**
- int **thumbnailSize**
- int **thumbnailGroupingRadius**
- int **markerGroupingRadius**
- bool **previewSingleItems**
- bool **previewGroupedItems**
- bool **showNumbersOnItems**
- int **sortKey**
- bool **modificationsAllowed**

### Current map state

- [GeoCoordinates::Pair](#) **selectionRectangle**
- bool **haveMovingCluster**
- [GeoMouseModes](#) **currentMouseMode**
- [GeoMouseModes](#) **availableMouseModes**
- [GeoMouseModes](#) **visibleMouseModes**
- bool **activeState**

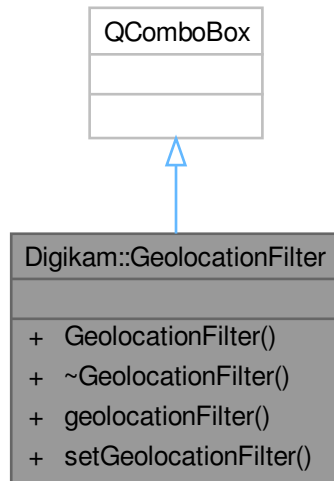
## 9.615.1 Member Function Documentation

### 9.615.1.1 hasRegionSelection()

```
bool Digikam::GeoIfaceSharedData::hasRegionSelection ( ) const [inline]
```

## 9.616 Digikam::GeolocationFilter Class Reference

Inheritance diagram for Digikam::GeolocationFilter:



### Signals

- void **signalFilterChanged** (const [ItemFilterSettings::GeolocationCondition](#) &condition)

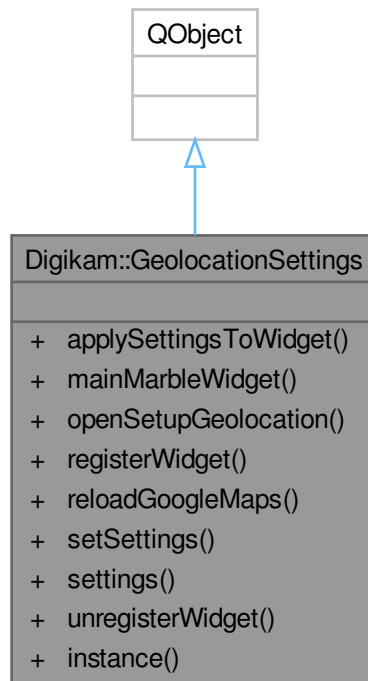
### Public Member Functions

- **GeolocationFilter** (QWidget \*const parent)
- [ItemFilterSettings::GeolocationCondition](#) **geolocationFilter** () const
- void **setGeolocationFilter** (const [ItemFilterSettings::GeolocationCondition](#) &condition)



## 9.617 Digikam::GeolocationSettings Class Reference

Inheritance diagram for Digikam::GeolocationSettings:



### Signals

- void **signalGeolocationSettingsChanged** (const [GeolocationSettingsContainer](#) &current, const [GeolocationSettingsContainer](#) &previous)
- void **signalSettingsChanged** ()
- void **signalSetupGeolocation** (int tab)

### Public Member Functions

- void **applySettingsToWidget** ([MapWidget](#) \*const widget)  
*Apply the current settings to a previously registered [MapWidget](#).*
- [MarbleWidget](#) \* **mainMarbleWidget** () const  
*Return the first registered [MarbleWidget](#) instance stored in the collection.*
- void **openSetupGeolocation** ([SetupGeolocation::GeolocationTab](#) tab)
- void **registerWidget** ([MapWidget](#) \*const widget)  
*Store one [MapWidget](#) instance in the collection.*
- void **reloadGoogleMaps** ()
- void **setSettings** (const [GeolocationSettingsContainer](#) &settings)  
*Sets the current Metadata settings and writes them to config.*
- [GeolocationSettingsContainer](#) **settings** () const  
*Returns the current Metadata settings.*
- void **unregisterWidget** ([MapWidget](#) \*const widget)  
*Remove one [MapWidget](#) instance in the collection.*

### Static Public Member Functions

- static [GeolocationSettings](#) \* `instance` ()  
*Global container for Metadata settings.*

### Friends

- class [GeolocationSettingsCreator](#)

## 9.617.1 Member Function Documentation

### 9.617.1.1 `instance()`

```
GeolocationSettings * Digikam::GeolocationSettings::instance ( ) [static]
```

All accessor methods are thread-safe.

### 9.617.1.2 `mainMarbleWidget()`

```
MarbleWidget * Digikam::GeolocationSettings::mainMarbleWidget ( ) const
```

If no valid instance is found, nullptr is returned.

## 9.618 Digikam::GeolocationSettingsContainer Class Reference

The class [GeolocationSettingsContainer](#) encapsulates all Marble related settings.

### Public Member Functions

- void **readFromConfig** (const KConfigGroup &group)
- void **writeToConfig** (KConfigGroup &group) const

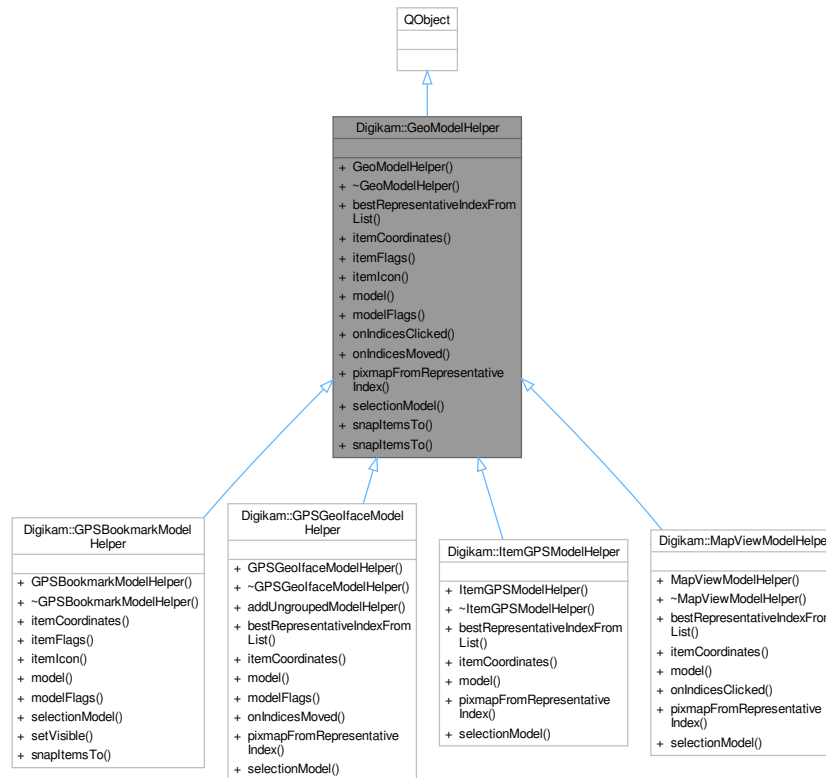
### Public Attributes

- Marble::AngleUnit **angleUnit** = Marble::DecimalDegree
- Marble::MapQuality **animationQuality** = Marble::LowQuality
- MarbleLocale::MeasurementSystem **distanceUnit** = MarbleLocale::MetricSystem
- bool **inertialRotation** = true
- QFont **mapFont**
- bool **mouseRotation** = true
- int **persistentTileCacheLimit** = 999999
- bool **showAtmos** = false
- bool **showCities** = true
- bool **showCross** = true
- bool **showGrid** = true
- bool **showRelief** = true
- bool **showSunShading** = false
- Marble::MapQuality **stillQuality** = Marble::HighQuality
- int **volatileTileCacheLimit** = 100

## 9.619 Digikam::GeoModelHelper Class Reference

Helper class to access data in models.

Inheritance diagram for Digikam::GeoModelHelper:



### Public Types

- enum **PropertyFlag** { **FlagNull** = 0 , **FlagVisible** = 1 , **FlagMovable** = 2 , **FlagSnaps** = 4 }
- typedef QFlags< PropertyFlag > **PropertyFlags**

### Signals

- void **signalModelChangedDrastically** ()
- void **signalThumbnailAvailableForIndex** (const QPersistentModelIndex &index, const QPixmap &pixmap)
- void **signalVisibilityChanged** ()

### Public Member Functions

- **GeoModelHelper** (QObject \*const parent=nullptr)
- virtual QPersistentModelIndex **bestRepresentativeIndexFromList** (const QList< QPersistentModelIndex > &list, const int sortKey)
- virtual bool **itemCoordinates** (const QModelIndex &index, GeoCoordinates \*const coordinates) const =0
- virtual PropertyFlags **itemFlags** (const QModelIndex &index) const

- virtual bool [itemIcon](#) (const QModelIndex &index, QPoint \*const offset, QSize \*const size, QPixmap \*const pixmap, QUrl \*const url) const  
*these are necessary for ungrouped models*
- virtual QAbstractItemModel \* [model](#) () const =0  
*these are necessary for grouped and ungrouped models*
- virtual PropertyFlags [modelFlags](#) () const
- virtual void [onIndicesClicked](#) (const QList< QPersistentModelIndex > &clickedIndices)
- virtual void [onIndicesMoved](#) (const QList< QPersistentModelIndex > &movedIndices, const [GeoCoordinates](#) &targetCoordinates, const QPersistentModelIndex &targetSnapIndex)
- virtual QPixmap [pixmapFromRepresentativeIndex](#) (const QPersistentModelIndex &index, const QSize &size)  
*these are used by MarkerModel for grouped models*
- virtual QItemSelectionModel \* [selectionModel](#) () const =0
- virtual void [snapItemsTo](#) (const QModelIndex &targetIndex, const QList< QModelIndex > &snappedIndices)
- void [snapItemsTo](#) (const QModelIndex &targetIndex, const QList< QPersistentModelIndex > &snappedIndices)

### 9.619.1 Detailed Description

[GeoModelHelper](#) is used to access data held in models, which is not suitable for transfer using the the Qt-style API, like coordinates or custom sized thumbnails.

The basic functions which have to be implemented are:

- [model\(\)](#): Returns a pointer to the model
- [selectionModel\(\)](#): Returns a pointer to the selection model. It may return a null-pointer if no selection model is used.
- [itemCoordinates\(\)](#): Returns the coordinates for a given item index, if it has any.
- [modelFlags\(\)](#): Returns flags for the model.

For ungrouped models, the following functions should also be implemented:

- [itemIcon\(\)](#): Returns an icon for an index, and an offset to the 'center' of the item.
- [itemFlags\(\)](#): Returns flags for individual items.
- [snapItemsTo\(\)](#): Grouped items have been moved and should snap to an index.

For grouped models which are accessed by `MarkerModel`, the following functions should be implemented:

- [bestRepresentativeIndexFromList\(\)](#): Find the item that should represent a group of items.
- [pixmapFromRepresentativeIndex\(\)](#): Find a thumbnail for an item.

### 9.619.2 Member Function Documentation

#### 9.619.2.1 [bestRepresentativeIndexFromList\(\)](#)

```
QPersistentModelIndex Digikam::GeoModelHelper::bestRepresentativeIndexFromList (
    const QList< QPersistentModelIndex > & list,
    const int sortKey ) [virtual]
```

Reimplemented in [Digikam::MapViewModelHelper](#).

### 9.619.2.2 itemCoordinates()

```
virtual bool Digikam::GeoModelHelper::itemCoordinates (
    const QModelIndex & index,
    GeoCoordinates *const coordinates ) const [pure virtual]
```

Implemented in [Digikam::MapViewModelHelper](#).

### 9.619.2.3 itemIcon()

```
bool Digikam::GeoModelHelper::itemIcon (
    const QModelIndex & index,
    QPoint *const offset,
    QSize *const size,
    QPixmap *const pixmap,
    QUrl *const url ) const [virtual]
```

Returns the icon for an ungrouped marker.

The icon can either be returned as a URL to an image, or as a QPixmap. If the caller can handle URLs (for example, to display them in HTML), he can provide the URL parameter. However, the [GeoModelHelper](#) may still choose to return a QPixmap instead, if no URL is available.

#### Parameters

<i>index</i>	Modelindex of the marker.
<i>offset</i>	Offset of the zero point in the icon, given from the top-left.
<i>size</i>	the size of the icon, only populated if a URL is returned.
<i>pixmap</i>	Holder for the pixmap of the icon.
<i>url</i>	URL of the icon if available.

Reimplemented in [Digikam::GPSBookmarkModelHelper](#).

### 9.619.2.4 model()

```
virtual QAbstractItemModel * Digikam::GeoModelHelper::model ( ) const [pure virtual]
```

Implemented in [Digikam::MapViewModelHelper](#), [Digikam::ItemGPSModelHelper](#), [Digikam::GPSBookmarkModelHelper](#), and [Digikam::GPSGeofaceModelHelper](#).

### 9.619.2.5 onIndicesClicked()

```
void Digikam::GeoModelHelper::onIndicesClicked (
    const QList< QPersistentModelIndex > & clickedIndices ) [virtual]
```

Reimplemented in [Digikam::MapViewModelHelper](#).

### 9.619.2.6 pixmapFromRepresentativeIndex()

```
QPixmap Digikam::GeoModelHelper::pixmapFromRepresentativeIndex (
    const QPersistentModelIndex & index,
    const QSize & size ) [virtual]
```

Reimplemented in [Digikam::MapViewModelHelper](#), [Digikam::ItemGPSModelHelper](#), and [Digikam::GPSGeofaceModelHelper](#).

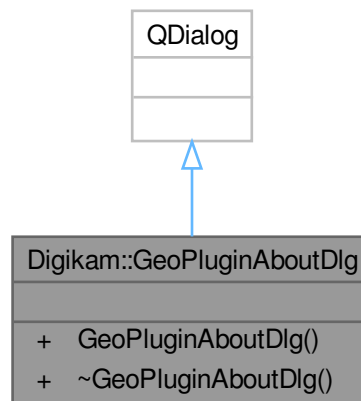
### 9.619.2.7 selectionModel()

```
virtual QItemSelectionModel * Digikam::GeoModelHelper::selectionModel ( ) const [pure virtual]
```

Implemented in [Digikam::MapViewModelHelper](#).

## 9.620 Digikam::GeoPluginAboutDlg Class Reference

Inheritance diagram for Digikam::GeoPluginAboutDlg:



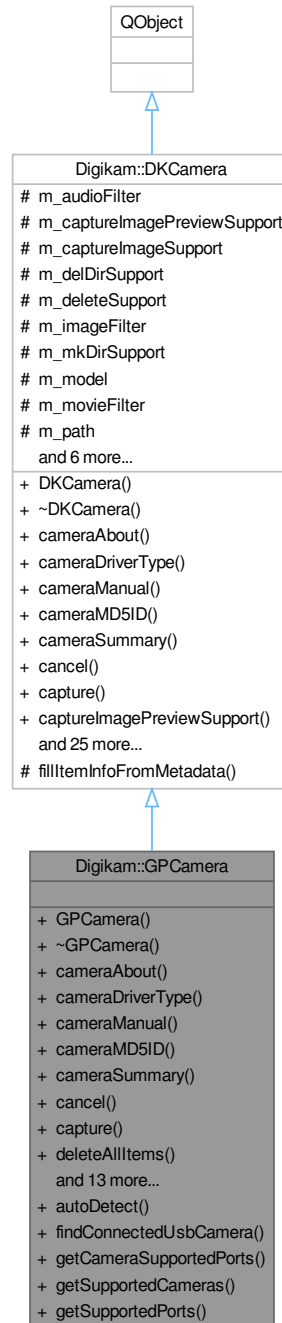
### Public Member Functions

- **GeoPluginAboutDlg** (`PluginInterface *const tool, QWidget *const parent=nullptr`)

## 9.621 Digikam::GPCamera Class Reference

Gphoto2 camera Implementation of abstract type [DKCamera](#).

Inheritance diagram for Digikam::GPCamera:



### Public Member Functions

- **GPCamera** (const QString &title, const QString &model, const QString &port, const QString &path)

- bool [cameraAbout](#) (QString &about) override
- DKCamera::CameraDriverType [cameraDriverType](#) () override
- bool [cameraManual](#) (QString &manual) override
- QByteArray [cameraMD5ID](#) () override
- bool [cameraSummary](#) (QString &summary) override
- void [cancel](#) () override
- bool [capture](#) (CamItemInfo &itemInfo) override
- bool [deleteAllItems](#) (const QString &folder)
  - recursively delete all items*
- bool [deleteItem](#) (const QString &folder, const QString &itemName) override
- bool [doConnect](#) () override
- bool [downloadItem](#) (const QString &folder, const QString &itemName, const QString &saveFile) override
- bool [getFolders](#) (const QString &folder) override
- bool [getFreeSpace](#) (qint64 &bytesSize, qint64 &bytesAvail) override
- void [getItemInfo](#) (const QString &folder, const QString &itemName, [CamItemInfo](#) &info, bool useMetadata) override
- bool [getItemsInfoList](#) (const QString &folder, bool useMetadata, [CamItemInfoList](#) &items) override
  - If getImageDimensions is false, the camera shall set width and height to -1 if the values are not immediately available.*
- bool [getItemsList](#) (const QString &folder, QStringList &itemsList)
- bool [getMetadata](#) (const QString &folder, const QString &itemName, [DMetadata](#) &meta) override
- bool [getPreview](#) (QImage &preview) override
- bool [getThumbnail](#) (const QString &folder, const QString &itemName, QImage &thumbnail) override
- bool [setLockItem](#) (const QString &folder, const QString &itemName, bool lock) override
- bool [uploadItem](#) (const QString &folder, const QString &itemName, const QString &localFile, [CamItemInfo](#) &itemInfo) override

## Public Member Functions inherited from [Digikam::DKCamera](#)

- [DKCamera](#) (const QString &title, const QString &model, const QString &port, const QString &path)
- bool [captureImagePreviewSupport](#) () const
- bool [captureImageSupport](#) () const
- bool [delDirSupport](#) () const
- bool [deleteSupport](#) () const
- QString [mimeType](#) (const QString &fileext) const
- bool [mkDirSupport](#) () const
- QString [model](#) () const
- QString [path](#) () const
- QString [port](#) () const
- void [printSupportedFeatures](#) ()
- bool [thumbnailSupport](#) () const
- QString [title](#) () const
- bool [uploadSupport](#) () const
- QString [uuid](#) () const

## Static Public Member Functions

- static int [autoDetect](#) (QString &model, QString &port)
- static bool [findConnectedUsbCamera](#) (int vendorId, int productId, QString &model, QString &port)
- static void [getCameraSupportedPorts](#) (const QString &model, QStringList &plist)
- static void [getSupportedCameras](#) (int &count, QStringList &cList)
- static void [getSupportedPorts](#) (QStringList &plist)



## Additional Inherited Members

## Public Types inherited from [Digikam::DKCamera](#)

- enum **CameraDriverType** { **GPhotoDriver** = 0 , **UMSDriver** }

## Signals inherited from [Digikam::DKCamera](#)

- void **signalFolderList** (const QStringList &)

## Protected Member Functions inherited from [Digikam::DKCamera](#)

- void **fillItemInfoFromMetadata** ([CamItemInfo](#) &item, const [DMetadata](#) &meta) const

## Protected Attributes inherited from [Digikam::DKCamera](#)

- QString **m\_audioFilter**
- bool **m\_captureImagePreviewSupport** = false
- bool **m\_captureImageSupport** = false
- bool **m\_delDirSupport** = false
- bool **m\_deleteSupport** = false
- QString **m\_imageFilter**
- bool **m\_mkDirSupport** = false
- QString **m\_model**
- QString **m\_movieFilter**
- QString **m\_path**
- QString **m\_port**
- QString **m\_rawFilter**
- bool **m\_thumbnailSupport** = false
- QString **m\_title**
- bool **m\_uploadSupport** = false
- QString **m\_uuid**

## 9.621.1 Member Function Documentation

### 9.621.1.1 cameraAbout()

```
bool Digikam::GPCamera::cameraAbout (
    QString & about ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

### 9.621.1.2 cameraDriverType()

```
DKCamera::CameraDriverType Digikam::GPCamera::cameraDriverType ( ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

### 9.621.1.3 cameraManual()

```
bool Digikam::GPCamera::cameraManual (
    QString & manual ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

### 9.621.1.4 cameraMD5ID()

```
QByteArray Digikam::GPCamera::cameraMD5ID ( ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

### 9.621.1.5 cameraSummary()

```
bool Digikam::GPCamera::cameraSummary (
    QString & summary ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

### 9.621.1.6 cancel()

```
void Digikam::GPCamera::cancel ( ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

### 9.621.1.7 capture()

```
bool Digikam::GPCamera::capture (
    CamItemInfo & itemInfo ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

### 9.621.1.8 deleteItem()

```
bool Digikam::GPCamera::deleteItem (
    const QString & folder,
    const QString & itemName ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

### 9.621.1.9 doConnect()

```
bool Digikam::GPCamera::doConnect ( ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

#### 9.621.1.10 downloadItem()

```
bool Digikam::GPCamera::downloadItem (
    const QString & folder,
    const QString & itemName,
    const QString & saveFile ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

#### 9.621.1.11 getFolders()

```
bool Digikam::GPCamera::getFolders (
    const QString & folder ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

#### 9.621.1.12 getFreeSpace()

```
bool Digikam::GPCamera::getFreeSpace (
    quint64 & bytesSize,
    quint64 & bytesAvail ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

#### 9.621.1.13 getItemInfo()

```
void Digikam::GPCamera::getItemInfo (
    const QString & folder,
    const QString & itemName,
    CamItemInfo & info,
    bool useMetadata ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

#### 9.621.1.14 getItemsInfoList()

```
bool Digikam::GPCamera::getItemsInfoList (
    const QString & folder,
    bool useMetadata,
    CamItemInfoList & infoList ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

#### 9.621.1.15 getMetadata()

```
bool Digikam::GPCamera::getMetadata (
    const QString & folder,
    const QString & itemName,
    DMetadata & meta ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

**9.621.1.16 getPreview()**

```
bool Digikam::GPCamera::getPreview (
    QImage & preview ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

**9.621.1.17 getThumbnail()**

```
bool Digikam::GPCamera::getThumbnail (
    const QString & folder,
    const QString & itemName,
    QImage & thumbnail ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

**9.621.1.18 setLockItem()**

```
bool Digikam::GPCamera::setLockItem (
    const QString & folder,
    const QString & itemName,
    bool lock ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

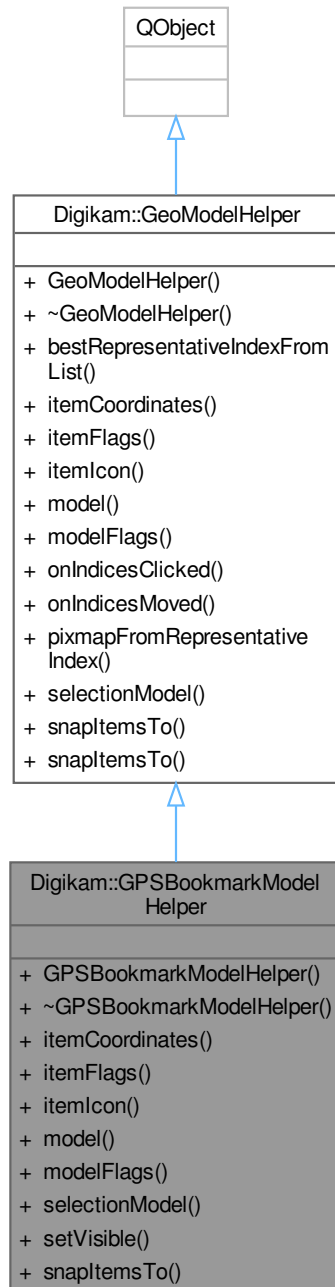
**9.621.1.19 uploadItem()**

```
bool Digikam::GPCamera::uploadItem (
    const QString & folder,
    const QString & itemName,
    const QString & localFile,
    CamItemInfo & itemInfo ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

## 9.622 Digikam::GPSBookmarkModelHelper Class Reference

Inheritance diagram for Digikam::GPSBookmarkModelHelper:



### Public Types

- enum **Constants** { **CoordinatesRole** = Qt::UserRole + 1 }

## Public Types inherited from [Digikam::GeoModelHelper](#)

- enum **PropertyFlag** { **FlagNull** = 0 , **FlagVisible** = 1 , **FlagMovable** = 2 , **FlagSnaps** = 4 }
- typedef QFlags< PropertyFlag > **PropertyFlags**

## Signals

- void **signalUndoCommand** ([GPSUndoCommand](#) \*undoCommand)

## Signals inherited from [Digikam::GeoModelHelper](#)

- void **signalModelChangedDrastically** ()
- void **signalThumbnailAvailableForIndex** (const QPersistentModelIndex &index, const QPixmap &pixmap)
- void **signalVisibilityChanged** ()

## Public Member Functions

- **GPSBookmarkModelHelper** ([BookmarksManager](#) \*const bookmarkManager, [GPSItemModel](#) \*const imageModel, QObject \*const parent=nullptr)
- bool **itemCoordinates** (const QModelIndex &index, [GeoCoordinates](#) \*const coordinates) const override
- PropertyFlags **itemFlags** (const QModelIndex &index) const override
- bool **itemIcon** (const QModelIndex &index, QPoint \*const offset, QSize \*const size, QPixmap \*const pixmap, QUrl \*const url) const override  
*these are necessary for ungrouped models*
- QAbstractItemModel \* **model** () const override  
*these are necessary for grouped and ungrouped models*
- PropertyFlags **modelFlags** () const override
- QItemSelectionModel \* **selectionModel** () const override
- void **setVisible** (const bool state)
- void **snapItemsTo** (const QModelIndex &targetIndex, const QList< QModelIndex > &snappedIndices) override

## Public Member Functions inherited from [Digikam::GeoModelHelper](#)

- **GeoModelHelper** (QObject \*const parent=nullptr)
- virtual QPersistentModelIndex **bestRepresentativeIndexFromList** (const QList< QPersistentModelIndex > &list, const int sortKey)
- virtual void **onIndicesClicked** (const QList< QPersistentModelIndex > &clickedIndices)
- virtual void **onIndicesMoved** (const QList< QPersistentModelIndex > &movedIndices, const [GeoCoordinates](#) &targetCoordinates, const QPersistentModelIndex &targetSnapIndex)
- virtual QPixmap **pixmapFromRepresentativeIndex** (const QPersistentModelIndex &index, const QSize &size)  
*these are used by MarkerModel for grouped models*
- void **snapItemsTo** (const QModelIndex &targetIndex, const QList< QPersistentModelIndex > &snappedIndices)

## 9.622.1 Member Function Documentation

### 9.622.1.1 itemCoordinates()

```
bool Digikam::GPSBookmarkModelHelper::itemCoordinates (
    const QModelIndex & index,
    GeoCoordinates *const coordinates ) const [override], [virtual]
```

Implements [Digikam::GeoModelHelper](#).

### 9.622.1.2 itemFlags()

```
GeoModelHelper::PropertyFlags Digikam::GPSBookmarkModelHelper::itemFlags (
    const QModelIndex & index ) const [override], [virtual]
```

Reimplemented from [Digikam::GeoModelHelper](#).

### 9.622.1.3 itemIcon()

```
bool Digikam::GPSBookmarkModelHelper::itemIcon (
    const QModelIndex & index,
    QPoint *const offset,
    QSize *const size,
    QPixmap *const pixmap,
    QUrl *const url ) const [override], [virtual]
```

Returns the icon for an ungrouped marker.

The icon can either be returned as a URL to an image, or as a QPixmap. If the caller can handle URLs (for example, to display them in HTML), he can provide the URL parameter. However, the [GeoModelHelper](#) may still choose to return a QPixmap instead, if no URL is available.

#### Parameters

<i>index</i>	Modelindex of the marker.
<i>offset</i>	Offset of the zero point in the icon, given from the top-left.
<i>size</i>	the size of the icon, only populated if a URL is returned.
<i>pixmap</i>	Holder for the pixmap of the icon.
<i>url</i>	URL of the icon if available.

Reimplemented from [Digikam::GeoModelHelper](#).

### 9.622.1.4 model()

```
QAbstractItemModel * Digikam::GPSBookmarkModelHelper::model ( ) const [override], [virtual]
```

Implements [Digikam::GeoModelHelper](#).

### 9.622.1.5 modelFlags()

```
GeoModelHelper::PropertyFlags Digikam::GPSBookmarkModelHelper::modelFlags ( ) const [override],
[virtual]
```

Reimplemented from [Digikam::GeoModelHelper](#).

### 9.622.1.6 selectionModel()

```
QItemSelectionModel * Digikam::GPSBookmarkModelHelper::selectionModel ( ) const [override],
[virtual]
```

Implements [Digikam::GeoModelHelper](#).

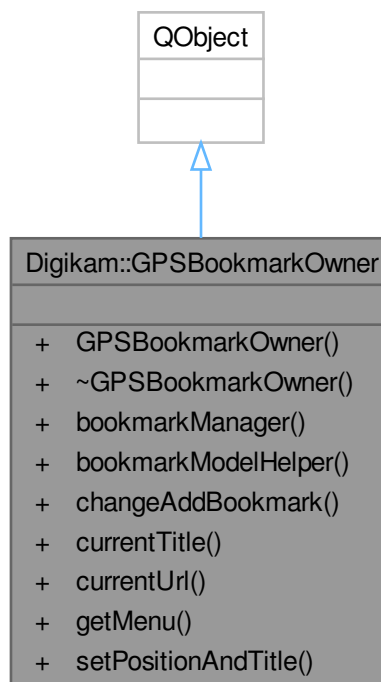
### 9.622.1.7 snapItemsTo()

```
void Digikam::GPSBookmarkModelHelper::snapItemsTo (
    const QModelIndex & targetIndex,
    const QList< QModelIndex > & snappedIndices ) [override], [virtual]
```

Reimplemented from [Digikam::GeoModelHelper](#).

## 9.623 Digikam::GPSBookmarkOwner Class Reference

Inheritance diagram for Digikam::GPSBookmarkOwner:



### Signals

- void **positionSelected** (const [GPSDataContainer](#) &position)

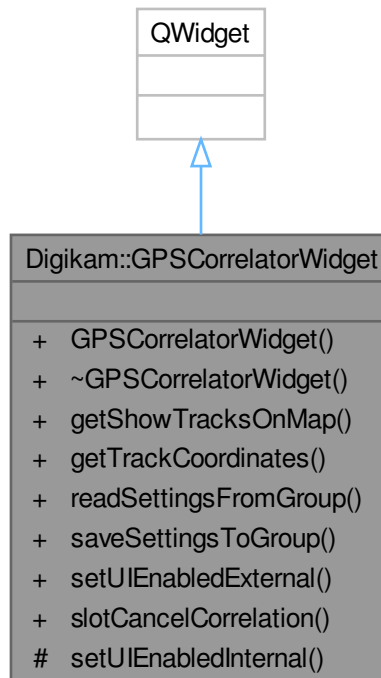
### Public Member Functions

- **GPSBookmarkOwner** ([GPSItemModel](#) \*const gpsItemModel, [QWidget](#) \*const parent)
- [BookmarksManager](#) \* **bookmarkManager** () const
- [GPSBookmarkModelHelper](#) \* **bookmarkModelHelper** () const
- void **changeAddBookmark** (const bool state)
- [QString](#) **currentTitle** () const
- [QString](#) **currentUrl** () const
- [QMenu](#) \* **getMenu** () const
- void **setPositionAndTitle** (const [GeoCoordinates](#) &coordinates, const [QString](#) &title)



## 9.624 Digikam::GPSCorrelatorWidget Class Reference

Inheritance diagram for Digikam::GPSCorrelatorWidget:



### Public Slots

- void **slotCancelCorrelation** ()

### Signals

- void **signalAllTrackFilesReady** ()
- void **signalProgressChanged** (const int currentProgress)
- void **signalProgressSetup** (const int maxProgress, const QString &progressText)
- void **signalSetUIEnabled** (const bool enabledState)
- void **signalSetUIEnabled** (const bool enabledState, QObject \*const cancelObject, const QString &cancelSlot)
- void **signalTrackListChanged** (const [Digikam::GeoCoordinates](#) &coordinate)
- void **signalUndoCommand** ([GPSUndoCommand](#) \*undoCommand)

### Public Member Functions

- **GPSCorrelatorWidget** (QWidget \*const parent, [GPSItemModel](#) \*const imageModel, QItemSelectionModel \*const selectionModel, [TrackManager](#) \*const trackManager)
- bool **getShowTracksOnMap** () const
- QList< [GeoCoordinates::List](#) > **getTrackCoordinates** () const
- void **readSettingsFromGroup** (const KConfigGroup \*const group)
- void **saveSettingsToGroup** (KConfigGroup \*const group)
- void **setUIEnabledExternal** (const bool state)

### Protected Member Functions

- void **setUIEnabledInternal** (const bool state)

## 9.625 Digikam::GPSDataContainer Class Reference

### Public Types

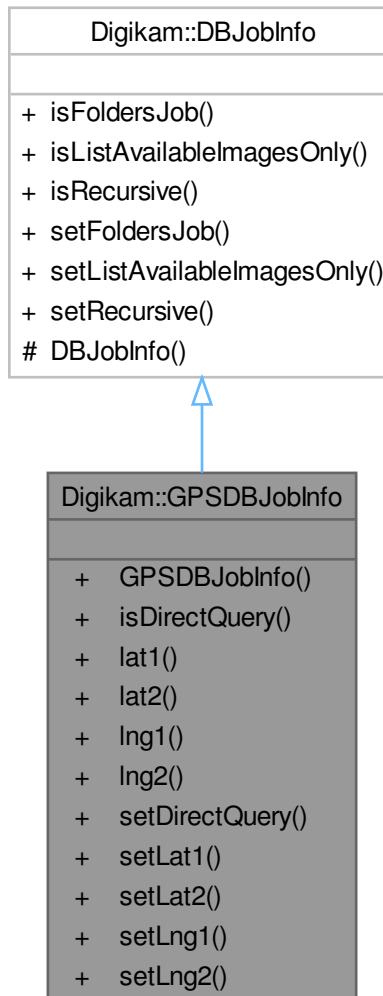
- typedef QFlags< HasFlagsEnum > **HasFlags**
- enum **HasFlagsEnum** {  
**HasCoordinates** = 1 , **HasAltitude** = 2 , **HasIsInterpolated** = 4 , **HasNSatellites** = 8 ,  
**HasDop** = 16 , **HasFixType** = 32 , **HasSpeed** = 64 }

### Public Member Functions

- void **clear** ()
- void **clearAltitude** ()
- void **clearDop** ()
- void **clearFixType** ()
- void **clearNonCoordinates** ()
- void **clearNSatellites** ()
- void **clearSpeed** ()
- HasFlags **flags** () const
- [GeoCoordinates](#) **getCoordinates** () const
- qreal **getDop** () const
- qreal **getFixType** () const
- int **getNSatellites** () const
- qreal **getSpeed** () const  
*Return the speed in m/s.*
- bool **hasAltitude** () const
- bool **hasCoordinates** () const
- bool **hasDop** () const
- bool **hasFixType** () const
- bool **hasNSatellites** () const
- bool **hasSpeed** () const
- bool **operator==** (const [GPSDataContainer](#) &b) const
- void **setAltitude** (const qreal alt)
- void **setCoordinates** (const [GeoCoordinates](#) &coordinates)
- void **setDop** (const qreal dop)
- void **setFixType** (const int fixType)
- void **setLatLon** (const qreal lat, const qreal lon)
- void **setNSatellites** (const int nSatellites)
- void **setSpeed** (const qreal speed)  
*Set the speed in m/s.*

## 9.626 Digikam::GPSDBJobInfo Class Reference

Inheritance diagram for Digikam::GPSDBJobInfo:



### Public Member Functions

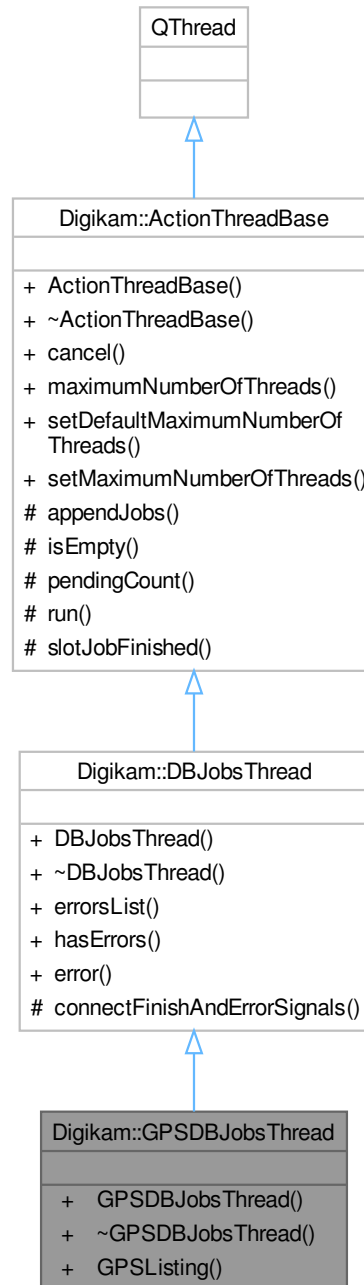
- bool **isDirectQuery** () const
- qreal **lat1** () const
- qreal **lat2** () const
- qreal **lng1** () const
- qreal **lng2** () const
- void **setDirectQuery** ()
- void **setLat1** (qreal lat)
- void **setLat2** (qreal lat)
- void **setLng1** (qreal lng)
- void **setLng2** (qreal lng)

**Public Member Functions inherited from [Digikam::DBJobInfo](#)**

- bool **isFoldersJob** () const
- bool **isListAvailableImagesOnly** () const
- bool **isRecursive** () const
- void **setFoldersJob** ()
- void **setListAvailableImagesOnly** ()
- void **setRecursive** ()

## 9.627 Digikam::GPSDBJobsThread Class Reference

Inheritance diagram for Digikam::GPSDBJobsThread:



### Signals

- void **directQueryData** (const QList< QVariant > &data)

## Signals inherited from [Digikam::DBJobsThread](#)

- void **data** (const QList< [ItemLISTERRecord](#) > &records)
- void **finished** ()

## Public Member Functions

- **GPSDBJobsThread** (QObject \*const parent)
- void [GPSListing](#) (const [GPSDBJobInfo](#) &info)  
*Starts GPS listing and scanning.*

## Public Member Functions inherited from [Digikam::DBJobsThread](#)

- **DBJobsThread** (QObject \*const parent)
- QList< QString > & [errorsList](#) ()  
*A method to get all errors reported from jobs.*
- bool [hasErrors](#) ()  
*hasErrors: a method to check for jobs errors*

## Public Member Functions inherited from [Digikam::ActionThreadBase](#)

- **ActionThreadBase** (QObject \*const parent=nullptr)
- void **cancel** (bool isCancel=true)  
*Cancel processing of current jobs under progress.*
- int **maximumNumberOfThreads** () const  
*Return the maximum number of threads used to parallelize collection of job processing.*
- void [setDefaultMaximumNumberOfThreads](#) ()  
*Reset maximum number of threads used to parallelize collection of job processing to max core detected on computer.*
- void **setMaximumNumberOfThreads** (int n)  
*Adjust maximum number of threads used to parallelize collection of job processing.*

## Additional Inherited Members

## Public Slots inherited from [Digikam::DBJobsThread](#)

- void [error](#) (const QString &errString)  
*Appends the error string to m\_errorsList.*

## Protected Slots inherited from [Digikam::ActionThreadBase](#)

- void [slotJobFinished](#) ()

## Protected Member Functions inherited from [Digikam::DBJobsThread](#)

- void [connectFinishAndErrorSignals](#) (DBJob \*const j)  
*Connects the signals of job to the signals of the thread.*

## Protected Member Functions inherited from [Digikam::ActionThreadBase](#)

- void [appendJobs](#) (const [ActionJobCollection](#) &jobs)  
*Append a collection of jobs to process into QThreadPool.*
- bool [isEmpty](#) () const  
*Return true if list of pending jobs to process is empty.*
- int [pendingCount](#) () const  
*Return the number of pending jobs to process.*
- void [run](#) () override  
*Main thread loop used to process jobs in todo list.*

### 9.627.1 Member Function Documentation

#### 9.627.1.1 GPSListing()

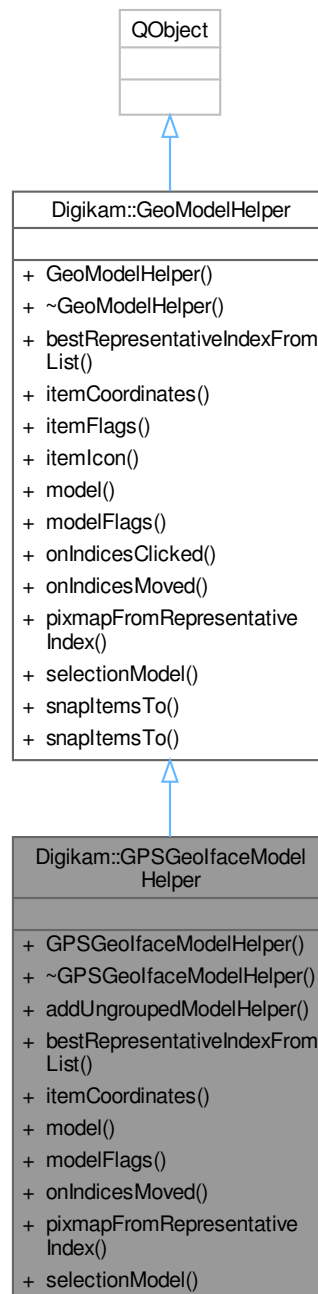
```
void Digikam::GPSDBJobsThread::GPSListing (  
    const GPSDBJobInfo & info )
```

##### Parameters

<i>info</i>	represents the GPS job info
-------------	-----------------------------

## 9.628 Digikam::GPSGeofaceModelHelper Class Reference

Inheritance diagram for Digikam::GPSGeofaceModelHelper:



### Signals

- void **signalUndoCommand** ([GPSUndoCommand](#) \*undoCommand)



## Signals inherited from [Digikam::GeoModelHelper](#)

- void **signalModelChangedDrastically** ()
- void **signalThumbnailAvailableForIndex** (const QPersistentModelIndex &index, const QPixmap &pixmap)
- void **signalVisibilityChanged** ()

## Public Member Functions

- **GPSGeofaceModelHelper** ([GPSItemModel](#) \*const **model**, QItemSelectionModel \*const **selectionModel**, QObject \*const **parent**=nullptr)
- void **addUngroupedModelHelper** ([GeoModelHelper](#) \*const **newModelHelper**)
- QPersistentModelIndex **bestRepresentativeIndexFromList** (const QList< QPersistentModelIndex > &**list**, const int **sortKey**) override
- bool **itemCoordinates** (const QModelIndex &**index**, [GeoCoordinates](#) \*const **coordinates**) const override
- QAbstractItemModel \* **model** () const override
 

*these are necessary for grouped and ungrouped models*
- PropertyFlags **modelFlags** () const override
- void **onIndicesMoved** (const QList< QPersistentModelIndex > &**movedMarkers**, const [GeoCoordinates](#) &**targetCoordinates**, const QPersistentModelIndex &**targetSnapIndex**) override
- QPixmap **pixmapFromRepresentativeIndex** (const QPersistentModelIndex &**index**, const QSize &**size**) override
 

*these are used by MarkerModel for grouped models*
- QItemSelectionModel \* **selectionModel** () const override

## Public Member Functions inherited from [Digikam::GeoModelHelper](#)

- **GeoModelHelper** (QObject \*const **parent**=nullptr)
- virtual PropertyFlags **itemFlags** (const QModelIndex &**index**) const
- virtual bool **itemIcon** (const QModelIndex &**index**, QPoint \*const **offset**, QSize \*const **size**, QPixmap \*const **pixmap**, QUrl \*const **url**) const
 

*these are necessary for ungrouped models*
- virtual void **onIndicesClicked** (const QList< QPersistentModelIndex > &**clickedIndices**)
- virtual void **snapItemsTo** (const QModelIndex &**targetIndex**, const QList< QModelIndex > &**snappedIndices**)
- void **snapItemsTo** (const QModelIndex &**targetIndex**, const QList< QPersistentModelIndex > &**snappedIndices**)

## Additional Inherited Members

## Public Types inherited from [Digikam::GeoModelHelper](#)

- enum **PropertyFlag** { **FlagNull** = 0 , **FlagVisible** = 1 , **FlagMovable** = 2 , **FlagSnaps** = 4 }
- typedef QFlags< PropertyFlag > **PropertyFlags**

### 9.628.1 Member Function Documentation

#### 9.628.1.1 bestRepresentativeIndexFromList()

```
QPersistentModelIndex Digikam::GPSGeoIfaceModelHelper::bestRepresentativeIndexFromList (
    const QList< QPersistentModelIndex > & list,
    const int sortKey ) [override], [virtual]
```

Reimplemented from [Digikam::GeoModelHelper](#).

### 9.628.1.2 itemCoordinates()

```
bool Digikam::GPSGeoIfaceModelHelper::itemCoordinates (
    const QModelIndex & index,
    GeoCoordinates *const coordinates ) const [override], [virtual]
```

Implements [Digikam::GeoModelHelper](#).

### 9.628.1.3 model()

```
QAbstractItemModel * Digikam::GPSGeoIfaceModelHelper::model ( ) const [override], [virtual]
```

Implements [Digikam::GeoModelHelper](#).

### 9.628.1.4 modelFlags()

```
GeoModelHelper::PropertyFlags Digikam::GPSGeoIfaceModelHelper::modelFlags ( ) const [override], [virtual]
```

Reimplemented from [Digikam::GeoModelHelper](#).

### 9.628.1.5 onIndicesMoved()

```
void Digikam::GPSGeoIfaceModelHelper::onIndicesMoved (
    const QList< QPersistentModelIndex > & movedMarkers,
    const GeoCoordinates & targetCoordinates,
    const QPersistentModelIndex & targetSnapIndex ) [override], [virtual]
```

Reimplemented from [Digikam::GeoModelHelper](#).

### 9.628.1.6 pixmapFromRepresentativeIndex()

```
QPixmap Digikam::GPSGeoIfaceModelHelper::pixmapFromRepresentativeIndex (
    const QPersistentModelIndex & index,
    const QSize & size ) [override], [virtual]
```

Reimplemented from [Digikam::GeoModelHelper](#).

### 9.628.1.7 selectionModel()

```
QItemSelectionModel * Digikam::GPSGeoIfaceModelHelper::selectionModel ( ) const [override], [virtual]
```

Implements [Digikam::GeoModelHelper](#).

## 9.629 Digikam::GPSItemContainer Class Reference

Inheritance diagram for Digikam::GPSItemContainer:



### Public Member Functions

- **GPSItemContainer** (const QUrl &url)

### Loading and saving

- virtual QString [saveChanges](#) ()
- virtual bool [loadImageData](#) ()
- bool **isDirty** () const
- QUrl **url** () const
- QDateTime **dateTime** () const

### GPS related functions

- void **setCoordinates** (const [GeoCoordinates](#) &newCoordinates)
- [GeoCoordinates](#) **coordinates** () const
- [GPSDataContainer](#) **gpsData** () const
- void **setGPSData** (const [GPSDataContainer](#) &container)
- void **restoreGPSData** (const [GPSDataContainer](#) &container)

*Restore the gps data to `container`.*

### Static Public Attributes

- static const int **ColumnAccuracy** = 6
- static const int **ColumnAltitude** = 5
- static const int **ColumnDateTime** = 2
- static const int **ColumnDOP** = 9
- static const int **ColumnFilename** = 1
- static const int **ColumnFixType** = 10
- static const int **ColumnGPSItemContainerCount** = 13
- static const int **ColumnLatitude** = 3
- static const int **ColumnLongitude** = 4
- static const int **ColumnNSatellites** = 11
- static const int **ColumnSpeed** = 12
- static const int **ColumnStatus** = 8
- static const int **ColumnTags** = 7
- static const int **ColumnThumbnail** = 0
- static const int **RoleCoordinates** = Qt::UserRole + 1

### Tag related functions

- [GPSItemModel](#) \* **m\_model** = nullptr
- QUrl **m\_url**
- QDateTime **m\_dateTime**
- bool **m\_dirty** = false
- [GPSDataContainer](#) **m\_gpsData**
- [GPSDataContainer](#) **m\_savedState**
- bool **m\_tagListDirty** = false
- QList< QList< [TagData](#) > > **m\_tagList**
- QList< QList< [TagData](#) > > **m\_savedTagList**
- bool **m\_writeXmpTags** = true
- bool **m\_writeMetaLoc** = true
- class **GPSItemModel**
- void **setTagList** (const QList< QList< [TagData](#) > > &externalTagList)

*The tags added in reverse geocoding process are stored in each image, before they end up in external tag model.*

- bool **isTagListDirty** () const
- QList< QList< [TagData](#) > > **getTagList** () const
- void **restoreRGTagList** (const QList< QList< [TagData](#) > > &tagList)

*Returns the tag list of the current image.*

- Replaces the current tag list with the one contained in tagList.*
- void **writeTagsToXmp** (const bool writeXmpTags)
  - Writes the current tags to XMP metadata.*
- void **writeLocations** (const bool writeMetaLoc)
  - Writes the current tags to the metadata location fields.*
- void **setLocationInfo** (const [TagData](#) &tagData, [IptcCoreLocationInfo](#) &locationInfo)
- QVariant **data** (const int column, const int role) const
  - these are only to be called by the [GPSItemModel](#)*
- void **setModel** ([GPSItemModel](#) \*const model)
- void **emitDataChanged** ()
- [DMetadata](#) \* **getMetadataForFile** () const
- [SaveProperties](#) **saveProperties** () const

### Functions used by the model

- bool **lessThan** (const [GPSItemContainer](#) \*const otherItem, const int column) const
- static void **setHeaderData** ([GPSItemModel](#) \*const model)

## 9.629.1 Member Function Documentation

### 9.629.1.1 isTagListDirty()

```
bool Digikam::GPSItemContainer::isTagListDirty ( ) const
```

#### Returns

Returns true is the current image has been modified and not saved.

### 9.629.1.2 loadImageData()

```
bool Digikam::GPSItemContainer::loadImageData ( ) [virtual]
```

### 9.629.1.3 restoreGPSData()

```
void Digikam::GPSItemContainer::restoreGPSData (
    const GPSDataContainer & container )
```

Sets m\_dirty to false if container equals savedState.

### 9.629.1.4 saveChanges()

```
QString Digikam::GPSItemContainer::saveChanges ( ) [virtual]
```

### 9.629.1.5 setTagList()

```
void Digikam::GPSItemContainer::setTagList (
    const QList< QList< TagData > > & externalTagList )
```

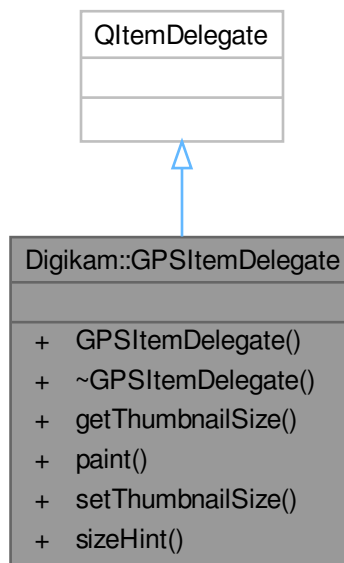
This function adds them.

## Parameters

<i>externalTagList</i>	A list containing tags.
------------------------	-------------------------

## 9.630 Digikam::GPSItemDelegate Class Reference

Inheritance diagram for Digikam::GPSItemDelegate:



### Public Member Functions

- **GPSItemDelegate** ([GPSItemList](#) \*const imageList, QObject \*const parent=nullptr)
- int **getThumbnailSize** () const
- void **paint** (QPainter \*painter, const QStyleOptionViewItem &option, const QModelIndex &sortMappedindex) const override
- void **setThumbnailSize** (const int size)
- QSize **sizeHint** (const QStyleOptionViewItem &option, const QModelIndex &sortMappedindex) const override

## 9.631 Digikam::GPSItemInfo Class Reference

### Public Types

- typedef QList< [GPSItemInfo](#) > **List**

**Static Public Member Functions**

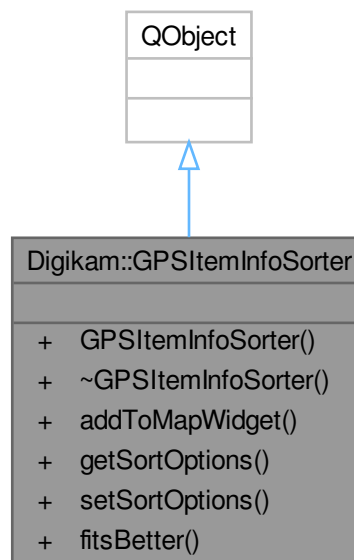
- static [GPSItemInfo](#) **fromIdCoordinatesRatingDateTime** (const qlonglong p\_id, const [GeoCoordinates](#) &p\_coordinates, const int p\_rating, const QDateTime &p\_creationDate)

**Public Attributes**

- [GeoCoordinates](#) **coordinates**
- QDateTime **dateTime**
- qlonglong **id** = -2
- int **rating** = -1
- QUrl **url**

**9.632 Digikam::GPSItemInfoSorter Class Reference**

Inheritance diagram for Digikam::GPSItemInfoSorter:

**Public Types**

- enum **SortOption** { **SortYoungestFirst** = 0 , **SortOldestFirst** = 1 , **SortRating** = 2 }
- typedef QFlags< SortOption > **SortOptions**

**Public Member Functions**

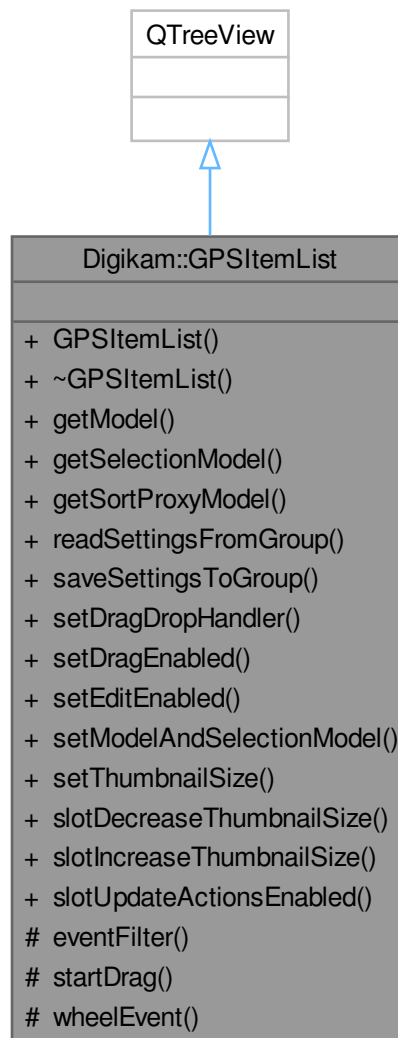
- **GPSItemInfoSorter** (QObject \*const parent)
- void **addToMapWidget** ([MapWidget](#) \*const mapWidget)
- SortOptions **getSortOptions** () const
- void **setSortOptions** (const SortOptions sortOptions)

### Static Public Member Functions

- static bool **fitsBetter** (const [GPSItemInfo](#) &oldInfo, const GeoGroupState oldState, const [GPSItemInfo](#) &newInfo, const GeoGroupState newState, const GeoGroupState globalGroupState, const SortOptions sortOptions)

## 9.633 Digikam::GPSItemList Class Reference

Inheritance diagram for Digikam::GPSItemList:



### Public Slots

- void **slotDecreaseThumbnailSize** ()
- void **slotIncreaseThumbnailSize** ()
- void **slotUpdateActionsEnabled** ()



## Signals

- void **signalImageActivated** (const QModelIndex &index)

## Public Member Functions

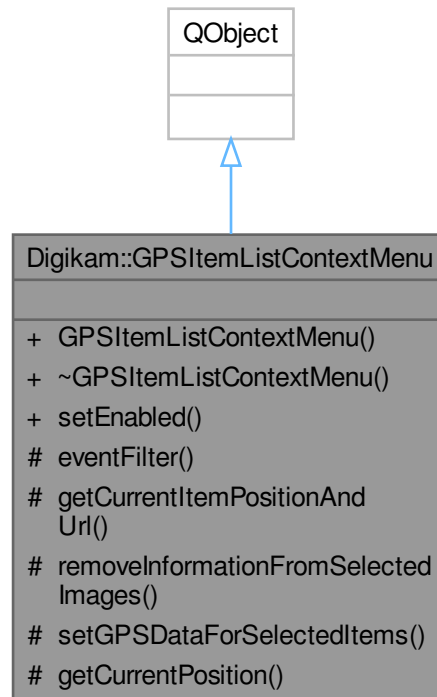
- **GPSItemList** (QWidget \*const parent=nullptr)
- [GPSItemModel](#) \* **getModel** () const
- QItemSelectionModel \* **getSelectionModel** () const
- [GPSItemSortProxyModel](#) \* **getSortProxyModel** () const
- void **readSettingsFromGroup** (const KConfigGroup \*const group)
- void **saveSettingsToGroup** (KConfigGroup \*const group)
- void **setDragDropHandler** ([ItemListDragDropHandler](#) \*const dragDropHandler)
- void **setDragEnabled** (const bool state)
- void **setEditEnabled** (const bool state)
- void **setModelAndSelectionModel** ([GPSItemModel](#) \*const model, QItemSelectionModel \*const selectionModel)
- void **setThumbnailSize** (const int size)

## Protected Member Functions

- bool **eventFilter** (QObject \*watched, QEvent \*event) override
- void **startDrag** (Qt::DropActions supportedActions) override
- void **wheelEvent** (QWheelEvent \*we) override

## 9.634 Digikam::GPSItemListContextMenu Class Reference

Inheritance diagram for Digikam::GPSItemListContextMenu:



### Signals

- void **signalProgressChanged** (const int currentProgress)
- void **signalProgressSetup** (const int maxProgress, const QString &progressText)
- void **signalSetUIEnabled** (const bool enabledState)
- void **signalSetUIEnabled** (const bool enabledState, QObject \*const cancelObject, const QString &cancelSlot)
- void **signalUndoCommand** ([GPSUndoCommand](#) \*undoCommand)

### Public Member Functions

- **GPSItemListContextMenu** ([GPSItemList](#) \*const imagesList, [GPSBookmarkOwner](#) \*const bookmarkOwner= nullptr)
- void **setEnabled** (const bool state)

### Protected Member Functions

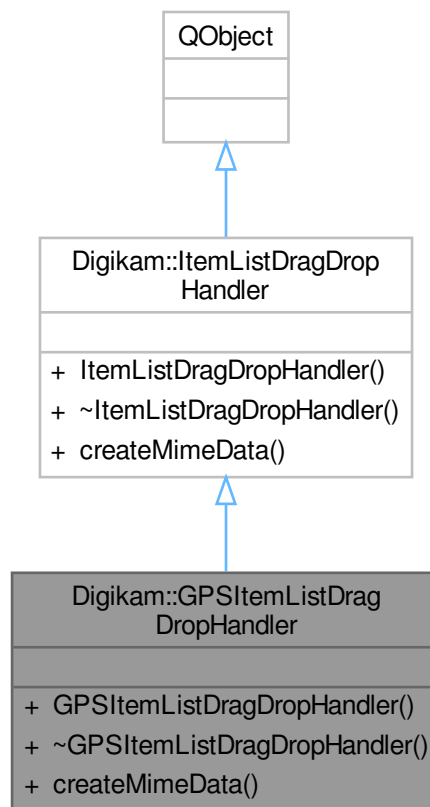
- bool **eventFilter** (QObject \*watched, QEvent \*event) override
- bool **getCurrentItemPositionAndUrl** ([GPSDataContainer](#) \*const gpsInfo, QUrl \*const itemUrl)
- void **removeInformationFromSelectedImages** (const [GPSDataContainer::HasFlags](#) flagsToClear, const QString &undoDescription)
- void **setGPSDataForSelectedItems** (const [GPSDataContainer](#) &gpsData, const QString &undoDescription)

### Static Protected Member Functions

- static bool **getCurrentPosition** ([GPSDataContainer](#) \*position, void \*mydata)

## 9.635 Digikam::GPSItemListDragDropHandler Class Reference

Inheritance diagram for Digikam::GPSItemListDragDropHandler:



### Public Member Functions

- **GPSItemListDragDropHandler** (`QObject *const parent=nullptr`)
- `QMimeData *` [createMimeData](#) (`const QList< QPersistentModelIndex > &modelIndices`) override

### Public Member Functions inherited from [Digikam::ItemListDragDropHandler](#)

- **ItemListDragDropHandler** (`QObject *const parent=nullptr`)

## 9.635.1 Member Function Documentation

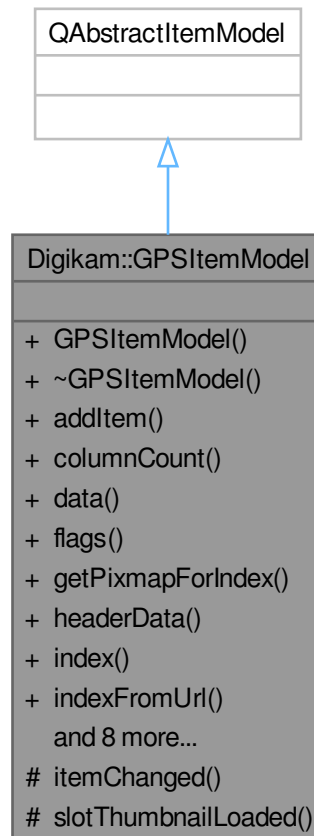
### 9.635.1.1 createMimeData()

```
QMimeData * Digikam::GPSItemListDragDropHandler::createMimeData (
    const QList< QPersistentModelIndex > & modelIndices ) [override], [virtual]
```

Implements [Digikam::ItemListDragDropHandler](#).

## 9.636 Digikam::GPSItemModel Class Reference

Inheritance diagram for Digikam::GPSItemModel:



### Signals

- void **signalThumbnailForIndexAvailable** (const QPersistentModelIndex &index, const QPixmap &pixmap)

## Public Member Functions

- **GPSItemModel** (QObject \*const parent=nullptr)
- void **addItem** ([GPSItemContainer](#) \*const newItem)
- int **columnCount** (const QModelIndex &parent=QModelIndex()) const override
- QVariant **data** (const QModelIndex &index, int role=Qt::DisplayRole) const override
- Qt::ItemFlags **flags** (const QModelIndex &index) const override
- QPixmap **getPixmapForIndex** (const QPersistentModelIndex &itemIndex, const int size)
- QVariant **headerData** (int section, Qt::Orientation orientation, int role) const override
- QModelIndex **index** (int row, int column, const QModelIndex &parent=QModelIndex()) const override
- QModelIndex **indexFromUrl** (const QUrl &url) const
- [GPSItemContainer](#) \* **itemFromIndex** (const QModelIndex &index) const
- [GPSItemContainer](#) \* **itemFromUrl** (const QUrl &url) const
- QModelIndex **parent** (const QModelIndex &index) const override
- int **rowCount** (const QModelIndex &parent=QModelIndex()) const override
- void **setColumnCount** (const int nColumns)
- bool **setData** (const QModelIndex &index, const QVariant &value, int role) override
- bool **setHeaderData** (int section, Qt::Orientation orientation, const QVariant &value, int role) override
- Qt::DropActions **supportedDragActions** () const override

## Protected Slots

- void **slotThumbnailLoaded** (const [LoadingDescription](#) &, const QPixmap &)

## Protected Member Functions

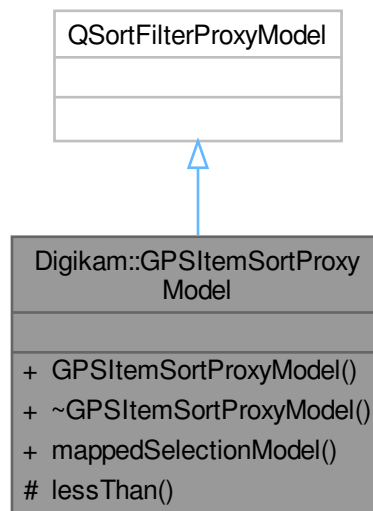
- void **itemChanged** ([GPSItemContainer](#) \*const changedItem)

## Friends

- class **GPSItemContainer**

## 9.637 Digikam::GPSItemSortProxyModel Class Reference

Inheritance diagram for Digikam::GPSItemSortProxyModel:



### Public Member Functions

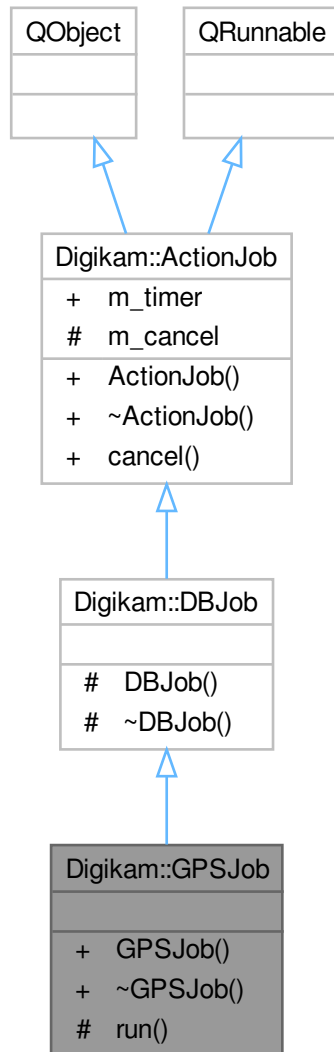
- **GPSItemSortProxyModel** ([GPSItemModel](#) \*const imageModel, QItemSelectionModel \*const source←→ SelectionModel)
- QItemSelectionModel \* **mappedSelectionModel** () const

### Protected Member Functions

- bool **lessThan** (const QModelIndex &left, const QModelIndex &right) const override

## 9.638 Digikam::GPSJob Class Reference

Inheritance diagram for Digikam::GPSJob:



### Signals

- void **directQueryData** (const QList< QVariant > &data)

### Signals inherited from [Digikam::DBJob](#)

- void **data** (const QList< [ItemListerRecord](#) > &records)
- void **error** (const QString &err)

## Signals inherited from [Digikam::ActionJob](#)

- void **signalDone** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job is done.*
- void **signalProgress** (int)  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job progress.*
- void **signalStarted** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager that job is started.*

## Public Member Functions

- **GPSJob** (const [GPSDBJobInfo](#) &jobInfo)

## Public Member Functions inherited from [Digikam::ActionJob](#)

- **ActionJob** (QObject \*const parent=nullptr)  
*Constructor which delegate deletion of QRunnable instance to [ActionThreadBase](#), not QThreadPool.*
- **~ActionJob** () override  
*Re-implement destructor in you implementation.*

## Protected Member Functions

- void **run** () override

## Additional Inherited Members

## Public Slots inherited from [Digikam::ActionJob](#)

- void **cancel** ()  
*Call this method to cancel job.*

## Public Attributes inherited from [Digikam::ActionJob](#)

- QElapsedTimer **m\_timer**  
*Timer to determine the running time of the job.*

## Protected Attributes inherited from [Digikam::ActionJob](#)

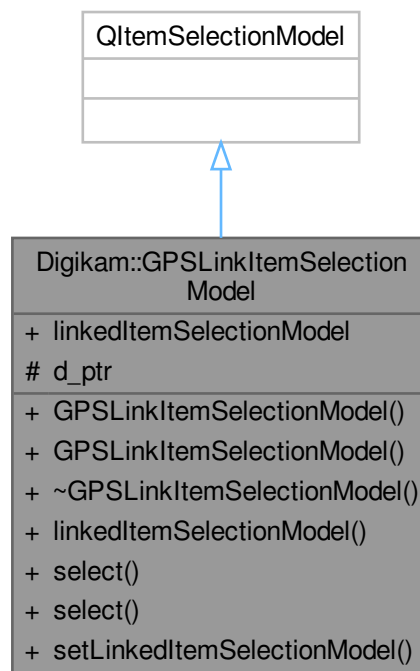
- bool **m\_cancel** = false  
*You can use this boolean in your implementation to know if job must be canceled.*



## 9.639 Digikam::GPSLinkItemSelectionModel Class Reference

Makes it possible to share a selection in multiple views which do not have the same source model.

Inheritance diagram for Digikam::GPSLinkItemSelectionModel:



### Signals

- void **linkedItemSelectionModelChanged** ()

### Public Member Functions

- **GPSLinkItemSelectionModel** (QAbstractItemModel \*const targetModel, QItemSelectionModel \*const linkedItemSelectionModel, QObject \*const parent=nullptr)
- **GPSLinkItemSelectionModel** (QObject \*const parent=nullptr)
- QItemSelectionModel \* **linkedItemSelectionModel** () const
- void **select** (const QItemSelection &selection, QItemSelectionModel::SelectionFlags command) override
- void **select** (const QModelIndex &index, QItemSelectionModel::SelectionFlags command) override
- void **setLinkedItemSelectionModel** (QItemSelectionModel \*const selectionModel)

### Protected Attributes

- GPSLinkItemSelectionModelPrivate \*const **d\_ptr**

## Properties

- QItemSelectionModel \* **linkedItemSelectionModel**

### 9.639.1 Detailed Description

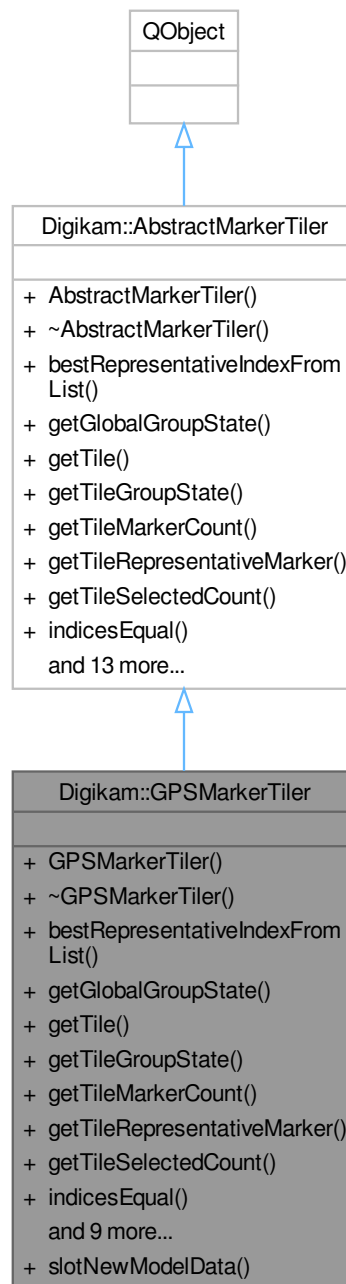
Although Qt documentation, multiple views can share the same QItemSelectionModel, the views then need to have the same source model.

If there is a proxy model between the model and one of the views, or different proxy models in each, this class makes it possible to share the selection between the views.

## 9.640 Digikam::GPSTiler Class Reference

Marker model for storing data needed to display markers on the map.

Inheritance diagram for Digikam::GPSMarkerTiler:



### Public Slots

- void `slotNewModelData` (const `QList< ItemInfo >` &infoList)  
*Receives notifications from the album model about new items.*

### Signals

- void `signalModelFilteredImages` (const `QList< qlonglong >` &imagesId)

## Signals inherited from [Digikam::AbstractMarkerTiler](#)

- void **signalThumbnailAvailableForIndex** (const QVariant &index, const QPixmap &pixmap)
- void **signalTilesOrSelectionChanged** ()

## Public Member Functions

- [GPSMarkerTiler](#) (QObject \*const parent, [ItemFilterModel](#) \*const imageFilterModel, QItemSelectionModel \*const selectionModel)  
*Constructor.*
- [~GPSMarkerTiler](#) () override  
*Destructor.*
- QVariant [bestRepresentativeIndexFromList](#) (const QList< QVariant > &indices, const int sortKey) override  
*This function finds the best representative marker from a group of markers.*
- GeoGroupState [getGlobalGroupState](#) () override
- [AbstractMarkerTiler::Tile](#) \* [getTile](#) (const [TileIndex](#) &tileIndex, const bool stopIfEmpty) override  
*Returns a pointer to a tile.*
- GeoGroupState [getTileGroupState](#) (const [TileIndex](#) &tileIndex) override
- int [getTileMarkerCount](#) (const [TileIndex](#) &tileIndex) override
- QVariant [getTileRepresentativeMarker](#) (const [TileIndex](#) &tileIndex, const int sortKey) override  
*This function finds the best representative marker from a tile of markers.*
- int [getTileSelectedCount](#) (const [TileIndex](#) &tileIndex) override
- bool [indicesEqual](#) (const QVariant &a, const QVariant &b) const override  
*This function compares two marker indices.*
- void [onIndicesClicked](#) (const [ClickInfo](#) &clickInfo) override  
*These can be implemented if you want to react to actions in geolocation interface.*
- QPixmap [pixmapFromRepresentativeIndex](#) (const QVariant &index, const QSize &size) override  
*This function retrieves the thumbnail for an index.*
- void [prepareTiles](#) (const [GeoCoordinates](#) &upperLeft, const [GeoCoordinates](#) &lowerRight, int level) override  
*Requests all images inside a given rectangle from the database.*
- void [regenerateTiles](#) () override
- void [removeCurrentRegionSelection](#) ()
- void [setActive](#) (const bool state) override  
*Sets the map active/inactive.*
- void [setPositiveFilterIsActive](#) (const bool state)
- void [setRegionSelection](#) (const [GeoCoordinates::Pair](#) &sel)
- [Tile](#) \* [tileNew](#) () override

## Public Member Functions inherited from [Digikam::AbstractMarkerTiler](#)

- [AbstractMarkerTiler](#) (QObject \*const parent=nullptr)
- bool [indicesEqual](#) (const QList &a, const QList &b, const int upToLevel) const
- bool [isDirty](#) () const
- virtual void [onIndicesMoved](#) (const [TileIndex::List](#) &tileIndicesList, const [GeoCoordinates](#) &target←Coordinates, const QPersistentModelIndex &targetSnapIndex)
- void [resetRootTile](#) ()
- [Tile](#) \* [rootTile](#) ()
- void [setDirty](#) (const bool state=true)
- virtual TilerFlags [tilerFlags](#) () const  
*These have to be implemented.*

## Additional Inherited Members

## Public Types inherited from [Digikam::AbstractMarkerTiler](#)

- enum **TilerFlag** { **FlagNull** = 0 , **FlagMovable** = 1 }
- typedef QFlags< TilerFlag > **TilerFlags**

### 9.640.1 Detailed Description

The data is retrieved from [Digikam's](#) database.

### 9.640.2 Constructor & Destructor Documentation

#### 9.640.2.1 GPSTiler()

```
Digikam::GPSTiler::GPSTiler (
    QObject *const parent,
    ItemFilterModel *const imageFilterModel,
    QItemSelectionModel *const selectionModel ) [explicit]
```

#### Parameters

<i>parent</i>	The parent object
<i>imageFilterModel</i>	The image filter instance
<i>selectionModel</i>	The selection model instance

### 9.640.3 Member Function Documentation

#### 9.640.3.1 bestRepresentativeIndexFromList()

```
QVariant Digikam::GPSTiler::bestRepresentativeIndexFromList (
    const QList< QVariant > & indices,
    const int sortKey ) [override], [virtual]
```

This is needed to display a thumbnail for a marker group.

#### Parameters

<i>indices</i>	A list containing markers, obtained by <code>getTileRepresentativeMarker</code> .
<i>sortKey</i>	Sets the criteria for selecting the representative thumbnail, a combination of the <code>SortOptions</code> bits.

#### Returns

Returns the internally used index of the marker.

Implements [Digikam::AbstractMarkerTiler](#).

**9.640.3.2 getGlobalGroupState()**

```
GeoGroupState Digikam::GPSMarkerTiler::getGlobalGroupState ( ) [override], [virtual]
```

Implements [Digikam::AbstractMarkerTiler](#).

**9.640.3.3 getTile()**

```
AbstractMarkerTiler::Tile * Digikam::GPSMarkerTiler::getTile (
    const TileIndex & tileIndex,
    const bool stopIfEmpty ) [override], [virtual]
```

**Parameters**

<i>tileIndex</i>	The index of a tile.
<i>stopIfEmpty</i>	Determines whether child tiles are also created for empty tiles.

Implements [Digikam::AbstractMarkerTiler](#).

**9.640.3.4 getTileGroupState()**

```
GeoGroupState Digikam::GPSMarkerTiler::getTileGroupState (
    const TileIndex & tileIndex ) [override], [virtual]
```

Implements [Digikam::AbstractMarkerTiler](#).

**9.640.3.5 getTileMarkerCount()**

```
int Digikam::GPSMarkerTiler::getTileMarkerCount (
    const TileIndex & tileIndex ) [override], [virtual]
```

Implements [Digikam::AbstractMarkerTiler](#).

**9.640.3.6 getTileRepresentativeMarker()**

```
QVariant Digikam::GPSMarkerTiler::getTileRepresentativeMarker (
    const TileIndex & tileIndex,
    const int sortKey ) [override], [virtual]
```

**Parameters**

<i>tileIndex</i>	Index of the tile from which the best marker should be found.
<i>sortKey</i>	Sets the criteria for selecting the representative thumbnail, a combination of the SortOptions bits.

**Returns**

Returns the internally used index of the marker.

Implements [Digikam::AbstractMarkerTiler](#).

### 9.640.3.7 getTileSelectedCount()

```
int Digikam::GPSTiler::getTileSelectedCount (
    const TileIndex & tileIndex ) [override], [virtual]
```

Implements [Digikam::AbstractMarkerTiler](#).

### 9.640.3.8 indicesEqual()

```
bool Digikam::GPSTiler::indicesEqual (
    const QVariant & a,
    const QVariant & b ) const [override], [virtual]
```

Implements [Digikam::AbstractMarkerTiler](#).

### 9.640.3.9 onIndicesClicked()

```
void Digikam::GPSTiler::onIndicesClicked (
    const ClickInfo & clickInfo ) [override], [virtual]
```

Reimplemented from [Digikam::AbstractMarkerTiler](#).

### 9.640.3.10 pixmapFromRepresentativeIndex()

```
QPixmap Digikam::GPSTiler::pixmapFromRepresentativeIndex (
    const QVariant & index,
    const QSize & size ) [override], [virtual]
```

#### Parameters

<i>index</i>	The marker's index.
<i>size</i>	The size of the thumbnail.

#### Returns

If the thumbnail has been loaded in the [ThumbnailLoadThread](#) instance, it is returned. If not, a [QPixmap](#) is returned and [ThumbnailLoadThread](#)'s signal named `signalThumbnailLoaded` is emitted when the thumbnail becomes available.

Implements [Digikam::AbstractMarkerTiler](#).

### 9.640.3.11 prepareTiles()

```
void Digikam::GPSTiler::prepareTiles (
    const GeoCoordinates & upperLeft,
```

```
const GeoCoordinates & lowerRight,
int level ) [override], [virtual]
```

This function calls the database for the images found inside a rectangle defined by upperLeft and lowerRight points. The images are returned from the database in batches.

#### Parameters

<i>upperLeft</i>	The North-West point.
<i>lowerRight</i>	The South-East point.
<i>level</i>	The requested tiling level.

Implements [Digikam::AbstractMarkerTiler](#).

#### 9.640.3.12 regenerateTiles()

```
void Digikam::GPSTiler::regenerateTiles ( ) [override], [virtual]
```

Implements [Digikam::AbstractMarkerTiler](#).

#### 9.640.3.13 setActive()

```
void Digikam::GPSTiler::setActive (
const bool state ) [override], [virtual]
```

#### Parameters

<i>state</i>	New state of the map, true means active.
--------------	--

Implements [Digikam::AbstractMarkerTiler](#).

#### 9.640.3.14 setPositiveFilterIsActive()

```
void Digikam::GPSTiler::setPositiveFilterIsActive (
const bool state )
```

#### 9.640.3.15 slotNewModelData

```
void Digikam::GPSTiler::slotNewModelData (
const QList< ItemInfo > & infoList ) [slot]
```

#### 9.640.3.16 tileNew()

```
AbstractMarkerTiler::Tile * Digikam::GPSTiler::tileNew ( ) [override], [virtual]
```

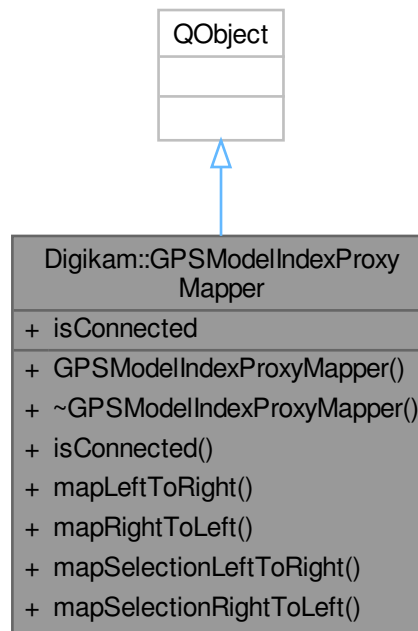
Implements [Digikam::AbstractMarkerTiler](#).



## 9.641 Digikam::GPSModelIndexProxyMapper Class Reference

This class facilitates easy mapping of indexes and selections through proxy models.

Inheritance diagram for Digikam::GPSModelIndexProxyMapper:



### Signals

- void **isConnectedChanged** ()

### Public Member Functions

- **GPSModelIndexProxyMapper** (const QAbstractItemModel \*const leftModel, const QAbstractItemModel \*const rightModel, QObject \*const parent=nullptr)
- bool **isConnected** () const
- QModelIndex **mapLeftToRight** (const QModelIndex &index) const  
*Maps the `index` from the left model to the right model.*
- QModelIndex **mapRightToLeft** (const QModelIndex &index) const  
*Maps the `index` from the right model to the left model.*
- QItemSelection **mapSelectionLeftToRight** (const QItemSelection &selection) const  
*Maps the `selection` from the left model to the right model.*
- QItemSelection **mapSelectionRightToLeft** (const QItemSelection &selection) const  
*Maps the `selection` from the right model to the left model.*

## Properties

- bool [isConnected](#)

*Indicates whether there is a chain that can be followed from leftModel to rightModel.*

### 9.641.1 Detailed Description

In a complex system of proxy models there can be a need to map indexes and selections between them, and sometimes to do so without knowledge of the path from one model to another.

If there is a need to map indexes between proxy 2 and proxy 4, a [GPSModelIndexProxyMapper](#) can be created to facilitate mapping of indexes between them.

Note that the aim is to achieve black box connections so that there is no need for application code to know the structure of proxy models in the path between left and right and attempt to manually map them.

The isConnected property indicates whether there is a path from the left side to the right side.

### 9.641.2 Property Documentation

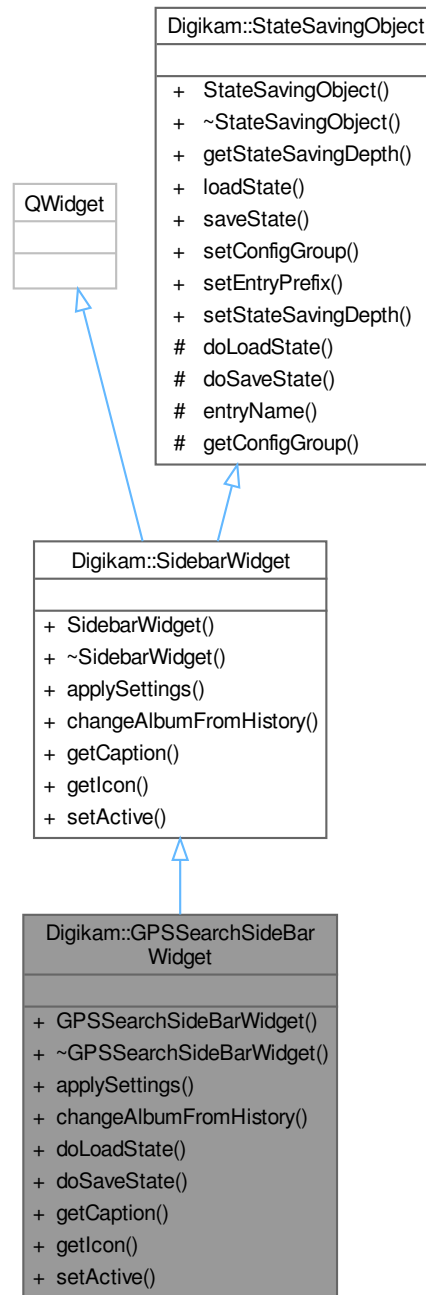
#### 9.641.2.1 isConnected

```
bool Digikam::GPSModelIndexProxyMapper::isConnected [read]
```

This value can change if the sourceModel of an intermediate proxy is changed.

## 9.642 Digikam::GPSSearchSideBarWidget Class Reference

Inheritance diagram for Digikam::GPSSearchSideBarWidget:



### Signals

- void **signalMapSololtems** (const QList< qlonglong > &, const QString &)

## Signals inherited from [Digikam::SidebarWidget](#)

- void **requestActiveTab** ([SidebarWidget](#) \*)  
*This signal can be emitted if this sidebar widget wants to be the one that is active.*
- void **signalNotificationError** (const QString &message, int type)  
*To dispatch error message to temporized pop-up notification widget hosted with icon-view.*

## Public Member Functions

- **GPSSearchSideBarWidget** (QWidget \*const parent, [SearchModel](#) \*const searchModel, [SearchModificationHelper](#) \*const searchModificationHelper, [ItemFilterModel](#) \*const imageFilterModel, [QItemSelectionModel](#) \*const itemSelectionModel)
- void **applySettings** () override  
*This method is invoked when the application settings should be (re-) applied to this widget.*
- void **changeAlbumFromHistory** (const QList< [Album](#) \* > &album) override  
*This is called on this widget when the history requires to move back to the specified album.*
- void **doLoadState** () override  
*Implement this hook method for state loading.*
- void **doSaveState** () override  
*Implement this hook method for state saving.*
- const QString **getCaption** () override  
*Must be implemented to return the title of this sidebar's tab.*
- const QIcon **getIcon** () override  
*Must be implemented and return the icon that shall be visible for this sidebar widget.*
- void **setActive** (bool active) override  
*This method is called if the visible sidebar widget is changed.*

## Public Member Functions inherited from [Digikam::SidebarWidget](#)

- [SidebarWidget](#) (QWidget \*const parent)  
*Constructor.*
- **~SidebarWidget** () override=default  
*Destructor.*

## Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (QObject \*const host)  
*Constructor.*
- virtual **~StateSavingObject** ()  
*Destructor.*
- [StateSavingDepth](#) **getStateSavingDepth** () const  
*Returns the depth used for state saving or loading.*
- void **loadState** ()  
*Invokes loading the class' state.*
- void **saveState** ()  
*Invokes saving the class' state.*
- virtual void **setConfigGroup** (const KConfigGroup &group)  
*Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void **setEntryPrefix** (const QString &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void **setStateSavingDepth** (const [StateSavingDepth](#) depth)  
*Sets the depth used for state saving or loading.*

## Additional Inherited Members

## Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }

*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

## Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- [QString](#) [entryName](#) (const [QString](#) &base) const  
*Always use this method to create config group entry names.*
- [KConfigGroup](#) [getConfigGroup](#) () const  
*Returns the config group that must be used for state saving and loading.*

## 9.642.1 Member Function Documentation

### 9.642.1.1 [applySettings\(\)](#)

```
void Digikam::GPSSearchSideBarWidget::applySettings ( ) [override], [virtual]
```

Implements [Digikam::SidebarWidget](#).

### 9.642.1.2 [changeAlbumFromHistory\(\)](#)

```
void Digikam::GPSSearchSideBarWidget::changeAlbumFromHistory (
    const QList< Album * > & album ) [override], [virtual]
```

Implements [Digikam::SidebarWidget](#).

### 9.642.1.3 [doLoadState\(\)](#)

```
void Digikam::GPSSearchSideBarWidget::doLoadState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

### 9.642.1.4 [doSaveState\(\)](#)

```
void Digikam::GPSSearchSideBarWidget::doSaveState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

### 9.642.1.5 getCaption()

```
const QString Digikam::GPSSearchSideBarWidget::getCaption ( ) [override], [virtual]
```

#### Returns

localized title string

Implements [Digikam::SidebarWidget](#).

### 9.642.1.6 getIcon()

```
const QIcon Digikam::GPSSearchSideBarWidget::getIcon ( ) [override], [virtual]
```

#### Returns

pixmap icon

Implements [Digikam::SidebarWidget](#).

### 9.642.1.7 setActive()

```
void Digikam::GPSSearchSideBarWidget::setActive (
    bool active ) [override], [virtual]
```

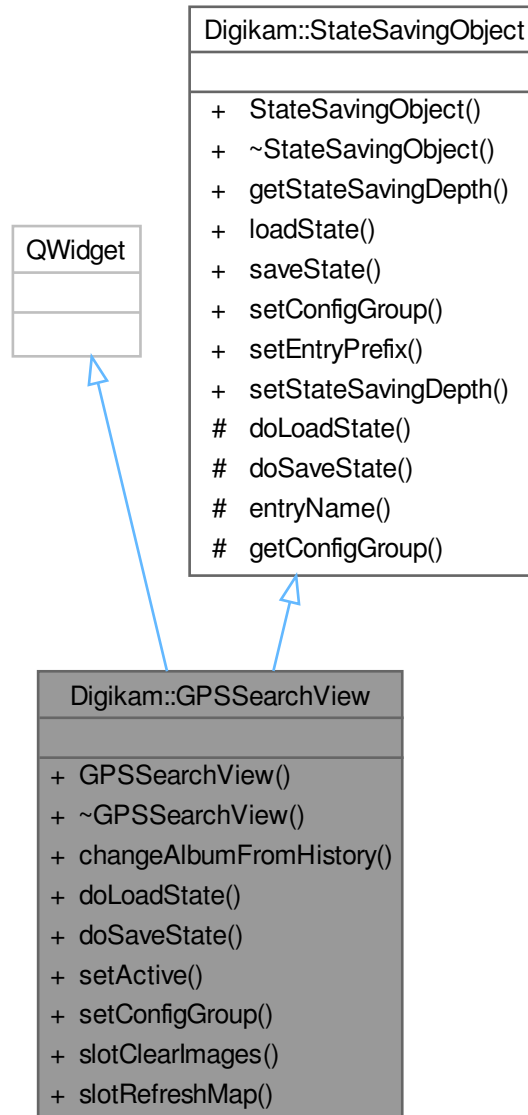
#### Parameters

<i>active</i>	if true, this widget is the new active widget, if false another widget is active
---------------	--

Implements [Digikam::SidebarWidget](#).

## 9.643 Digikam::GPSSearchView Class Reference

Inheritance diagram for Digikam::GPSSearchView:



### Public Slots

- void **slotClearImages** ()
- void **slotRefreshMap** ()

### Signals

- void **signalMapSolItems** (const QList< qlonglong > &idList, const QString &id)

## Public Member Functions

- [GPSSearchView](#) (QWidget \*const parent, [searchModel](#) \*const searchModel, [searchModificationHelper](#) \*const searchModificationHelper, [itemFilterModel](#) \*const imageFilterModel, [QItemSelectionModel](#) \*const itemSelectionModel)

*Constructor.*

- void **changeAlbumFromHistory** ([SAlbum](#) \*const album)

- void **doLoadState** () override

*Implement this hook method for state loading.*

- void **doSaveState** () override

*Implement this hook method for state saving.*

- void **setActive** (bool state)

*Sets the widget active or inactive.*

- void **setConfigGroup** (const [KConfigGroup](#) &group) override

*Sets a dedicated config group that will be used to store and reload the state from.*

## Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (QObject \*const host)

*Constructor.*

- virtual **~StateSavingObject** ()

*Destructor.*

- [StateSavingDepth](#) **getStateSavingDepth** () const

*Returns the depth used for state saving or loading.*

- void **loadState** ()

*Invokes loading the class' state.*

- void **saveState** ()

*Invokes saving the class' state.*

- virtual void **setEntryPrefix** (const [QString](#) &prefix)

*Define a prefix that will be used for every entry in the config group.*

- void **setStateSavingDepth** (const [StateSavingDepth](#) depth)

*Sets the depth used for state saving or loading.*

## Additional Inherited Members

## Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }

*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

## Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- [QString](#) **entryName** (const [QString](#) &base) const

*Always use this method to create config group entry names.*

- [KConfigGroup](#) **getConfigGroup** () const

*Returns the config group that must be used for state saving and loading.*



## 9.643.1 Constructor & Destructor Documentation

### 9.643.1.1 GPSSearchView()

```
Digikam::GPSSearchView::GPSSearchView (
    QWidget *const parent,
    SearchModel *const searchModel,
    SearchModificationHelper *const searchModificationHelper,
    ItemFilterModel *const imageFilterModel,
    QItemSelectionModel *const itemSelectionModel ) [explicit]
```

#### Parameters

<i>parent</i>	The parent object.
<i>searchModel</i>	The model that stores the searches.
<i>searchModificationHelper</i>	The helper instance to perform the searches.
<i>imageFilterModel</i>	The image model used by displaying the selected images on map.
<i>itemSelectionModel</i>	The selection model corresponding to the imageFilterModel.

## 9.643.2 Member Function Documentation

### 9.643.2.1 doLoadState()

```
void Digikam::GPSSearchView::doLoadState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

### 9.643.2.2 doSaveState()

```
void Digikam::GPSSearchView::doSaveState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

### 9.643.2.3 setActive()

```
void Digikam::GPSSearchView::setActive (
    bool state )
```

Called when the GPSSearch tab becomes the current/not current tab.

#### Parameters

<i>state</i>	When true, the widget is enabled.
--------------	-----------------------------------

### 9.643.2.4 setConfigGroup()

```
void Digikam::GPSSearchView::setConfigGroup (
    const KConfigGroup & group ) [override], [virtual]
```

If this method is not called, a group based on the object name is used.

You can re-implement this method to pass the group set here to child objects. Don't forget to call this method in your implementation.

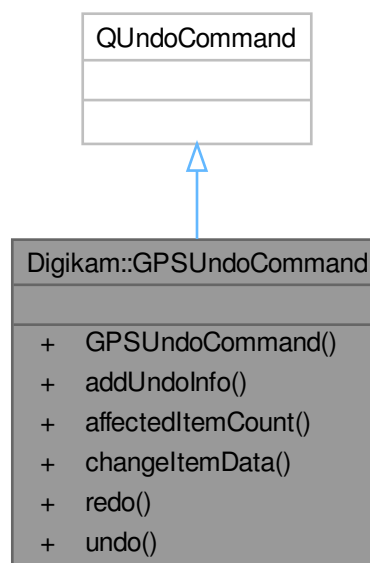
#### Parameters

<i>group</i>	config group to use for state saving and restoring
--------------	--

Reimplemented from [Digikam::StateSavingObject](#).

## 9.644 Digikam::GPSUndoCommand Class Reference

Inheritance diagram for Digikam::GPSUndoCommand:



#### Classes

- class [UndoInfo](#)

### Public Member Functions

- **GPSUndoCommand** (QUndoCommand \*const parent=nullptr)
- void **addUndoInfo** (const [UndoInfo](#) &info)
- int **affectedItemCount** () const
- void **changeItemData** (const bool redoIt)
- void **redo** () override
- void **undo** () override

## 9.645 Digikam::GPSUndoCommand::UndoInfo Class Reference

### Public Types

- typedef QList< [UndoInfo](#) > **List**

### Public Member Functions

- **UndoInfo** (const QPersistentModelIndex &pModelIndex)
- void **readNewDataFromItem** (const [GPSItemContainer](#) \*const imageItem)
- void **readOldDataFromItem** (const [GPSItemContainer](#) \*const imageItem)

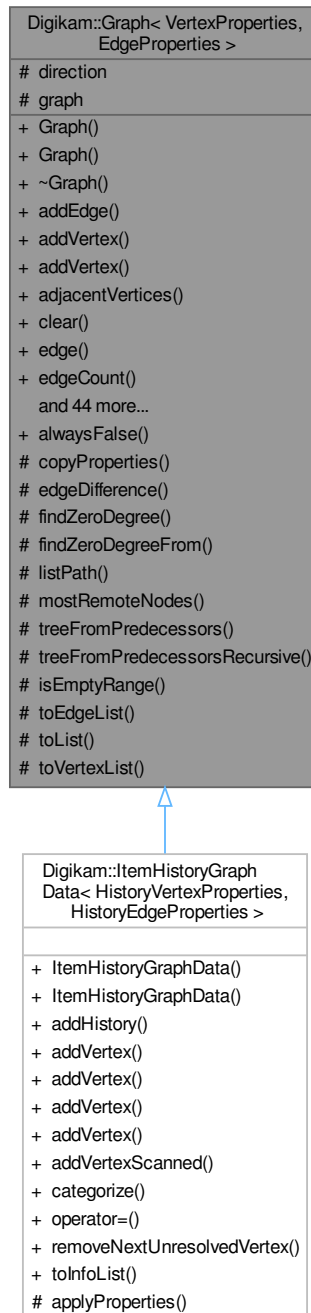
### Public Attributes

- [GPSDataContainer](#) **dataAfter**
- [GPSDataContainer](#) **dataBefore**
- QPersistentModelIndex **modelIndex**
- QList< QList< [TagData](#) > > **newTagList**
- QList< QList< [TagData](#) > > **oldTagList**

## 9.646 Digikam::Graph< VertexProperties, EdgeProperties > Class Template Reference

The graph base class template.

Inheritance diagram for Digikam::Graph< VertexProperties, EdgeProperties >:



## Classes

- class [DominatorTree](#)
- class [Edge](#)
- class [GraphSearch](#)
- class [Path](#)

*Helper class to find paths through the graph.*

- class [Vertex](#)

*These two classes provide source-compatible wrappers for the vertex and edge descriptors, providing default construction to null and the isNull() method.*

## Public Types

- typedef graph\_traits::adjacency\_iterator **adjacency\_iter**
- typedef std::pair< adjacency\_iter, adjacency\_iter > **adjacency\_vertex\_range\_t**
- enum [AdjacencyFlags](#) {  
**OutboundEdges** = 1 << 0 , **InboundEdges** = 1 << 1 , **EdgesToLeaf** = 1 << 2 , **EdgesToRoot** = 1 << 3 ,  
**AllEdges** = InboundEdges | OutboundEdges }
- typedef boost::property\_map< GraphContainer, edge\_properties\_t >::const\_type **const\_edge\_property\_map\_t**
- typedef boost::property\_map< GraphContainer, boost::vertex\_index\_t >::const\_type **const\_vertex\_index\_map\_t**
- typedef boost::property\_map< GraphContainer, vertex\_properties\_t >::const\_type **const\_vertex\_property\_map\_t**
- typedef graph\_traits::degree\_size\_type **degree\_t**
- typedef graph\_traits::edge\_iterator **edge\_iter**
- typedef boost::property\_map< GraphContainer, edge\_properties\_t >::type **edge\_property\_map\_t**
- typedef std::pair< edge\_iter, edge\_iter > **edge\_range\_t**
- typedef graph\_traits::edge\_descriptor **edge\_t**
- typedef QPair< [Edge](#), [Edge](#) > **EdgePair**
- typedef boost::graph\_traits< GraphContainer > **graph\_traits**  
*a bunch of graph-specific typedefs that make the long boost types manageable.*
- typedef boost::adjacency\_list< boost::vecS, boost::vecS, boost::bidirectionalS, boost::property< boost::vertex\_index\_t, int, boost::property< vertex\_properties\_t, VertexProperties > >, boost::property< edge\_properties\_t, EdgeProperties > > > **GraphContainer**
- enum **GraphCopyFlags** { **CopyVertexProperties** = 1 << 0 , **CopyEdgeProperties** = 1 << 1 , **CopyAllProperties** = CopyVertexProperties | CopyEdgeProperties }
- typedef graph\_traits::in\_edge\_iterator **in\_edge\_iter**
- typedef boost::inv\_adjacency\_iterator\_generator< GraphContainer, vertex\_t, in\_edge\_iter >::type **inv\_adjacency\_iter**
- typedef std::pair< inv\_adjacency\_iter, inv\_adjacency\_iter > **inv\_adjacency\_vertex\_range\_t**
- typedef graph\_traits::out\_edge\_iterator **out\_edge\_iter**
- typedef std::pair< out\_edge\_iter, out\_edge\_iter > **out\_edge\_range\_t**
- enum **ReturnOrder** { **BreadthFirstOrder** , **DepthFirstOrder** }
- typedef boost::property\_map< GraphContainer, boost::vertex\_index\_t >::type **vertex\_index\_map\_t**
- typedef graph\_traits::vertex\_iterator **vertex\_iter**
- typedef boost::property\_map< GraphContainer, vertex\_properties\_t >::type **vertex\_property\_map\_t**
- typedef std::pair< vertex\_iter, vertex\_iter > **vertex\_range\_t**
- typedef graph\_traits::vertex\_descriptor **vertex\_t**
- typedef [QMapForAdaptors](#)< [Vertex](#), int > **VertexIntMap**
- typedef boost::associative\_property\_map< [VertexIntMap](#) > **VertexIntMapAdaptor**
- typedef QPair< [Vertex](#), [Vertex](#) > **VertexPair**
- typedef [QMapForAdaptors](#)< [Vertex](#), [Vertex](#) > **VertexVertexMap**
- typedef boost::associative\_property\_map< [VertexVertexMap](#) > **VertexVertexMapAdaptor**

## Public Member Functions

- **Graph** (const [Graph](#) &g)
- **Graph** ([MeaningOfDirection](#) dir=[ParentToChild](#))
- **Edge addEdge** (const [Vertex](#) &v1, const [Vertex](#) &v2)
- **Vertex addVertex** ()
- **Vertex addVertex** (const [VertexProperties](#) &properties)
- **QList< [Vertex](#) > adjacentVertices** (const [Vertex](#) &v, [AdjacencyFlags](#) flags=[AllEdges](#)) const
- void **clear** ()
- **Edge edge** (const [Vertex](#) &v1, const [Vertex](#) &v2) const
- int **edgeCount** () const
- **QList< [VertexPair](#) > edgePairs** () const
- **QList< [Edge](#) > edges** () const
- **QList< [Edge](#) > edges** (const [Vertex](#) &v, [AdjacencyFlags](#) flags=[AllEdges](#)) const
- template<class T >  
**Vertex findVertexByProperties** (const T &value) const
- const [GraphContainer](#) & **getGraph** () const  
*Accessing vertices and edges.*
- bool **hasEdge** (const [Vertex](#) &v1, const [Vertex](#) &v2) const
- bool **hasEdges** () const
- bool **hasEdges** (const [Vertex](#) &v, [AdjacencyFlags](#) flags=[AllEdges](#)) const
- int **inDegree** (const [Vertex](#) &v) const
- bool **isConnected** (const [Vertex](#) &v1, const [Vertex](#) &v2) const
- bool **isEmpty** () const
- bool **isLeaf** (const [Vertex](#) &v) const
- bool **isRoot** (const [Vertex](#) &v) const
- **QList< [Vertex](#) > leaves** () const  
*Returns all leaves, i.e.*
- **QList< [Vertex](#) > leavesFrom** (const [Vertex](#) &v) const
- **QList< [Vertex](#) > longestPathTouching** (const [Vertex](#) &v) const  
*Returns the longest path through the graph, starting from a vertex in [roots\(\)](#), ending on a vertex in [leaves\(\)](#), and passing vertex v.*
- template<typename LessThan >  
**QList< [Vertex](#) > longestPathTouching** (const [Vertex](#) &v, LessThan lessThan) const
- [MeaningOfDirection](#) **meaningOfDirection** () const
- [Graph](#) & **operator=** (const [Graph](#) &other)
- int **outDegree** (const [Vertex](#) &v) const
- [EdgeProperties](#) & **properties** (const [Edge](#) &e)
- const [EdgeProperties](#) & **properties** (const [Edge](#) &e) const
- [VertexProperties](#) & **properties** (const [Vertex](#) &v)
- const [VertexProperties](#) & **properties** (const [Vertex](#) &v) const
- [EdgeProperties](#) **properties** (const [Vertex](#) &v1, const [Vertex](#) &v2) const
- void **remove** (const [Vertex](#) &v)
- **QList< [Vertex](#) > roots** () const  
*Returns all roots, i.e.*
- **QList< [Vertex](#) > rootsOf** (const [Vertex](#) &v) const  
*Returns all roots of vertex v.*
- void **setProperties** (const [Edge](#) &e, const [EdgeProperties](#) &props)
- void **setProperties** (const [Vertex](#) &v, const [VertexProperties](#) &props)
- **QMap< [Vertex](#), int > shortestDistancesFrom** (const [Vertex](#) &v) const  
*Returns the shortest distances from [Vertex](#) to all vertices in the graph.*
- **QList< [Vertex](#) > shortestPath** (const [Vertex](#) &v1, const [Vertex](#) &v2) const  
*Returns the shortestPath between id1 and id2.*
- [Vertex](#) **source** (const [Edge](#) &e) const

- **Vertex target** (const [Edge](#) &e) const
- [QList](#)< [Vertex](#) > **topologicalSort** () const  
*Returns the vertex ids of this graph, in topological order.*
- **Graph transitiveClosure** (GraphCopyFlags flags=CopyAllProperties) const  
*Returns a copy of this graph with all edges added to form the transitive closure.*
- **Graph transitiveReduction** ([QList](#)< [Edge](#) > \*removedEdges=0, GraphCopyFlags flags=CopyAllProperties) const  
*Returns a copy of this graph, with edges removed so that the transitive reduction is formed.*
- int **vertexCount** () const
- [QList](#)< [Vertex](#) > **vertices** () const
- [QList](#)< [Vertex](#) > **verticesBreadthFirst** (const [Vertex](#) &givenRef=[Vertex](#)()) const  
*Orders all vertices of the graph in a breadth-first manner.*
- [template](#)<typename LessThan >  
[QList](#)< [Vertex](#) > **verticesDepthFirstSorted** (const [Vertex](#) &givenRef, LessThan lessThan) const  
*Orders all vertices of the graph in a depth-first manner.*
- [QList](#)< [Vertex](#) > **verticesDominatedBy** (const [Vertex](#) &v, const [Vertex](#) &root, const [QList](#)< [Vertex](#) > &presortedVertices) const  
*For a vertex v reachable from a vertex root returns all vertices dominated by v starting from root.*
- [QList](#)< [Vertex](#) > **verticesDominatedBy** (const [Vertex](#) &v, const [Vertex](#) &root, ReturnOrder order=Breadth↔FirstOrder) const  
*For a vertex v reachable from a vertex root, returns, in depth-first or breadth-first order, all vertices dominated by v starting from root.*
- [template](#)<typename LessThan >  
[QList](#)< [Vertex](#) > **verticesDominatedByDepthFirstSorted** (const [Vertex](#) &v, const [Vertex](#) &root, LessThan lessThan) const  
*For a vertex v reachable from a vertex root all vertices dominated by v starting from root.*

### Static Public Member Functions

- [template](#)<typename T >  
static bool **alwaysFalse** (const T &, const T &)

### Protected Member Functions

- void **copyProperties** ([Graph](#) &other, GraphCopyFlags flags, const std::vector< vertex\_t > &copiedVertices) const  
*According to the given flags and based on the map, copies vertex and edge properties from this to the other graph.*
- [QList](#)< [Edge](#) > **edgeDifference** (const [Graph](#) &other, const std::vector< vertex\_t > &copiedVertices) const  
*Returns a list of edges of this graph that have been removed in other.*
- [QList](#)< [Vertex](#) > **findZeroDegree** (bool inOrOut) const  
*Finds vertex ids of all vertices with zero in- our out-degree.*
- [QList](#)< [Vertex](#) > **findZeroDegreeFrom** (const [Vertex](#) &v, bool inOrOut) const
- [QList](#)< [Vertex](#) > **listPath** (const [Vertex](#) &root, const [Vertex](#) &target, const [VertexVertexMap](#) &predecessors, MeaningOfDirection dir=ParentToChild) const  
*Get a list of vertex ids for the path from root to target, using the given predecessors.*
- [QList](#)< [Vertex](#) > **mostRemoteNodes** (const [VertexIntMap](#) &distances) const  
*Get the list of vertices with the largest value in the given distance map.*
- [QList](#)< [Vertex](#) > **treeFromPredecessors** (const [Vertex](#) &v, const [VertexVertexMap](#) &predecessors) const
- void **treeFromPredecessorsRecursive** (const [Vertex](#) &v, [QList](#)< [Vertex](#) > &vertices, const [VertexVertexMap](#) &predecessors) const

## Static Protected Member Functions

- `template<typename range_t >`  
static bool **isEmptyRange** (const range\_t &range)
- `template<typename range_t >`  
static QList< [Edge](#) > **toEdgeList** (const range\_t &range)
- `template<typename Value , typename range_t >`  
static QList< Value > **toList** (const range\_t &range)  
*Returns a list of vertex ids of vertices in the given range.*
- `template<typename range_t >`  
static QList< [Vertex](#) > **toVertexList** (const range\_t &range)

## Protected Attributes

- [MeaningOfDirection](#) **direction** = [ParentToChild](#)
- GraphContainer **graph**

## 9.646.1 Member Enumeration Documentation

### 9.646.1.1 AdjacencyFlags

```
template<class VertexProperties , class EdgeProperties >
enum Digikam::Graph::AdjacencyFlags
```

#### Enumerator

EdgesToLeaf	These resolve to one of the flags above, depending on MeaningOfDirection.
-------------	---

## 9.646.2 Member Function Documentation

### 9.646.2.1 edgeDifference()

```
template<class VertexProperties , class EdgeProperties >
QList< Edge > Digikam::Graph< VertexProperties, EdgeProperties >::edgeDifference (
    const Graph< VertexProperties, EdgeProperties > & other,
    const std::vector< vertex_t > & copiedVertices ) const [inline], [protected]
```

copiedVertices maps the vertices of this graph to other.

### 9.646.2.2 leaves()

```
template<class VertexProperties , class EdgeProperties >
QList< Vertex > Digikam::Graph< VertexProperties, EdgeProperties >::leaves ( ) const [inline]
```

vertices with no children Takes the graph direction into account.



### 9.646.2.3 listPath()

```
template<class VertexProperties , class EdgeProperties >
QList< Vertex > Digikam::Graph< VertexProperties, EdgeProperties >::listPath (
    const Vertex & root,
    const Vertex & target,
    const VertexVertexMap & predecessors,
    MeaningOfDirection dir = ParentToChild ) const [inline], [protected]
```

Depending on MeaningOfDirection, the ids are listed inverted, from target to root.

### 9.646.2.4 longestPathTouching()

```
template<class VertexProperties , class EdgeProperties >
QList< Vertex > Digikam::Graph< VertexProperties, EdgeProperties >::longestPathTouching (
    const Vertex & v ) const [inline]
```

The returned list is given in that order, root - v - leave. If there is more than one candidate for root or leave, lessThan is used to determine the first candidate.

### 9.646.2.5 roots()

```
template<class VertexProperties , class EdgeProperties >
QList< Vertex > Digikam::Graph< VertexProperties, EdgeProperties >::roots ( ) const [inline]
```

vertices with no parents. Takes the graph direction into account.

### 9.646.2.6 rootsOf()

```
template<class VertexProperties , class EdgeProperties >
QList< Vertex > Digikam::Graph< VertexProperties, EdgeProperties >::rootsOf (
    const Vertex & v ) const [inline]
```

Subset of [roots\(\)](#). I case any leaves have roots that are not roots of v, they will not be contained in this list.

### 9.646.2.7 shortestDistancesFrom()

```
template<class VertexProperties , class EdgeProperties >
QMap< Vertex, int > Digikam::Graph< VertexProperties, EdgeProperties >::shortestDistancesFrom (
    const Vertex & v ) const [inline]
```

If the value is -1, a vertex is not reachable from v.

### 9.646.2.8 shortestPath()

```
template<class VertexProperties , class EdgeProperties >
QList< Vertex > Digikam::Graph< VertexProperties, EdgeProperties >::shortestPath (
    const Vertex & v1,
    const Vertex & v2 ) const [inline]
```

If s2 is not reachable from s1, the path is searched from s2 to s1. The returned list always starts with s1, contains the intermediate vertices, and ends with s2. If no path is available, an empty list is returned.

**9.646.2.9 transitiveReduction()**

```
template<class VertexProperties , class EdgeProperties >
Graph Digikam::Graph< VertexProperties, EdgeProperties >::transitiveReduction (
    QList< Edge > * removedEdges = 0,
    GraphCopyFlags flags = CopyAllProperties ) const [inline]
```

Optionally, a list of edges of this graph that have been removed in the returned graph is given.

**9.646.2.10 vertexCount()**

```
template<class VertexProperties , class EdgeProperties >
int Digikam::Graph< VertexProperties, EdgeProperties >::vertexCount ( ) const [inline]
```

**Note**

for "hasAdjacentVertices", simply use hasEdges(v, flags).

**9.646.2.11 verticesBreadthFirst()**

```
template<class VertexProperties , class EdgeProperties >
QList< Vertex > Digikam::Graph< VertexProperties, EdgeProperties >::verticesBreadthFirst (
    const Vertex & givenRef = Vertex() ) const [inline]
```

A single vertex is taken as reference to distinguish main root and side paths. Otherwise the first root is taken as reference.

**9.646.2.12 verticesDepthFirstSorted()**

```
template<class VertexProperties , class EdgeProperties >
template<typename LessThan >
QList< Vertex > Digikam::Graph< VertexProperties, EdgeProperties >::verticesDepthFirstSorted
(
    const Vertex & givenRef,
    LessThan lessThan ) const [inline]
```

When discovering a vertex, the adjacent vertices are sorted with the given lessThan. A single vertex is taken as starting point. If null, the first root is taken as reference.

**9.646.2.13 verticesDominatedBy()**

```
template<class VertexProperties , class EdgeProperties >
QList< Vertex > Digikam::Graph< VertexProperties, EdgeProperties >::verticesDominatedBy (
    const Vertex & v,
    const Vertex & root,
    const QList< Vertex > & presortedVertices ) const [inline]
```

The order is the same as in the given, sorted list of all vertices in this graph (or all vertices expected to be returned). The returned list is the intersection of the dominated vertices and presortedVertices, in order of presortedVertices). Remove all vertices from the DFS of v that are not in the dominated tree.

### 9.646.2.14 verticesDominatedByDepthFirstSorted()

```
template<class VertexProperties , class EdgeProperties >
template<typename LessThan >
QList< Vertex > Digikam::Graph< VertexProperties, EdgeProperties >::verticesDominatedByDepthFirstSorted (
    const Vertex & v,
    const Vertex & root,
    LessThan lessThan ) const [inline]
```

The returned list is in depth-first order, using root as starting point, and when discovering a vertex, sorting the adjacent vertices with the given lessThan.

## 9.647 Digikam::Graph< VertexProperties, EdgeProperties >::DominatorTree Class Reference

### Public Member Functions

- template<class GraphType > void **enter** (const GraphType &graph, const Vertex &v, MeaningOfDirection direction=ParentToChild)

### Public Attributes

- [VertexVertexMap](#) predecessors

## 9.648 Digikam::Graph< VertexProperties, EdgeProperties >::Edge Class Reference

### Public Member Functions

- **Edge** (const edge\_t &e)
- bool **isNull** () const
- **operator const edge\_t &** () const
- **operator edge\_t &** ()
- **Edge** & **operator=** (const edge\_t &other)
- bool **operator==** (const edge\_t &other) const
- edge\_t & **toEdge** ()
- const edge\_t & **toEdge** () const

### Protected Attributes

- edge\_t **e**
- bool **null** = true

## 9.649 Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch Class Reference

### Classes

- class [BreadthFirstSearchVisitor](#)
- class [CommonVisitor](#)
- class [DepthFirstSearchVisitor](#)
- class [lessThanMapEdgeToTarget](#)

### Public Member Functions

- template<class GraphType >  
void **breadthFirstSearch** (const GraphType &graph, const [Vertex](#) &v, bool invertGraph)
- template<class GraphType >  
void **depthFirstSearch** (const GraphType &graph, const [Vertex](#) &v, bool invertGraph)
- template<class GraphType , typename LessThan >  
void **depthFirstSearchSorted** (const GraphType &graph, const [Vertex](#) &v, bool invertGraph, LessThan lessThan)

### Public Attributes

- QList< [Vertex](#) > **vertices**

### Protected Member Functions

- template<class IncidenceGraph , class DFSVisitor , class ColorMap , typename LessThan >  
void **depth\_first\_search\_sorted** (const IncidenceGraph &g, [Vertex](#) u, DFSVisitor &vis, ColorMap color, LessThan lessThan)

*This is boost's simple, old, recursive DFS algorithm adapted with lessThan.*

## 9.649.1 Member Function Documentation

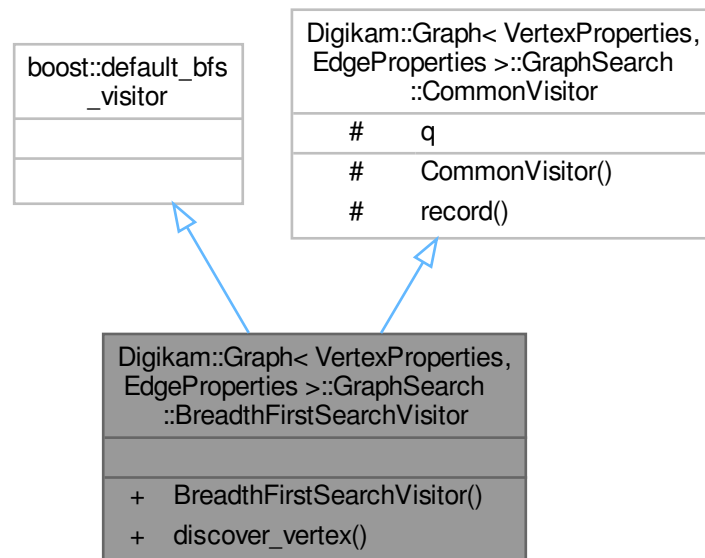
### 9.649.1.1 depth\_first\_search\_sorted()

```
template<class VertexProperties , class EdgeProperties >
template<class IncidenceGraph , class DFSVisitor , class ColorMap , typename LessThan >
void Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch::depth_first_search_↵
sorted (
    const IncidenceGraph & g,
    Vertex u,
    DFSVisitor & vis,
    ColorMap color,
    LessThan lessThan ) [inline], [protected]
```

Sort edges. The lessThan we have takes vertices, so we use a lessThan which maps the given edges to their targets, and calls our vertex lessThan.

## 9.650 Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch::BreadthFirstSearchVisitor Class Reference

Inheritance diagram for Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch::BreadthFirstSearchVisitor:



### Public Member Functions

- **BreadthFirstSearchVisitor** ([GraphSearch](#) \*const q)
- `template<typename VertexType , typename GraphType >`  
 void **discover\_vertex** (VertexType u, const GraphType &) const

### Additional Inherited Members

### Protected Member Functions inherited from

#### [Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch::CommonVisitor](#)

- **CommonVisitor** ([GraphSearch](#) \*const qq)
- void **record** (const [Vertex](#) &v) const

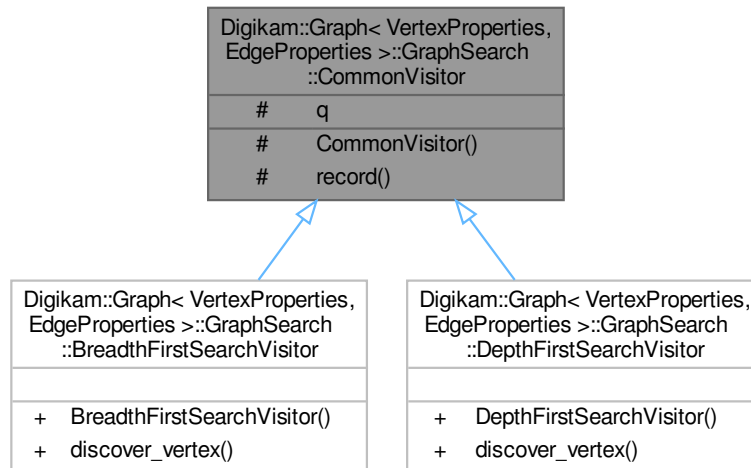
### Protected Attributes inherited from

#### [Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch::CommonVisitor](#)

- [GraphSearch](#) \*const q = nullptr

## 9.651 Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch::CommonVisitor Class Reference

Inheritance diagram for Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch::CommonVisitor:



### Protected Member Functions

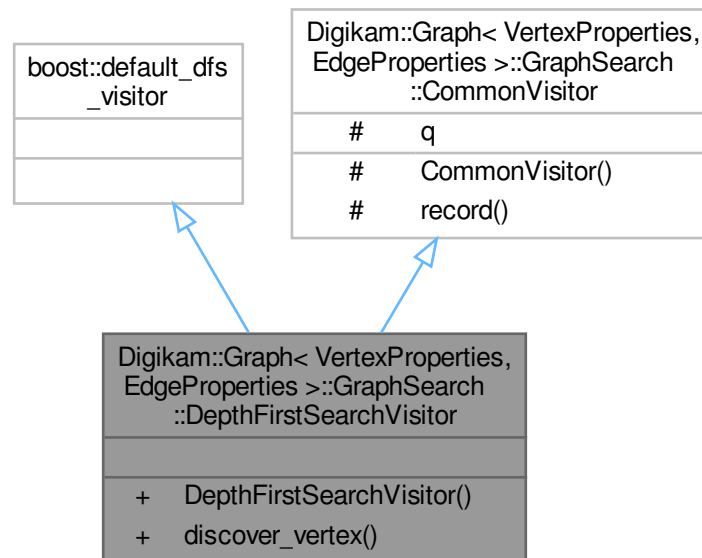
- **CommonVisitor** ([GraphSearch](#) \*const qq)
- void **record** (const [Vertex](#) &v) const

### Protected Attributes

- [GraphSearch](#) \*const **q** = nullptr

## 9.652 Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch::DepthFirstSearchVisitor Class Reference

Inheritance diagram for Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch::DepthFirstSearchVisitor:



### Public Member Functions

- **DepthFirstSearchVisitor** ([GraphSearch](#) \*const q)
- template<typename VertexType , typename GraphType >  
void **discover\_vertex** (VertexType u, const GraphType &) const

### Additional Inherited Members

#### Protected Member Functions inherited from

#### [Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch::CommonVisitor](#)

- **CommonVisitor** ([GraphSearch](#) \*const qq)
- void **record** (const [Vertex](#) &v) const

#### Protected Attributes inherited from

#### [Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch::CommonVisitor](#)

- [GraphSearch](#) \*const q = nullptr

## 9.653 Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch::lessThanMapEdgeToTarget< GraphType, VertexLessThan > Class Template Reference

### Public Member Functions

- **lessThanMapEdgeToTarget** (const GraphType &gg, VertexLessThan vertexLessThan)
- bool **operator()** (const edge\_descriptor &a, const edge\_descriptor &b)

### Public Attributes

- const GraphType & **g**
- VertexLessThan **vertexLessThan**

## 9.654 Digikam::Graph< VertexProperties, EdgeProperties >::Path Class Reference

Helper class to find paths through the graph.

### Public Member Functions

- bool **isReachable** (const [Vertex](#) &v) const
- template<class GraphType >  
void **longestPath** (const GraphType &graph, const [Vertex](#) &v)
- template<class GraphType >  
void **shortestPath** (const GraphType &graph, const [Vertex](#) &v)

### Public Attributes

- [VertexIntMap](#) **distances**
- [VertexVertexMap](#) **predecessors**

### 9.654.1 Detailed Description

```
template<class VertexProperties, class EdgeProperties>
class Digikam::Graph< VertexProperties, EdgeProperties >::Path
```

Call one of the methods and then read the maps.



## 9.654.2 Member Function Documentation

### 9.654.2.1 longestPath()

```
template<class VertexProperties , class EdgeProperties >
template<class GraphType >
void Digikam::Graph< VertexProperties, EdgeProperties >::Path::longestPath (
    const GraphType & graph,
    const Vertex & v ) [inline]
```

We provide a constant weight of 1.

Invert the default compare method: With greater, we get the longest path.

Will be returned if a node is unreachable.

Store distance and predecessors in QMap, wrapped to serve as property maps.

### 9.654.2.2 shortestPath()

```
template<class VertexProperties , class EdgeProperties >
template<class GraphType >
void Digikam::Graph< VertexProperties, EdgeProperties >::Path::shortestPath (
    const GraphType & graph,
    const Vertex & v ) [inline]
```

we provide a constant weight of 1.

Store distance and predecessors in QMap, wrapped to serve as property maps.

## 9.655 Digikam::Graph< VertexProperties, EdgeProperties >::Vertex Class Reference

These two classes provide source-compatible wrappers for the vertex and edge descriptors, providing default construction to null and the isNull() method.

### Public Member Functions

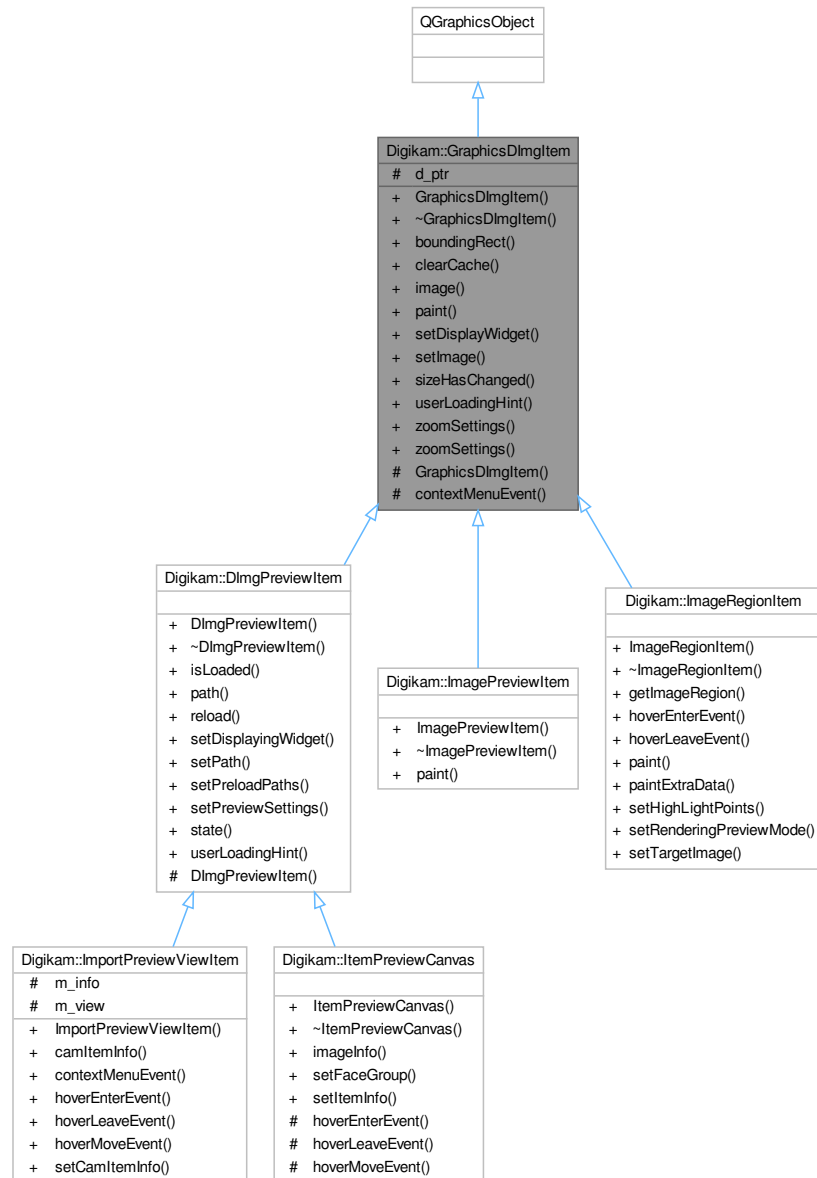
- **Vertex** (const vertex\_t &vv)
- bool **isNull** () const
- **operator const vertex\_t & ()** const
- **operator vertex\_t & ()**
- bool **operator!=** (const vertex\_t &other) const
- **Vertex** & **operator=** (const vertex\_t &other)
- bool **operator==** (const vertex\_t &other) const

### Protected Attributes

- vertex\_t v

## 9.656 Digikam::GraphicsDImgItem Class Reference

Inheritance diagram for Digikam::GraphicsDImgItem:



### Signals

- void **imageChanged** ()
- void **imageSizeChanged** (const QSizeF &size)
- void **showContextMenu** (QGraphicsSceneContextMenuEvent \*e)

### Public Member Functions

- **GraphicsDImgItem** (QGraphicsItem \*const parent=nullptr)

- QRectF **boundingRect** () const override
- void **clearCache** ()
- [DImg image](#) () const
- void **paint** (QPainter \*painter, const QStyleOptionGraphicsItem \*option, QWidget \*widget) override
- void **setDisplayWidget** (QWidget \*const widget)
- void **setImage** (const [DImg](#) &img)  
*Sets the [DImg](#) to be drawn by this item.*
- void **sizeHasChanged** ()
- virtual QString **userLoadingHint** () const
- [ImageZoomSettings](#) \* **zoomSettings** ()
- const [ImageZoomSettings](#) \* **zoomSettings** () const

### Protected Member Functions

- **GraphicsDImgItem** (GraphicsDImgItemPrivate &dd, QGraphicsItem \*const parent)
- void **contextMenuEvent** (QGraphicsSceneContextMenuEvent \*e) override

### Protected Attributes

- GraphicsDImgItemPrivate \*const **d\_ptr**

## 9.656.1 Member Function Documentation

### 9.656.1.1 setImage()

```
void Digikam::GraphicsDImgItem::setImage (  
    const DImg & img )
```

Note: [DImg](#) is explicitly shared, and no copy is automatically taken here.

## 9.657 Digikam::GraphicsDImgView Class Reference

Inheritance diagram for Digikam::GraphicsDImgView:



### Signals

- void **activated** ()
- void **contentsMoved** (bool panningFinished)

- void **contentsMoving** (int, int)
- void **leftButtonClicked** ()
- void **leftButtonDoubleClicked** ()
- void **resized** ()
- void **rightButtonClicked** ()
- void **toNextImage** ()
- void **toPreviousImage** ()
- void **viewportRectChanged** (const QRectF &viewportRect)

### Public Member Functions

- **GraphicsDImgView** (QWidget \*const parent=nullptr)
- int **contentsX** () const
- int **contentsY** () const
- void **drawText** (QPainter \*p, const QRectF &rect, const QString &text)
- void **fitToWindow** ()
- [GraphicsDImgItem](#) \* **item** () const  
*Return the instance of item set by [setItem\(\)](#).*
- [SinglePhotoPreviewLayout](#) \* **layout** () const
- [DImgPreviewItem](#) \* **previewItem** () const  
*Return a cast of item instance of item set by [setItem\(\)](#) as [DImgPreviewItem](#) Note: if you store a [GraphicsDImgItem](#) object using [setItem\(\)](#), this method will return 0.*
- void **scrollPointOnPoint** (const QPointF &scenePos, const QPoint &viewportPos)  
*Scrolls the view such that scenePos (in scene coordinates is displayed on the viewport at viewportPos (in viewport coordinates).*
- void **setContentPos** (int x, int y)
- void **setItem** ([GraphicsDImgItem](#) \*const item)  
*Store internal instance of item as [GraphicsDImgItem](#).*
- void **toggleFullScreen** (bool set)
- QRect **visibleArea** () const

### Protected Slots

- void **slotContentsMoved** ()
- void **slotCornerButtonPressed** ()
- void **slotPanIconHidden** ()
- virtual void **slotPanIconSelectionMoved** (const QRect &, bool)

### Protected Member Functions

- virtual bool **acceptsMouseClicked** (QMouseEvent \*e)
- void **continuePanning** (const QPoint &pos)
- void **drawForeground** (QPainter \*painter, const QRectF &rect) override
- void **finishPanning** ()
- void **installPanIcon** ()
- void **mouseDoubleClickEvent** (QMouseEvent \*) override
- void **mouseMoveEvent** (QMouseEvent \*) override
- void **mousePressEvent** (QMouseEvent \*) override
- void **mouseReleaseEvent** (QMouseEvent \*) override
- void **resizeEvent** (QResizeEvent \*) override
- void **scrollContentsBy** (int dx, int dy) override
- void **setScaleFitToWindow** (bool value)
- void **setShowText** (bool value)
- void **startPanning** (const QPoint &pos)
- void **wheelEvent** (QWheelEvent \*) override

## 9.657.1 Member Function Documentation

### 9.657.1.1 scrollPointOnPoint()

```
void Digikam::GraphicsDImageView::scrollPointOnPoint (
    const QPointF & scenePos,
    const QPoint & viewportPos )
```

E.g., calling `scrollPointOnPoint(scenePos, viewport()->rect().center())` is equivalent to calling `centerOn(scenePos)`.

### 9.657.1.2 setItem()

```
void Digikam::GraphicsDImageView::setItem (
    GraphicsDImgItem *const item )
```

You can store [DImgPreviewItem](#) object also by this method. Use `item()` or `previewItem()` to get right version. Note: if you store a [GraphicsDImgItem](#) object, `previewItem()` will return 0.

## 9.658 Digikam::GreycstorationContainer Class Reference

### Public Types

- enum **INTERPOLATION** { **NearestNeighbor** = 0 , **Linear** , **RungeKutta** }

### Public Member Functions

- void **setInpaintingDefaultSettings** ()
- void **setResizeDefaultSettings** ()
- void **setRestorationDefaultSettings** ()

### Public Attributes

- float **alpha** = 0.6F
- float **amplitude** = 60.0F
- float **anisotropy** = 0.3F
- int **btile** = 4
- float **da** = 30.0F
- float **dl** = 0.6F
- bool **fastApprox** = true
- float **gaussPrec** = 2.0F
- uint **interp** = NearestNeighbor
- uint **nbIter** = 1
- float **sharpness** = 0.7F
- float **sigma** = 1.1F
- int **tile** = 256

## 9.659 Digikam::GreycstorationFilter Class Reference

Inheritance diagram for Digikam::GreycstorationFilter:



### Public Types

- enum `MODE` { `Restore = 0`, `InPainting`, `Resize`, `SimpleResize` }

## Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

## Public Member Functions

- [GreycstorationFilter](#) ([DImg](#) \*const orgImage, const [GreycstorationContainer](#) &settings, int mode=Restore, int newWidth=0, int newHeight=0, const [QImage](#) &inPaintingMask=[QImage](#)(), [QObject](#) \*const parent=nullptr)  
*Constructor with all arguments.*
- [GreycstorationFilter](#) ([QObject](#) \*const parent=nullptr)  
*Constructor without argument.*
- void [cancelFilter](#) () override  
*Cancel the threaded computation.*
- [FilterAction](#) [filterAction](#) () override  
*Returns the action description corresponding to currently set options.*
- [QString](#) [filterIdentifier](#) () const override  
*Return the identifier for this filter in the image history.*
- void [readParameters](#) (const [FilterAction](#) &action) override
- void [setInPaintingMask](#) (const [QImage](#) &inPaintingMask)
- void [setMode](#) (int mode, int newWidth=0, int newHeight=0)
- void [setSettings](#) (const [GreycstorationContainer](#) &settings)
- void [setup](#) ()

## Public Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImg](#) \*const orgImage, [QObject](#) \*const parent, const [QString](#) &name=[QString](#)())  
*Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter](#) ([QObject](#) \*const parent=nullptr, const [QString](#) &name=[QString](#)())  
*Constructs a filter without argument.*
- const [QString](#) & [filterName](#) ()
- int [filterVersion](#) () const
- [DImg](#) [getTargetImage](#) ()
- [QList](#)< int > [multithreadedSteps](#) (int stop, int start=0) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool [parametersSuccessfullyRead](#) () const  
*Optional: error handling for readParameters.*
- virtual [QString](#) [readParametersError](#) (const [FilterAction](#) &actionThatFailed) const
- void [setFilterName](#) (const [QString](#) &name)
- void [setFilterVersion](#) (int version)  
*Replaying a filter action: Set the filter version.*
- void [setOriginalImage](#) (const [DImg](#) &orgImage)
- void [setupAndStartDirectly](#) (const [DImg](#) &orgImage, [DImgThreadedFilter](#) \*const master, int progress←Begin=0, int progressEnd=100)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void [setupFilter](#) (const [DImg](#) &orgImage)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void [startFilter](#) ()  
*Start the threaded computation.*
- virtual void [startFilterDirectly](#) ()  
*Start computation of this filter, directly in this thread.*
- virtual [QList](#)< int > [supportedVersions](#) () const



## Public Member Functions inherited from Digikam::DynamicThread

- [DynamicThread](#) (QObject \*const parent=nullptr)  
*This class extends QRunnable, so you have to reimplement virtual void [run\(\)](#).*
- [~DynamicThread](#) () override  
*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool [isFinished](#) () const
- bool [isRunning](#) () const
- QThread::Priority [priority](#) () const
- void [setEmitSignals](#) (bool emitThem)
- void [setPriority](#) (QThread::Priority priority)  
*Sets the priority for this dynamic thread.*
- State [state](#) () const

## Static Public Member Functions

- static QString [cimgVersionString](#) ()
- static int [CurrentVersion](#) ()
- static QString [DisplayableName](#) ()
- static QString [FilterIdentifier](#) ()
- static QList< int > [SupportedVersions](#) ()

## Additional Inherited Members

## Public Slots inherited from Digikam::DynamicThread

- void [start](#) ()
- void [stop](#) ()  
*Stop computation, sets the running flag to false.*
- void [wait](#) ()  
*Waits until the thread finishes.*

## Signals inherited from Digikam::DImgThreadedFilter

- void [finished](#) (bool success)  
*Emitted when the computation has completed.*
- void [progress](#) (int progress)  
*Emitted when progress info from the calculation is available.*
- void [started](#) ()  
*This signal is emitted when image data is available and the computation has started.*

## Signals inherited from Digikam::DynamicThread

- void [finished](#) ()
- void [starting](#) ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from [Digikam::DImgThreadedFilter](#)

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.659.1 Member Enumeration Documentation

### 9.659.1.1 MODE

```
enum Digikam::GreycstorationFilter::MODE
```

Enumerator

SimpleResize	Mode to resize image without to use Greycstoration algorithm.
--------------	---

## 9.659.2 Constructor & Destructor Documentation

### 9.659.2.1 GreycstorationFilter() [1/2]

```
Digikam::GreycstorationFilter::GreycstorationFilter (
    QObject *const parent = nullptr ) [explicit]
```

Before to use it, you need to call in order: `setSettings()`, `setMode()`, optionally `setInPaintingMask()`, `setOriginalImage()`, and necessary `setup()` at end.

### 9.659.2.2 GreycstorationFilter() [2/2]

```
Digikam::GreycstorationFilter::GreycstorationFilter (
    DImg *const orgImage,
    const GreycstorationContainer & settings,
    int mode = Restore,
    int newWidth = 0,
    int newHeight = 0,
    const QImage & inPaintingMask = QImage(),
    QObject *const parent = nullptr )
```

Ready to use.

## 9.659.3 Member Function Documentation

### 9.659.3.1 cancelFilter()

```
void Digikam::GreycstorationFilter::cancelFilter ( ) [override], [virtual]
```

Reimplemented from [Digikam::DImgThreadedFilter](#).

### 9.659.3.2 filterAction()

```
FilterAction Digikam::GreycstorationFilter::filterAction ( ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

### 9.659.3.3 filterIdentifier()

```
QString Digikam::GreycstorationFilter::filterIdentifier ( ) const [inline], [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

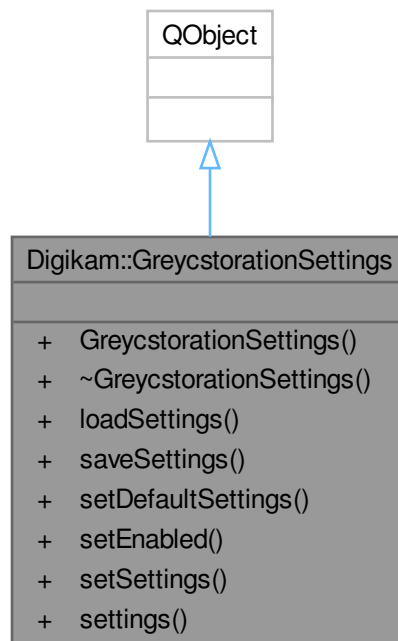
### 9.659.3.4 readParameters()

```
void Digikam::GreycstorationFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

## 9.660 Digikam::GreycstorationSettings Class Reference

Inheritance diagram for Digikam::GreycstorationSettings:



### Public Member Functions

- **GreycstorationSettings** (QTabWidget \*const parent)
- bool **loadSettings** (QFile &file, const QString &header)
- void **saveSettings** (QFile &file, const QString &header)
- void **setDefaultSettings** (const [GreycstorationContainer](#) &settings)
- void **setEnabled** (bool)
- void **setSettings** (const [GreycstorationContainer](#) &settings)
- [GreycstorationContainer](#) **settings** () const

## 9.661 Digikam::GroupedImagesFinder Class Reference

### Public Member Functions

- [GroupedImagesFinder](#) (const QList< [ItemInfo](#) > &source)

*TODO: Groups should not be resolved in dio, it should be handled in views.*

### Public Attributes

- QList< [ItemInfo](#) > infos

### 9.661.1 Constructor & Destructor Documentation

#### 9.661.1.1 GroupedImagesFinder()

```
Digikam::GroupedImagesFinder::GroupedImagesFinder (  
    const QList< ItemInfo > & source ) [explicit]
```

This is already done for most things except for drag&drop, which is hard :)

## 9.662 Digikam::GroupIndicatorOverlay Class Reference

Inheritance diagram for Digikam::GroupIndicatorOverlay:



### Signals

- void **showButtonContextMenu** (const QModelIndex &index, QContextMenuEvent \*event)
- void **toggleGroupOpen** (const QModelIndex &index)

## Signals inherited from [Digikam::ItemDelegateOverlay](#)

- void **hideNotification** ()
- void **requestNotification** (const QModelIndex &index, const QString &message)
- void **update** (const QModelIndex &index)

## Public Member Functions

- **GroupIndicatorOverlay** (QObject \*const parent)
- [GroupIndicatorOverlayWidget](#) \* **buttonWidget** () const

## Public Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- [AbstractWidgetDelegateOverlay](#) (QObject \*const parent)  
*This class provides functionality for using a widget in an overlay.*

## Public Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- **ItemDelegateOverlay** (QObject \*const parent=nullptr)
- virtual bool **acceptsDelegate** (QAbstractItemDelegate \*) const
- QAbstractItemDelegate \* **delegate** () const
- virtual void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)  
*Only these two methods are implemented as virtual methods.*
- virtual void **paint** (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index)
- void **setDelegate** (QAbstractItemDelegate \*delegate)
- void **setView** (QAbstractItemView \*view)
- QAbstractItemView \* **view** () const

## Protected Slots

- void **slotButtonClicked** ()
- void **slotButtonContextMenu** (QContextMenuEvent \*event)

## Protected Slots inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- virtual void **slotLayoutChanged** ()
- virtual void **slotReset** ()  
*Default implementations of these three slots call `hide()`*
- virtual void **slotRowsRemoved** (const QModelIndex &parent, int start, int end)
- virtual void **slotViewportEntered** ()

## Protected Slots inherited from [Digikam::ItemDelegateOverlay](#)

### Protected Member Functions

- bool [checkIndex](#) (const QModelIndex &index) const override  
*Return true here if you want to show the overlay for the given index.*
- QWidget \* [createWidget](#) () override  
*Create your widget here.*
- void [setActive](#) (bool) override  
*If active is true, this will call [createWidget\(\)](#), initialize the widget for use, and setup connections for the virtual slots.*
- void [slotEntered](#) (const QModelIndex &index) override  
*Default implementation shows the widget iff the index is valid and checkIndex returns true.*
- void [updatePosition](#) ()
- void [updateRating](#) ()
- void [visualChange](#) () override  
*Called when any change from the delegate occurs - when the overlay is installed, when size hints, styles or fonts change.*

## Protected Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- bool [checkIndexOnEnter](#) (const QModelIndex &index) const  
*Utility method called from slotEntered.*
- bool [eventFilter](#) (QObject \*obj, QEvent \*event) override
- virtual void [hide](#) ()  
*Called when the widget shall be hidden (mouse cursor left index, viewport, uninstalled etc.).*
- virtual QString [notifyMultipleMessage](#) (const QModelIndex &, int number)
- QWidget \* [parentWidget](#) () const  
*Returns the widget to be used as parent for your widget created in [createWidget\(\)](#)*
- virtual void [viewportLeaveEvent](#) (QObject \*obj, QEvent \*event)  
*Called when a QEvent::Leave of the viewport is received.*
- virtual void [widgetEnterEvent](#) ()  
*Called when a QEvent::Enter resp.*
- void [widgetEnterNotifyMultiple](#) (const QModelIndex &index)  
*A sample implementation for above methods.*
- virtual void [widgetLeaveEvent](#) ()
- void [widgetLeaveNotifyMultiple](#) ()

## Protected Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- QList< QModelIndex > [affectedIndexes](#) (const QModelIndex &index) const
- bool [affectsMultiple](#) (const QModelIndex &index) const  
*For the context that an overlay can affect multiple items: Assuming the currently overlaid index is given.*
- int [numberOfAffectedIndexes](#) (const QModelIndex &index) const
- bool [viewHasMultiSelection](#) () const  
*Utility method.*

### Protected Attributes

- QPersistentModelIndex [m\\_index](#)



## Protected Attributes inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- bool `m_mouseButtonPressedOnWidget` = false
- `QWidget * m_widget` = nullptr

## Protected Attributes inherited from [Digikam::ItemDelegateOverlay](#)

- `QAbstractItemDelegate * m_delegate` = nullptr
- `QAbstractItemView * m_view` = nullptr

## 9.662.1 Member Function Documentation

### 9.662.1.1 `checkIndex()`

```
bool Digikam::GroupIndicatorOverlay::checkIndex (
    const QModelIndex & index ) const [override], [protected], [virtual]
```

The default implementation returns true.

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.662.1.2 `createWidget()`

```
QWidget * Digikam::GroupIndicatorOverlay::createWidget ( ) [override], [protected], [virtual]
```

When creating the object, pass `parentWidget()` as parent widget. Ownership of the object is passed. It will be deleted in `setActive(false)`.

Implements [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.662.1.3 `setActive()`

```
void Digikam::GroupIndicatorOverlay::setActive (
    bool active ) [override], [protected], [virtual]
```

If active is false, this will delete the widget and disconnect all signal from model and view to this object (!)

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.662.1.4 `slotEntered()`

```
void Digikam::GroupIndicatorOverlay::slotEntered (
    const QModelIndex & index ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

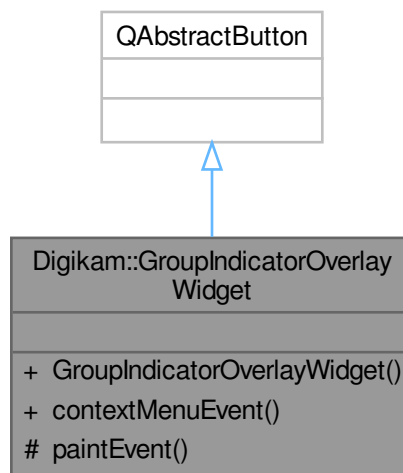
### 9.662.1.5 visualChange()

```
void Digikam::GroupIndicatorOverlay::visualChange ( ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ItemDelegateOverlay](#).

## 9.663 Digikam::GroupIndicatorOverlayWidget Class Reference

Inheritance diagram for Digikam::GroupIndicatorOverlayWidget:



### Signals

- void **contextMenu** (QContextMenuEvent \*event)

### Public Member Functions

- **GroupIndicatorOverlayWidget** (QWidget \*const parent=nullptr)
- void **contextMenuEvent** (QContextMenuEvent \*event) override

### Protected Member Functions

- void **paintEvent** (QPaintEvent \*) override

## 9.664 Digikam::GroupingViewImplementation Class Reference

Inheritance diagram for Digikam::GroupingViewImplementation:



### Public Member Functions

- [ItemInfoList](#) **getHiddenGroupedInfos** (const [ItemInfoList](#) &infos) const
- virtual bool **hasHiddenGroupedImages** (const [ItemInfo](#) &) const

*must be implemented by parent view*

- bool **needGroupResolving** ([OperationType](#) type, const [ItemInfoList](#) &infos) const
- [ItemInfoList](#) **resolveGrouping** (const [ItemInfoList](#) &infos) const

## 9.664.1 Member Function Documentation

### 9.664.1.1 hasHiddenGroupedImages()

```
virtual bool Digikam::GroupingViewImplementation::hasHiddenGroupedImages (
    const ItemInfo & ) const [inline], [virtual]
```

Reimplemented in [Digikam::ItemThumbnailBar](#), [Digikam::DigikamItemView](#), and [Digikam::TableViewTreeView](#).

## 9.665 Digikam::GroupItemFilterSettings Class Reference

### Public Member Functions

- bool **isAllOpen** () const
- bool **isFiltering** () const
  - Returns if images will be filtered by these criteria at all.*
- bool **isOpen** (qlonglong group) const
- bool **matches** (const [ItemInfo](#) &info) const
  - Returns true if the given [ItemInfo](#) matches the filter criteria.*
- bool **operator==** (const [GroupItemFilterSettings](#) &other) const
- void **setAllOpen** (bool open)
  - Open all groups.*
- void **setOpen** (qlonglong group, bool open)
  - Open or close a group.*
- [DatabaseFields::Set](#) **watchFlags** () const

### Protected Attributes

- bool **m\_allOpen** = false
- QSet< qlonglong > **m\_openGroups**

## 9.666 Digikam::GroupStateComputer Class Reference

### Public Member Functions

- void **addFilteredPositiveState** (const GeoGroupState state)
- void **addRegionSelectedState** (const GeoGroupState state)
- void **addSelectedState** (const GeoGroupState state)
- void **addState** (const GeoGroupState state)
- void **clear** ()
- GeoGroupState **getState** () const

## 9.667 Digikam::Haar::Calculator Class Reference

### Public Member Functions

- int `calcHaar` (`ImageData` \*const imageData, `SignatureData` \*const sigData)  
*Determines a total of NUM\_COEFS positions in the image that have the largest magnitude (absolute value) in color value.*
- void `transform` (`ImageData` \*const data)  
*Do the Haar tensorial 2d transform itself.*

### 9.667.1 Member Function Documentation

#### 9.667.1.1 calcHaar()

```
int Digikam::Haar::Calculator::calcHaar (
    ImageData *const data,
    SignatureData *const sigData )
```

Returns linearized coordinates in sig1, sig2, and sig3. avgl are the [0,0] values. The order of occurrence of the coordinates in sig doesn't matter. Complexity is  $3 \times \text{NUM\_PIXELS}^2 \times 2\log(\text{NUM\_COEFS})$ .

#### 9.667.1.2 transform()

```
void Digikam::Haar::Calculator::transform (
    ImageData *const data )
```

Here input is RGB data [0..255] in Unit arrays. Results are available in a, b, and c. Fully inplace calculation; order of result is interleaved though, but we don't care about that.

## 9.668 Digikam::Haar::ImageData Class Reference

### Public Member Functions

- void `fillImageData` (const `DImg` &image)  
*Write pixels of a DImg in three arrays (one per color channel, pixels linearly)*
- void `fillImageData` (const `QImage` &image)  
*Write pixels of a QImage in three arrays (one per color channel, pixels linearly)*

### Public Attributes

- Unit `data1` [NumberOfPixelsSquared] = { 0.0 }
- Unit `data2` [NumberOfPixelsSquared] = { 0.0 }
- Unit `data3` [NumberOfPixelsSquared] = { 0.0 }

## 9.669 Digikam::Haar::SignatureData Class Reference

### Public Attributes

- double **avg** [3] = { 0.0 }  
*YIQ for position [0,0].*
- Haar::Idx **sig** [3][Haar::NumberOfCoefficients] = { { 0 } }  
*Y/I/Q positions with largest magnitude.*

## 9.670 Digikam::Haar::SignatureMap Class Reference

This class provides very fast lookup if a certain pixel is set (positive or negative) in the loaded coefficient set.

### Public Types

- typedef bool **MapIndexType**

### Public Member Functions

- void **fill** (const Haar::Idx \*const coefs)  
*Load a set of coefficients.*
- bool **operator[]** (Haar::Idx index) const  
*Query if the given index is set. Index must be in the range -16383..16383.*

### Public Attributes

- MapIndexType \* **m\_indexList** = nullptr

## 9.671 Digikam::Haar::WeightBin Class Reference

### Public Member Functions

- **WeightBin** ()  
*Setup initial fixed Haar weights that each coefficient represents.*
- unsigned char **bin** (int index) const
- unsigned char **binAbs** (int index) const

### Public Attributes

- unsigned char **m\_bin** [16384] = { 0 }  
*Fixed weight mask for pixel positions (i,j).*

## 9.671.1 Member Data Documentation

### 9.671.1.1 m\_bin

```
unsigned char Digikam::Haar::WeightBin::m_bin[16384] = { 0 }
```

Each entry  $x = i * \text{NUM\_PIXELS} + j$ , gets value  $\max(i,j)$  saturated at 5. To be treated as a constant.

## 9.672 Digikam::Haar::Weights Class Reference

### Public Types

- enum **SketchType** { **ScannedSketch** = 0 , **PaintedSketch** = 1 }

### Public Member Functions

- **Weights** (SketchType type=ScannedSketch)
- float **weight** (int weight, int channel) const
- float **weightForAverage** (int channel) const

## 9.673 Digikam::Haarface Class Reference

### Public Types

- enum **AlbumTagRelation** {  
**NoMix** = 0 , **Union** = 1 , **Intersection** = 2 , **AlbumExclusive** = 3 ,  
**TagExclusive** = 4 }
- using **DuplicatesResultsMap** = QMap< qlonglong, QPair< double, QList< qlonglong > > >
- enum **DuplicatesSearchRestrictions** { **None** = 0 , **SameAlbum** = 1 , **DifferentAlbum** = 2 }
- enum class **RefImageSelMethod** : unsigned int {  
**OlderOrLarger** = 0 , **PreferFolder** = 1 , **ExcludeFolder** = 2 , **NewerCreationDate** = 3 ,  
**NewerModificationDate** = 4 }

*The RefImageSelMethod enum Selection method to determine which image will be the reference in the duplicate search.*

- enum **SketchType** { **ScannedSketch** = 0 , **HanddrawnSketch** = 1 }

## Public Member Functions

- **Haarface** (const QSet< qlonglong > &images2Scan)
- QPair< double, QMap< qlonglong, double > > **bestMatchesForImageWithThreshold** (const QString &imagePath, double requiredPercentage, double maximumPercentage, const QList< int > &targetAlbums, DuplicatesSearchRestrictions searchResultRestriction=DuplicatesSearchRestrictions::None, SketchType type=ScannedSketch)
 

*Searches the database for the best matches for the specified query image.*
- QPair< double, QMap< qlonglong, double > > **bestMatchesForImageWithThreshold** (qlonglong imageid, double requiredPercentage, double maximumPercentage, const QList< int > &targetAlbums, DuplicatesSearchRestrictions searchResultRestriction=DuplicatesSearchRestrictions::None, SketchType type=ScannedSketch)
 

*Searches the database for the best matches for the specified query image.*
- QMap< qlonglong, double > **bestMatchesForSignature** (const QString &signature, const QList< int > &targetAlbums, int numberOfResults=20, SketchType type=ScannedSketch)
- DuplicatesResultsMap **findDuplicates** (const QSet< qlonglong > &images2Scan, const QSet< qlonglong >::const\_iterator &rangeBegin, const QSet< qlonglong >::const\_iterator &rangeEnd, RefImageSelMethod reflImageSelectionMethod, const QSet< qlonglong > &refs, double requiredPercentage, double maximumPercentage, DuplicatesSearchRestrictions searchResultRestriction=DuplicatesSearchRestrictions::None, HaarProgressObserver \*const observer=nullptr)
 

*Fill a map of duplicates images found over a list of images to scan.*
- bool **fulfillsRestrictions** (qlonglong imageid, int albumId, qlonglong originalImageid, int originalAlbumId, const QList< int > &targetAlbums, DuplicatesSearchRestrictions searchResultRestriction)
 

*Checks whether the image with the given imageid fulfills all restrictions given in targetAlbums and in respect to searchResultRestriction.*
- void **getBestAndWorstPossibleScore** (Haar::SignatureData \*const querySig, SketchType type, double \*const lowestAndBestScore, double \*const highestAndWorstScore)
 

*For a given signature, find out the highest and lowest possible score that any other signature could reach, compared to the given signature.*
- bool **indexImage** (const QString &filename)
 

*Adds an image to the index in the database.*
- bool **indexImage** (const QString &filename, const DImg &image)
- bool **indexImage** (const QString &filename, const QImage &image)
- bool **indexImage** (qlonglong imageid, const DImg &image)
- bool **indexImage** (qlonglong imageid, const QImage &image)
- QImage **loadQImage** (const QString &filename)
 

*This method loads a QImage from the given filename.*
- bool **retrieveSignatureFromDB** (qlonglong imageid, Haar::SignatureData &sig)
 

*Retrieve the Haar signature from database using image id.*
- void **setAlbumRootsToSearch** (const QList< int > &albumRootIds)
 

*Give a list of albumRoots to which the search shall be limited.*
- void **setAlbumRootsToSearch** (const QSet< int > &albumRootIds)
- QString **signatureAsText** (const QImage &image)
 

*Calculates the Haar signature, bring it in a form as stored in the DB, and encode it to Ascii data.*

## Static Public Member Functions

- static QSet< qlonglong > **imagesFromAlbumsAndTags** (const QList< int > &albums2Scan, const QList< int > &tags2Scan, AlbumTagRelation relation)
 

*Collects all images from the given album and tag ids according to their relation.*
- static int **preferredSize** ()
- static void **rebuildDuplicatesAlbums** (const DuplicatesResultsMap &results, bool isAlbumUpdate)
 

*This method rebuilds the given SAlbums using the given results.*



## 9.673.1 Member Enumeration Documentation

### 9.673.1.1 RefImageSelMethod

```
enum class Digikam::HaarIface::RefImageSelMethod : unsigned int [strong]
```

When adding method here, update also [Haarface::findDuplicates\(\)](#)

Enumerator

OlderOrLarger	Original.
PreferFolder	Prefer select folder to be the reference.
ExcludeFolder	Prefer image not in the selected folder.
NewerCreationDate	Prefer newer creation date image.
NewerModificationDate	Prefer newer modification date image.

## 9.673.2 Member Function Documentation

### 9.673.2.1 bestMatchesForImageWithThreshold() [1/2]

```
QPair< double, QMap< qlonglong, double > > Digikam::HaarIface::bestMatchesForImageWith↵
Threshold (
    const QString & imagePath,
    double requiredPercentage,
    double maximumPercentage,
    const QList< int > & targetAlbums,
    DuplicatesSearchRestrictions searchResultRestriction = DuplicatesSearchRestrictions↵
::None,
    SketchType type = ScannedSketch )
```

All matches with a similarity in a given threshold interval are returned. The threshold is in the range required↵Percentage..maximumPercentage.

### 9.673.2.2 bestMatchesForImageWithThreshold() [2/2]

```
QPair< double, QMap< qlonglong, double > > Digikam::HaarIface::bestMatchesForImageWith↵
Threshold (
    qlonglong imageid,
    double requiredPercentage,
    double maximumPercentage,
    const QList< int > & targetAlbums,
    DuplicatesSearchRestrictions searchResultRestriction = DuplicatesSearchRestrictions↵
::None,
    SketchType type = ScannedSketch )
```

All matches with a similarity in a given threshold interval are returned. The threshold is in the range required↵Percentage..maximumPercentage.

### 9.673.2.3 findDuplicates()

```

HaarIface::DuplicatesResultsMap Digikam::HaarIface::findDuplicates (
    const QSet< qlonglong > & images2Scan,
    const QSet< qlonglong >::const_iterator & rangeBegin,
    const QSet< qlonglong >::const_iterator & rangeEnd,
    RefImageSelMethod refImageSelectionMethod,
    const QSet< qlonglong > & refs,
    double requiredPercentage,
    double maximumPercentage,
    DuplicatesSearchRestrictions searchResultRestriction = DuplicatesSearchRestrictions←
    ::None,
    HaarProgressObserver *const observer = nullptr )

```

For each map item, the result values is list of candidate images which are duplicates of the key image. All images are referenced by id from database. The threshold is in the range 0..1, with 1 meaning identical signature.

### 9.673.2.4 loadQImage()

```

QImage Digikam::HaarIface::loadQImage (
    const QString & filename )

```

#### Parameters

<i>filename</i>	the name of the file (path)
-----------------	-----------------------------

#### Returns

A QImage, non-null on success.

### 9.673.2.5 rebuildDuplicatesAlbums()

```

void Digikam::HaarIface::rebuildDuplicatesAlbums (
    const DuplicatesResultsMap & results,
    bool isAlbumUpdate ) [static]

```

#### Parameters

<i>results</i>	Map of duplicates images found over a list of images.
<i>isAlbumUpdate</i>	if true update the SAlbums in the database.

### 9.673.2.6 retrieveSignatureFromDB()

```

bool Digikam::HaarIface::retrieveSignatureFromDB (
    qlonglong imageid,
    Haar::SignatureData & sig )

```

Return true if item signature exist else false.

### 9.673.2.7 setAlbumRootsToSearch()

```
void Digikam::HaarIface::setAlbumRootsToSearch (
    const QList< int > & albumRootIds )
```

Calling with an empty list will disable filtering.

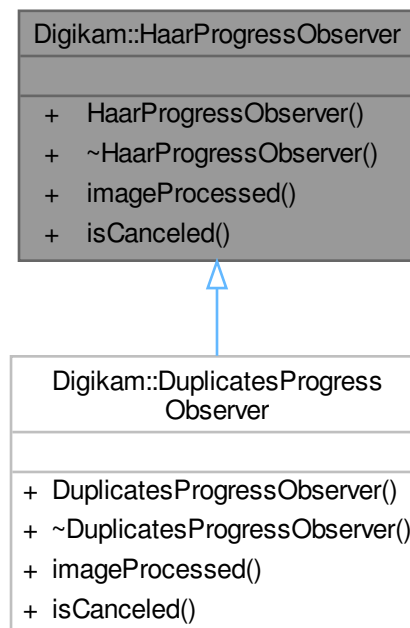
### 9.673.2.8 signatureAsText()

```
QString Digikam::HaarIface::signatureAsText (
    const QImage & image )
```

Can be used for bestMatchesForSignature.

## 9.674 Digikam::HaarProgressObserver Class Reference

Inheritance diagram for Digikam::HaarProgressObserver:

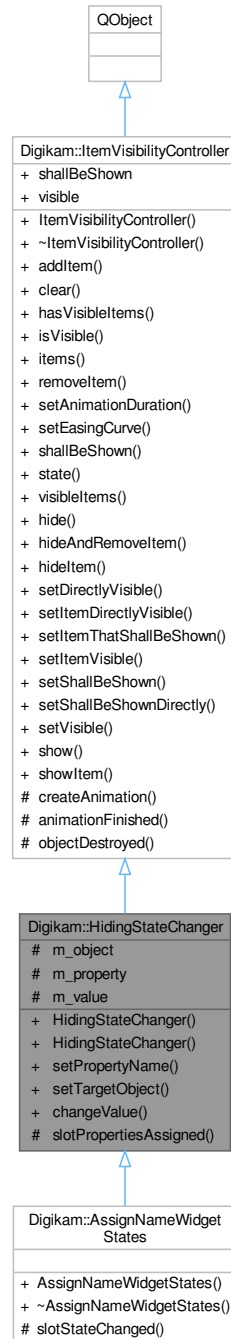


### Public Member Functions

- virtual void **imageProcessed** (const [ItemInfo](#) &, const QImage &, int)=0
- virtual bool **isCanceled** ()

## 9.675 Digikam::HidingStateChanger Class Reference

Inheritance diagram for Digikam::HidingStateChanger:



### Public Slots

- void **changeValue** (const QVariant &value)

## Public Slots inherited from [Digikam::ItemVisibilityController](#)

- void **hide** ()
- void **hideAndRemoveItem** (QObject \*item)
  - Hide the item, and then remove it.*
- void **hideItem** (QObject \*item)
- void **setDirectlyVisible** (bool visible)
- void **setItemDirectlyVisible** (QObject \*item, bool visible)
- void **setItemThatShallBeShown** (QObject \*item)
  - Sets a single item to be shown.*
- void **setItemVisible** (QObject \*item, bool visible)
- void **setShallBeShown** (bool shallBeShown)
  - Adjusts the first condition - the items are shown if shallBeShown is true and isVisible is true.*
- void **setShallBeShownDirectly** (bool shallBeShown)
- void **setVisible** (bool visible)
- void **show** ()
  - Adjusts the main condition.*
- void **showItem** (QObject \*item)
  - Shows or hides a single item.*

## Signals

- void **finished** ()
  - Emitted when the items were hidden, the target object's property changed, and the items shown again.*
- void **stateChanged** ()
  - Emitted when the items were hidden and the target object's property changed.*

## Signals inherited from [Digikam::ItemVisibilityController](#)

- void **hiddenAndRemoved** (QObject \*item)
  - Emitted when hideAndRemoveItem has finished.*
- void **propertiesAssigned** (bool visible)
  - Emitted when the (main) transition has finished.*
- void **propertiesAssigned** (QObject \*item, bool visible)
  - Emitted when a transition for a single item finished (see setItemVisible())*

## Public Member Functions

- [HidingStateChanger](#) (QObject \*const parent=nullptr)
  - This class provides a state change while fading in and out: When changeValue is called, first the items are hidden, when this is finished, the property is assigned to the object.*
- **HidingStateChanger** (QObject \*const target, const QByteArray &property, QObject \*const parent=nullptr)
  - Convenience constructor: Sets target and property name.*
- void **setPropertyName** (const QByteArray &propertyName)
- void **setTargetObject** (QObject \*const object)

## Public Member Functions inherited from [Digikam::ItemVisibilityController](#)

- **ItemVisibilityController** (QObject \*const parent=nullptr)
- void **addItem** (QObject \*const object)
  - Add and remove objects.*
- void **clear** ()
  - Remove all animations.*
- bool **hasVisibleItems** ([IncludeFadingOutMode](#) mode=[IncludeFadingOut](#)) const
  - This returns the "result" of isVisible and shallBeShown: Something is indeed visible on the scene.*
- bool **isVisible** () const
- QList< QObject \* > **items** () const
  - Returns all items under control.*
- void **removeItem** (QObject \*const object)
- void **setAnimationDuration** (int msec)
- void **setEasingCurve** (const QEasingCurve &easing)
  - Allows to change the default parameters of all animations.*
- bool **shallBeShown** () const
- [State](#) **state** () const
- QList< QObject \* > **visibleItems** ([IncludeFadingOutMode](#) mode=[IncludeFadingOut](#)) const
  - Returns all currently visible items.*

## Protected Slots

- void **slotPropertiesAssigned** (bool)

## Protected Slots inherited from [Digikam::ItemVisibilityController](#)

- void **animationFinished** ()
- void **objectDestroyed** (QObject \*)

## Protected Attributes

- QObject \* **m\_object** = nullptr
- QByteArray **m\_property**
- QVariant **m\_value**

## Additional Inherited Members

## Public Types inherited from [Digikam::ItemVisibilityController](#)

- enum [IncludeFadingOutMode](#) { [IncludeFadingOut](#) , [ExcludeFadingOut](#) }
- enum [State](#) { [Hidden](#) , [FadingIn](#) , [Visible](#) , [FadingOut](#) }

*This class handles complex visibility situations for items.*

## Protected Member Functions inherited from [Digikam::ItemVisibilityController](#)

- virtual QPropertyAnimation \* **createAnimation** (QObject \*item)
  - Creates the animation for showing and hiding the given item.*

## Properties inherited from [Digikam::ItemVisibilityController](#)

- bool **shallBeShown**
- bool **visible**

### 9.675.1 Constructor & Destructor Documentation

#### 9.675.1.1 HidingStateChanger()

```
Digikam::HidingStateChanger::HidingStateChanger (
    QObject *const parent = nullptr ) [explicit]
```

Afterwards, the items are shown again. Note that the targetObject is not necessarily a controlled item!

## 9.676 Digikam::Highlighter Class Reference

Inheritance diagram for Digikam::Highlighter:



### Public Member Functions

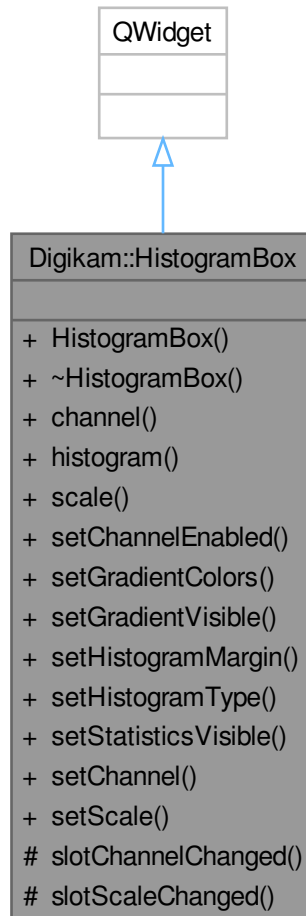
- **Highlighter** (`QTextDocument *const document`, `Parser *const _parser`)

### Protected Member Functions

- void **highlightBlock** (`const QString &text`) override

## 9.677 Digikam::HistogramBox Class Reference

Inheritance diagram for Digikam::HistogramBox:



### Public Slots

- void **setChannel** (ChannelType channel)
- void **setScale** ([HistogramScale](#) scale)

### Signals

- void **signalChannelChanged** (ChannelType channel)
- void **signalScaleChanged** ([HistogramScale](#) scale)



**Public Member Functions**

- **HistogramBox** (QWidget \*const parent=nullptr, HistogramBoxType type=Digikam::LRGB, bool select←→ Mode=false)
- ChannelType **channel** () const
- [HistogramWidget](#) \* **histogram** () const
- [HistogramScale](#) **scale** () const
- void **setChannelEnabled** (bool enabled)
- void **setGradientColors** (const QColor &from, const QColor &to)
- void **setGradientVisible** (bool visible)
- void **setHistogramMargin** (int)
- void **setHistogramType** (HistogramBoxType type)
- void **setStatisticsVisible** (bool b)

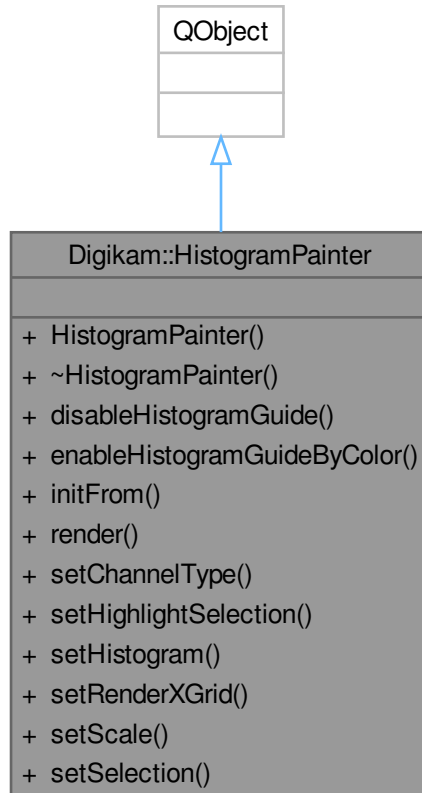
**Protected Slots**

- void **slotChannelChanged** ()
- void **slotScaleChanged** ()

## 9.678 Digikam::HistogramPainter Class Reference

A class that paints a histogram on a QPixmap.

Inheritance diagram for Digikam::HistogramPainter:



## Public Member Functions

- [HistogramPainter](#) (QObject \*const parent)  
*Constructor.*
- [~HistogramPainter](#) () override  
*Destructor.*
- void [disableHistogramGuide](#) ()  
*Disables the rendering of the color guide.*
- void [enableHistogramGuideByColor](#) (const [DColor](#) &color)  
*Starts rendering a guide that indicates where in the histogram a specified color can be found.*
- void [initFrom](#) (QWidget \*const widget)  
*Stores a widget that is used to initialize the painter used in the next call to render.*
- void [render](#) (QPixmap &paintDevice)  
*Renders the given histogram on the pixmap.*
- void [setChannelType](#) (ChannelType channelType)  
*Set the channel type to render with the next call to render.*
- void [setHighlightSelection](#) (bool highlightSelection)  
*Decide whether to highlight a specified selection in the histogram or not.*
- void [setHistogram](#) ([ImageHistogram](#) \*const histogram)  
*Set the histogram to paint with the next call to render.*
- void [setRenderXGrid](#) (bool renderXGrid)  
*Decide whether to render a separation of the histogram in x direction.*
- void [setScale](#) ([HistogramScale](#) scale)  
*Set the scale to paint the histogram with.*
- void [setSelection](#) (double selectionMin, double selectionMax)  
*Sets the selection to highlight.*

### 9.678.1 Detailed Description

Warning: before first usage of the render method, you must call [initFrom\(\)](#) to initialize the painter.

### 9.678.2 Constructor & Destructor Documentation

#### 9.678.2.1 HistogramPainter()

```
Digikam::HistogramPainter::HistogramPainter (
    QObject *const parent ) [explicit]
```

#### Parameters

<i>parent</i>	the parent for Qt's destruction mechanism
---------------	---

### 9.678.3 Member Function Documentation

#### 9.678.3.1 enableHistogramGuideByColor()

```
void Digikam::HistogramPainter::enableHistogramGuideByColor (
    const DColor & color )
```

**Parameters**

<i>color</i>	the color to highlight in the histogram
--------------	---

**9.678.3.2 initFrom()**

```
void Digikam::HistogramPainter::initFrom (
    QWidget *const widget )
```

Therefore you must ensure that this widget will not be destroyed as long as you want to use the render method without a new call to this method!!!

**Parameters**

<i>widget</i>	the widget to initialize painting from
---------------	--

**9.678.3.3 render()**

```
void Digikam::HistogramPainter::render (
    QPixmap & paintDevice )
```

The whole size of the pixmap is used for the histogram.

You must ensure that once before using this method a call to `initFrom` was made and the widget given in that call is still present.

**Parameters**

<i>paintDevice</i>	pixmap to paint the histogram on
--------------------	----------------------------------

**9.678.3.4 setChannelType()**

```
void Digikam::HistogramPainter::setChannelType (
    ChannelType channelType )
```

**Parameters**

<i>channelType</i>	channel type to render
--------------------	------------------------

**9.678.3.5 setHighlightSelection()**

```
void Digikam::HistogramPainter::setHighlightSelection (
    bool highlightSelection )
```

The selection must be defined with `setHighlightSelection`.

## Parameters

<i>highlightSelection</i>	if true, a selection will be highlighted
---------------------------	--

**9.678.3.6 setHistogram()**

```
void Digikam::HistogramPainter::setHistogram (
    ImageHistogram *const histogram )
```

## Parameters

<i>histogram</i>	an existing pointer to a histogram to paint on next call to render. The histogram must still exist at that call.
------------------	--

**9.678.3.7 setRenderXGrid()**

```
void Digikam::HistogramPainter::setRenderXGrid (
    bool renderXGrid )
```

## Parameters

<i>renderXGrid</i>	if true, a separation at some significant value in x direction is rendered.
--------------------	---

**9.678.3.8 setScale()**

```
void Digikam::HistogramPainter::setScale (
    HistogramScale scale )
```

## Parameters

<i>scale</i>	scal to paint histogram with
--------------	------------------------------

**9.678.3.9 setSelection()**

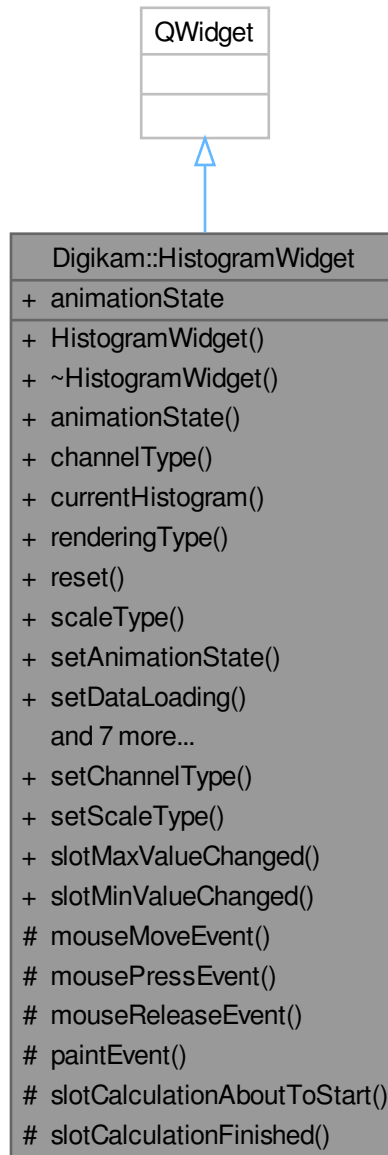
```
void Digikam::HistogramPainter::setSelection (
    double selectionMin,
    double selectionMax )
```

## Parameters

<i>selectionMin</i>	0 <= value <= 1, percent of the histogram width to start highlighting as percent. Ensure that this value is smaller then selectionMax.
<i>selectionMax</i>	0 <= value <= 1, percent of the histogram width to end highlighting as percent. Ensure that this value is greater then selectionMin.

## 9.679 Digikam::HistogramWidget Class Reference

Inheritance diagram for Digikam::HistogramWidget:



### Public Slots

- void **setChannelType** (ChannelType channel)
- void **setScaleType** (HistogramScale scale)
- void **slotMaxValueChanged** (int max)
- void **slotMinValueChanged** (int min)

## Signals

- void **signalHistogramComputationDone** (bool)
- void **signalHistogramComputationFailed** ()
- void **signalIntervalChanged** (int min, int max)
- void **signalMaximumValueChanged** (int)

## Public Member Functions

- [HistogramWidget](#) (int w, int h, QWidget \*const parent=nullptr, bool selectMode=true, bool showProgress=true, bool statisticsVisible=false)  
*Standard constructor.*
- int **animationState** () const
- ChannelType **channelType** () const
- [ImageHistogram](#) \* **currentHistogram** () const  
*Currently rendered histogram, depending on current rendering type.*
- [HistogramRenderingType](#) **renderingType** () const
- void **reset** ()
- [HistogramScale](#) **scaleType** () const
- void **setAnimationState** (int animationState)
- void **setDataLoading** ()
- void **setHistogramGuideByColor** (const [DColor](#) &color)
- void **setLoadingFailed** ()
- void **setRenderingType** ([HistogramRenderingType](#) type)
- void **setStatisticsVisible** (bool b)
- void **stopHistogramComputation** ()  
*Stop current histogram computations.*
- void **updateData** (const [DImg](#) &img, const [DImg](#) &sel=[DImg](#)(), bool showProgress=true)  
*Update full image histogram data methods.*
- void **updateSelectionData** (const [DImg](#) &sel, bool showProgress=true)  
*Update image selection histogram data methods.*

## Protected Slots

- void **slotCalculationAboutToStart** ()
- void **slotCalculationFinished** (bool success)

## Protected Member Functions

- void **mouseMoveEvent** (QMouseEvent \*) override
- void **mousePressEvent** (QMouseEvent \*) override
- void **mouseReleaseEvent** (QMouseEvent \*) override
- void **paintEvent** (QPaintEvent \*) override

## Properties

- int **animationState**

## 9.679.1 Constructor & Destructor Documentation

### 9.679.1.1 HistogramWidget()

```
Digikam::HistogramWidget::HistogramWidget (
    int w,
    int h,
    QWidget *const parent = nullptr,
    bool selectMode = true,
    bool showProgress = true,
    bool statisticsVisible = false )
```

Needed to use [updateData\(\)](#) methods after to create valid instance.

## 9.680 Digikam::HistoryEdgeProperties Class Reference

Every edge has one associated object of this class.

### Public Member Functions

- [HistoryEdgeProperties](#) & **operator+=** (const [FilterAction](#) &action)

### Public Attributes

- `QList< FilterAction > actions`

### 9.680.1 Detailed Description

For two vertices v1, v2 with and edge e, v1 -> v2, describes the actions necessary to create v2 from v1: v1 -> actions[0] -> ... -> actions[n] = v2.

## 9.681 Digikam::HistoryImageld Class Reference

### Public Types

- enum [Type](#) {
  - InvalidType** = 0 , **Original** = 1 << 0 , **Intermediate** = 1 << 1 , **Source** = 1 << 2 ,
  - Current** = 1 << 3 }
- typedef `QFlags< Type > Types`

*Note: In this class, the [Type](#) is used as a simple enum, but it is also prepared for usage as flags.*

## Public Member Functions

- **HistoryImageId** ()=default  
*Creates an invalid [HistoryImageId](#).*
- **HistoryImageId** (const QString &uuid, [Type](#) type=[Current](#))  
*Creates an id with the given UUID and type.*
- QDateTime **creationDate** () const
- QString **fileName** () const  
*If a file on disk is referenced: Returns the file name (without folder)*
- QString **filePath** () const  
*If a file on disk is referenced: Returns the full file path (folder + filename)*
- qlonglong **fileSize** () const
- bool **hasCreationDate** () const
- bool **hasFileName** () const
- bool **hasFileOnDisk** () const
- bool **hasUniqueHashIdentifier** () const
- bool **hasUuid** () const
- bool **isCurrentFile** () const
- bool **isIntermediateFile** () const
- bool **isOriginalFile** () const
- bool **isSourceFile** () const
- bool **isValid** () const  
*A valid id needs at least a valid type and a UUID or a filename.*
- bool **operator==** (const [HistoryImageId](#) &other) const
- QString **originalUuid** () const
- QString **path** () const  
*If a file on disk is referenced: Returns the path, without filename, with a trailing slash.*
- void **setCreationDate** (const QDateTime &creationDate)
- void **setFileName** (const QString &fileName)
- void **setPath** (const QString &path)
- void **setPathOnDisk** (const QString &filePath)
- void **setType** ([HistoryImageId::Type](#) type)
- void **setUniqueHash** (const QString &uniqueHash, qlonglong fileSize)
- void **setUuid** (const QString &uuid)
- [Type](#) **type** () const
- QString **uniqueHash** () const
- QString **uuid** () const

## Public Attributes

- QDateTime **m\_creationDate**  
*The creationDate of the original image.*
- QString **m\_fileName**  
*The filename of the referred file.*
- QString **m\_filePath**  
*The path of the referred file (NOTE: without file name!, including trailing slash)*
- qlonglong **m\_fileSize** = 0  
*The file size of the referred file.*
- QString **m\_originalUUID**  
*A unique identifier designating the original image from which the referred image was created.*
- [Type](#) **m\_type** = [InvalidType](#)  
*Type of this History Image Id.*
- QString **m\_uniqueHash**  
*The uniqueHash of the referred file.*
- QString **m\_uuid**  
*A unique identifier for the referred file.*



## 9.681.1 Member Enumeration Documentation

### 9.681.1.1 Type

enum `Digikam::HistoryImageId::Type`

#### Enumerator

Original	The original file (typically created by a camera)
Intermediate	A file created during the editing the history, between the original file and the current file.
Source	When a file is created from multiple files, there can be no direct original (panorama) but multiple sources, or one direct original and some other, additional source files. To record source files outside of the direct history, this type is used.
Current	The "current" file. This is a special entry: It refers to the file from which this history was read. It need not be written to the file, because it describes the file itself. There is typically exactly one current entry if the history is associated with an image; there can be no current entry.

## 9.681.2 Member Data Documentation

### 9.681.2.1 m\_originalUUID

`QString Digikam::HistoryImageId::m_originalUUID`

Typically, this is a RAW or JPEG created by the camera in the moment of taking the photograph.

### 9.681.2.2 m\_uuid

`QString Digikam::HistoryImageId::m_uuid`

This id shall be changed each time the image is edited.

## 9.682 Digikam::HistoryVertexProperties Class Reference

Every vertex has one associated object of this class.

### Public Member Functions

- bool `alwaysMarkedAs` (`HistoryImageId::Type`) const
- `ItemInfo firstItemInfo` () const
- bool `markedAs` (`HistoryImageId::Type`) const
- `HistoryVertexProperties` & `operator+=` (const `HistoryImageId` &info)
- `HistoryVertexProperties` & `operator+=` (const `ItemInfo` &info)
- `HistoryVertexProperties` & `operator+=` (const `QString` &uuid)
- bool `operator==` (const `HistoryImageId` &info) const
- bool `operator==` (const `ItemInfo` &info) const
- bool `operator==` (const `QString` &uuid) const
- bool `operator==` (qulonglong id) const

**Public Attributes**

- QList< [ItemInfo](#) > **infos**
- QList< [HistoryImageId](#) > **referredImages**
- QString **uuid**

**9.682.1 Detailed Description**

All entries in a vertex refer to *identical* images. There can be multiple referred images in a history entry. Each single [HistoryImageId](#) can resolve into none, one, or multiple [ItemInfos](#). So there is no mapping between the two fields here.

If an image is created from multiple source images (panorama etc.), there will be one vertex per source image!

**9.683 Digikam::HotPixelContainer Class Reference****Public Types**

- enum **Direction** { **TWODIM\_DIRECTION** = 0 , **VERTICAL\_DIRECTION** = 1 , **HORIZONTAL\_DIRECTION** = 2 }
- enum **InterpolationMethod** { **AVERAGE\_INTERPOLATION** = 0 , **LINEAR\_INTERPOLATION** = 1 , **QUADRATIC\_INTERPOLATION** = 2 , **CUBIC\_INTERPOLATION** = 3 }

**Public Member Functions**

- bool **isDefault** () const
- bool **operator==** (const [HotPixelContainer](#) &other) const
- void **writeToFilterAction** ([FilterAction](#) &action, const QString &prefix=QString()) const

**Static Public Member Functions**

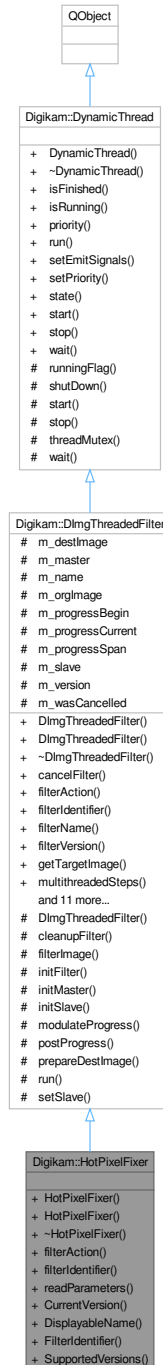
- static [HotPixelContainer](#) **fromFilterAction** (const [FilterAction](#) &action, const QString &prefix=QString())

**Public Attributes**

- QUrl **blackFrameUrl**
- InterpolationMethod **filterMethod**
- QList< [HotPixelProps](#) > **hotPixelsList**

## 9.684 Digikam::HotPixelFixer Class Reference

Inheritance diagram for Digikam::HotPixelFixer:



### Public Member Functions

- **HotPixelFixer** (`Dlmg *const orgImage`, `QObject *const parent`, `const HotPixelContainer &settings`)
- **HotPixelFixer** (`QObject *const parent=nullptr`)

- [Digikam::FilterAction filterAction \(\)](#) override  
*Returns the action description corresponding to currently set options.*
- [QString filterIdentifier \(\)](#) const override  
*Return the identifier for this filter in the image history.*
- void [readParameters \(const FilterAction &action\)](#) override

## Public Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter \(DImg \\*const orgImage, QObject \\*const parent, const QString &name=QString\(\)\)](#)  
*Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter \(QObject \\*const parent=nullptr, const QString &name=QString\(\)\)](#)  
*Constructs a filter without argument.*
- virtual void [cancelFilter \(\)](#)  
*Cancel the threaded computation.*
- const [QString &filterName \(\)](#)
- int [filterVersion \(\)](#) const
- [DImg getTargetImage \(\)](#)
- [QList< int > multithreadedSteps \(int stop, int start=0\)](#) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool [parametersSuccessfullyRead \(\)](#) const  
*Optional: error handling for readParameters.*
- virtual [QString readParametersError \(const FilterAction &actionThatFailed\)](#) const
- void [setFilterName \(const QString &name\)](#)
- void [setFilterVersion \(int version\)](#)  
*Replaying a filter action: Set the filter version.*
- void [setOriginalImage \(const DImg &orgImage\)](#)
- void [setupAndStartDirectly \(const DImg &orgImage, DImgThreadedFilter \\*const master, int progress←Begin=0, int progressEnd=100\)](#)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void [setupFilter \(const DImg &orgImage\)](#)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void [startFilter \(\)](#)  
*Start the threaded computation.*
- virtual void [startFilterDirectly \(\)](#)  
*Start computation of this filter, directly in this thread.*
- virtual [QList< int > supportedVersions \(\)](#) const

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread \(QObject \\*const parent=nullptr\)](#)  
*This class extends QRunnable, so you have to reimplement virtual void [run\(\)](#).*
- [~DynamicThread \(\)](#) override  
*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool [isFinished \(\)](#) const
- bool [isRunning \(\)](#) const
- [QThread::Priority priority \(\)](#) const
- void [setEmitSignals \(bool emitThem\)](#)
- void [setPriority \(QThread::Priority priority\)](#)  
*Sets the priority for this dynamic thread.*
- State [state \(\)](#) const

### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from [Digikam::DImgThreadedFilter](#)

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.684.1 Member Function Documentation

### 9.684.1.1 filterAction()

`Digikam::FilterAction` `Digikam::HotPixelFixer::filterAction ( )` [override], [virtual]

Implements `Digikam::DImgThreadedFilter`.

### 9.684.1.2 filterIdentifier()

`QString` `Digikam::HotPixelFixer::filterIdentifier ( )` const [inline], [override], [virtual]

Implements `Digikam::DImgThreadedFilter`.

### 9.684.1.3 readParameters()

```
void Digikam::HotPixelFixer::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements `Digikam::DImgThreadedFilter`.

## 9.685 Digikam::HotPixelProps Class Reference

### Public Member Functions

- bool **fromString** (const `QString` &str)
- int **height** () const
- bool **operator==** (const `HotPixelProps` &p) const
- `QString` **toString** () const
- int **width** () const
- int **x** () const
- int **y** () const

### Static Public Member Functions

- static `QList`< `HotPixelProps` > **fromStringList** (const `QStringList` &hplst)
- static `QStringList` **toStringList** (const `QList`< `HotPixelProps` > &lst)

### Public Attributes

- int **luminosity**
- `QRect` **rect**

## 9.685.1 Member Function Documentation

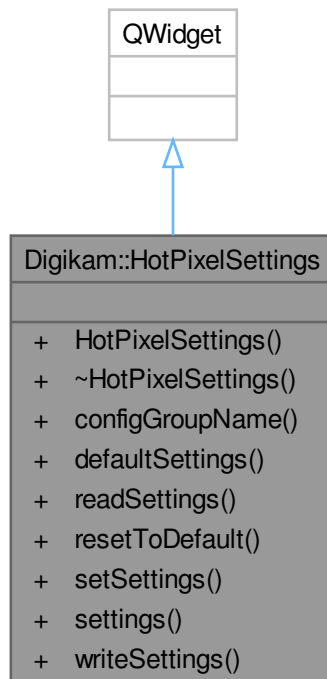
### 9.685.1.1 operator==( )

```
bool Digikam::HotPixelProps::operator==(
    const HotPixelProps & p ) const
```

NOTE:we can say they're same hotpixel spot if they touch (next to) each other horizontally or vertically, not diagonal corners

## 9.686 Digikam::HotPixelSettings Class Reference

Inheritance diagram for Digikam::HotPixelSettings:



### Signals

- void **signalHotPixels** (const `QPolygon` &pointList)
- void **signalSettingsChanged** ( )



### Public Member Functions

- **HotPixelSettings** (QWidget \*const parent)
- QString **configGroupName** () const
- [HotPixelContainer](#) **defaultSettings** () const
- void **readSettings** (const KConfigGroup &group)
- void **resetToDefault** ()
- void **setSettings** (const [HotPixelContainer](#) &settings)
- [HotPixelContainer](#) **settings** () const
- void **writeSettings** (KConfigGroup &group)

## 9.687 Digikam::HotPixelsWeights Class Reference

### Public Member Functions

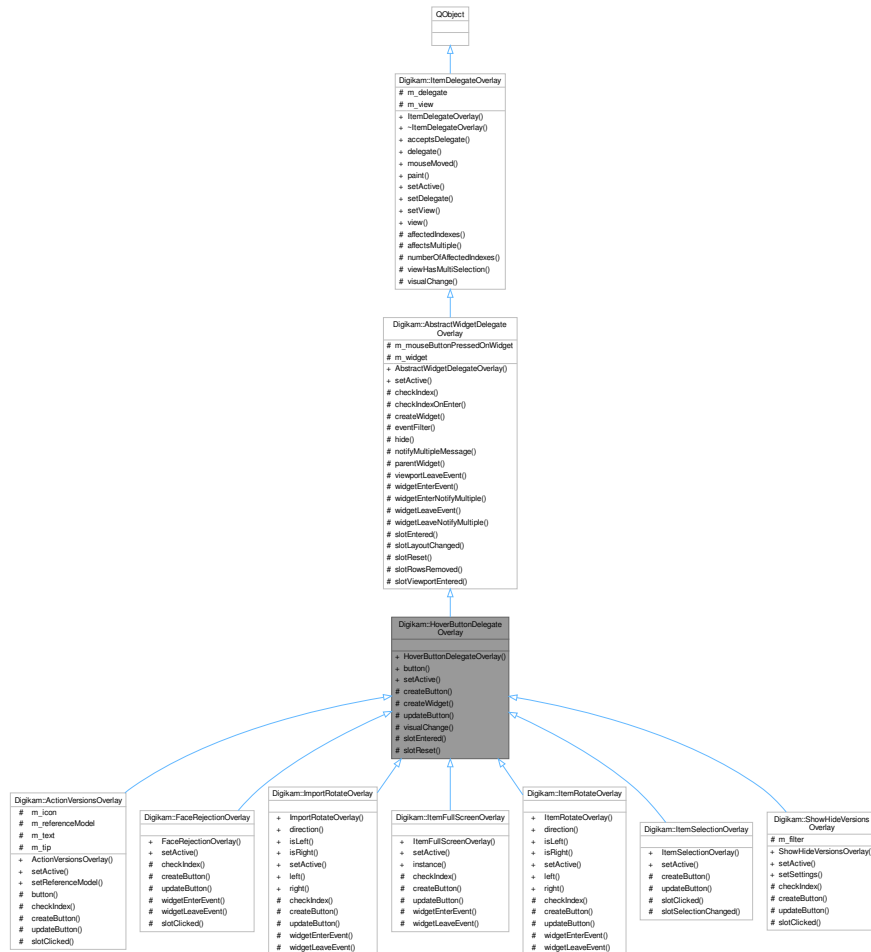
- **HotPixelsWeights** (const [HotPixelsWeights](#) &w)
- void **calculateHotPixelsWeights** ()
- unsigned int **height** () const
- [HotPixelsWeights](#) & **operator=** (const [HotPixelsWeights](#) &w)
- bool **operator==** (const [HotPixelsWeights](#) &ws) const
- double \*\* **operator[]** (int n) const
- unsigned int **polynomeOrder** () const
- const QList< QPoint > **positions** () const
- void **setHeight** (int h)
- void **setPolynomeOrder** (int order)
- void **setTwoDim** (bool td)
- void **setWidth** (int w)
- bool **twoDim** () const
- unsigned int **width** () const

### Protected Member Functions

- int **coefficientNumber** () const
- double \*\*\* **weightMatrices** () const

## 9.688 Digikam::HoverButtonDelegateOverlay Class Reference

Inheritance diagram for Digikam::HoverButtonDelegateOverlay:



### Public Member Functions

- **HoverButtonDelegateOverlay** (QObject \*const parent)
- **ItemViewHoverButton** \* **button** () const
- void **setActive** (bool active) override  
Will call *createButton()*.

### Public Member Functions inherited from Digikam::AbstractWidgetDelegateOverlay

- **AbstractWidgetDelegateOverlay** (QObject \*const parent)  
This class provides functionality for using a widget in an overlay.

## Public Member Functions inherited from Digikam::ItemDelegateOverlay

- **ItemDelegateOverlay** (QObject \*const parent=nullptr)
- virtual bool **acceptsDelegate** (QAbstractItemDelegate \*) const
- QAbstractItemDelegate \* **delegate** () const
- virtual void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)  
*Only these two methods are implemented as virtual methods.*
- virtual void **paint** (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index)
- void **setDelegate** (QAbstractItemDelegate \*delegate)
- void **setView** (QAbstractItemView \*view)
- QAbstractItemView \* **view** () const

## Protected Slots

- void **slotEntered** (const QModelIndex &index) override
- void **slotReset** () override

## Protected Slots inherited from Digikam::AbstractWidgetDelegateOverlay

- virtual void **slotEntered** (const QModelIndex &index)  
*Default implementation shows the widget iff the index is valid and checkIndex returns true.*
- virtual void **slotLayoutChanged** ()
- virtual void **slotReset** ()  
*Default implementations of these three slots call `hide()`*
- virtual void **slotRowsRemoved** (const QModelIndex &parent, int start, int end)
- virtual void **slotViewportEntered** ()

## Protected Slots inherited from Digikam::ItemDelegateOverlay

### Protected Member Functions

- virtual **ItemViewHoverButton** \* **createButton** ()=0  
*Create your widget here.*
- QWidget \* **createWidget** () override  
*Create your widget here.*
- virtual void **updateButton** (const QModelIndex &index)=0  
*Called when a new index is entered.*
- void **visualChange** () override  
*Called when any change from the delegate occurs - when the overlay is installed, when size hints, styles or fonts change.*

## Protected Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- virtual bool [checkIndex](#) (const QModelIndex &index) const  
*Return true here if you want to show the overlay for the given index.*
- bool [checkIndexOnEnter](#) (const QModelIndex &index) const  
*Utility method called from slotEntered.*
- bool [eventFilter](#) (QObject \*obj, QEvent \*event) override
- virtual void [hide](#) ()  
*Called when the widget shall be hidden (mouse cursor left index, viewport, uninstalled etc.).*
- virtual QString [notifyMultipleMessage](#) (const QModelIndex &, int number)
- QWidget \* [parentWidget](#) () const  
*Returns the widget to be used as parent for your widget created in [createWidget\(\)](#)*
- virtual void [viewportLeaveEvent](#) (QObject \*obj, QEvent \*event)  
*Called when a QEvent::Leave of the viewport is received.*
- virtual void [widgetEnterEvent](#) ()  
*Called when a QEvent::Enter resp.*
- void [widgetEnterNotifyMultiple](#) (const QModelIndex &index)  
*A sample implementation for above methods.*
- virtual void [widgetLeaveEvent](#) ()
- void [widgetLeaveNotifyMultiple](#) ()

## Protected Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- QList< QModelIndex > [affectedIndexes](#) (const QModelIndex &index) const
- bool [affectsMultiple](#) (const QModelIndex &index) const  
*For the context that an overlay can affect multiple items: Assuming the currently overlaid index is given.*
- int [numberOfAffectedIndexes](#) (const QModelIndex &index) const
- bool [viewHasMultiSelection](#) () const  
*Utility method.*

## Additional Inherited Members

## Signals inherited from [Digikam::ItemDelegateOverlay](#)

- void [hideNotification](#) ()
- void [requestNotification](#) (const QModelIndex &index, const QString &message)
- void [update](#) (const QModelIndex &index)

## Protected Attributes inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- bool [m\\_mouseButtonPressedOnWidget](#) = false
- QWidget \* [m\\_widget](#) = nullptr

## Protected Attributes inherited from [Digikam::ItemDelegateOverlay](#)

- QAbstractItemDelegate \* [m\\_delegate](#) = nullptr
- QAbstractItemView \* [m\\_view](#) = nullptr

## 9.688.1 Member Function Documentation

### 9.688.1.1 createButton()

```
virtual ItemViewHoverButton * Digikam::HoverButtonDelegateOverlay::createButton ( ) [protected],  
[pure virtual]
```

Pass view() as parent.

Implemented in [Digikam::FaceRejectionOverlay](#), [Digikam::ItemFullScreenOverlay](#), [Digikam::ItemRotateOverlay](#), [Digikam::ItemSelectionOverlay](#), [Digikam::ShowHideVersionsOverlay](#), [Digikam::ActionVersionsOverlay](#), and [Digikam::ImportRotateOverlay](#).

### 9.688.1.2 createWidget()

```
QWidget * Digikam::HoverButtonDelegateOverlay::createWidget ( ) [override], [protected], [virtual]
```

When creating the object, pass [parentWidget\(\)](#) as parent widget. Ownership of the object is passed. It will be deleted in [setActive\(false\)](#).

Implements [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.688.1.3 setActive()

```
void Digikam::HoverButtonDelegateOverlay::setActive (   
    bool active ) [override], [virtual]
```

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

Reimplemented in [Digikam::ItemFullScreenOverlay](#), [Digikam::ItemRotateOverlay](#), [Digikam::ItemSelectionOverlay](#), [Digikam::ShowHideVersionsOverlay](#), and [Digikam::ImportRotateOverlay](#).

### 9.688.1.4 updateButton()

```
virtual void Digikam::HoverButtonDelegateOverlay::updateButton (   
    const QModelIndex & index ) [protected], [pure virtual]
```

Reposition your button here, adjust and store state.

Implemented in [Digikam::FaceRejectionOverlay](#), [Digikam::ItemFullScreenOverlay](#), [Digikam::ItemRotateOverlay](#), [Digikam::ItemSelectionOverlay](#), [Digikam::ShowHideVersionsOverlay](#), [Digikam::ActionVersionsOverlay](#), and [Digikam::ImportRotateOverlay](#).

### 9.688.1.5 visualChange()

```
void Digikam::HoverButtonDelegateOverlay::visualChange ( ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ItemDelegateOverlay](#).

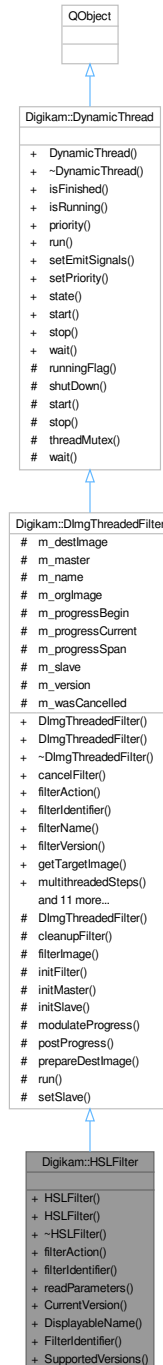
## 9.689 Digikam::HSLContainer Class Reference

### Public Attributes

- double **hue** = 0.0
- double **lightness** = 0.0
- double **saturation** = 0.0
- double **vibrance** = 0.0

## 9.690 Digikam::HSLFilter Class Reference

Inheritance diagram for Digikam::HSLFilter:



### Public Member Functions

- **HSLFilter** (`DImg *const orgImage`, `QObject *const parent=nullptr`, `const HSLContainer &settings=HSLContainer()`)
- **HSLFilter** (`QObject *const parent=nullptr`)

- [FilterAction filterAction \(\)](#) override  
*Returns the action description corresponding to currently set options.*
- [QString filterIdentifier \(\)](#) const override  
*Return the identifier for this filter in the image history.*
- void [readParameters \(const FilterAction &action\)](#) override

## Public Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter \(DImg \\*const orgImage, QObject \\*const parent, const QString &name=QString\(\)\)](#)  
*Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter \(QObject \\*const parent=nullptr, const QString &name=QString\(\)\)](#)  
*Constructs a filter without argument.*
- virtual void [cancelFilter \(\)](#)  
*Cancel the threaded computation.*
- const [QString &filterName \(\)](#)
- int [filterVersion \(\)](#) const
- [DImg getTargetImage \(\)](#)
- [QList< int > multithreadedSteps \(int stop, int start=0\)](#) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool [parametersSuccessfullyRead \(\)](#) const  
*Optional: error handling for readParameters.*
- virtual [QString readParametersError \(const FilterAction &actionThatFailed\)](#) const
- void [setFilterName \(const QString &name\)](#)
- void [setFilterVersion \(int version\)](#)  
*Replaying a filter action: Set the filter version.*
- void [setOriginalImage \(const DImg &orgImage\)](#)
- void [setupAndStartDirectly \(const DImg &orgImage, DImgThreadedFilter \\*const master, int progress←Begin=0, int progressEnd=100\)](#)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void [setupFilter \(const DImg &orgImage\)](#)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void [startFilter \(\)](#)  
*Start the threaded computation.*
- virtual void [startFilterDirectly \(\)](#)  
*Start computation of this filter, directly in this thread.*
- virtual [QList< int > supportedVersions \(\)](#) const

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread \(QObject \\*const parent=nullptr\)](#)  
*This class extends QRunnable, so you have to reimplement virtual void [run\(\)](#).*
- [~DynamicThread \(\)](#) override  
*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool [isFinished \(\)](#) const
- bool [isRunning \(\)](#) const
- [QThread::Priority priority \(\)](#) const
- void [setEmitSignals \(bool emitThem\)](#)
- void [setPriority \(QThread::Priority priority\)](#)  
*Sets the priority for this dynamic thread.*
- State [state \(\)](#) const



### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from [Digikam::DImgThreadedFilter](#)

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.690.1 Member Function Documentation

### 9.690.1.1 filterAction()

`FilterAction` Digikam::HSLFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.690.1.2 filterIdentifier()

`QString` Digikam::HSLFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

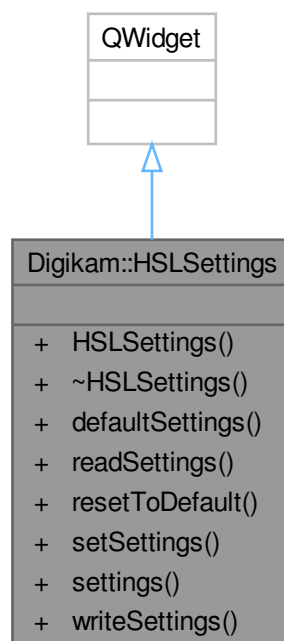
### 9.690.1.3 readParameters()

```
void Digikam::HSLFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

## 9.691 Digikam::HSLSettings Class Reference

Inheritance diagram for Digikam::HSLSettings:



## Signals

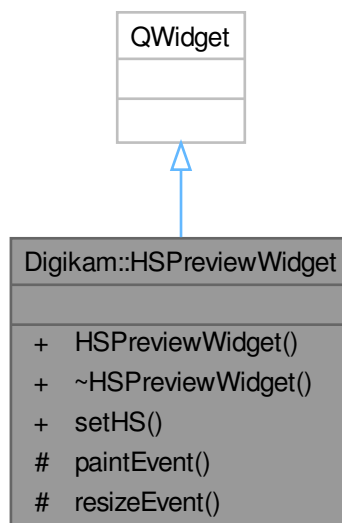
- void **signalSettingsChanged** ()

## Public Member Functions

- **HSLSettings** (QWidget \*const parent)
- **HSLContainer defaultSettings** () const
- void **readSettings** (const KConfigGroup &group)
- void **resetToDefault** ()
- void **setSettings** (const **HSLContainer** &settings)
- **HSLContainer settings** () const
- void **writeSettings** (KConfigGroup &group)

## 9.692 Digikam::HSPreviewWidget Class Reference

Inheritance diagram for Digikam::HSPreviewWidget:



## Public Member Functions

- **HSPreviewWidget** (QWidget \*const parent=nullptr)
- void **setHS** (double hue, double sat)

## Protected Member Functions

- void **paintEvent** (QPaintEvent \*) override
- void **resizeEvent** (QResizeEvent \*) override

## 9.693 Digikam::HTMLWidget Class Reference

Inheritance diagram for Digikam::HTMLWidget:



### Signals

- void **selectionHasBeenMade** (const Digikam::GeoCoordinates::Pair &coordinatesRect)
- void **signalHTMLEvents** (const QStringList &events)
- void **signalJavaScriptReady** ()
- void **signalMessageEvent** (const QString &message)

### Public Member Functions

- **HTMLWidget** (QWidget \*const parent=nullptr)
- void **centerOn** (const qreal west, const qreal north, const qreal east, const qreal south, const bool use↔ SaneZoomLevel=true)
- void **loadInitialHTML** (const QString &initialHTML)
- void **mouseModeChanged** (const GeoMouseModes mouseMode)
- void **removeSelectionRectangle** ()
- QVariant **runScript** (const QString &scriptCode, bool async=true)

*Wrapper around executeScript to catch more errors.*

- bool **runScript2Coordinates** (const QString &scriptCode, [GeoCoordinates](#) \*const coordinates)  
*Execute a script which returns coordinates and parse these.*
- void **setSelectionRectangle** (const GeoCoordinates::Pair &searchCoordinates)
- void **setSharedGeofaceObject** ([GeofaceSharedData](#) \*const sharedData)

### Protected Slots

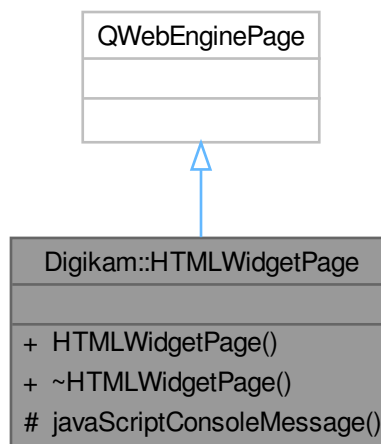
- void **progress** (int progress)
- void **slotHTMLCompleted** (bool ok)

### Protected Member Functions

- bool **eventFilter** (QObject \*, QEvent \*) override

## 9.694 Digikam::HTMLWidgetPage Class Reference

Inheritance diagram for Digikam::HTMLWidgetPage:



### Signals

- void **signalHTMLEvents** (const QStringList &events)
- void **signalMessageEvent** (const QString &message)

### Public Member Functions

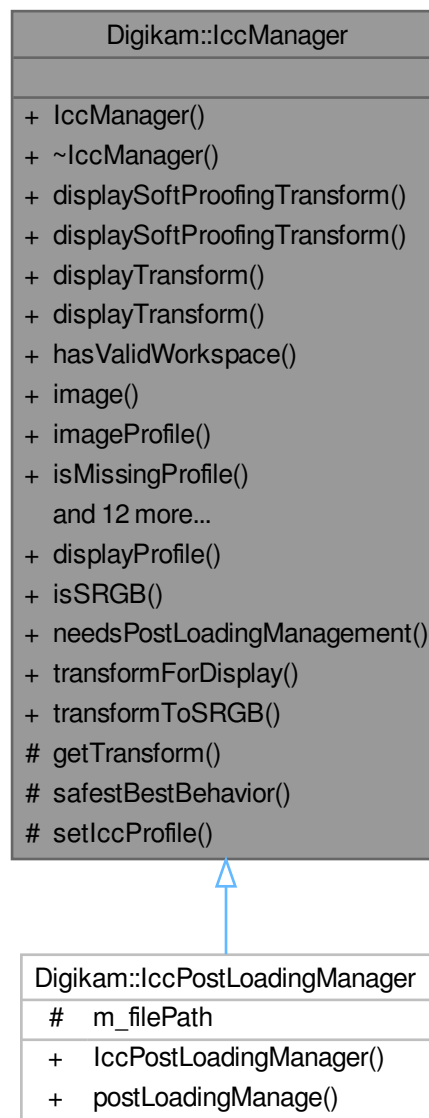
- **HTMLWidgetPage** ([HTMLWidget](#) \*const parent=nullptr)

**Protected Member Functions**

- void **javascriptConsoleMessage** (JavaScriptConsoleMessageLevel, const QString &, int, const QString &) override

**9.695 Digikam::IccManager Class Reference**

Inheritance diagram for Digikam::IccManager:

**Public Member Functions**

- `IccManager` (const `DImg` &image, const `ICCSettingsContainer` &settings=`IccSettings::instance()` ->settings())

Constructs an *IccManager* object.

- **IccTransform displaySoftProofingTransform** (const [IccProfile](#) &deviceProfile, const [IccProfile](#) &displayProfile) Profile)
- **IccTransform displaySoftProofingTransform** (const [IccProfile](#) &deviceProfile, QWidget \*const displayingWidget=nullptr) Widget=nullptr)
  - Returns a display transform, with soft-proofing enabled for the given device profile.
- **IccTransform displayTransform** (const [IccProfile](#) &displayProfile)
- **IccTransform displayTransform** (QWidget \*const displayingWidget=nullptr)
- bool **hasValidWorkspace** () const
- **DImg image** () const
- **IccProfile imageProfile** (ICCSettingsContainer::Behavior behavior, const [IccProfile](#) &specifiedProfile=[IccProfile](#)())
  - Returns the profile that will be used to interpret the image, using the given behavior.
- bool **isMissingProfile** () const
- bool **isProfileMismatch** () const
- bool **isUncalibratedColor** () const
- **DImgLoaderObserver \* observer** () const
- void **setObserver** ([DImgLoaderObserver](#) \*const observer)
- **ICCSettingsContainer settings** () const
- void **transform** (ICCSettingsContainer::Behavior behavior, const [IccProfile](#) &specifiedProfile=[IccProfile](#)())
  - Same as above, but not using default settings but the given settings.
- void **transformDefault** ()
  - Transforms the image for full editing, using default settings.
- void **transformForDisplay** ()
  - Transforms the image for display on screen.
- void **transformForDisplay** (const [IccProfile](#) &displayProfile)
- void **transformForDisplay** (QWidget \*const widget)
- void **transformForOutput** (const [IccProfile](#) &outputProfile)
  - Transforms the image for output to the specified output profile.
- void **transformToSRGB** ()
  - Transforms the image to sRGB.

### Static Public Member Functions

- static [IccProfile](#) **displayProfile** (QWidget \*const displayingWidget=nullptr)
- static bool **isSRGB** (const [DImg](#) &img)
  - Returns true if a call to [transformToSRGB\(\)](#) would have an effect.
- static bool **needsPostLoadingManagement** (const [DImg](#) &img)
  - Returns true if the given image is marked as needing user interaction for further color management decision after loading.
- static void **transformForDisplay** (QImage &qimage, const [IccProfile](#) &displayProfile1=[displayProfile](#)())
  - Transforms the given QImage from sRGB to given display profile.
- static void **transformToSRGB** (QImage &qimage, const [IccProfile](#) &inputProfile)
  - Transforms the given QImage from the given inputProfile to sRGB.

### Protected Member Functions

- void **getTransform** ([IccTransform](#) &trans, ICCSettingsContainer::Behavior behavior, const [IccProfile](#) &specifiedProfile)
- ICCSettingsContainer::Behavior **safestBestBehavior** () const
- void **setIccProfile** (const [IccProfile](#) &profile)



## 9.695.1 Constructor & Destructor Documentation

### 9.695.1.1 IccManager()

```
Digikam::IccManager::IccManager (
    const DImg & image,
    const ICCSettingsContainer & settings = IccSettings::instance\(\)->settings() )
[explicit]
```

The [DImg](#) will be edited. The `filePath` is for display only.

## 9.695.2 Member Function Documentation

### 9.695.2.1 needsPostLoadingManagement()

```
bool Digikam::IccManager::needsPostLoadingManagement (
    const DImg & img ) [static]
```

If this returns true, use [IccPostLoadingManager](#) to do this.

### 9.695.2.2 transformDefault()

```
void Digikam::IccManager::transformDefault ( )
```

If the default settings require showing a dialog, the image is marked as such but no action is taken. See [IccPostLoadingManager](#).

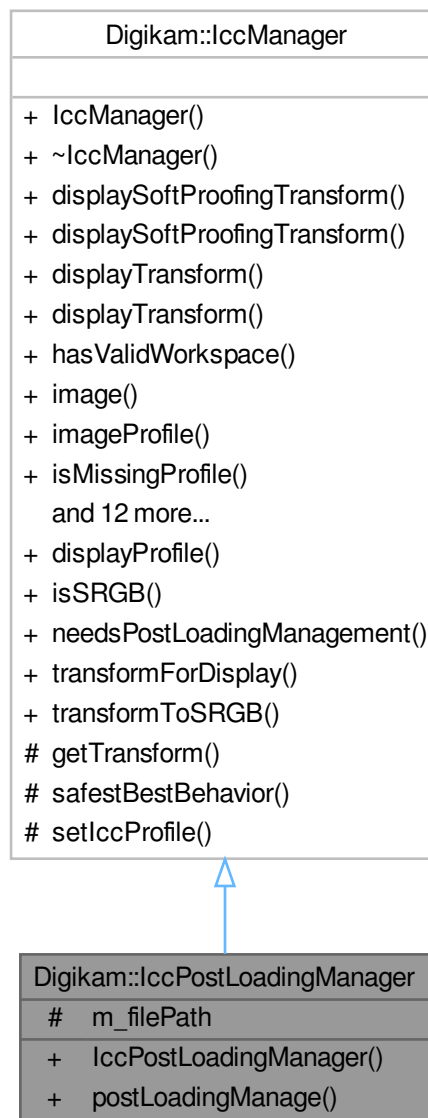
### 9.695.2.3 transformForDisplay()

```
void Digikam::IccManager::transformForDisplay ( )
```

The result is not suitable for editing or storage. You can specify the widget in which the image will be displayed, or specify the display profile yourself. You can retrieve the profile with `displayProfile()` and pass it to [transformForDisplay\(\)](#) later (in a thread), or you can get a transform from `displayTransform` and apply it yourself.

## 9.696 Digikam::IccPostLoadingManager Class Reference

Inheritance diagram for Digikam::IccPostLoadingManager:



### Public Member Functions

- [IccPostLoadingManager](#) (DImg &image, const QString &filePath=QString(), const [ICCSettingsContainer](#) &settings=[IccSettings::instance\(\)](#) ->settings())  
*Constructs an [IccPostLoadingManager](#) object.*
- [IccTransform](#) [postLoadingManage](#) (QWidget \*const parent=nullptr)  
*Carries out color management asking the user for his decision.*

## Public Member Functions inherited from Digikam::lccManager

- **lccManager** (const **DImg** &image, const **ICCSettingsContainer** &settings=**lccSettings::instance()** ->settings())  
*Constructs an **lccManager** object.*
- **lccTransform displaySoftProofingTransform** (const **lccProfile** &deviceProfile, const **lccProfile** &displayProfile)
- **lccTransform displaySoftProofingTransform** (const **lccProfile** &deviceProfile, **QWidget** \*const displayingWidget=**nullptr**)  
*Returns a display transform, with soft-proofing enabled for the given device profile.*
- **lccTransform displayTransform** (const **lccProfile** &displayProfile)
- **lccTransform displayTransform** (**QWidget** \*const displayingWidget=**nullptr**)
- bool **hasValidWorkspace** () const
- **DImg image** () const
- **lccProfile imageProfile** (**ICCSettingsContainer::Behavior** behavior, const **lccProfile** &specifiedProfile=**lccProfile()**)  
*Returns the profile that will be used to interpret the image, using the given behavior.*
- bool **isMissingProfile** () const
- bool **isProfileMismatch** () const
- bool **isUncalibratedColor** () const
- **DImgLoaderObserver \* observer** () const
- void **setObserver** (**DImgLoaderObserver** \*const observer)
- **ICCSettingsContainer settings** () const
- void **transform** (**ICCSettingsContainer::Behavior** behavior, const **lccProfile** &specifiedProfile=**lccProfile()**)  
*Same as above, but not using default settings but the given settings.*
- void **transformDefault** ()  
*Transforms the image for full editing, using default settings.*
- void **transformForDisplay** ()  
*Transforms the image for display on screen.*
- void **transformForDisplay** (const **lccProfile** &displayProfile)
- void **transformForDisplay** (**QWidget** \*const widget)
- void **transformForOutput** (const **lccProfile** &outputProfile)  
*Transforms the image for output to the specified output profile.*
- void **transformToSRGB** ()  
*Transforms the image to sRGB.*

## Protected Attributes

- **QString m\_filePath**

## Additional Inherited Members

## Static Public Member Functions inherited from Digikam::lccManager

- static **lccProfile displayProfile** (**QWidget** \*const displayingWidget=**nullptr**)
- static bool **isSRGB** (const **DImg** &img)  
*Returns true if a call to **transformToSRGB()** would have an effect.*
- static bool **needsPostLoadingManagement** (const **DImg** &img)  
*Returns true if the given image is marked as needing user interaction for further color management decision after loading.*
- static void **transformForDisplay** (**QImage** &qimage, const **lccProfile** &displayProfile1=**displayProfile()**)  
*Transforms the given QImage from sRGB to given display profile.*
- static void **transformToSRGB** (**QImage** &qimage, const **lccProfile** &inputProfile)  
*Transforms the given QImage from the given inputProfile to sRGB.*

## Protected Member Functions inherited from [Digikam::IccManager](#)

- void **getTransform** ([IccTransform](#) &trans, [ICCSettingsContainer::Behavior](#) behavior, const [IccProfile](#) &specifiedProfile)
- [ICCSettingsContainer::Behavior](#) **safestBestBehavior** () const
- void **setIccProfile** (const [IccProfile](#) &profile)

## 9.696.1 Constructor & Destructor Documentation

### 9.696.1.1 IccPostLoadingManager()

```
Digikam::IccPostLoadingManager::IccPostLoadingManager (
    DImg & image,
    const QString & filePath = QString\(\),
    const ICCSettingsContainer & settings = IccSettings::instance\(\)->settings() )
[explicit]
```

The [DImg](#) will be edited. The filePath is for display only.

## 9.696.2 Member Function Documentation

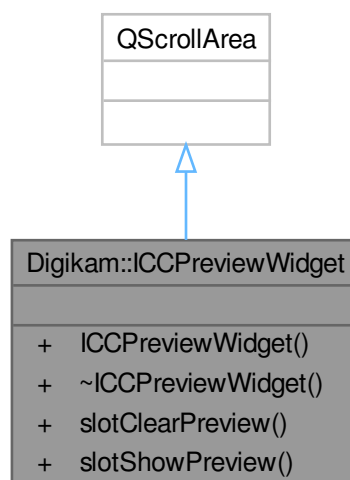
### 9.696.2.1 postLoadingManage()

```
IccTransform Digikam::IccPostLoadingManager::postLoadingManage (
    QWidget *const parent = nullptr )
```

Afterwards, needsPostLoadingManagement will return false.

## 9.697 Digikam::ICCPreviewWidget Class Reference

Inheritance diagram for [Digikam::ICCPreviewWidget](#):



**Public Slots**

- void **slotClearPreview** ()
- void **slotShowPreview** (const QUrl &url)

**Public Member Functions**

- **ICCPreviewWidget** (QWidget \*const parent=nullptr)

**9.698 Digikam::IccProfile Class Reference****Public Types**

- enum **ProfileType** {  
**InvalidType** , **Input** , **Output** , **Display** ,  
**Abstract** , **ColorSpace** , **DeviceLink** , **NamedColor** }

**Public Member Functions**

- **IccProfile** ()  
*Creates a null profile.*
- **IccProfile** (const **IccProfile** &other)
- **IccProfile** (const QByteArray &data)  
*Creates a profile from the given data in memory.*
- **IccProfile** (const QString &filePath)  
*Creates a profile from the given file.*
- void **close** ()  
*Close the profile, freeing resources.*
- QByteArray **data** ()  
*Returns the raw profile data.*
- QString **description** ()  
*Reads the profile description.*
- QString **filePath** () const  
*Returns the filename that this profile was read from.*
- void \* **handle** () const  
*Access to the LCMS cmsHPROFILE handle.*
- bool **isNull** () const
- bool **isOpen** () const  
*Returns if the profile is opened.*
- bool **isSameProfileAs** (**IccProfile** &other)  
*This method compares the actual profile data bit by bit.*
- bool **open** ()  
*Open this profile.*
- **operator void** \* () const
- bool **operator!=** (const **IccProfile** &other) const
- **IccProfile** & **operator=** (const **IccProfile** &other)
- bool **operator==** (const **IccProfile** &other) const  
*Returns true if both profiles are null, if both profiles are created from the same file profile, or if the loaded profile data is identical.*
- **ProfileType** **type** ()
- bool **writeToFile** (const QString &filePath)  
*Writes the profile to the given file.*

## Static Public Member Functions

- static [IccProfile](#) **adobeRGB** ()
- static void **considerOriginalAdobeRGB** (const QString &filePath)
- static QList< [IccProfile](#) > **defaultProfiles** ()  
*Returns a list with the profiles above.*
- static QStringList **defaultSearchPaths** ()  
*Returns the default search paths for ICC profiles.*
- static [IccProfile](#) **proPhotoRGB** ()
- static QList< [IccProfile](#) > **scanDirectories** (const QStringList &dirs)
- static [IccProfile](#) **sRGB** ()  
*Returns the profiles available with RawEngine.*
- static [IccProfile](#) **wideGamutRGB** ()

## 9.698.1 Member Enumeration Documentation

### 9.698.1.1 ProfileType

```
enum Digikam::IccProfile::ProfileType
```

#### Enumerator

InvalidType	Returned for a null profile or an unknown (non-standard) profile type.
Input	For an input device like a scanner or digital camera.
Output	For an output device like a printer.
Display	For a display device like a monitor.

## 9.698.2 Member Function Documentation

### 9.698.2.1 close()

```
void Digikam::IccProfile::close ( )
```

You can re-open. Called automatically at destruction.

### 9.698.2.2 data()

```
QByteArray Digikam::IccProfile::data ( )
```

Reads the data from disk if loaded from disk and not yet loaded.

### 9.698.2.3 defaultSearchPaths()

```
QStringList Digikam::IccProfile::defaultSearchPaths ( ) [static]
```

This does not include any user-specified settings.

#### 9.698.2.4 description()

```
QString Digikam::IccProfile::description ( )
```

Opens the profile if necessary.

#### 9.698.2.5 filePath()

```
QString Digikam::IccProfile::filePath ( ) const
```

returns a null QString() if this profile was loaded from memory.

#### 9.698.2.6 open()

```
bool Digikam::IccProfile::open ( )
```

Returns true if the operation succeeded or the profile is already open. Returns false if the profile is null or the operation failed. You need to open each profile after construction.

#### 9.698.2.7 operator==( )

```
bool Digikam::IccProfile::operator== (
    const IccProfile & other ) const
```

Note: This will not ensure that the data is loaded. Use `isSameProfile()`.

#### 9.698.2.8 sRGB()

```
IccProfile Digikam::IccProfile::sRGB ( ) [static]
```

You still need to call `open()` on them.

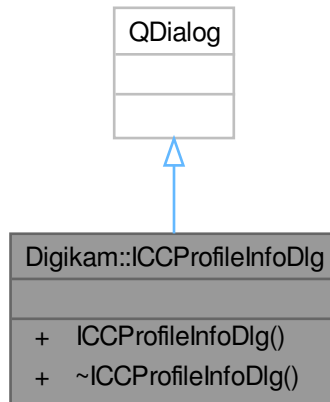
#### 9.698.2.9 type()

```
IccProfile::ProfileType Digikam::IccProfile::type ( )
```

< 'nkb', proprietary in Nikon profiles

## 9.699 Digikam::ICCPProfileInfoDlg Class Reference

Inheritance diagram for Digikam::ICCPProfileInfoDlg:



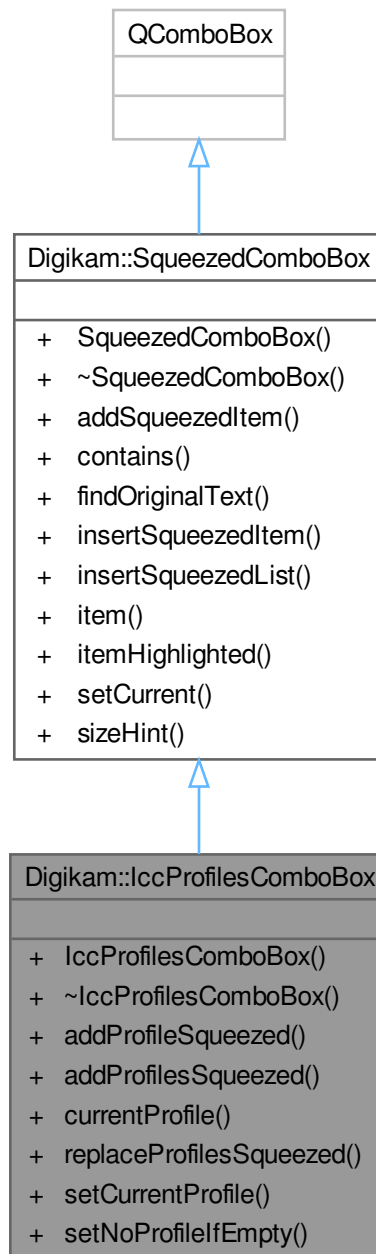
### Public Member Functions

- **ICCPProfileInfoDlg** (`QWidget *const parent`, `const QString &profilePath`, `const IccProfile &profile` ←  
Data=`IccProfile()`)



## 9.700 Digikam::IccProfilesComboBox Class Reference

Inheritance diagram for Digikam::IccProfilesComboBox:



### Public Member Functions

- `IccProfilesComboBox` (`QWidget *const parent=nullptr`)
- void `addProfileSqueezed` (`const IccProfile &profile, const QString &description=QString()`)

- Add the given profile with the given description, or, if null, a standard description.*
- void **addProfilesSqueezed** (const QList< [IccProfile](#) > &profiles)
    - Checks the given profiles for validity, creates a suitable description (ICC profile description, file path), removes duplicates by file path, sorts them and adds them in sorted order.*
  - [IccProfile](#) **currentProfile** () const
    - Retrieves the current profile, or a null profile if none is selected.*
  - void **replaceProfilesSqueezed** (const QList< [IccProfile](#) > &profiles)
    - Clears, does the same as addProfilesSqueezed, and restores the current entry if possible.*
  - void **setCurrentProfile** (const [IccProfile](#) &profile)
    - Sets the current profile.*
  - void **setNoProfileIfEmpty** (const QString &message)
    - Sets a message the is displayed in the combo box and disables the combo box, if the combo box is currently empty.*

## Public Member Functions inherited from [Digikam::SqueezedComboBox](#)

- [SqueezedComboBox](#) (QWidget \*const parent=nullptr, const char \*name=nullptr)
  - Constructor.*
- [~SqueezedComboBox](#) () override
  - destructor*
- void **addSqueezedItem** (const QString &newItem, const QVariant &userData=QVariant())
  - Append an item.*
- bool **contains** (const QString &text) const
  - Returns true if the combobox contains the original (not-squeezed) version of text.*
- int **findOriginalText** (const QString &text, Qt::CaseSensitivity cs=Qt::CaseSensitive) const
  - Returns the index of the combobox if found the original (not-squeezed) version of text.*
- void **insertSqueezedItem** (const QString &newItem, int index, const QVariant &userData=QVariant())
  - This inserts a item to the list.*
- void **insertSqueezedList** (const QStringList &newItems, int index)
  - This inserts items to the list.*
- QString **item** (int index) const
  - This method returns the full text (not squeezed) for the index.*
- QString **itemHighlighted** () const
  - This method returns the full text (not squeezed) of the currently highlighted item.*
- void **setCurrent** (const QString &itemText)
  - Set the current item to the one matching the given text.*
- QSize **sizeHint** () const override
  - Sets the [sizeHint\(\)](#) of this widget.*

## Additional Inherited Members

## Signals inherited from [Digikam::SqueezedComboBox](#)

- void **signalItemActivated** (const QString &)

## 9.700.1 Constructor & Destructor Documentation

### 9.700.1.1 IccProfilesComboBox()

```
Digikam::IccProfilesComboBox::IccProfilesComboBox (
    QWidget *const parent = nullptr ) [explicit]
```

#### Note

Use the signal `currentIndexChanged(int)` for change notification

## 9.700.2 Member Function Documentation

### 9.700.2.1 addProfileSqueezed()

```
void Digikam::IccProfilesComboBox::addProfileSqueezed (
    const IccProfile & profile,
    const QString & description = QString() )
```

Does not test for duplicity, does not sort into existing profiles.

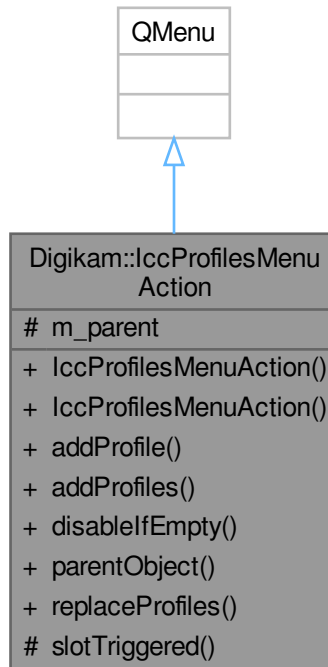
### 9.700.2.2 setCurrentProfile()

```
void Digikam::IccProfilesComboBox::setCurrentProfile (
    const IccProfile & profile )
```

If profile is not in the list, sets no current item (-1)

## 9.701 Digikam::IccProfilesMenuAction Class Reference

Inheritance diagram for Digikam::IccProfilesMenuAction:



### Signals

- void **triggered** (const [IccProfile](#) &profile)

### Public Member Functions

- **IccProfilesMenuAction** (const QIcon &icon, const QString &text, QObject \*const parent)
- **IccProfilesMenuAction** (const QString &text, QObject \*const parent)
- void **addProfile** (const [IccProfile](#) &profile, const QString &description=QString())
  - Add the given profile with the given description, or, if null, a standard description.*
- void **addProfiles** (const QList< [IccProfile](#) > &profile)
  - Checks the given profiles for validity, creates a suitable description (ICC profile description, file path), removes duplicates (in newly added list) by file path, sorts them and adds them in sorted order.*
- void **disableIfEmpty** ()
  - Disables if the menu is currently empty.*
- QObject \* **parentObject** () const
  - Return the parent QObject.*
- void **replaceProfiles** (const QList< [IccProfile](#) > &profile)
  - Equivalent to calling clear() and addProfiles().*

### Protected Slots

- void **slotTriggered** (QObject \*)

### Protected Attributes

- QObject \* **m\_parent** = nullptr

## 9.701.1 Member Function Documentation

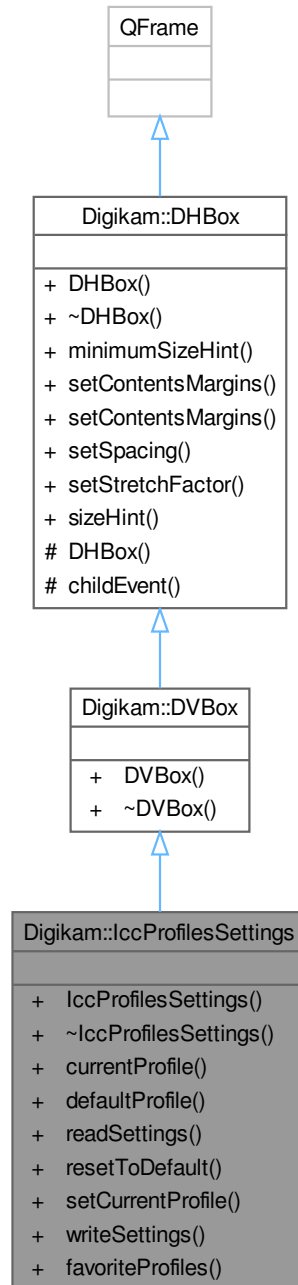
### 9.701.1.1 addProfile()

```
void Digikam::IccProfilesMenuAction::addProfile (
    const IccProfile & profile,
    const QString & description = QString() )
```

Does not test for duplicity, does not sort into existing profiles.

## 9.702 Digikam::IccProfilesSettings Class Reference

Inheritance diagram for Digikam::IccProfilesSettings:



### Signals

- void **signalSettingsChanged** ()

### Public Member Functions

- **IccProfilesSettings** (QWidget \*const parent=nullptr)
- **IccProfile currentProfile** () const
- **IccProfile defaultProfile** () const
- void **readSettings** (KConfigGroup &group)
- void **resetToDefault** ()
- void **setCurrentProfile** (const IccProfile &prof)
- void **writeSettings** (KConfigGroup &group)

### Public Member Functions inherited from Digikam::DVBox

- **DVBox** (QWidget \*const parent=nullptr)

### Public Member Functions inherited from Digikam::DHBox

- **DHBox** (QWidget \*const parent=nullptr)
- QSize **minimumSizeHint** () const override
- void **setContentsMargins** (const QMargins &margins)
- void **setContentsMargins** (int left, int top, int right, int bottom)
- void **setSpacing** (int space)
- void **setStretchFactor** (QWidget \*const widget, int stretch)
- QSize **sizeHint** () const override

### Static Public Member Functions

- static QStringList **favoriteProfiles** (KConfigGroup &group)

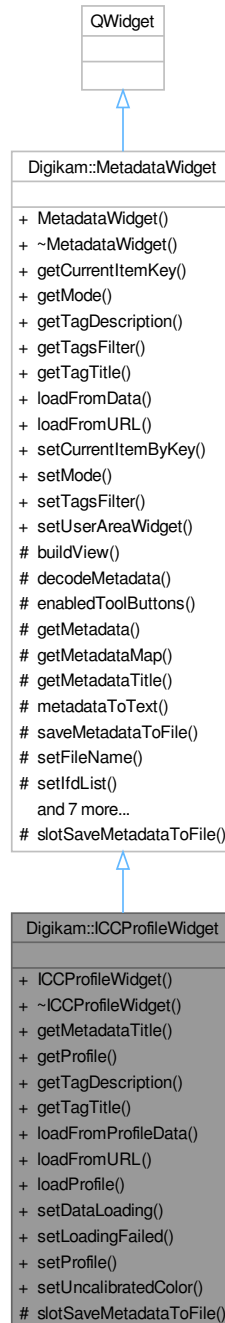
### Additional Inherited Members

### Protected Member Functions inherited from Digikam::DHBox

- **DHBox** (bool vertical, QWidget \*const parent)
- void **childEvent** (QChildEvent \*e) override

## 9.703 Digikam::ICCPProfileWidget Class Reference

Inheritance diagram for Digikam::ICCPProfileWidget:



### Public Member Functions

- **ICCPProfileWidget** (QWidget \*const parent, int w=256, int h=256)
- QString [getMetadataTitle](#) () const override



- [IccProfile](#) **getProfile** () const
- QString **getTagDescription** (const QString &key) override
- QString **getTagTitle** (const QString &key) override
- bool **loadFromProfileData** (const QString &fileName, const QByteArray &data)
- bool **loadFromURL** (const QUrl &url) override
- bool **loadProfile** (const QString &fileName, const [IccProfile](#) &data)
- void **setDataLoading** ()
- void **setLoadingFailed** ()
- bool **setProfile** (const [IccProfile](#) &profile)
- void **setUncalibratedColor** ()

### Public Member Functions inherited from [Digikam::MetadataWidget](#)

- **MetadataWidget** (QWidget \*const parent, const QString &name=QString())
- QString **getCurrentItemKey** () const
- int **getMode** () const
- QStringList **getTagsFilter** () const
- virtual bool **loadFromData** (const QString &fileName, const [DMetadata](#) &data=[DMetadata](#)())
- void **setCurrentItemByKey** (const QString &itemKey)
- void **setMode** (int mode)
- void **setTagsFilter** (const QStringList &list)
- void **setUserAreaWidget** (QWidget \*const w)

### Protected Slots

- void **slotSaveMetadataToFile** () override

### Protected Slots inherited from [Digikam::MetadataWidget](#)

- virtual void **slotSaveMetadataToFile** ()=0

### Additional Inherited Members

### Public Types inherited from [Digikam::MetadataWidget](#)

- enum **TagFilters** { **NONE** = 0 , **PHOTO** , **CUSTOM** }

### Signals inherited from [Digikam::MetadataWidget](#)

- void **signalSetupMetadataFilters** ()

## Protected Member Functions inherited from [Digikam::MetadataWidget](#)

- void **enabledToolButtons** (bool)
- [DMetadata](#) \* **getMetadata** () const
- const [DMetadata::MetaDataMap](#) & **getMetadataMap** ()
- QString **metadataToText** () const
- QUrl **saveMetadataToFile** (const QString &caption, const QString &fileFilter)
- void **setFileName** (const QString &fileName)
- void **setIfdList** (const [DMetadata::MetaDataMap](#) &ifds, const QStringList &keysFilter, const QStringList &tagsFilter)
- void **setIfdList** (const [DMetadata::MetaDataMap](#) &ifds, const QStringList &tagsFilter=QStringList())
- bool **setMetadata** (const [DMetadata](#) &data=[DMetadata](#)())
- virtual void **setMetadataEmpty** ()
- void **setMetadataMap** (const [DMetadata::MetaDataMap](#) &data=[DMetadata::MetaDataMap](#)())
- void **setup** ()
  - Call this method in children class constructors to init signal/slots connections.*
- bool **storeMetadataToFile** (const QUrl &url, const QByteArray &metaData)
- [MetadataListView](#) \* **view** () const

### 9.703.1 Member Function Documentation

#### 9.703.1.1 getMetadataTitle()

```
QString Digikam::ICCPProfileWidget::getMetadataTitle ( ) const [override], [virtual]
```

Implements [Digikam::MetadataWidget](#).

#### 9.703.1.2 getTagDescription()

```
QString Digikam::ICCPProfileWidget::getTagDescription (
    const QString & key ) [override], [virtual]
```

Reimplemented from [Digikam::MetadataWidget](#).

#### 9.703.1.3 getTagTitle()

```
QString Digikam::ICCPProfileWidget::getTagTitle (
    const QString & key ) [override], [virtual]
```

Reimplemented from [Digikam::MetadataWidget](#).

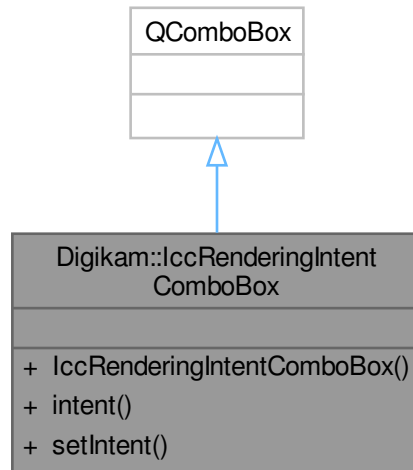
#### 9.703.1.4 loadFromURL()

```
bool Digikam::ICCPProfileWidget::loadFromURL (
    const QUrl & url ) [override], [virtual]
```

Implements [Digikam::MetadataWidget](#).

## 9.704 Digikam::IccRenderingIntentComboBox Class Reference

Inheritance diagram for Digikam::IccRenderingIntentComboBox:

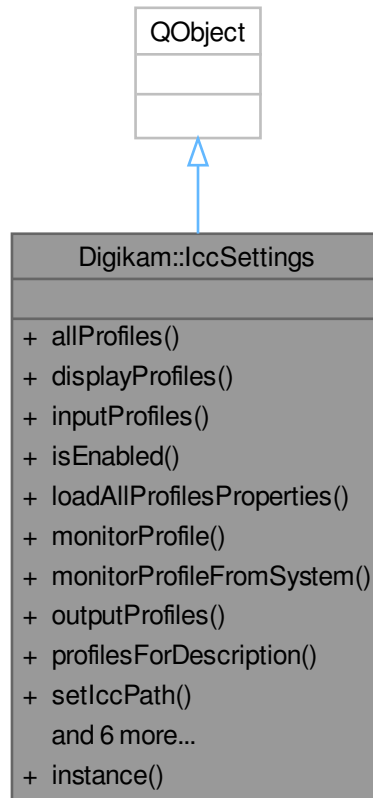


### Public Member Functions

- **IccRenderingIntentComboBox** (`QWidget *const parent=nullptr`)
- `int intent ()` const
- `void setIntent (int intent)`

## 9.705 Digikam::IccSettings Class Reference

Inheritance diagram for Digikam::IccSettings:



### Signals

- void **signalICCSettingsChanged** (const [ICCSettingsContainer](#) &current, const [ICCSettingsContainer](#) &previous)
- void **signalSettingsChanged** ()

### Public Member Functions

- [QList< IccProfile >](#) **allProfiles** ()
- [QList< IccProfile >](#) **displayProfiles** ()  
*Get available profiles suitable as monitor/display profile.*
- [QList< IccProfile >](#) **inputProfiles** ()  
*Get available profiles suitable as input profile.*
- bool **isEnabled** () const  
*Returns if color management is enabled.*
- void [loadAllProfilesProperties](#) ()

- IccProfile* caches some of its properties (description, type) when it was read once.

  - **IccProfile** `monitorProfile` (QWidget \*const widget=nullptr)
 

Returns the monitor profile (for color managed view).
  - bool **monitorProfileFromSystem** () const
 

Returns if the monitor profile (as returned by `monitorProfile()`) is set system-wide, so that the `monitorProfile` field of the current settings need not be set and will not be used by `monitorProfile()`.
  - QList< **IccProfile** > **outputProfiles** ()
 

Get available profiles suitable as proof/output profiles.
  - QList< **IccProfile** > **profilesForDescription** (const QString &description)
 

Returns a list of profiles with the given description()
  - void **setIccPath** (const QString &path)
  - void **setSettings** (const **ICCSettingsContainer** &settings)
 

Sets the current ICC settings and writes them to config.
  - **ICCSettingsContainer** **settings** ()
 

Returns the current ICC settings.
  - void **setUseManagedPreviews** (bool `useManagedPreviews`)
  - void **setUseManagedView** (bool `useManagedView`)
 

Set single parts of the settings.
  - bool **useManagedPreviews** () const
 

Returns if color management for previews is enabled.
  - QList< **IccProfile** > **workspaceProfiles** ()
 

Get available profiles suitable as workspace profile.

### Static Public Member Functions

- static **IccSettings** \* **instance** ()
 

Global container for ICC settings.

### Friends

- class **IccSettingsCreator**
- class **Private**

## 9.705.1 Member Function Documentation

### 9.705.1.1 instance()

```
IccSettings * Digikam::IccSettings::instance ( ) [static]
```

All accessor methods are thread-safe.

### 9.705.1.2 loadAllProfilesProperties()

```
void Digikam::IccSettings::loadAllProfilesProperties ( )
```

Subsequently, to read these values no opening is needed. This ensures that all profiles have these values read. May imply scanning and opening all profiles.

### 9.705.1.3 monitorProfile()

```

IccProfile Digikam::IccSettings::monitorProfile (
    QWidget *const widget = nullptr )

```

If there are multiple screens, a system-wide settings specifies the monitor profile, and the widget parameter is passed, the returned profile is for the widget's screen. If no settings is specified, the default sRGB profile is returned.

## 9.706 Digikam::IccSettingsContainer Class Reference

### Public Types

- typedef QFlags< BehaviorEnum > **Behavior**
- enum BehaviorEnum {
  - InvalidBehavior** = 0 , **UseEmbeddedProfile** = 1 << 0 , **UseSRGB** = 1 << 1 , **UseWorkspace** = 1 << 2 , **UseDefaultInputProfile** = 1 << 3 , **UseSpecifiedProfile** = 1 << 4 , **AutomaticColors** = 1 << 5 , **DoNotInterpret** = 1 << 6 ,
  - KeepProfile** = 1 << 10 , **ConvertToWorkspace** = 1 << 11 , **LeaveFileUntagged** = 1 << 18 , **AskUser** = 1 << 20 ,
  - SafestBestAction** = 1 << 21 , **PreserveEmbeddedProfile** = UseEmbeddedProfile | KeepProfile ,
  - EmbeddedToWorkspace** = UseEmbeddedProfile | ConvertToWorkspace , **SRGBToWorkspace** = UseSRGB | ConvertToWorkspace ,
  - AutoToWorkspace** = AutomaticColors | ConvertToWorkspace , **InputToWorkspace** = UseDefaultInputProfile | ConvertToWorkspace , **SpecifiedToWorkspace** = UseSpecifiedProfile | ConvertToWorkspace ,
  - NoColorManagement** = DoNotInterpret | LeaveFileUntagged }

### Public Member Functions

- void **readFromConfig** (KConfigGroup &group)
- void **writeManagedPreviewsToConfig** (KConfigGroup &group) const
- void **writeManagedViewToConfig** (KConfigGroup &group) const
- void **writeToConfig** (KConfigGroup &group) const

### Public Attributes

- QString **defaultInputProfile**
- Behavior **defaultMismatchBehavior** = EmbeddedToWorkspace
- Behavior **defaultMissingProfileBehavior** = SRGBToWorkspace
- QString **defaultProofProfile**
- Behavior **defaultUncalibratedBehavior** = AutoToWorkspace
- bool **doGamutCheck** = false
- bool **enableCM** = true
- QColor **gamutCheckMaskColor** = QColor(126, 255, 255)
- QString **iccFolder**
- Behavior **lastMismatchBehavior** = EmbeddedToWorkspace
- Behavior **lastMissingProfileBehavior** = SRGBToWorkspace
- QString **lastSpecifiedAssignProfile**
- QString **lastSpecifiedInputProfile**
- Behavior **lastUncalibratedBehavior** = AutoToWorkspace
- QString **monitorProfile**
- int **proofingRenderingIntent** = IccTransform::AbsoluteColorimetric
  - Settings specific for soft proofing.*
- int **renderingIntent** = IccTransform::Perceptual
- bool **useBPC** = true
- bool **useManagedPreviews** = true
- bool **useManagedView** = true
- QString **workspaceProfile**

## 9.706.1 Member Enumeration Documentation

### 9.706.1.1 BehaviorEnum

enum `Digikam::ICCSettingsContainer::BehaviorEnum`

Enumerator

<code>InvalidBehavior</code>	Note: Values are stored in config - keep them constant.
<code>UseEmbeddedProfile</code>	Interpretation of the image data.
<code>KeepProfile</code>	Transformation / target profile.
<code>LeaveFileUntagged</code>	Special flags and values.
<code>PreserveEmbeddedProfile</code>	ready combinations for convenience

## 9.707 Digikam::lccTransform Class Reference

### Public Types

- enum `RenderingIntent` { `Perceptual` = 0 , `RelativeColorimetric` = 1 , `Saturation` = 2 , `AbsoluteColorimetric` = 3 }

### Public Member Functions

- `lccTransform` (const `lccTransform` &other)
- bool `apply` (`DImage` &image, `DImageLoaderObserver` \*const observer=nullptr)
  - Apply this transform with the set profiles and options to the image.*
- bool `apply` (`QImage` &qimage)
  - Apply this transform to the QImage.*
- `QColor` `checkGamutMaskColor` () const
- void `close` ()
  - Closes the transform, not the profiles.*
- `lccProfile` `effectiveInputProfile` () const
  - Returns the embedded profile; if none is set, the input profile; if none is set, sRGB.*
- `lccProfile` `embeddedProfile` () const
  - Returns the contained profiles.*
- `lccProfile` `inputProfile` () const
- `RenderingIntent` `intent` () const
- bool `isCheckingGamut` () const
- bool `isUsingBlackPointCompensation` () const
- `lccTransform` & `operator=` (const `lccTransform` &other)
- `lccProfile` `outputProfile` () const
- `RenderingIntent` `proofIntent` () const
- `lccProfile` `proofProfile` () const
- void `setCheckGamut` (bool checkGamut)
- void `setCheckGamutMaskColor` (const `QColor` &color)
- void `setDoNotEmbedOutputProfile` (bool doNotEmbed)
  - Call this with 'true' if you do not want the output profile to be set as embedded profile after `apply()` did a transformation.*
- void `setEmbeddedProfile` (const `DImage` &image)

*Sets the input profiles of this transform.*

- void **setInputProfile** (const [IccProfile](#) &profile)
- void **setIntent** (int intent)
- void **setIntent** (RenderingIntent intent)

*Set options.*

- void **setOutputProfile** (const [IccProfile](#) &profile)

*Sets the output transform.*

- void **setProofIntent** (int intent)
- void **setProofIntent** (RenderingIntent intent)
- void **setProofProfile** (const [IccProfile](#) &profile)

*Makes this transform a proofing transform, if profile is not null.*

- void **setUseBlackPointCompensation** (bool useBPC)
- bool **willHaveEffect** ()

*Returns if this transformation will have an effect, i.e.*

## Static Public Member Functions

- static void **init** ()

*Initialize LittleCMS library.*

## 9.707.1 Member Function Documentation

### 9.707.1.1 **apply()** [1/2]

```
bool Digikam::IccTransform::apply (
    DImg & image,
    DImgLoaderObserver *const observer = nullptr )
```

Optionally pass an observer to get progress information.

### 9.707.1.2 **apply()** [2/2]

```
bool Digikam::IccTransform::apply (
    QImage & qimage )
```

This has only basic functionality.

### 9.707.1.3 **close()**

```
void Digikam::IccTransform::close ( )
```

Called at destruction.

### 9.707.1.4 **setDoNotEmbedOutputProfile()**

```
void Digikam::IccTransform::setDoNotEmbedOutputProfile (
    bool doNotEmbed )
```

Default is to set the output profile as embedded profile (false).



### 9.707.1.5 setEmbeddedProfile()

```
void Digikam::IccTransform::setEmbeddedProfile (
    const DImg & image )
```

You can call both setEmbeddedProfile and setInputProfile. If the image contains an embedded profile this profile is used and takes precedence over the set input profile, which is used without an embedded profile. If none is set, sRGB is used.

### 9.707.1.6 willHaveEffect()

```
bool Digikam::IccTransform::willHaveEffect ( )
```

if effective input profile and output profile are different.

## 9.708 Digikam::IccTransformFilter Class Reference

Inheritance diagram for Digikam::IccTransformFilter:



### Public Member Functions

- **IccTransformFilter** (**Dimg** \*const orgImage, **QObject** \*const parent, const **IccTransform** &transform)
- **IccTransformFilter** (**QObject** \*const parent=nullptr)

- [FilterAction filterAction \(\)](#) override  
*Returns the action description corresponding to currently set options.*
- [QString filterIdentifier \(\)](#) const override  
*Return the identifier for this filter in the image history.*
- [bool parametersSuccessfullyRead \(\)](#) const override  
*Optional: error handling for readParameters.*
- [void readParameters \(const FilterAction &action\)](#) override
- [QString readParametersError \(const FilterAction &actionThatFailed\)](#) const override

## Public Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter \(DImg \\*const orgImage, QObject \\*const parent, const QString &name=QString\(\)\)](#)  
*Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter \(QObject \\*const parent=nullptr, const QString &name=QString\(\)\)](#)  
*Constructs a filter without argument.*
- [virtual void cancelFilter \(\)](#)  
*Cancel the threaded computation.*
- [const QString &filterName \(\)](#)
- [int filterVersion \(\)](#) const
- [DImg getTargetImage \(\)](#)
- [QList< int > multithreadedSteps \(int stop, int start=0\)](#) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- [void setFilterName \(const QString &name\)](#)
- [void setFilterVersion \(int version\)](#)  
*Replaying a filter action: Set the filter version.*
- [void setOriginalImage \(const DImg &orgImage\)](#)
- [void setupAndStartDirectly \(const DImg &orgImage, DImgThreadedFilter \\*const master, int progress←Begin=0, int progressEnd=100\)](#)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- [void setupFilter \(const DImg &orgImage\)](#)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- [virtual void startFilter \(\)](#)  
*Start the threaded computation.*
- [virtual void startFilterDirectly \(\)](#)  
*Start computation of this filter, directly in this thread.*
- [virtual QList< int > supportedVersions \(\)](#) const

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread \(QObject \\*const parent=nullptr\)](#)  
*This class extends QRunnable, so you have to reimplement virtual void [run\(\)](#).*
- [~DynamicThread \(\)](#) override  
*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- [bool isFinished \(\)](#) const
- [bool isRunning \(\)](#) const
- [QThread::Priority priority \(\)](#) const
- [void setEmitSignals \(bool emitThem\)](#)
- [void setPriority \(QThread::Priority priority\)](#)  
*Sets the priority for this dynamic thread.*
- [State state \(\)](#) const

## Public Member Functions inherited from [Digikam::DImgLoaderObserver](#)

- virtual bool **continueQuery** ()  
*Queries whether the image IO operation shall be continued.*
- virtual float **granularity** ()  
*Return a relative value which determines the granularity, the frequency with which the [DImgLoaderObserver](#) is checked and progress is posted.*

## Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

## Protected Member Functions

- void **filterImage** () override  
*Main image filter method.*
- void **progressInfo** (float **progress**) override  
*Posts progress information about image IO.*

## Protected Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void **cleanupFilter** ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void **initFilter** ()  
*Start filter operation before threaded method.*
- void **initMaster** ()
- void **initSlave** ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int **modulateProgress** (int **progress**)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void **postProgress** (int **progress**)  
*Emit progress info.*
- virtual void **prepareDestImage** ()
- void **run** () override  
*List of threaded operations by filter.*
- void **setSlave** ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from Digikam::DynamicThread

- bool **runningFlag** () const volatile  
*In you [run\(\)](#) method, you shall regularly check for [runningFlag\(\)](#) and cleanup and return if false.*
- void **shutDown** ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call [stop\(\)](#) and [wait\(\)](#), knowing that nothing will call [start\(\)](#) anymore after this 3) Be sure the thread will never be running at destruction.*
- void **start** (QMutexLocker< QMutex > &locker)  
*Doing the same as [start\(\)](#), [stop\(\)](#) and [wait](#) above, provide it with a locked QMutexLocker on mutex().*
- void **stop** (const QMutexLocker< QMutex > &locker)
- QMutex \* **threadMutex** () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void **wait** (QMutexLocker< QMutex > &locker)

## Additional Inherited Members

## Public Types inherited from Digikam::DynamicThread

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

## Public Slots inherited from Digikam::DynamicThread

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

## Signals inherited from Digikam::DlmgThreadedFilter

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

## Signals inherited from Digikam::DynamicThread

- void **finished** ()
- void **starting** ()  
*Emitted if [emitSignals](#) is enabled.*

## Protected Attributes inherited from [Digikam::DImgThreadedFilter](#)

- [DImg m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter \\* m\\_master](#) = nullptr  
*The master of this slave filter.*
- [QString m\\_name](#)  
*Filter name.*
- [DImg m\\_orgImage](#)  
*Copy of original Image data.*
- [int m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- [int m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in [postProgress\(\)](#).*
- [int m\\_progressSpan](#) = 0
- [DImgThreadedFilter \\* m\\_slave](#) = nullptr  
*The current slave.*
- [int m\\_version](#) = 1
- [bool m\\_wasCancelled](#) = false

### 9.708.1 Member Function Documentation

#### 9.708.1.1 [filterAction\(\)](#)

[FilterAction](#) [Digikam::IccTransformFilter::filterAction \( \)](#) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

#### 9.708.1.2 [filterIdentifier\(\)](#)

[QString](#) [Digikam::IccTransformFilter::filterIdentifier \( \)](#) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

#### 9.708.1.3 [filterImage\(\)](#)

[void](#) [Digikam::IccTransformFilter::filterImage \( \)](#) [override], [protected], [virtual]

Override in subclass.

Implements [Digikam::DImgThreadedFilter](#).

#### 9.708.1.4 [parametersSuccessfullyRead\(\)](#)

[bool](#) [Digikam::IccTransformFilter::parametersSuccessfullyRead \( \)](#) const [override], [virtual]

When [readParameters\(\)](#) has been called, this method will return true if the call was successful, and false if not. If returning false, [readParametersError\(\)](#) will give an error message. The default implementation always returns success. You only need to reimplement when a filter is likely to fail in a different environment, e.g. depending on availability of installed files. These methods have an undefined return value if [readParameters\(\)](#) was not called previously.

Reimplemented from [Digikam::DImgThreadedFilter](#).

### 9.708.1.5 progressInfo()

```
void Digikam::IccTransformFilter::progressInfo (
    float progress ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::DImgLoaderObserver](#).

### 9.708.1.6 readParameters()

```
void Digikam::IccTransformFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

### 9.708.1.7 readParametersError()

```
QString Digikam::IccTransformFilter::readParametersError (
    const FilterAction & actionThatFailed ) const [override], [virtual]
```

Reimplemented from [Digikam::DImgThreadedFilter](#).

## 9.709 Digikam::Identity Class Reference

### Public Member Functions

- [Identity](#) ()  
*Wraps a face recognition [Identity](#).*
- [Identity](#) (const [Identity](#) &other)
- QString [attribute](#) (const QString &att) const  
*Attribute value accessor.*
- QMap< QString, QString > [attributesMap](#) () const  
*Attributes map accessor.*
- int [id](#) () const  
*Id value accessor.*
- bool [isNull](#) () const
- [Identity](#) & [operator=](#) (const [Identity](#) &other)
- bool [operator==](#) (const [Identity](#) &other) const
- void [setAttribute](#) (const QString &att, const QString &val)
- void [setAttributesMap](#) (const QMap< QString, QString > &attributes)
- void [setId](#) (int [id](#))

## 9.709.1 Constructor & Destructor Documentation

### 9.709.1.1 Identity()

```
Digikam::Identity::Identity ( )
```

An identity refers to a natural person. There is an internal id which is used the [FacesEngine](#) storage, and a number of attributes which map the identity to the outside. Prespecified attributes: "fullName" The full name as on the ID card, e.g. "Peter Brown" "name" The person's name without further specification, e.g. "Peter" or "Dad" "uuid" A UUID that is assigned to each new identity at creation.

For fullName and name, multiple values are allowed.

Attributes can be used to map an identity to other fields and services where natural persons play a role.

## 9.710 Digikam::IdentityProvider Class Reference

### Public Member Functions

- [Identity](#) **addIdentity** (const QMap< QString, QString > &attributes)  
*Adds a new identity with the specified attributes.*
- [Identity](#) **addIdentityDebug** (const QMap< QString, QString > &attributes)  
*This is the debug version of addIdentity, so the identity is only added to identityCache, but not into the recognition database.*
- int **addTraining** (const [Identity](#) &identity, const QString &hash, const cv::Mat &feature)  
*add the face features and hash to the recognition DB returns the ID of the new row*
- QList< [Identity](#) > **allIdentities** () const  
*Returns all identities known to the database.*
- void **clearAllTraining** ()  
*clears all identities and face training from the recognition DB*
- bool **clearTraining** (const QString &hash)  
*Deletes the training image for the given hash, leaving the identity as such in the database.*
- void **deleteIdentities** (QList< [Identity](#) > identitiesToBeDeleted)  
*Deletes a list of identities from the database.*
- void **deleteIdentity** (const [Identity](#) &identityToBeDeleted)  
*Deletes an identity from the database.*
- [Identity](#) **findIdentity** (const QMap< QString, QString > &attributes) const  
*Finds the identity matching the given attributes.*
- [Identity](#) **findIdentity** (const QString &attribute, const QString &value) const  
*Finds the first identity with matching attribute - value.*
- [Identity](#) **identity** (int id) const
- bool **integrityCheck** ()  
*Checks the integrity and returns true if everything is fine.*
- bool **isValidId** (int label) const  
*checks if the id exists in the recognition DB*
- void **vacuum** ()  
*Shrinks the database.*

### Static Public Member Functions

- static [IdentityProvider](#) \* **instance** ()

### Protected Member Functions

- bool **addIdentityFace** (const [Identity](#) &identity, QString &hash, cv::Mat embedding)
- bool **deleteIdentityFace** (const [Identity](#) &identity, QString &hash)
- cv::Ptr< cv::ml::TrainData > **getTrainingData** () const  
*Deletes a list of identities from the database.*
- bool **initialize** ()

### Friends

- class **FaceClassifier**
- class **Identity**
- class **IdentityProviderCreator**



## 9.710.1 Member Function Documentation

### 9.710.1.1 addIdentity()

```
Identity Digikam::IdentityProvider::addIdentity (
    const QMap< QString, QString > & attributes )
```

Please note that a UUID is automatically generated.

### 9.710.1.2 findIdentity() [1/2]

```
Identity Digikam::IdentityProvider::findIdentity (
    const QMap< QString, QString > & attributes ) const
```

Attributes are first checked with knowledge of their meaning. Secondly, all unknown attributes are used. Returns a null [Identity](#) if no match is possible or the map is empty.

### 9.710.1.3 findIdentity() [2/2]

```
Identity Digikam::IdentityProvider::findIdentity (
    const QString & attribute,
    const QString & value ) const
```

Returns a null identity if no match is found or attribute is empty.

## 9.711 Digikam::ImageChangeset Class Reference

### Public Member Functions

- [ImageChangeset](#) ()=default  
*An [ImageChangeset](#) covers adding or changing any properties of an image.*
- [ImageChangeset](#) (const QList< qlonglong > &ids, const [DatabaseFields::Set](#) &changes)
- [ImageChangeset](#) (qlonglong id, const [DatabaseFields::Set](#) &changes)
- [DatabaseFields::Set](#) **changes** () const
- bool **containsImage** (qlonglong id) const
- QList< qlonglong > **ids** () const
- [ImageChangeset](#) & **operator**<< (const QDBusArgument &argument)
- const [ImageChangeset](#) & **operator**>> (QDBusArgument &argument) const

### 9.711.1 Constructor & Destructor Documentation

#### 9.711.1.1 ImageChangeset()

```
Digikam::ImageChangeset::ImageChangeset ( ) [default]
```

It is described by a list of affected image ids, and a set of affected database fields. There is no guarantee that information in the database has actually been changed.

## 9.712 Digikam::ImageCommonContainer Class Reference

### Public Attributes

- int **colorDepth** = 0  
*bits per channel, 8/16*
- QString **colorModel**
- QDateTime **creationDate**
- QDateTime **digitizationDate**
- QDateTime **fileModificationDate**
- QString **fileName**
- qint64 **fileSize** = 0
- QString **format**
- int **height** = 0
- QString **orientation**
- int **rating** = -1
- int **width** = 0

## 9.713 Digikam::ImageCurves Class Reference

### Public Types

- typedef double **CRMatrix**[4][4]
- enum **CurveType** { **CURVE\_SMOOTH** = 0 , **CURVE\_FREE** }

### Public Member Functions

- **ImageCurves** (bool sixteenBit)
- **ImageCurves** (const [CurvesContainer](#) &container)
- **ImageCurves** (const [ImageCurves](#) &other)
- QByteArray **channelToBinary** (int channel) const  
*Writes the given channel to a raw binary representation.*
- void **curvesCalculateAllCurves** ()
- void **curvesCalculateCurve** (int channel)
- void **curvesChannelReset** (int channel)
- float **curvesLutFunc** (int n\_channels, int channel, float value)
- void **curvesLutProcess** (uchar \*const srcPR, uchar \*const destPR, int w, int h)
- void **curvesLutSetup** (int nchannels)
- void **curvesReset** ()  
*Methods to manipulate the curves data.*
- void **fillFromOtherCurves** (const [ImageCurves](#) \*const otherCurves)  
*Fills this curves with the data supplied by another curves object.*
- [CurvesContainer](#) **getContainer** () const  
*Returns a container with the settings for all channels of this Curves object.*
- [CurvesContainer](#) **getContainer** (int channel) const  
*Returns a container containing the values of this Curves object for the given channel, and linear values for all other channels.*
- QPoint **getCurvePoint** (int channel, int point) const
- QPolygon **getCurvePoints** (int channel) const
- int **getCurvePointX** (int channel, int point) const

- int **getCurvePointY** (int channel, int point) const
- [CurveType](#) **getCurveType** (int channel) const
- int **getCurveValue** (int channel, int bin) const
- QPolygon **getCurveValues** (int channel) const
- bool **isCurvePointEnabled** (int channel, int point) const
- bool **isDirty** () const
  - Curves properties.*
- bool **isLinear** () const
- bool **isLinear** (int channel) const
  - Returns true if the curve is linear for the given channel, or all channels.*
- bool **isSixteenBits** () const
- bool **loadCurvesFromGimpCurvesFile** (const QUrl &fileUrl)
- [ImageCurves](#) & **operator=** (const [ImageCurves](#) &other)
- bool **saveCurvesToGimpCurvesFile** (const QUrl &fileUrl) const
  - Methods to save/load the curves values to/from a Gimp curves text file.*
- bool **setChannelFromBinary** (int channel, const QByteArray &array)
  - Set the channel from the given raw binary representation.*
- void **setContainer** (const [CurvesContainer](#) &container)
- void **setCurvePoint** (int channel, int point, const QPoint &val)
- void **setCurvePoints** (int channel, const QPolygon &vals)
- void **setCurvePointX** (int channel, int point, int x)
- void **setCurvePointY** (int channel, int point, int y)
- void **setCurveType** ([CurveType](#) type)
- void **setCurveType** (int channel, [CurveType](#) type)
- void **setCurveValue** (int channel, int bin, int val)
  - Methods to set manually the curves values.*
- void **setCurveValues** (int channel, const QPolygon &vals)
- void **unsetCurvePoint** (int channel, int point)

### Static Public Member Functions

- static QPoint **getDisabledValue** ()

### Static Public Attributes

- static const int **MULTIPLIER\_16BIT** = 255
  - Curve points have to multiplied with this value for 16 bit images.*
- static const int **NUM\_CHANNELS** = 5
  - Number of channels in a curve.*
- static const int **NUMBER\_OF\_POINTS** = 17
  - The max number of points contained in a curve.*

## 9.713.1 Member Enumeration Documentation

### 9.713.1.1 CurveType

enum [Digikam::ImageCurves::CurveType](#)

## Enumerator

CURVE_SMOOTH	Smooth curve type.
CURVE_FREE	Freehand curve type.

## 9.713.2 Member Function Documentation

### 9.713.2.1 channelToBinary()

```
QByteArray Digikam::ImageCurves::channelToBinary (
    int channel ) const
```

Binary format:

Note that 16bit free curves take a lot of memory (~85kB) while all other forms take less than 400 bytes.

Version 1 :16 Type 0,1,2 : 8 Bytes depth 1,2 : 8 reserved :32 count :32

Type 0 (linear curve): Type 1 (smooth curve): for (0...count) point.x :32 point.y :32 Type 2 (free curve): for (0...count) if (Bytes depth == 1) value : 8 else if (Bytes depth == 2) value :16

In Big Endian byte order. Data then converted to base64.

### 9.713.2.2 fillFromOtherCurves()

```
void Digikam::ImageCurves::fillFromOtherCurves (
    const ImageCurves *const otherCurves )
```

This ensures that 8 and 16 bit curves are properly converted.

#### Parameters

<i>otherCurves</i>	other curves object to adapt config from
--------------------	--

### 9.713.2.3 setChannelFromBinary()

```
bool Digikam::ImageCurves::setChannelFromBinary (
    int channel,
    const QByteArray & array )
```

The data is checked for validity, only on valid data true is returned. Note that the bytes depth (isSixteenBits()) of the encoded representation must match the depth of this curves object.

### 9.713.2.4 setContainer()

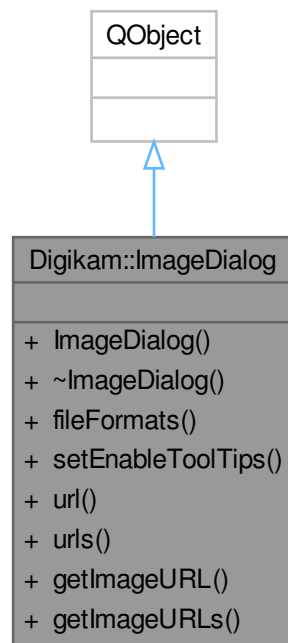
```
void Digikam::ImageCurves::setContainer (
    const CurvesContainer & container )
```

## Note

bits depth must match

## 9.714 Digikam::ImageDialog Class Reference

Inheritance diagram for Digikam::ImageDialog:



### Public Member Functions

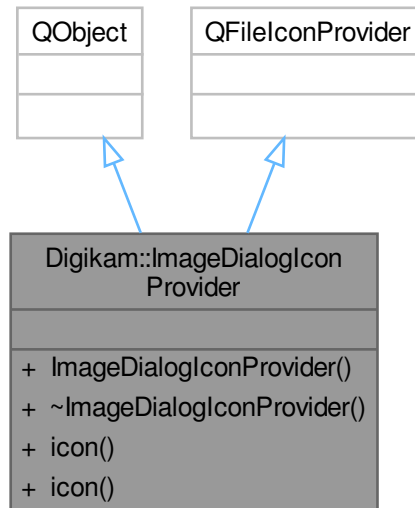
- **ImageDialog** (`QWidget *const parent`, `const QUrl &url`, `bool singleSelect=false`, `const QString &caption=QString()`)
- `QStringList fileFormats () const`
- `void setEnabledToolTips (bool val)`
- `QUrl url () const`
- `QList< QUrl > urls () const`

### Static Public Member Functions

- `static QUrl getImageURL (QWidget *const parent, const QUrl &url, const QString &caption=QString())`
- `static QList< QUrl > getImageURLs (QWidget *const parent, const QUrl &url, const QString &caption=QString())`

## 9.715 Digikam::ImageDialogIconProvider Class Reference

Inheritance diagram for Digikam::ImageDialogIconProvider:



### Signals

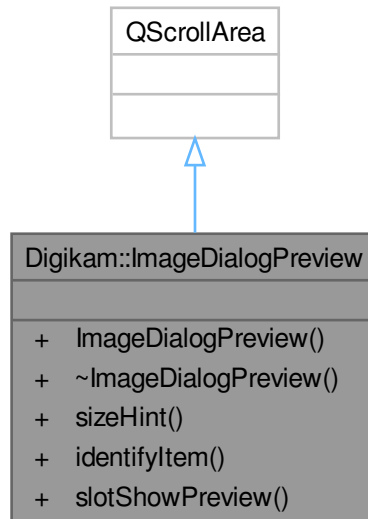
- void **signalThumbnailRefresh** ()

### Public Member Functions

- QIcon **icon** (const QFileInfo &info) const override
- QIcon **icon** (QAbstractFileIconProvider::IconType type) const override

## 9.716 Digikam::ImageDialogPreview Class Reference

Inheritance diagram for Digikam::ImageDialogPreview:



### Public Slots

- void **slotShowPreview** (const `QUrl` &url)

### Public Member Functions

- **ImageDialogPreview** (`QWidget` \*const parent=nullptr)
- `QSize` **sizeHint** () const override

### Static Public Member Functions

- static `QString` **identifyItem** (const `QUrl` &url, const `QImage` &preview=`QImage`())

## 9.717 Digikam::ImageDialogToolTip Class Reference

Inheritance diagram for Digikam::ImageDialogToolTip:



### Public Member Functions

- void **setData** (QAbstractItemView \*const view, const QModelIndex &index, const QUrl &url)

### Public Member Functions inherited from [Digikam::DItemToolTip](#)

- **DItemToolTip** (QWidget \*const parent=nullptr)



### Additional Inherited Members

### Protected Member Functions inherited from [Digikam::DItemToolTip](#)

- bool **event** (QEvent \*) override
- void **paintEvent** (QPaintEvent \*) override
- void **renderArrows** ()
- void **reposition** ()
- void **resizeEvent** (QResizeEvent \*) override
- bool **toolTipsEmpty** () const
- void **updateToolTip** ()

## 9.718 Digikam::ImageGuideWidget Class Reference

Inheritance diagram for Digikam::ImageGuideWidget:



### Public Types

- enum `ColorPointSrc` { `OriginalImage` = 0 , `PreviewImage` , `TargetPreviewImage` }
- enum `GuideToolMode` { `HVGuideMode` = 0 , `PickColorMode` }

## Public Slots

- void **slotChangeGuideColor** (const QColor &color)
- void **slotChangeGuideSize** (int size)
- void **slotPreviewModeChanged** (int mode)

## Signals

- void **signalResized** ()
- void **spotPositionChangedFromOriginal** (const Digikam::DColor &color, const QPoint &position)
- void **spotPositionChangedFromTarget** (const Digikam::DColor &color, const QPoint &position)

## Public Member Functions

- **ImageGuideWidget** (QWidget \*const parent=nullptr, bool spotVisible=true, int guideMode=PickColor←Mode, const QColor &guideColor=Qt::red, int guideSize=1, bool blink=false, [Imagelface::PreviewType](#) type=[Imagelface::FullImage](#))
- void **exposureSettingsChanged** ()
- QImage **getMask** () const
- DColor **getSpotColor** (int getColorFrom) const
- QPoint **getSpotPosition** () const
- void **ICCSettingsChanged** ()
- [Imagelface](#) \* **imagelface** () const
- int **previewMode** () const
- void **resetPoints** ()
- void **resetSpotPosition** ()
- void **setBackgroundColor** (const QColor &)
- void **setEraseMode** (bool erase)
- void **setMaskCursor** ()
- void **setMaskEnabled** (bool enabled)
- void **setMaskPenSize** (int size)
- void **setPaintColor** (const QColor &color)
- void **setPoints** (const QPolygon &p, bool drawLine=false)
- void **setSpotVisible** (bool spotVisible, bool blink=false)
- void **setSpotVisibleNoUpdate** (bool spotVisible)
- void **updatePreview** ()

## Protected Member Functions

- void **drawLineTo** (const QPoint &endPoint)
- void **drawLineTo** (int width, bool erase, const QColor &color, const QPoint &start, const QPoint &end)
- void **drawText** (QPainter \*const p, const QPoint &corner, const QString &text)
- void **enterEvent** (QEnterEvent \*) override
- void **leaveEvent** (QEvent \*) override
- void **mouseMoveEvent** (QMouseEvent \*) override
- void **mousePressEvent** (QMouseEvent \*) override
- void **mouseReleaseEvent** (QMouseEvent \*) override
- void **paintEvent** (QPaintEvent \*) override
- void **resizeEvent** (QResizeEvent \*) override
- void **setSpotPosition** (const QPoint &point)
- void **timerEvent** (QTimerEvent \*) override
- QPoint **translateItemPosition** (const QPoint &point, bool src) const
- QPoint **translatePointPosition** (const QPoint &point) const
- void **updateMaskCursor** ()
- void **updatePixmap** ()
- void **updateSpotPosition** (int x, int y)

## 9.719 Digikam::ImageHistogram Class Reference

Inheritance diagram for Digikam::ImageHistogram:



### Signals

- void **calculationAboutToStart** ()  
when calculation in thread is initiated, from other thread

- void **calculationFinished** (bool success)
- void **calculationStarted** ()  
*emitted from calculation thread*

## Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Public Member Functions

- **ImageHistogram** (const [DImg](#) &img, QObject \*const parent=nullptr)
- void **calculate** ()  
*Started computation: synchronous or threaded.*
- void **calculateInThread** ()
- double **getCount** (int channel, int start, int end) const
- int **getHistogramSegments** () const
- double **getMaximum** (int channel, int start, int end) const
- int **getMaxSegmentIndex** () const
- double **getMean** (int channel, int start, int end) const
- int **getMedian** (int channel, int start, int end) const
- double **getPixels** () const
- double **getStdDev** (int channel, int start, int end) const
- double **getValue** (int channel, int bin) const
- bool **isCalculating** () const
- bool **isSixteenBit** () const  
*Methods to access the histogram data.*
- bool **isValid** () const
- void **stopCalculation** ()  
*Stop threaded computation.*

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread](#) (QObject \*const parent=nullptr)  
*This class extends QRunnable, so you have to reimplement virtual void [run\(\)](#).*
- [~DynamicThread](#) () override  
*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool **isFinished** () const
- bool **isRunning** () const
- QThread::Priority **priority** () const
- void **setEmitSignals** (bool emitThem)
- void **setPriority** (QThread::Priority priority)  
*Sets the priority for this dynamic thread.*
- State **state** () const

## Protected Member Functions

- void **run** () override  
*Implement this pure virtual function in your subclass.*

## Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool **runningFlag** () const volatile  
*In you [run\(\)](#) method, you shall regularly check for [runningFlag\(\)](#) and cleanup and return if false.*
- void **shutDown** ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call [stop\(\)](#) and [wait\(\)](#), knowing that nothing will call [start\(\)](#) anymore after this 3) Be sure the thread will never be running at destruction.*
- void **start** (QMutexLocker< QMutex > &locker)  
*Doing the same as [start\(\)](#), [stop\(\)](#) and [wait](#) above, provide it with a locked QMutexLocker on [mutex\(\)](#).*
- void **stop** (const QMutexLocker< QMutex > &locker)
- QMutex \* **threadMutex** () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void **wait** (QMutexLocker< QMutex > &locker)

## Additional Inherited Members

## Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

## Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

## 9.719.1 Member Function Documentation

### 9.719.1.1 [run\(\)](#)

```
void Digikam::ImageHistogram::run ( ) [override], [protected], [virtual]
```

Implements [Digikam::DynamicThread](#).

## 9.720 [Digikam::ImageHistoryEntry](#) Class Reference

### Public Member Functions

- bool **isNull** () const

### Public Attributes

- QString **history**
- qlonglong **imageId** = 0
- QString **uuid**

## 9.721 Digikam::Imagelface Class Reference

### Public Types

- enum [PreviewType](#) { [FullImage](#) , [ImageSelection](#) }

### Public Member Functions

- [Imagelface](#) (const QSize &size=QSize(0, 0))  
*Standard constructor.*
- [DColor](#) **colorInfoFromOriginal** (const QPoint &point) const  
*Get colors from original, (unchanged) preview or target preview (set by setPreviewImage) image.*
- [DColor](#) **colorInfoFromPreview** (const QPoint &point) const
- [DColor](#) **colorInfoFromTargetPreview** (const QPoint &point) const
- void **convertOriginalColorDepth** (int depth)  
*Convert depth of original image.*
- QPixmap **convertToPixmap** (const [DImg](#) &img) const  
*Convert a [DImg](#) image to a pixmap for screen using color managed view if necessary.*
- void **crop** (const QRect &region)  
*Crop the original image currently hosted by editor to the region.*
- [DImg](#) \* **original** () const  
*Return a pointer to the [DImg](#) object representing the original image.*
- bool **originalHasAlpha** () const
- [IccProfile](#) **originalIccProfile** () const  
*Original image meta-data management methods.*
- [MetaEngineData](#) **originalMetadata** () const
- [PhotoInfoContainer](#) **originalPhotoInfo** () const
- bool **originalSixteenBit** () const
- QSize **originalSize** () const  
*Methods to get/set original image information.*
- void **paint** (QPaintDevice \*const device, const QRect &rect, QPainter \*const painter=nullptr)  
*Paint the current target preview image (or the preview image, if setPreview has not been called) on the given paint device.*
- [DImg](#) **preview** () const  
*Return a [DImg](#) object representing the preview image.*
- bool **previewHasAlpha** () const
- [DImg](#) \* **previewReference** ()  
*Return a pointer to the [DImg](#) object representing the preview image.*
- bool **previewSixteenBit** () const
- QSize **previewSize** () const  
*Methods to get/set preview image information.*
- [PreviewType](#) **previewType** () const
- [DImg](#) **selection** () const  
*Return a [DImg](#) object representing the current original image selection.*
- QRect **selectionRect** () const  
*Return current image selection position and size into original image coordinates.*
- void **setOriginal** (const QString &caller, const [FilterAction](#) &action, const [DImg](#) &img)  
*Replace the data of the original with the given image.*
- void **setOriginalIccProfile** (const [IccProfile](#) &profile)  
*Set the color profile of the original image.*
- void **setOriginalMetadata** (const [MetaEngineData](#) &meta)

- void `setPreview` (const `DImg` &img)  
*Replace the stored target preview with the given image.*
- void `setPreviewIccProfile` (const `IccProfile` &profile)  
*Set the color profile of the preview image.*
- `DImg` `setPreviewSize` (const `QSize` &size) const  
*Sets preview size and returns new preview as with `getPreview`.*
- void `setPreviewType` (`PreviewType` type=`FullImage`)  
*If `useSelection` is true, preview will be rendered using current selection in editor instead the full image.*
- void `setSelection` (const `QString` &caller, const `FilterAction` &action, const `DImg` &img)  
*Replace the data of the current original image selection with the given data.*

## 9.721.1 Member Enumeration Documentation

### 9.721.1.1 PreviewType

```
enum Digikam::ImageIface::PreviewType
```

Enumerator

FullImage	Preview will be rendered using full image.
ImageSelection	Preview will be rendered using current selection from editor canvas.

## 9.721.2 Constructor & Destructor Documentation

### 9.721.2.1 ImageIface()

```
Digikam::ImageIface::ImageIface (
    const QSize & size = QSize(0, 0) ) [explicit]
```

Size is the constrain dimension of preview. This can be null size.

## 9.721.3 Member Function Documentation

### 9.721.3.1 original()

```
DImg * Digikam::ImageIface::original ( ) const
```

This object may not be modified or stored. Make copies if you need.

### 9.721.3.2 paint()

```
void Digikam::ImageIface::paint (
    QPaintDevice *const device,
    const QRect & rect,
    QPainter *const painter = nullptr )
```

at x|y, with given maximum width and height taken from rectangle rect.



### 9.721.3.3 previewReference()

```
DImg * Digikam::ImageIface::previewReference ( )
```

This function is a backdoor for preview editing.

### 9.721.3.4 setOriginal()

```
void Digikam::ImageIface::setOriginal (
    const QString & caller,
    const FilterAction & action,
    const DImg & img )
```

The characteristics of the data must match the characteristics of the original image as returned by the original...() methods, The size of image can be changed. Caller is an i18n'ed string that will be shown as the undo/redo action name.

### 9.721.3.5 setPreview()

```
void Digikam::ImageIface::setPreview (
    const DImg & img )
```

The characteristics of the data must match the characteristics of the current as returned by the preview...() methods. The target preview image is used by the paint() and colorInfoFromTargetPreview() methods. The image returned by getPreview() is unaffected.

### 9.721.3.6 setPreviewSize()

```
DImg Digikam::ImageIface::setPreviewSize (
    const QSize & size ) const
```

The parameters are only hints, previewSize() may differ from size.

### 9.721.3.7 setPreviewType()

```
void Digikam::ImageIface::setPreviewType (
    PreviewType type = FullImage )
```

Default preview is FullImage.

### 9.721.3.8 setSelection()

```
void Digikam::ImageIface::setSelection (
    const QString & caller,
    const FilterAction & action,
    const DImg & img )
```

The characteristics of the data must match the characteristics of the current selection as returned by the selection←→Width(), selectionHeight(), originalSixteenBit() and originalHasAlpha() methods. Caller is an i18n'ed string that will be shown as the undo/redo action name.

## 9.722 Digikam::ImageLevels Class Reference

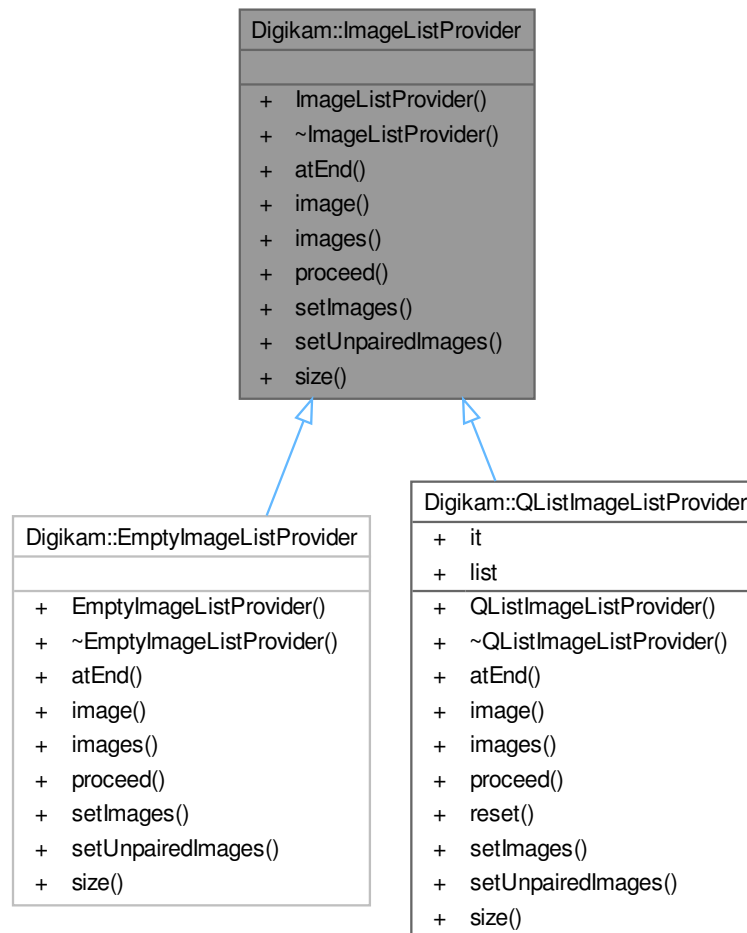
### Public Member Functions

- **ImageLevels** (bool sixteenBit)
- double **getLevelGammaValue** (int channel)
- int **getLevelHighInputValue** (int channel)
- int **getLevelHighOutputValue** (int channel)
- int **getLevelLowInputValue** (int channel)
- int **getLevelLowOutputValue** (int channel)
- bool **isDirty** ()
- bool **isSixteenBits** ()
- void **levelsAuto** (const [ImageHistogram](#) \*const hist)
- void **levelsBlackToneAdjustByColors** (int channel, const [DColor](#) &color)
- void **levelsCalculateTransfers** ()
- void **levelsChannelAuto** (const [ImageHistogram](#) \*const hist, int channel)
- void **levelsChannelReset** (int channel)
  - Methods to manipulate the levels data.*
- void **levelsGrayToneAdjustByColors** (int channel, const [DColor](#) &color)
- int **levelsInputFromColor** (int channel, const [DColor](#) &color)
- float **levelsLutFunc** (int nchannels, int channel, float value)
- void **levelsLutProcess** (uchar \*const srcPR, uchar \*const destPR, uint w, uint h)
- void **levelsLutSetup** (int nchannels)
- void **levelsWhiteToneAdjustByColors** (int channel, const [DColor](#) &color)
- bool **loadLevelsFromGimpLevelsFile** (const [QUrl](#) &fileUrl)
- void **reset** ()
- bool **saveLevelsToGimpLevelsFile** (const [QUrl](#) &fileUrl)
  - Methods to save/load the levels values to/from a Gimp levels text file.*
- void **setLevelGammaValue** (int channel, double val)
  - Methods to set manually the levels values.*
- void **setLevelHighInputValue** (int channel, int val)
- void **setLevelHighOutputValue** (int channel, int val)
- void **setLevelLowInputValue** (int channel, int val)
- void **setLevelLowOutputValue** (int channel, int val)

## 9.723 Digikam::ImageListProvider Class Reference

This class provides access to a list of unspecified entities, where for each entry a QImage can be provided.

Inheritance diagram for Digikam::ImageListProvider:



## Public Member Functions

- virtual bool **atEnd** () const =0
- virtual QPair< QImage \*, QString > **image** ()=0
- virtual QList< QPair< QImage \*, QString > > **images** ()=0
- virtual void **proceed** (int steps=1)=0
- virtual void **setImages** (const QList< QPair< QImage \*, QString > > &)=0
- virtual void **setUnpairedImages** (const QList< QImage \* > &)=0
- virtual int **size** () const =0

### 9.723.1 Detailed Description

Only forward iteration is required.

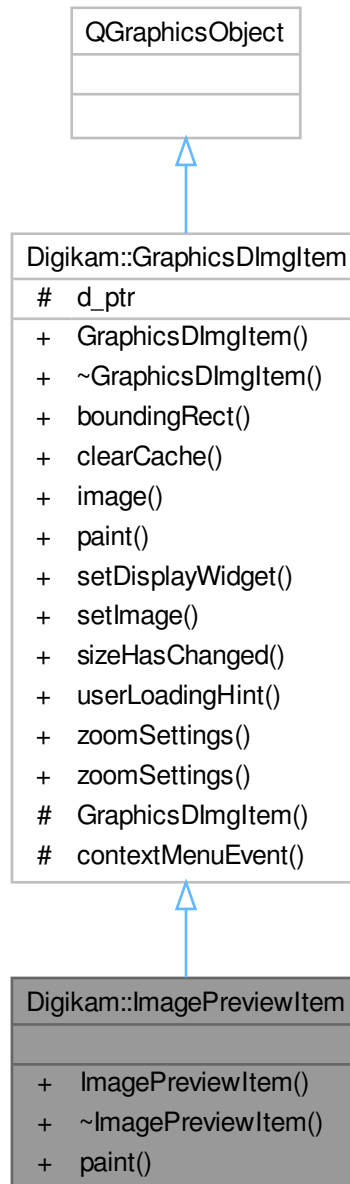
## 9.724 Digikam::ImageMetadataContainer Class Reference

### Public Attributes

- bool **allFieldsNull** = true
- QString **aperture**
- QString **exposureMode**
- QString **exposureProgram**
- QString **exposureTime**
- QString **flashMode**
- QString **focalLength**
- QString **focalLength35**
- QString **lens**
- QString **make**
- QString **meteringMode**
- QString **model**
- QString **sensitivity**
- QString **subjectDistance**
- QString **subjectDistanceCategory**
- QString **whiteBalance**
- QString **whiteBalanceColorTemperature**

## 9.725 Digikam::ImagePreviewItem Class Reference

Inheritance diagram for Digikam::ImagePreviewItem:



### Public Member Functions

- void **paint** (QPainter \*painter, const QStyleOptionGraphicsItem \*option, QWidget \*widget) override

## Public Member Functions inherited from [Digikam::GraphicsDImgItem](#)

- **GraphicsDImgItem** (QGraphicsItem \*const parent=nullptr)
- QRectF **boundingRect** () const override
- void **clearCache** ()
- [DImg](#) **image** () const
- void **paint** (QPainter \*painter, const QStyleOptionGraphicsItem \*option, QWidget \*widget) override
- void **setDisplayWidget** (QWidget \*const widget)
- void **setImage** (const [DImg](#) &img)
  - Sets the [DImg](#) to be drawn by this item.*
- void **sizeHasChanged** ()
- virtual QString **userLoadingHint** () const
- [ImageZoomSettings](#) \* **zoomSettings** ()
- const [ImageZoomSettings](#) \* **zoomSettings** () const

## Additional Inherited Members

## Signals inherited from [Digikam::GraphicsDImgItem](#)

- void **imageChanged** ()
- void **imageSizeChanged** (const QSizeF &size)
- void **showContextMenu** (QGraphicsSceneContextMenuEvent \*e)

## Protected Member Functions inherited from [Digikam::GraphicsDImgItem](#)

- **GraphicsDImgItem** (GraphicsDImgItemPrivate &dd, QGraphicsItem \*const parent)
- void **contextMenuEvent** (QGraphicsSceneContextMenuEvent \*e) override

## Protected Attributes inherited from [Digikam::GraphicsDImgItem](#)

- GraphicsDImgItemPrivate \*const **d\_ptr**

## 9.726 Digikam::ImageQualityCalculator Class Reference

### Classes

- struct [ResultDetection](#)

### Public Member Functions

- void **addDetectionResult** (const QString &name, const float score, const float weight) const
- float **calculateQuality** () const

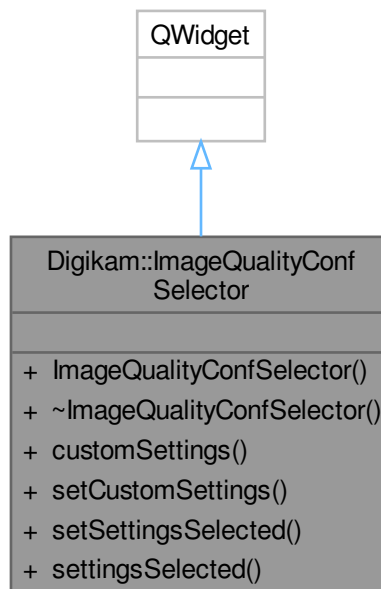
## 9.727 Digikam::ImageQualityCalculator::ResultDetection Struct Reference

### Public Attributes

- QString **detetionType**
- float **score**
- float **weight**

## 9.728 Digikam::ImageQualityConfSelector Class Reference

Inheritance diagram for Digikam::ImageQualityConfSelector:



### Public Types

- enum `SettingsType` { `GlobalSettings = 0` , `CustomSettings` }

### Signals

- void `signalQualitySetup ()`
- void `signalSettingsChanged ()`

## Public Member Functions

- **ImageQualityConfSelector** (QWidget \*const parent=nullptr)
- **ImageQualityContainer** **customSettings** () const
- void **setCustomSettings** (const **ImageQualityContainer** &settings)
- void **setSettingsSelected** (**SettingsType** type)
- **SettingsType** **settingsSelected** () const

## 9.728.1 Member Enumeration Documentation

### 9.728.1.1 SettingsType

```
enum Digikam::ImageQualityConfSelector::SettingsType
```

#### Enumerator

GlobalSettings	Global settings available in setup dialog.
CustomSettings	Settings customized by end-user.

## 9.729 Digikam::ImageQualityContainer Class Reference

### Public Member Functions

- **ImageQualityContainer** (const **ImageQualityContainer** &other)
- **ImageQualityContainer** & **operator=** (const **ImageQualityContainer** &other)
- void **readFromConfig** ()
- void **readFromConfig** (const KConfigGroup &)
- void **writeToConfig** ()
- void **writeToConfig** (KConfigGroup &)

### Public Attributes

- int **acceptedThreshold**  
*Item accepted threshold.*
- int **blurWeight**  
*Item blur level.*
- int **compressionWeight**  
*Item compression level.*
- bool **detectAesthetic**  
*Enable image aesthetic detection.*
- bool **detectBlur**  
*Enable image blur detection.*
- bool **detectCompression**  
*Enable image compression detection.*
- bool **detectExposure**  
*Enable image over and under exposure detection.*
- bool **detectNoise**

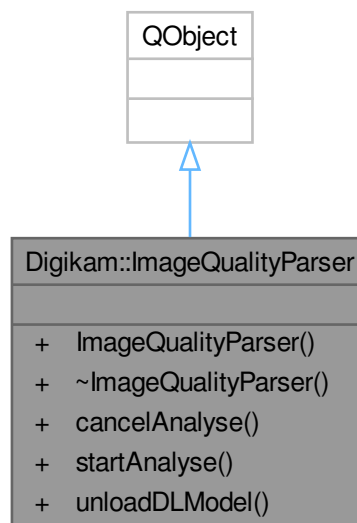


- Enable image noise detection.*

  - int **exposureWeight**  
*Item exposure level.*
  - bool **highQAccepted**  
*Assign Accepted property to high quality.*
  - bool **lowQRejected**  
*Assign Rejected property to low quality.*
  - bool **mediumQPending**  
*Assign Pending property to medium quality.*
  - int **noiseWeight**  
*Item noise level.*
  - int **pendingThreshold**  
*Item pending threshold.*
  - int **rejectedThreshold**  
*Item rejection threshold.*

## 9.730 Digikam::ImageQualityParser Class Reference

Inheritance diagram for Digikam::ImageQualityParser:



### Public Member Functions

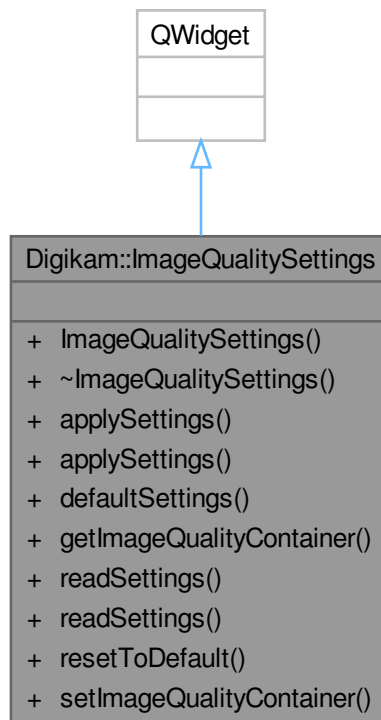
- **ImageQualityParser** (const [DImg](#) &image, const [ImageQualityContainer](#) &settings, PickLabel \*const label)  
*Standard constructor with picklabel container to fill at end of analyze.*
- void **cancelAnalyse** ()
- void **startAnalyse** ()  
*Perform quality estimation and fill Pick Label value accordingly.*

### Static Public Member Functions

- static void **unloadDLModel** ()

## 9.731 Digikam::ImageQualitySettings Class Reference

Inheritance diagram for Digikam::ImageQualitySettings:



### Signals

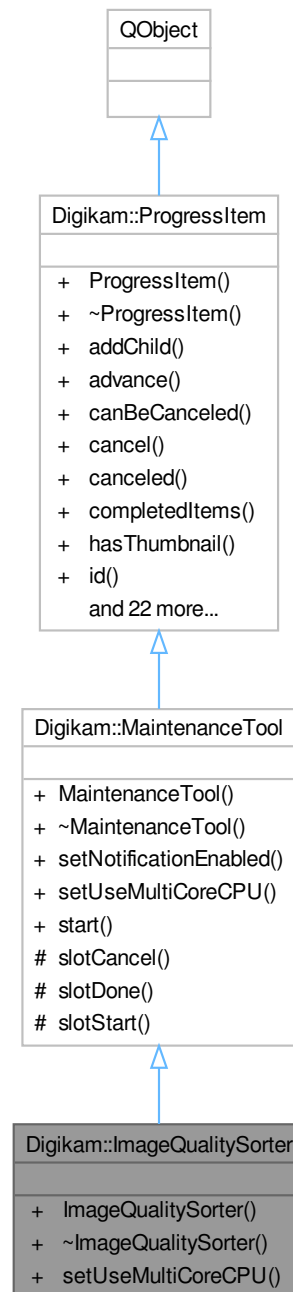
- void **signalSettingsChanged** ()

### Public Member Functions

- **ImageQualitySettings** (`QWidget *const parent=nullptr`)
- void **applySettings** ()
- void **applySettings** (`KConfigGroup &`)
- `ImageQualityContainer` **defaultSettings** () const
- `ImageQualityContainer` **getImageQualityContainer** () const
- void **readSettings** ()
- void **readSettings** (`const KConfigGroup &`)
- void **resetToDefault** ()
- void **setImageQualityContainer** (`const ImageQualityContainer &imq`)

## 9.732 Digikam::ImageQualitySorter Class Reference

Inheritance diagram for Digikam::ImageQualitySorter:



### Public Types

- enum `QualityScanMode` { `AllItems` = 0 , `NonAssignedItems` }

### Public Member Functions

- [ImageQualitySorter](#) ([QualityScanMode](#) mode, const [AlbumList](#) &list=[AlbumList](#)(), const [ImageQualityContainer](#) &quality=[ImageQualityContainer](#)(), [ProgressItem](#) \*const parent=nullptr)  
*Constructor using AlbumList as argument.*
- void [setUseMultiCoreCPU](#) (bool b) override  
*Re-implement this method if your tool is able to use multi-core CPU to process item in parallel.*

### Public Member Functions inherited from [Digikam::MaintenanceTool](#)

- [MaintenanceTool](#) (const [QString](#) &id, [ProgressItem](#) \*const parent=nullptr)
- void [setNotificationEnabled](#) (bool b)  
*If true, show a notification message on desktop notification manager with time elapsed to run process.*

### Public Member Functions inherited from [Digikam::ProgressItem](#)

- [ProgressItem](#) ([ProgressItem](#) \*const parent, const [QString](#) &id, const [QString](#) &label, const [QString](#) &status, bool canBeCanceled, bool hasThumb)
- void [addChild](#) ([ProgressItem](#) \*const kiddo)
- bool [advance](#) (unsigned int v)  
*Advance total items processed by n values and update percentage in progressbar.*
- bool [canBeCanceled](#) () const
- void [cancel](#) ()
- bool [canceled](#) () const
- unsigned int [completedItems](#) () const
- bool [hasThumbnail](#) () const
- const [QString](#) & [id](#) () const
- bool [incCompletedItems](#) (unsigned int v=1)
- void [incTotalItems](#) (unsigned int v=1)
- const [QString](#) & [label](#) () const
- [ProgressItem](#) \* [parent](#) () const
- unsigned int [progress](#) () const
- void [removeChild](#) ([ProgressItem](#) \*const kiddo)
- void [reset](#) ()  
*Reset the progress value of this item to 0 and the status string to the empty string.*
- void [setComplete](#) ()  
*Tell the item it has finished.*
- bool [setCompletedItems](#) (unsigned int v)
- void [setLabel](#) (const [QString](#) &v)
- void [setProgress](#) (unsigned int v)  
*Set the progress (percentage of completion) value of this item.*
- void [setShowAtStart](#) (bool showAtStart)  
*Set the property to pop-up item when it's added in progress manager.*
- void [setStatus](#) (const [QString](#) &v)  
*Set the string to be used for showing this item's current status.*
- void [setThumbnail](#) (const [QIcon](#) &icon)  
*Sets whether this item has a thumbnail.*
- void [setTotalItems](#) (unsigned int v)
- void [setUsesBusyIndicator](#) (bool useBusyIndicator)  
*Sets whether this item uses a busy indicator instead of real progress for its progress bar.*
- bool [showAtStart](#) () const
- const [QString](#) & [status](#) () const
- bool [totalCompleted](#) () const
- unsigned int [totalItems](#) () const
- void [updateProgress](#) ()  
*Recalculate progress according to total/completed items and update.*
- bool [usesBusyIndicator](#) () const

## Additional Inherited Members

### Public Slots inherited from [Digikam::MaintenanceTool](#)

- void **start** ()

### Signals inherited from [Digikam::MaintenanceTool](#)

- void **signalCanceled** ()  
*Emit when process is canceled.*
- void **signalComplete** ()  
*Emit when process is done (not canceled).*

### Signals inherited from [Digikam::ProgressItem](#)

- void [progressItemAdded](#) ([ProgressItem](#) \*item)  
*Emitted when a new [ProgressItem](#) is added.*
- void [progressItemCanceled](#) ([ProgressItem](#) \*item)  
*Emitted when an item was canceled.*
- void **progressItemCanceledById** (const QString &id)
- void [progressItemCompleted](#) ([ProgressItem](#) \*item)  
*Emitted when a progress item was completed.*
- void [progressItemLabel](#) ([ProgressItem](#) \*item, const QString &label)  
*Emitted when the label of an item changed.*
- void [progressItemProgress](#) ([ProgressItem](#) \*item, unsigned int v)  
*Emitted when the progress value of an item changes.*
- void [progressItemStatus](#) ([ProgressItem](#) \*item, const QString &mess)  
*Emitted when the status message of an item changed.*
- void [progressItemThumbnail](#) ([ProgressItem](#) \*item, const QPixmap &thumb)  
*Emitted when the thumbnail data must be set in item.*
- void [progressItemUsesBusyIndicator](#) ([ProgressItem](#) \*item, bool value)  
*Emitted when the busy indicator state of an item changes.*

### Protected Slots inherited from [Digikam::MaintenanceTool](#)

- virtual void **slotCancel** ()
- virtual void **slotDone** ()
- virtual void **slotStart** ()

## 9.732.1 Member Enumeration Documentation

### 9.732.1.1 QualityScanMode

```
enum Digikam::ImageQualitySorter::QualityScanMode
```

## Enumerator

AllItems	Clean all Pick Labels assignments and re-scan all items.
NonAssignedItems	Scan only items with no Pick Labels assigned.

## 9.732.2 Constructor & Destructor Documentation

### 9.732.2.1 ImageQualitySorter()

```
Digikam::ImageQualitySorter::ImageQualitySorter (
    QualityScanMode mode,
    const AlbumList & list = AlbumList(),
    const ImageQualityContainer & quality = ImageQualityContainer(),
    ProgressItem *const parent = nullptr ) [explicit]
```

If list is empty, whole Albums collection is processed.

## 9.732.3 Member Function Documentation

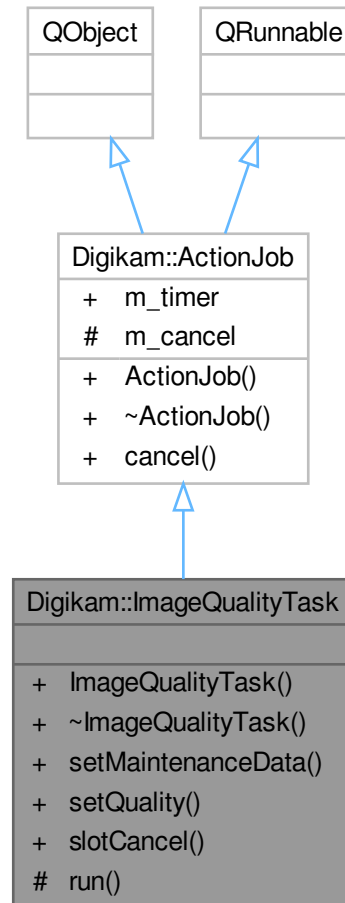
### 9.732.3.1 setUseMultiCoreCPU()

```
void Digikam::ImageQualitySorter::setUseMultiCoreCPU (
    bool ) [override], [virtual]
```

Reimplemented from [Digikam::MaintenanceTool](#).

## 9.733 Digikam::ImageQualityTask Class Reference

Inheritance diagram for Digikam::ImageQualityTask:



### Public Slots

- void **slotCancel** ()

### Public Slots inherited from [Digikam::ActionJob](#)

- void **cancel** ()  
*Call this method to cancel job.*

### Signals

- void **signalFinished** (const [ItemInfo](#) &, const QImage &, int)

## Signals inherited from [Digikam::ActionJob](#)

- void **signalDone** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job is done.*
- void **signalProgress** (int)  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job progress.*
- void **signalStarted** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager that job is started.*

## Public Member Functions

- void **setMaintenanceData** ([MaintenanceData](#) \*const data=nullptr)
- void **setQuality** (const [ImageQualityContainer](#) &quality)

## Public Member Functions inherited from [Digikam::ActionJob](#)

- **ActionJob** (QObject \*const parent=nullptr)  
*Constructor which delegate deletion of QRunnable instance to [ActionThreadBase](#), not QThreadPool.*
- **~ActionJob** () override  
*Re-implement destructor in you implementation.*

## Protected Member Functions

- void **run** () override

## Additional Inherited Members

## Public Attributes inherited from [Digikam::ActionJob](#)

- QElapsedTimer **m\_timer**  
*Timer to determine the running time of the job.*

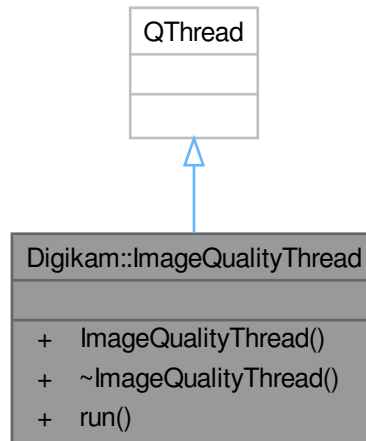
## Protected Attributes inherited from [Digikam::ActionJob](#)

- bool **m\_cancel** = false  
*You can use this boolean in your implementation to know if job must be canceled.*



## 9.734 Digikam::ImageQualityThread Class Reference

Inheritance diagram for Digikam::ImageQualityThread:

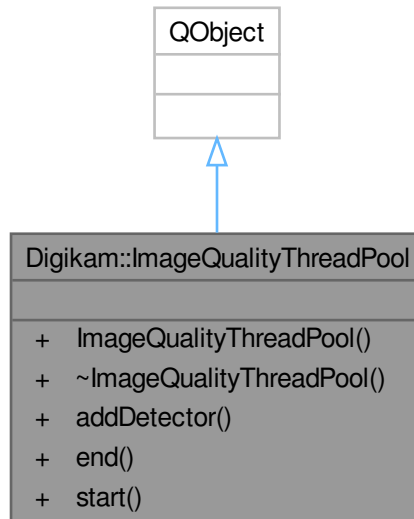


### Public Member Functions

- **ImageQualityThread** (`QObject *const parent`, `AbstractDetector *const detector`, `const cv::Mat &image`, `ImageQualityCalculator *const calculator`, `float weight_quality`)
- `void run ()` override

## 9.735 Digikam::ImageQualityThreadPool Class Reference

Inheritance diagram for Digikam::ImageQualityThreadPool:

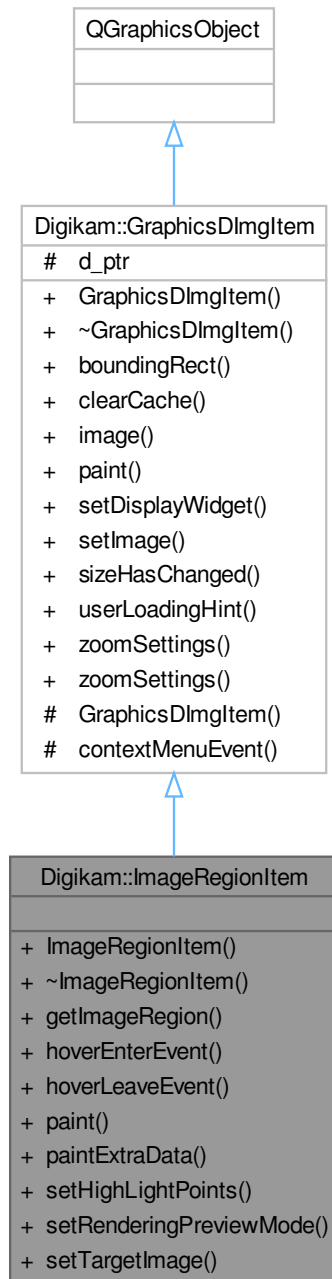


### Public Member Functions

- **ImageQualityThreadPool** (`QObject *const parent`, `ImageQualityCalculator *const calculator`)
- void **addDetector** (`const cv::Mat &image`, `float weight_quality`, `AbstractDetector *const detector`)
- void **end** ()
- void **start** ()

## 9.736 Digikam::ImageRegionItem Class Reference

Inheritance diagram for Digikam::ImageRegionItem:



### Public Member Functions

- **ImageRegionItem** ([ImageRegionWidget](#) \*const view, bool paintExtras=true)
- `QRect` **getImageRegion** () const

- void **hoverEnterEvent** (QGraphicsSceneHoverEvent \*) override
- void **hoverLeaveEvent** (QGraphicsSceneHoverEvent \*) override
- void **paint** (QPainter \*painter, const QStyleOptionGraphicsItem \*option, QWidget \*widget) override
- void **paintExtraData** (QPainter \*const painter)
- void **setHighLightPoints** (const QPolygon &pointsList)
- void **setRenderingPreviewMode** (int mode)
- void **setTargetImage** (const [DImg](#) &img)

### Public Member Functions inherited from [Digikam::GraphicsDImgItem](#)

- **GraphicsDImgItem** (QGraphicsItem \*const parent=nullptr)
- QRectF **boundingRect** () const override
- void **clearCache** ()
- [DImg](#) **image** () const
- void **paint** (QPainter \*painter, const QStyleOptionGraphicsItem \*option, QWidget \*widget) override
- void **setDisplayWidget** (QWidget \*const widget)
- void **setImage** (const [DImg](#) &img)
  - *Sets the [DImg](#) to be drawn by this item.*
- void **sizeHasChanged** ()
- virtual QString **userLoadingHint** () const
- [ImageZoomSettings](#) \* **zoomSettings** ()
- const [ImageZoomSettings](#) \* **zoomSettings** () const

### Additional Inherited Members

### Signals inherited from [Digikam::GraphicsDImgItem](#)

- void **imageChanged** ()
- void **imageSizeChanged** (const QSizeF &size)
- void **showContextMenu** (QGraphicsSceneContextMenuEvent \*e)

### Protected Member Functions inherited from [Digikam::GraphicsDImgItem](#)

- **GraphicsDImgItem** (GraphicsDImgItemPrivate &dd, QGraphicsItem \*const parent)
- void **contextMenuEvent** (QGraphicsSceneContextMenuEvent \*e) override

### Protected Attributes inherited from [Digikam::GraphicsDImgItem](#)

- GraphicsDImgItemPrivate \*const **d\_ptr**

## 9.737 Digikam::ImageRegionWidget Class Reference

Inheritance diagram for Digikam::ImageRegionWidget:



### Public Slots

- void **slotOriginalImageRegionChanged** (bool targetDone=true)
- void **slotPreviewModeChanged** (int mode)

## Signals

- void **signalCapturedPointFromOriginal** (const [Digikam::DColor](#) &, const QPoint &)
- void **signalOriginalClipFocusChanged** ()

## Signals inherited from [Digikam::GraphicsDImgView](#)

- void **activated** ()
- void **contentsMoved** (bool panningFinished)
- void **contentsMoving** (int, int)
- void **leftButtonClicked** ()
- void **leftButtonDoubleClicked** ()
- void **resized** ()
- void **rightButtonClicked** ()
- void **toNextImage** ()
- void **toPreviousImage** ()
- void **viewportRectChanged** (const QRectF &viewportRect)

## Public Member Functions

- **ImageRegionWidget** (QWidget \*const parent=nullptr, bool paintExtras=true)
- bool **capturePointMode** () const
- void **exposureSettingsChanged** ()
- [DImg](#) **getOriginalImage** () const
- QRect **getOriginalImageRegionToRender** () const  
*To get target image region area to render.*
- [DImg](#) **getOriginalRegionImage** (bool useDownscaledImage=false) const  
*To get target image region image to use for render operations If the bool parameter is true a downscaled version of the image region at screen resolution will be sent.*
- void **ICCSettingsChanged** ()
- void **setCapturePointMode** (bool b)
- void **setHighLightPoints** (const QPolygon &pointsList)
- void **setPreviewImage** (const [DImg](#) &img)
- void **updateImage** (const [DImg](#) &img)

## Public Member Functions inherited from [Digikam::GraphicsDImgView](#)

- **GraphicsDImgView** (QWidget \*const parent=nullptr)
- int **contentsX** () const
- int **contentsY** () const
- void **drawText** (QPainter \*p, const QRectF &rect, const QString &text)
- void **fitToWindow** ()
- [GraphicsDImgItem](#) \* **item** () const  
*Return the instance of item set by [setItem\(\)](#).*
- [SinglePhotoPreviewLayout](#) \* **layout** () const
- [DImgPreviewItem](#) \* **previewItem** () const  
*Return a cast of item instance of item set by [setItem\(\)](#) as [DImgPreviewItem](#) Note: if you store a [GraphicsDImgItem](#) object using [setItem\(\)](#), this method will return 0.*
- void **scrollPointOnPoint** (const QPointF &scenePos, const QPoint &viewportPos)  
*Scrolls the view such that scenePos (in scene coordinates is displayed on the viewport at viewportPos (in viewport coordinates).*
- void **setContentsPos** (int x, int y)
- void **setItem** ([GraphicsDImgItem](#) \*const item)  
*Store internal instance of item as [GraphicsDImgItem](#).*
- void **toggleFullScreen** (bool set)
- QRect **visibleArea** () const

### Protected Member Functions

- void **mousePressEvent** (QMouseEvent \*) override
- void **mouseReleaseEvent** (QMouseEvent \*) override

### Protected Member Functions inherited from [Digikam::GraphicsDImgView](#)

- virtual bool **acceptsMouseClicked** (QMouseEvent \*e)
- void **continuePanning** (const QPoint &pos)
- void **drawForeground** (QPainter \*painter, const QRectF &rect) override
- void **finishPanning** ()
- void **installPanIcon** ()
- void **mouseDoubleClickEvent** (QMouseEvent \*) override
- void **mouseMoveEvent** (QMouseEvent \*) override
- void **mousePressEvent** (QMouseEvent \*) override
- void **mouseReleaseEvent** (QMouseEvent \*) override
- void **resizeEvent** (QResizeEvent \*) override
- void **scrollContentsBy** (int dx, int dy) override
- void **setScaleFitToWindow** (bool value)
- void **setShowText** (bool value)
- void **startPanning** (const QPoint &pos)
- void **wheelEvent** (QWheelEvent \*) override

### Additional Inherited Members

### Protected Slots inherited from [Digikam::GraphicsDImgView](#)

- void **slotContentsMoved** ()
- void **slotCornerButtonPressed** ()
- void **slotPanIconHidden** ()
- virtual void **slotPanIconSelectionMoved** (const QRect &, bool)

## 9.737.1 Member Function Documentation

### 9.737.1.1 `getOriginalRegionImage()`

```
DImg Digikam::ImageRegionWidget::getOriginalRegionImage (
    bool useDownscaledImage = false ) const
```

Should be use to increase preview speed for the effects whose behaviour is a function of each pixel.

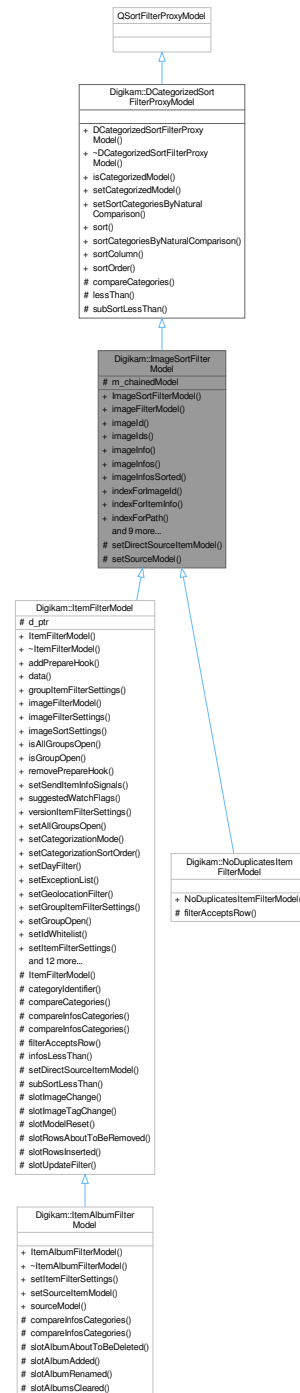
## 9.738 Digikam::ImageRelation Class Reference

### Public Attributes

- qlonglong **objectId** = 0
- qlonglong **subjectId** = 0
- DatabaseRelation::Type **type** = DatabaseRelation::UndefinedType

## 9.739 Digikam::ImageSortFilterModel Class Reference

Inheritance diagram for Digikam::ImageSortFilterModel:



### Public Member Functions

- **ImageSortFilterModel** (QObject \*const parent=nullptr)
- virtual **ItemFilterModel** \* **imageFilterModel** () const



Returns this, any chained [ItemFilterModel](#), or 0.

- qlonglong **imageId** (const QModelIndex &index) const
- QList< qlonglong > **imageIds** (const QList< QModelIndex > &indexes) const
- [ItemInfo](#) **imageInfo** (const QModelIndex &index) const
- QList< [ItemInfo](#) > **imageInfos** (const QList< QModelIndex > &indexes) const
- QList< [ItemInfo](#) > **imageInfosSorted** () const

Returns a list of all image infos, sorted according to this model.

- QModelIndex **indexForImageId** (qlonglong id) const
- QModelIndex **indexForItemInfo** (const [ItemInfo](#) &info) const
- QModelIndex **indexForPath** (const QString &filePath) const
- QModelIndex **mapFromDirectSourceToSourceItemModel** (const QModelIndex &sourceModel\_index) const
- QModelIndex **mapFromSourceItemModel** (const QModelIndex &imagemodel\_index) const
- QList< QModelIndex > **mapListFromSource** (const QList< QModelIndex > &sourceIndexes) const
- QList< QModelIndex > **mapListToSource** (const QList< QModelIndex > &indexes) const

Convenience methods mapped to [ItemModel](#).

- QModelIndex **mapToSourceItemModel** (const QModelIndex &index) const
- void **setSourceFilterModel** ([ImageSortFilterModel](#) \*const model)
- void **setSourceItemModel** ([ItemModel](#) \*const model)
- [ImageSortFilterModel](#) \* **sourceFilterModel** () const
- [ItemModel](#) \* **sourceItemModel** () const

## Public Member Functions inherited from [Digikam::DCategorizedSortFilterProxyModel](#)

- [DCategorizedSortFilterProxyModel](#) (QObject \*const parent=nullptr)
- bool **isCategorizedModel** () const
- void **setCategorizedModel** (bool categorizedModel)
- void **setSortCategoriesByNaturalComparison** (bool [sortCategoriesByNaturalComparison](#))
- void **sort** (int column, Qt::SortOrder order=Qt::AscendingOrder) override
- bool **sortCategoriesByNaturalComparison** () const
- int **sortColumn** () const
- Qt::SortOrder **sortOrder** () const

Enables or disables the categorization feature.

Set if the sorting using [CategorySortRole](#) will use a natural comparison in the case that strings were returned.

Overridden from [QSortFilterProxyModel](#).

## Protected Member Functions

- virtual void **setDirectSourceItemModel** ([ItemModel](#) \*const model)
- void **setSourceModel** ([QAbstractItemModel](#) \*const model) override

Reimplement if needed.

## Protected Member Functions inherited from [Digikam::DCategorizedSortFilterProxyModel](#)

- virtual int **compareCategories** (const QModelIndex &left, const QModelIndex &right) const
- bool **lessThan** (const QModelIndex &left, const QModelIndex &right) const override
- virtual bool **subSortLessThan** (const QModelIndex &left, const QModelIndex &right) const

This method compares the category of the *left* index with the category of the *right* index.

Overridden from [QSortFilterProxyModel](#).

This method has a similar purpose as [lessThan\(\)](#) has on [QSortFilterProxyModel](#).

**Protected Attributes**

- [ImageSortFilterModel](#) \* `m_chainedModel` = nullptr

**Additional Inherited Members****Public Types inherited from [Digikam::DCategorizedSortFilterProxyModel](#)**

- enum [AdditionalRoles](#) { [CategoryDisplayRole](#) = 0x17CE990A , [CategorySortRole](#) = 0x27857E60 }

**9.739.1 Member Function Documentation****9.739.1.1 imageFilterModel()**

```
ItemFilterModel * Digikam::ImageSortFilterModel::imageFilterModel ( ) const [virtual]
```

Reimplemented in [Digikam::ItemFilterModel](#).

**9.739.1.2 imageInfosSorted()**

```
QList< ItemInfo > Digikam::ImageSortFilterModel::imageInfosSorted ( ) const
```

If you do not need a sorted list, use [ItemModel](#)'s `imageInfos()` method.

**9.739.1.3 mapListToSource()**

```
QList< QModelIndex > Digikam::ImageSortFilterModel::mapListToSource (
    const QList< QModelIndex > & indexes ) const
```

Mentioned indexes returned come from the source image model.

**9.739.1.4 setDirectSourceItemModel()**

```
void Digikam::ImageSortFilterModel::setDirectSourceItemModel (
    ItemModel *const model ) [protected], [virtual]
```

Called only when model shall be set as (direct) sourceModel.

Reimplemented in [Digikam::ItemFilterModel](#).

**9.739.1.5 setSourceModel()**

```
void Digikam::ImageSortFilterModel::setSourceModel (
    QAbstractItemModel *const model ) [override], [protected]
```

**Note**

made protected

## 9.740 Digikam::ImageTagChangeset Class Reference

### Public Types

- enum [Operation](#) {  
**Unknown** , **Added** , **Moved** , **Removed** ,  
**RemovedAll** , **PropertiesChanged** }

An [ImageTagChangeset](#) covers adding and removing the association of a tag with an image.

### Public Member Functions

- **ImageTagChangeset** (const QList< qlonglong > &ids, const QList< int > &tags, [Operation](#) operation)
- **ImageTagChangeset** (qlonglong id, const QList< int > &tags, [Operation](#) operation)
- **ImageTagChangeset** (qlonglong id, int tag, [Operation](#) operation)
- bool **containsImage** (qlonglong id) const
- bool **containsTag** (int id) const
- QList< qlonglong > **ids** () const
- [Operation](#) **operation** () const
- [ImageTagChangeset](#) & **operator<<** (const [ImageTagChangeset](#) &other)  
*Combines two ImageTagChangesets.*
- [ImageTagChangeset](#) & **operator<<** (const QDBusArgument &argument)
- const [ImageTagChangeset](#) & **operator>>** (QDBusArgument &argument) const
- bool **propertiesWereChanged** () const
- QList< int > **tags** () const
- bool **tagsWereAdded** () const
- bool **tagsWereRemoved** () const

### 9.740.1 Member Enumeration Documentation

#### 9.740.1.1 Operation

```
enum Digikam::ImageTagChangeset::Operation
```

It is described by a list of affected image ids, a list of affected tags, and an operation. There is no guarantee that information in the database has actually been changed.

### 9.740.2 Member Function Documentation

#### 9.740.2.1 operator<<()

```
ImageTagChangeset & Digikam::ImageTagChangeset::operator<< (  

    const ImageTagChangeset & other )
```

The operations shall not differ between the two sets; the operation is set to Unknown if it differs. This is especially not suitable for RemovedAll changesets.

## 9.741 Digikam::ImageTagProperty Class Reference

### Public Member Functions

- bool **isNull** () const

### Public Attributes

- qlonglong **imageId** = -1
- QString **property**
- int **tagId** = -1
- QString **value**

## 9.742 Digikam::ImageTagPropertyName Class Reference

### Static Public Member Functions

- static QLatin1String **autodetectedFace** ()
- static QLatin1String **autodetectedPerson** ()
- static QLatin1String **faceToTrain** ()
- static QLatin1String **ignoredFace** ()
- static QLatin1String **tagRegion** ()

## 9.743 Digikam::ImageWindow Class Reference

Inheritance diagram for Digikam::ImageWindow:



### Public Slots

- void **loadItemInfos** (const [ItemInfoList](#) &imageInfoList, const [ItemInfo](#) &imageInfoCurrent, const QString &caption)

- void **openImage** (const [ItemInfo](#) &info)
- void **slotAssignColorLabel** (int colorId)
- void **slotAssignPickLabel** (int pickId)
- void **slotAssignRating** (int rating)
- void **slotSetup** () override
- void **slotSetupChanged** ()
- void **slotSetupICC** () override

### Public Slots inherited from [Digikam::EditorWindow](#)

- void **slotSetup** () override=0
- virtual void **slotSetupICC** ()=0

### Signals

- void **loadCurrentLater** ()
- void **signalSavingDialogProgress** (float value)
- void **signalURLChanged** (const [QUrl](#) &url)

### Signals inherited from [Digikam::EditorWindow](#)

- void **signalNoCurrentItem** ()
- void **signalPreviewModeChanged** (int)
- void **signalSelectionChanged** (const [QRect](#) &)
- void **signalToolApplied** ()

### Public Member Functions

- [DInfoInterface](#) \* **infoface** ([DPluginAction](#) \*const ac) override  
*Return the interface instance to access to items information.*
- bool **queryClose** () override
- void **toggleTag** (int tagID)
- [VersionManager](#) \* **versionManager** () const override

### Public Member Functions inherited from [Digikam::EditorWindow](#)

- **EditorWindow** (const [QString](#) &name, [QWidget](#) \*const parent=nullptr)
- bool **actionEnabledState** () const
- void **loadTool** ([EditorTool](#) \*const tool)
- void **registerExtraPluginsActions** ([QString](#) &dom) override

## Public Member Functions inherited from Digikam::DXmlGuiWindow

- **DXmlGuiWindow** (QWidget \*const parent=nullptr, Qt::WindowFlags f=Qt::WindowFlags())
- QList< QAction \* > **allActions** () const  
*Return all actions from internal collection.*
- void **cleanupActions** ()  
*Cleanup unwanted actions from action collection.*
- QString **configGroupName** () const
- void **createFullScreenAction** (const QString &name)  
*Create Full-screen action to action collection instance from managed window set through setManagedWindow().*
- void **createHelpActions** (const QString &handbookSection, bool coreOptions=true)  
*Create common actions from Help menu for all digiKam main windows.*
- void **createSettingsActions** ()  
*Create common actions to setup all digiKam main windows.*
- void **createSidebarActions** ()  
*Create common actions to handle side-bar through keyboard shortcuts.*
- bool **fullScreensActive** () const  
*Return true if managed window is currently in Full Screen Mode.*
- void **readFullScreenSettings** (const KConfigGroup &group)  
*Read full-screen settings from KDE config file.*
- void **registerPluginsActions** ()  
*Register all generic plugins action to this instance.*
- void **setConfigGroupName** (const QString &name)  
*Manage config group name used by window instance to get/set settings from config file.*
- void **setFullScreenOptions** (int options)  
*Set full-screen options to managed window.*
- void **unminimizeAndActivateWindow** ()

## Static Public Member Functions

- static [ImageWindow](#) \* **imageWindow** ()
- static bool **imageWindowCreated** ()

## Static Public Member Functions inherited from Digikam::DXmlGuiWindow

- static QAction \* **buildStdAction** (StdActionType type, const QObject \*const recvr, const char \*const slot, QObject \*const parent)
- static QString **configFullScreenHideSideBarsEntry** ()
- static QString **configFullScreenHideStatusBarEntry** ()
- static QString **configFullScreenHideThumbBarEntry** ()
- static QString **configFullScreenHideToolBarsEntry** ()  
*Shared with [FullScreenSettings](#).*
- static void **restoreWindowSize** (QWindow \*const win, const KConfigGroup &group)
- static void **saveWindowSize** (QWindow \*const win, KConfigGroup &group)
- static void **setGoodDefaultWindowSize** (QWindow \*const win)
- static void **setupIconTheme** ()  
*If we have some local breeze icon resource, prefer it.*

### Protected Member Functions

- void **closeEvent** (QCloseEvent \*e) override
- void **dragMoveEvent** (QDragMoveEvent \*e) override
- void **dropEvent** (QDropEvent \*e) override
- void **showEvent** (QShowEvent \*e) override

### Protected Member Functions inherited from [Digikam::EditorWindow](#)

- void **addServicesMenuForUrl** (const QUrl &url)
  - void **applyColorManagementSettings** ()
  - void **applyIOSettings** ()
  - void **applyStandardSettings** ()
  - bool **checkOverwrite** (const QUrl &url)
  - bool **checkPermissions** (const QUrl &url)
  - void **colorManage** ()
  - [EditorStackView](#) \* **editorStackView** () const
  - void **execSavingProgressDialog** ()
  - [ExposureSettingsContainer](#) \* **exposureSettings** () const
  - virtual void **finishSaving** (bool success)
  - virtual void **moveFile** ()
  - bool **moveLocalFile** (const QString &src, const QString &dest)
  - void **movingSaveFileFinished** (bool successful)
  - void **openWith** (const QUrl &url, QAction \*action)
  - bool **promptForOverWrite** ()
  - bool **promptUserDelete** (const QUrl &url)
  - bool **promptUserSave** (const QUrl &url, SaveAskMode mode=AskIfNeeded, bool allowCancel=true)
  - virtual void **readSettings** ()
  - void **readStandardSettings** ()
  - void **resetOrigin** ()
  - void **resetOriginSwitchFile** ()
  - [VersionFileOperation](#) **saveAsVersionFileOperation** (const QUrl &url, const QUrl &saveLocation, const QString &format)
  - [VersionFileOperation](#) **saveInFormatVersionFileOperation** (const QUrl &url, const QString &format)
  - virtual void **saveSettings** ()
  - void **saveStandardSettings** ()
  - [VersionFileOperation](#) **saveVersionFileOperation** (const QUrl &url, bool fork)
  - void **setupContextMenu** ()
  - void **setupSelectToolsAction** ()
  - void **setupStandardActions** ()
  - void **setupStandardConnections** ()
  - void **setupStatusBar** ()
  - [SidebarSplitter](#) \* **sidebarSplitter** () const
  - void **startingSave** (const QUrl &url)
  - bool **startingSaveAs** (const QUrl &url)
  - bool **startingSaveCurrentVersion** (const QUrl &url)
  - bool **startingSaveNewVersion** (const QUrl &url)
  - bool **startingSaveNewVersionAs** (const QUrl &url)
  - bool **startingSaveNewVersionInFormat** (const QUrl &url, const QString &format)
  - virtual void **toggleActions** (bool val)
  - void **toggleNonDestructiveActions** ()
  - void **toggleStandardActions** (bool val)
  - void **toggleToolActions** ([EditorTool](#) \*tool=nullptr)
  - void **toggleZoomActions** (bool val)
- Method used by Editor Tools.*
- bool **waitForSavingToComplete** ()



## Protected Member Functions inherited from [Digikam::DXmlGuiWindow](#)

- void **closeEvent** (QCloseEvent \*e) override
- void **editKeyboardShortcuts** (KActionCollection \*const extraac=nullptr, const QString &actitle=QString())  
*Call this method from your main window to show keyboard shortcut config dialog with an extra action collection to configure.*
- bool **eventFilter** (QObject \*obj, QEvent \*ev) override
- void **keyPressEvent** (QKeyEvent \*e) override
- QAction \* **showMenuBarAction** () const
- QAction \* **showStatusBarAction** () const

## Additional Inherited Members

## Public Types inherited from [Digikam::EditorWindow](#)

- enum **TransformType** { RotateLeft , RotateRight , FlipHorizontal , FlipVertical }

## Static Public Attributes inherited from [Digikam::EditorWindow](#)

- static const QString **CONFIG\_GROUP\_NAME**

## Protected Types inherited from [Digikam::EditorWindow](#)

- enum **SaveAskMode** { AskIfNeeded , OverwriteWithoutAsking , AlwaysSaveAs , SaveVersionWithoutAsking = OverwriteWithoutAsking , AlwaysNewVersion = AlwaysSaveAs }

## Protected Slots inherited from [Digikam::EditorWindow](#)

- virtual bool **saveOrSaveAs** ()
- void **slotAboutToShowRedoMenu** ()
- void **slotAboutToShowUndoMenu** ()
- virtual void **slotAddedDroppedItems** (QDropEvent \*e)=0
- virtual void **slotBackward** ()=0
- virtual void **slotChanged** ()=0
- void **slotComponentsInfo** () override
- virtual void **slotContextMenu** ()=0
- virtual void **slotDeleteCurrentItem** ()=0
- virtual void **slotDiscardChanges** ()
- virtual void **slotFileOriginChanged** (const QString &filePath)
- virtual void **slotFileWithDefaultApplication** ()=0
- virtual void **slotFirst** ()=0
- virtual void **slotForward** ()=0
- virtual void **slotLast** ()=0
- virtual void **slotLoadingFinished** (const QString &filename, bool success)
- void **slotLoadingProgress** (const QString &filePath, float progress)
- virtual void **slotLoadingStarted** (const QString &filename)
- void **slotNameLabelCancelButtonPressed** ()
- virtual void **slotOpenOriginal** ()
- virtual void **slotOpenWith** (QAction \*action=nullptr)=0
- virtual void **slotPrepareToLoad** ()
- virtual void **slotRevert** ()=0
- void **slotSavingProgress** (const QString &filePath, float progress)
- virtual void **slotSavingStarted** (const QString &filename)
- void **slotSelected** (bool)
- virtual void **slotUpdateItemInfo** ()=0

### Protected Slots inherited from [Digikam::DXmlGuiWindow](#)

- bool `slotClose ()`

### Protected Attributes inherited from [Digikam::EditorWindow](#)

- bool `m_actionEnabledState` = false
- QAction \* `m_applyToolAction` = nullptr
- QAction \* `m_backwardAction` = nullptr
- QColor `m_bgColor`
- [Canvas](#) \* `m_canvas` = nullptr
- QAction \* `m_closeToolAction` = nullptr
- QMenu \* `m_contextMenu` = nullptr
- QAction \* `m_discardChangesAction` = nullptr
- bool `m_editingOriginalImage` = true
- QAction \* `m_exportAction` = nullptr
- QAction \* `m_fileDeleteAction` = nullptr
- QAction \* `m_firstAction` = nullptr
- QString `m_formatForRAWVersioning`
- QString `m_formatForSubversions`
- QAction \* `m_forwardAction` = nullptr
- [IOFileSettings](#) \* `m_IOFileSettings` = nullptr
- QAction \* `m_lastAction` = nullptr
- [StatusProgressBar](#) \* `m_nameLabel` = nullptr
- bool `m_nonDestructive` = true
- QAction \* `m_openVersionAction` = nullptr
- KToolBarPopupAction \* `m_redoAction` = nullptr
- [DAdjustableLabel](#) \* `m_resLabel` = nullptr
- QAction \* `m_revertAction` = nullptr
- QAction \* `m_saveAction` = nullptr
- QAction \* `m_saveAsAction` = nullptr
- QAction \* `m_saveCurrentVersionAction` = nullptr
- KToolBarPopupAction \* `m_saveNewVersionAction` = nullptr
- QAction \* `m_saveNewVersionAsAction` = nullptr
- QMenu \* `m_saveNewVersionInFormatAction` = nullptr
- [SavingContext](#) `m_savingContext`
- QPointer< QProgressDialog > `m_savingProgressDialog` = nullptr
- QAction \* `m_serviceAction` = nullptr
- QMenu \* `m_servicesMenu` = nullptr
- bool `m_setExifOrientationTag` = true
- QAction \* `m_showBarAction` = nullptr
- [SidebarSplitter](#) \* `m_splitter` = nullptr
- [EditorStackView](#) \* `m_stackView` = nullptr
- QVector< TransformType > `m_transformQue`
- KToolBarPopupAction \* `m_undoAction` = nullptr

### Protected Attributes inherited from [Digikam::DXmlGuiWindow](#)

- [DLogoAction](#) \* `m_animLogo` = nullptr

## 9.743.1 Member Function Documentation

### 9.743.1.1 infoIface()

```
DInfoInterface * Digikam::ImageWindow::infoIface (
    DPluginAction *const ac ) [override], [virtual]
```

Implements [Digikam::DXmlGuiWindow](#).

### 9.743.1.2 versionManager()

```
VersionManager * Digikam::ImageWindow::versionManager ( ) const [override], [virtual]
```

Reimplemented from [Digikam::EditorWindow](#).

## 9.744 Digikam::ImageZoomSettings Class Reference

### Public Types

- enum **FitToSizeMode** { **AlwaysFit** , **OnlyScaleDown** }

### Public Member Functions

- **ImageZoomSettings** (const QSize &imageSize, const QSize &originalSize=QSize())
- void **fitToSize** (const QSizeF &frameSize, FitToSizeMode=AlwaysFit)
 

*Sets the current zoom factor to the factor needed to fit the current (original) image size into the given view size.*
- double **fitToSizeZoomFactor** (const QSizeF &frameSize, FitToSizeMode=AlwaysFit) const
 

*Returns the zoom factor that would be used by [fitToSize\(\)](#) called with the given frameSize.*
- QSizeF **imageSize** () const
 

*Returns the (available) image size.*
- bool **isFitToSize** (const QSizeF &frameSize) const
- QPointF **mapImageToZoom** (const QPointF &imagePoint) const
 

*For a given point (in (0,0), [imageSize\(\)](#)) returns the corresponding point in (0,0),[zoomedSize\(\)](#).*
- QRectF **mapImageToZoom** (const QRectF &imagePoint) const
 

*For a given rect contained in ((0,0), [imageSize\(\)](#)) returns the corresponding rectangle in (0,0),[zoomedSize\(\)](#).*
- QPointF **mapZoomToImage** (const QPointF &zoomedPoint) const
 

*For a given point (in (0,0), [zoomedSize\(\)](#)) returns the corresponding point in (0,0),[imageSize\(\)](#).*
- QRectF **mapZoomToImage** (const QRectF &imageRect) const
- QSizeF **originalImageSize** () const
 

*Return the original image size.*
- double **realZoomFactor** () const
 

*Return the real zoom factor dependent on device pixel ratio.*
- void **setDisplayWidget** (QWidget \*const widget)
 

*Set the graphics view widget to track the device pixel ratio.*
- void **setImageSize** (const QSize &size, const QSize &originalSize=QSize())
 

*Sets the size of the (available) image data.*
- void **setZoomFactor** (double zoom)
 

*Sets the current zoom factor, relative to (original) size.*

- double **snappedZoomFactor** (double newZoom, const QSizeF &frameSize) const  
*When setting a new zoom factor (absolute value), the new value may be very close to a special value.*
- double **snappedZoomStep** (double nextZoom, const QSizeF &frameSize) const  
*When changing the zoom from current zoom to given nextZoom, sometimes a special value may be crossed, and this could then be used instead of nextZoom.*
- QRectF **sourceRect** (const QRectF &imageRect) const  
*For a given rectangle contained in ((0,0), zoomedSize()) returns the corresponding rectangle in (0,0),imageSize().*
- QSizeF **zoomedSize** () const  
*Return the size of the image when the current zoom factor is applied.*
- double **zoomFactor** () const  
*Return the currently set zoom factor.*

### Static Public Member Functions

- static bool **getImageSmoothScale** ()
- static void **setImageSmoothScale** (bool enable)  
*Static functions to define the smooth scaling of the image.*

### Protected Attributes

- QWidget \* **m\_displayWidget** = nullptr
- QSizeF **m\_size**
- double **m\_zoom** = 1.0
- double **m\_zoomConst** = 1.0

## 9.744.1 Member Function Documentation

### 9.744.1.1 fitToSize()

```
void Digikam::ImageZoomSettings::fitToSize (
    const QSizeF & frameSize,
    FitToSizeMode mode = AlwaysFit )
```

Aspect ratio will be respected, that means the frameSize may not be completely filled in one dimension, and [zoomedSize\(\)](#) can differ from frameSize in one dimension.

### 9.744.1.2 originalImageSize()

```
QSizeF Digikam::ImageZoomSettings::originalImageSize ( ) const
```

Can be identical to size().

### 9.744.1.3 setImageSize()

```
void Digikam::ImageZoomSettings::setImageSize (
    const QSize & size,
    const QSize & originalSize = QSize() )
```

Optionally, you can specify an original size, if the available image data is a scaled-down version. In this case, zoom factors will refer to the original size. The zoom factor is unchanged, you need to call fitToSize again.

#### 9.744.1.4 snappedZoomFactor()

```
double Digikam::ImageZoomSettings::snappedZoomFactor (
    double newZoom,
    const QSizeF & frameSize ) const
```

Returns this special value if this is the case, returns *newZoom* if not applicable.

#### 9.744.1.5 snappedZoomStep()

```
double Digikam::ImageZoomSettings::snappedZoomStep (
    double nextZoom,
    const QSizeF & frameSize ) const
```

Returns this special zoom, or *nextZoom* if not applicable.

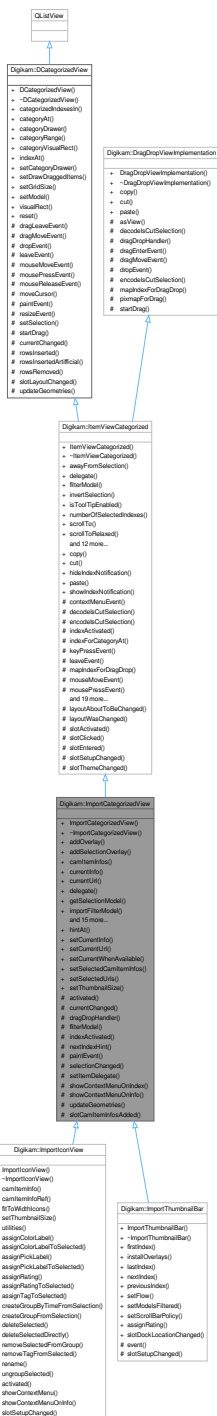
#### 9.744.1.6 zoomedSize()

```
QSizeF Digikam::ImageZoomSettings::zoomedSize ( ) const
```

This is the size the image should be displayed at.

## 9.745 Digikam::ImportCategorizedView Class Reference

Inheritance diagram for Digikam::ImportCategorizedView:



### Public Slots

- void **hintAt** (const [CamItemInfo](#) &info)

*Does something to gain attention for info, but not changing current selection.*

- void **setCurrentInfo** (const [CamItemInfo](#) &info)  
*Set as current item the item identified by the [CamItemInfo](#).*
- void **setCurrentUrl** (const QUrl &url)  
*Set as current item the item identified by its file url.*
- void **setCurrentWhenAvailable** (qulonglong camItemId)  
*Scroll the view to the given item when it becomes available.*
- void **setSelectedCamItemInfos** (const QList< [CamItemInfo](#) > &infos)  
*Set selected items.*
- void **setSelectedUrls** (const QList< QUrl > &urlList)  
*Set selected items identified by their file urls.*
- void **setThumbnailSize** (int size)

### Public Slots inherited from [Digikam::ItemViewCategorized](#)

- void **copy** () override
- void **cut** () override
- void **hideIndexNotification** ()
- void **paste** () override
- void **showIndexNotification** (const QModelIndex &index, const QString &message)

### Public Slots inherited from [Digikam::DCategorizedView](#)

- void **reset** () override

### Signals

- void **camItemInfoActivated** (const [CamItemInfo](#) &info)  
*Emitted when the given [CamItemInfo](#) is activated.*
- void **currentChanged** (const [CamItemInfo](#) &info)
- void **deselected** (const QList< [CamItemInfo](#) > &nowDeselectedInfos)  
*Emitted when items are deselected.*
- void **modelChanged** ()  
*Emitted when a new model is set.*
- void **selected** (const QList< [CamItemInfo](#) > &newSelectedInfos)  
*Emitted when new items are selected.*

### Signals inherited from [Digikam::ItemViewCategorized](#)

- void **clicked** (const QMouseEvent \*e, const QModelIndex &index)  
*For overlays: Like the respective parent class signals, but with additional info.*
- void **entered** (const QMouseEvent \*e, const QModelIndex &index)
- void **keyPressed** (QKeyEvent \*e)  
*Remember you may want to check if the event is accepted or ignored.*
- void **selectionChanged** ()  
*Emitted when any selection change occurs.*
- void **selectionCleared** ()  
*Emitted when the selection is completely cleared.*
- void **viewportClicked** (const QMouseEvent \*e)  
*While [clicked\(\)](#) is emitted with a valid index, this corresponds to clicking on empty space.*
- void **zoomInStep** ()
- void **zoomOutStep** ()

## Public Member Functions

- **ImportCategorizedView** (QWidget \*const parent=nullptr)
- void **addOverlay** (ItemDelegateOverlay \*overlay, ImportDelegate \*delegate=nullptr)
  - Add and remove an overlay.*
- void **addSelectionOverlay** (ImportDelegate \*delegate=nullptr)
- QList< CamItemInfo > **camItemInfos** () const
- CamItemInfo **currentInfo** () const
- QUrl **currentUrl** () const
- ImportDelegate \* **delegate** () const
- QItemSelectionModel \* **getSelectionModel** () const
- ImportFilterModel \* **importFilterModel** () const
  - Returns any ImportFilterModel in chain.*
- ImportItemModel \* **importItemModel** () const
- ImportSortFilterModel \* **importSortFilterModel** () const
- ImportThumbnailModel \* **importThumbnailModel** () const
  - Returns 0 if the ImportItemModel is not an ImportThumbnailModel.*
- CamItemInfo **nextInfo** (const CamItemInfo &info)
- CamItemInfo **nextInOrder** (const CamItemInfo &startingPoint, int nth)
  - Returns the n-th info after the given one.*
- CamItemInfo **previousInfo** (const CamItemInfo &info)
- void **removeOverlay** (ItemDelegateOverlay \*overlay)
- QList< CamItemInfo > **selectedCamItemInfos** () const
- QList< CamItemInfo > **selectedCamItemInfosCurrentFirst** () const
- QList< QUrl > **selectedUrls** () const
- void **setModels** (ImportItemModel \*model, ImportSortFilterModel \*filterModel)
- virtual void **setThumbnailSize** (const ThumbnailSize &size)
- ThumbnailSize **thumbnailSize** () const
- void **toIndex** (const QUrl &url)
  - Selects the index as current and scrolls to it.*
- QList< QUrl > **urls** () const

## Public Member Functions inherited from Digikam::ItemViewCategorized

- **ItemViewCategorized** (QWidget \*const parent=nullptr)
- void **awayFromSelection** ()
- DItemDelegate \* **delegate** () const
- void **invertSelection** ()
- bool **isToolTipEnabled** () const
- int **numberOfSelectedIndexes** () const
- void **scrollTo** (const QModelIndex &index, ScrollHint hint=EnsureVisible) override
- void **scrollToRelaxed** (const QModelIndex &index, ScrollHint hint=EnsureVisible)
  - Like scrollTo, but only scrolls if the index is not visible, regardless of hint.*
- void **setInitialSelectedItem** (bool enabled)
  - Ensure a initial selected item.*
- void **setScrollCurrentToCenter** (bool enabled)
  - Scroll automatically the current index to center of the view.*
- void **setScrollStepGranularity** (int factor)
  - Determine a step size for scrolling: The larger this number, the smaller and more precise is the scrolling.*
- void **setSelectedIndexes** (const QList< QModelIndex > &indexes)
- void **setSpacing** (int spacing)
  - Sets the spacing.*



- void **setToolTipEnabled** (bool enabled)
- void **setUsePointingHandCursor** (bool useCursor)  
*Set if the PointingHand Cursor should be shown over the activation area.*
- void **toFirstIndex** ()  
*Selects the index as current and scrolls to it.*
- void **toIndex** (const QModelIndex &index)
- void **toLastIndex** ()
- void **toNextIndex** ()
- void **toPreviousIndex** ()

## Public Member Functions inherited from Digikam::DCategorizedView

- **DCategorizedView** (QWidget \*const parent=nullptr)
- virtual QModelIndexList **categorizedIndexesIn** (const QRect &rect) const  
*This method will return all indexes whose visual rect intersects rect.*
- virtual QModelIndex **categoryAt** (const QPoint &point) const  
*This method will return the first index of the category in the region of which point is found.*
- **DCategoryDrawer** \* **categoryDrawer** () const
- virtual QItemSelectionRange **categoryRange** (const QModelIndex &index) const  
*This method returns the range of indexes contained in the category in which index is sorted.*
- virtual QRect **categoryVisualRect** (const QModelIndex &index) const  
*This method will return the visual rect of the header of the category in which index is sorted.*
- QModelIndex **indexAt** (const QPoint &point) const override
- void **setCategoryDrawer** (DCategoryDrawer \*categoryDrawer)
- void **setDrawDraggedItems** (bool drawDraggedItems)  
*Switch on drawing of dragged items.*
- void **setGridSize** (const QSize &size)
- void **setModel** (QAbstractItemModel \*model) override
- QRect **visualRect** (const QModelIndex &index) const override

## Public Member Functions inherited from Digikam::DragDropViewImplementation

- virtual void **copy** ()
- virtual void **cut** ()
- virtual void **paste** ()

## Protected Slots

- void **slotCamItemInfosAdded** ()

## Protected Slots inherited from Digikam::ItemViewCategorized

- void **layoutAboutToBeChanged** ()
- void **layoutWasChanged** ()
- void **slotActivated** (const QModelIndex &index)
- void **slotClicked** (const QModelIndex &index)
- void **slotEntered** (const QModelIndex &index)
- virtual void **slotSetupChanged** ()
- virtual void **slotThemeChanged** ()

## Protected Slots inherited from [Digikam::DCategorizedView](#)

- void **currentChanged** (const QModelIndex &current, const QModelIndex &previous) override
- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- virtual void **rowsInsertedArtificial** (const QModelIndex &parent, int start, int end)
- virtual void **slotLayoutChanged** ()
- void **updateGeometries** () override

## Protected Member Functions

- virtual void **activated** (const [CamItemInfo](#) &info, Qt::KeyboardModifiers modifiers)
  - Reimplement these in a subclass.*
- void **currentChanged** (const QModelIndex &index, const QModelIndex &previous) override
- [AbstractItemDragDropHandler](#) \* **dragDropHandler** () const override
  - You need to implement these three methods Returns the drag drop handler.*
- QSortFilterProxyModel \* **filterModel** () const override
  - reimplemented from parent class*
- void **indexActivated** (const QModelIndex &index, Qt::KeyboardModifiers modifiers) override
- QModelIndex **nextIndexHint** (const QModelIndex &indexToAnchor, const QItemSelectionRange &removed) const override
  - Assuming the given indexes would be removed (hypothetically!), return the index to be selected instead, starting from anchor.*
- void **paintEvent** (QPaintEvent \*e) override
- void **selectionChanged** (const QItemSelection &, const QItemSelection &) override
- void **setItemDelegate** ([ImportDelegate](#) \*delegate)
- void **showContextMenuOnIndex** (QContextMenuEvent \*event, const QModelIndex &index) override
  - Reimplement these in a subclass.*
- virtual void **showContextMenuOnInfo** (QContextMenuEvent \*event, const [CamItemInfo](#) &info)
- void **updateGeometries** () override

## Protected Member Functions inherited from [Digikam::ItemViewCategorized](#)

- void **contextMenuEvent** (QContextMenuEvent \*event) override
  - reimplemented from parent class*
- bool **decodelsCutSelection** (const QMimeData \*mimeData)
- void **encodelsCutSelection** (QMimeData \*mime, bool isCutSelection)
- QModelIndex **indexForCategoryAt** (const QPoint &pos) const
  - Returns an index that is representative for the category at position pos.*
- void **keyPressEvent** (QKeyEvent \*event) override
- void **leaveEvent** (QEvent \*event) override
- QModelIndex **mapIndexForDragDrop** (const QModelIndex &index) const override
  - Note: pure virtual [dragDropHandler\(\)](#) still open from [DragDropViewImplementation](#).*
- void **mouseMoveEvent** (QMouseEvent \*event) override
- void **mousePressEvent** (QMouseEvent \*event) override
- void **mouseReleaseEvent** (QMouseEvent \*event) override
- QModelIndex **moveCursor** (CursorAction cursorAction, Qt::KeyboardModifiers modifiers) override
- QPixmap **pixmapForDrag** (const QList< QModelIndex > &indexes) const override
  - Creates a pixmap for dragging the given indexes.*
- void **reset** () override
- void **resizeEvent** (QResizeEvent \*e) override
- void **rowsAboutToBeRemoved** (const QModelIndex &parent, int start, int end) override
- void **rowsInserted** (const QModelIndex &parent, int start, int end) override

- void **rowsRemoved** (const QModelIndex &parent, int start, int end) override
- void **selectionChanged** (const QItemSelection &, const QItemSelection &) override
- void **setItemDelegate** (DItemDelegate \*delegate)
- void **setToolTip** (ItemViewToolTip \*tip)
- virtual void **showContextMenu** (QContextMenuEvent \*event)
- virtual bool **showToolTip** (const QModelIndex &index, QStyleOptionViewItem &option, QHelpEvent \*e=nullptr)
  - Provides default behavior, can reimplement in a subclass.*
- void **updateDelegateSizes** ()
- void **userInteraction** ()
- bool **viewportEvent** (QEvent \*event) override
- void **wheelEvent** (QWheelEvent \*event) override

### Protected Member Functions inherited from Digikam::DCategorizedView

- void **dragLeaveEvent** (QDragLeaveEvent \*event) override
- void **dragMoveEvent** (QDragMoveEvent \*event) override
- void **dropEvent** (QDropEvent \*event) override
- void **leaveEvent** (QEvent \*event) override
- void **mouseMoveEvent** (QMouseEvent \*event) override
- void **mousePressEvent** (QMouseEvent \*event) override
- void **mouseReleaseEvent** (QMouseEvent \*event) override
- QModelIndex **moveCursor** (CursorAction cursorAction, Qt::KeyboardModifiers modifiers) override
- void **paintEvent** (QPaintEvent \*event) override
- void **resizeEvent** (QResizeEvent \*event) override
- void **setSelection** (const QRect &rect, QItemSelectionModel::SelectionFlags flags) override
- void **startDrag** (Qt::DropActions supportedActions) override

### Protected Member Functions inherited from Digikam::DragDropViewImplementation

- virtual QAbstractItemView \* **asView** ()=0
  - This one is implemented by DECLARE\_VIEW\_DRAG\_DROP\_METHODS.*
- bool **decodelsCutSelection** (const QMimeData \*mimeData)
- void **dragEnterEvent** (QDragEnterEvent \*event)
  - Implements the relevant QAbstractItemView methods for drag and drop.*
- void **dragMoveEvent** (QDragMoveEvent \*e)
- void **dropEvent** (QDropEvent \*e)
- void **encodelsCutSelection** (QMimeData \*mime, bool isCutSelection)
- void **startDrag** (Qt::DropActions supportedActions)

## 9.745.1 Member Function Documentation

### 9.745.1.1 activated()

```
void Digikam::ImportCategorizedView::activated (
    const CamItemInfo & info,
    Qt::KeyboardModifiers modifiers ) [protected], [virtual]
```

Reimplemented in [Digikam::ImportIconView](#).

### 9.745.1.2 addOverlay()

```
void Digikam::ImportCategorizedView::addOverlay (
    ItemDelegateOverlay * overlay,
    ImportDelegate * delegate = nullptr )
```

It will as well be removed automatically when destroyed. Unless you pass a different delegate, the current delegate will be used.

### 9.745.1.3 camItemInfoActivated

```
void Digikam::ImportCategorizedView::camItemInfoActivated (
    const CamItemInfo & info ) [signal]
```

Info is never null.

### 9.745.1.4 deselected

```
void Digikam::ImportCategorizedView::deselected (
    const QList< CamItemInfo > & nowDeselectedInfos ) [signal]
```

There may be other selected infos left. This signal is not emitted when the model is reset; then only selectionCleared is emitted.

### 9.745.1.5 dragDropHandler()

```
AbstractItemDragDropHandler * Digikam::ImportCategorizedView::dragDropHandler ( ) const [override],
[protected], [virtual]
```

Implements [Digikam::DragDropViewImplementation](#).

### 9.745.1.6 filterModel()

```
QSortFilterProxyModel * Digikam::ImportCategorizedView::filterModel ( ) const [override],
[protected], [virtual]
```

Implements [Digikam::ItemViewCategorized](#).

### 9.745.1.7 importFilterModel()

```
ImportFilterModel * Digikam::ImportCategorizedView::importFilterModel ( ) const
```

May not be sourceModel()

### 9.745.1.8 indexActivated()

```
void Digikam::ImportCategorizedView::indexActivated (
    const QModelIndex & index,
    Qt::KeyboardModifiers modifiers ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ItemViewCategorized](#).

### 9.745.1.9 nextIndexHint()

```
QModelIndex Digikam::ImportCategorizedView::nextIndexHint (
    const QModelIndex & indexToAnchor,
    const QItemSelectionRange & removed ) const [override], [protected], [virtual]
```

The default implementation returns the next remaining sibling.

Reimplemented from [Digikam::ItemViewCategorized](#).

### 9.745.1.10 nextInOrder()

```
CamItemInfo Digikam::ImportCategorizedView::nextInOrder (
    const CamItemInfo & startingPoint,
    int nth )
```

Specifically, return the previous info for  $nth = -1$  and the next info for  $n = 1$ . Returns a null info if either *startingPoint* or the *nth* info are not contained in the model

### 9.745.1.11 selected

```
void Digikam::ImportCategorizedView::selected (
    const QList< CamItemInfo > & newSelectedInfos ) [signal]
```

The parameter includes only the newly selected infos, there may be other already selected infos.

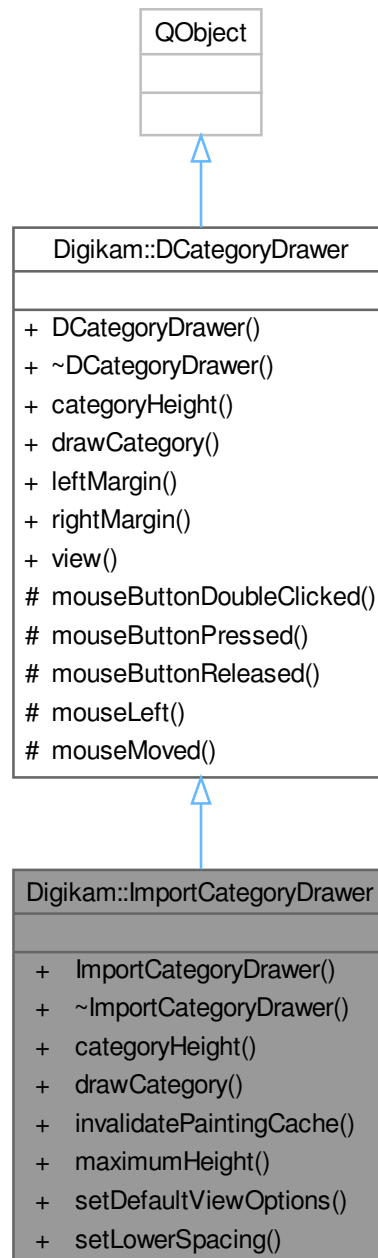
### 9.745.1.12 showContextMenuOnIndex()

```
void Digikam::ImportCategorizedView::showContextMenuOnIndex (
    QContextMenuEvent * event,
    const QModelIndex & index ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ItemViewCategorized](#).

## 9.746 Digikam::ImportCategoryDrawer Class Reference

Inheritance diagram for Digikam::ImportCategoryDrawer:



### Public Member Functions

- **ImportCategoryDrawer** ([ImportCategorizedView](#) \*const parent)
- int [categoryHeight](#) (const [QModelIndex](#) &index, const [QStyleOption](#) &option) const override

- void [drawCategory](#) (const QModelIndex &index, int sortRole, const QStyleOption &option, QPainter \*painter) const override  
*This method purpose is to draw a category represented by the given.*
- void [invalidatePaintingCache](#) ()
- virtual int [maximumHeight](#) () const
- void [setDefaultViewOptions](#) (const QStyleOptionViewItem &option)
- void [setLowerSpacing](#) (int spacing)

## Public Member Functions inherited from Digikam::DCategoryDrawer

- [DCategoryDrawer](#) ([DCategorizedView](#) \*const view)  
*Construct a category drawer for a given view.*
- virtual int [leftMargin](#) () const
- virtual int [rightMargin](#) () const
- [DCategorizedView](#) \* view () const

## Additional Inherited Members

## Signals inherited from Digikam::DCategoryDrawer

- void [actionRequested](#) (int action, const QModelIndex &index)  
*Emit this signal on your subclass implementation to notify that something happened.*
- void [collapseOrExpandClicked](#) (const QModelIndex &index)  
*This signal becomes emitted when collapse or expand has been clicked.*

## Protected Member Functions inherited from Digikam::DCategoryDrawer

- virtual void [mouseButtonDoubleClicked](#) (const QModelIndex &index, const QRect &blockRect, QMouseEvent \*event)  
*Method called when the mouse button has been double clicked.*
- virtual void [mouseButtonPressed](#) (const QModelIndex &index, const QRect &blockRect, QMouseEvent \*event)  
*Method called when the mouse button has been pressed.*
- virtual void [mouseButtonReleased](#) (const QModelIndex &index, const QRect &blockRect, QMouseEvent \*event)  
*Method called when the mouse button has been released.*
- virtual void [mouseLeft](#) (const QModelIndex &index, const QRect &blockRect)  
*Method called when the mouse button has left this block.*
- virtual void [mouseMoved](#) (const QModelIndex &index, const QRect &blockRect, QMouseEvent \*event)  
*Method called when the mouse has been moved.*

## 9.746.1 Member Function Documentation

### 9.746.1.1 categoryHeight()

```
int Digikam::ImportCategoryDrawer::categoryHeight (
    const QModelIndex & index,
    const QStyleOption & option ) const [override], [virtual]
```

#### Returns

The category height for the category represented by index `index` with style options `option`.

Reimplemented from [Digikam::DCategoryDrawer](#).

### 9.746.1.2 drawCategory()

```
void Digikam::ImportCategoryDrawer::drawCategory (
    const QModelIndex & index,
    int sortRole,
    const QStyleOption & option,
    QPainter * painter ) const [override], [virtual]
```

#### Parameters

<i>index</i>	with the given
<i>sortRole</i>	sorting role
<i>option</i>	painter style options
<i>painter</i>	painter instance

#### Note

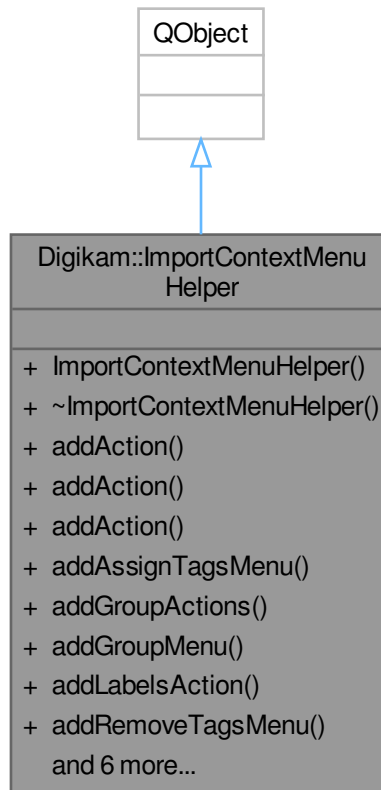
This method will be called one time per category, always with the first element in that category

Reimplemented from [Digikam::DCategoryDrawer](#).



## 9.747 Digikam::ImportContextMenuHelper Class Reference

Inheritance diagram for Digikam::ImportContextMenuHelper:



### Public Types

- typedef const QList< qlonglong > `itemIds`

### Signals

- void `signalAddNewTagFromABCMenu` (const QString &)
- void `signalAssignColorLabel` (int)
- void `signalAssignPickLabel` (int)
- void `signalAssignRating` (int)

### Public Member Functions

- [ImportContextMenuHelper](#) (QMenu \*const parent, KActionCollection \*const actionCollection=nullptr)  
*Constructs the helper class.*
- void [addAction](#) (const QString &name, bool addDisabled=false)

- Add an action from the actionCollection.*

  - void [addAction](#) (QAction \*action, bool addDisabled=false)

*Add a temporary action.*

  - void [addAction](#) (QAction \*action, QObject \*recv, const char \*slot, bool addDisabled=false)

*Add a temporary action and assign it to a custom slot.*

  - void [addAssignTagsMenu](#) (itemIds &ids)

*Add actions to add, remove or edit a tag.*

  - void **addGroupActions** (itemIds &ids)
  - void [addGroupMenu](#) (itemIds &ids)

*Add a "Group" menu.*

  - void [addLabelsAction](#) ()

*Add "Pick/Color/Rating Labels" action.*

  - void [addRemoveTagsMenu](#) (itemIds &ids)

*Add "Remove Tags" menu.*

  - void [addRotateMenu](#) (itemIds &ids)

*Add a menu to rotate item.*

  - void **addSeparator** ()

*Add a separator to the context menu.*

  - void [addServicesMenu](#) (const QList< QUrl > &selectedItems)

*Add the services menu to the menu.*

  - void [addSubMenu](#) (QMenu \*subMenu)

*Add a submenu to the parent context menu.*

  - QAction \* [exec](#) (const QPoint &pos, QAction \*at=nullptr)

*Execute the registered parent menu and evaluate the triggered actions.*

  - void [setImportFilterModel](#) ([ImportFilterModel](#) \*model)

*Set a filter model.*

## 9.747.1 Constructor & Destructor Documentation

### 9.747.1.1 ImportContextMenuHelper()

```
Digikam::ImportContextMenuHelper::ImportContextMenuHelper (
    QMenu *const parent,
    QActionCollection *const actionCollection = nullptr ) [explicit]
```

#### Parameters

<i>parent</i>	the menu the helper class is linked to
<i>actionCollection</i>	the actionCollection that should be used. If not set, the standard action from <a href="#">DigikamApp</a> is used

## 9.747.2 Member Function Documentation

### 9.747.2.1 addAction() [1/3]

```
void Digikam::ImportContextMenuHelper::addAction (
    const QString & name,
    bool addDisabled = false )
```

This method adds actions from the `actionCollection`. The `actionCollection` can be set in the constructor of the `ImportContextMenuHelper` class.

#### Parameters

<i>name</i>	the name of the action in the <code>actionCollection</code>
<i>addDisabled</i>	if set, disabled actions are added to the menu

#### 9.747.2.2 addAction() [2/3]

```
void Digikam::ImportContextMenuHelper::addAction (
    QAction * action,
    bool addDisabled = false )
```

Sometimes it is necessary to define actions that only exist in the current context menu content. Use this method to add such an action.

#### Parameters

<i>action</i>	the action to add
<i>addDisabled</i>	if set, disabled actions are added to the menu

#### 9.747.2.3 addAction() [3/3]

```
void Digikam::ImportContextMenuHelper::addAction (
    QAction * action,
    QObject * recv,
    const char * slot,
    bool addDisabled = false )
```

Use this method if you want to add a temporary action and immediately connect it to the receiving slot.

#### Parameters

<i>action</i>	the action to add
<i>recv</i>	the receiver of the triggered action
<i>slot</i>	the slot to connect the triggered action to
<i>addDisabled</i>	if set, disabled actions are added to the menu

#### 9.747.2.4 addAssignTagsMenu()

```
void Digikam::ImportContextMenuHelper::addAssignTagsMenu (
    itemIds & ids )
```

The tag modification helper is used to execute the action. You must set the parent tag to use on modification helper. Add "Assign Tags" menu.

This menu will provide a list of all tags available so that they can be assigned to the current selected items.

To make this menu work, you need to run [exec\(\)](#) from this class, otherwise the signals are not emitted and you will not be able to react on triggered actions from this menu. Make sure to connect the signals to the appropriate slots in the context menu handling method.

#### Parameters

<i>ids</i>	the selected items
------------	--------------------

#### See also

[exec\(\)](#)  
[signalAssignTag\(\)](#)

#### 9.747.2.5 addGroupMenu()

```
void Digikam::ImportContextMenuHelper::addGroupMenu (
    itemIds & ids )
```

This menu will provide actions open, close, add to, remove from, or split a group.

`addGroupActions` will add the actions as a flat list, not in a submenu. Note: Call `setItemFilterModel` before to have Open/Close group actions.

#### 9.747.2.6 addLabelsAction()

```
void Digikam::ImportContextMenuHelper::addLabelsAction ( )
```

This action will provide methods to assign pick/color/rating labels to the currently selected items.

To make this menu work, you need to run [exec\(\)](#) from this class, otherwise the signals are not emitted and you will not be able to react on triggered actions from this menu. Make sure to connect the signals to the appropriate slots in the context menu handling method.

#### See also

[exec\(\)](#)  
[signalAssignPickLabel\(\)](#)  
[signalAssignColorLabel\(\)](#)  
[signalAssignRating\(\)](#)

#### 9.747.2.7 addRemoveTagsMenu()

```
void Digikam::ImportContextMenuHelper::addRemoveTagsMenu (
    itemIds & ids )
```

This menu will provide a list of all tags assigned to the current items. Actions triggered in here will remove the selected tag from the items.

To make this menu work, you need to run [exec\(\)](#) from this class, otherwise the signals are not emitted and you will not be able to react on triggered actions from this menu. Make sure to connect the signals to the appropriate slots in the context menu handling method.

**Parameters**

<i>ids</i>	the selected items
------------	--------------------

**See also**

[exec\(\)](#)  
[signalRemoveTag\(\)](#)

**9.747.2.8 addRotateMenu()**

```
void Digikam::ImportContextMenuHelper::addRotateMenu (
    itemIds & ids )
```

**Parameters**

<i>ids</i>	the selected items
------------	--------------------

**9.747.2.9 addServicesMenu()**

```
void Digikam::ImportContextMenuHelper::addServicesMenu (
    const QList< QUrl > & selectedItems )
```

The services menu is used to open the selected items in a different application. It will query the item for registered services and provide them in a submenu. The menu will be titled "Open With...".

**Parameters**

<i>selectedItems</i>	the list of selected items
----------------------	----------------------------

**9.747.2.10 addSubMenu()**

```
void Digikam::ImportContextMenuHelper::addSubMenu (
    QMenu * subMenu )
```

**Parameters**

<i>subMenu</i>	the submenu to be added
----------------	-------------------------

**9.747.2.11 exec()**

```
QAction * Digikam::ImportContextMenuHelper::exec (
    const QPoint & pos,
    QAction * at = nullptr )
```

Always use this method instead the one from the parent menu. It will ensure that the signals are emitted and special cases are handled.

**Parameters**

<i>pos</i>	position of the triggered action in the registered menu
<i>at</i>	the action that should be at the position pos

**Returns**

the triggered action

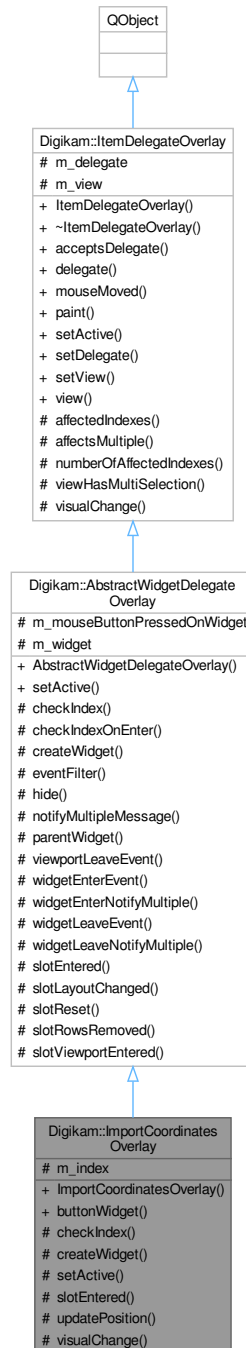
**9.747.2.12 setImportFilterModel()**

```
void Digikam::ImportContextMenuHelper::setImportFilterModel (  
    ImportFilterModel * model )
```

Some of the group actions will operate directly on the model.

## 9.748 Digikam::ImportCoordinatesOverlay Class Reference

Inheritance diagram for Digikam::ImportCoordinatesOverlay:



### Public Member Functions

- **ImportCoordinatesOverlay** (QObject \*const parent)
- **ImportOverlayWidget** \* **buttonWidget** () const

## Public Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- [AbstractWidgetDelegateOverlay](#) (QObject \*const parent)  
*This class provides functionality for using a widget in an overlay.*

## Public Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- **ItemDelegateOverlay** (QObject \*const parent=nullptr)
- virtual bool **acceptsDelegate** (QAbstractItemDelegate \*) const
- QAbstractItemDelegate \* **delegate** () const
- virtual void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)  
*Only these two methods are implemented as virtual methods.*
- virtual void **paint** (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index)
- void **setDelegate** (QAbstractItemDelegate \*delegate)
- void **setView** (QAbstractItemView \*view)
- QAbstractItemView \* **view** () const

## Protected Member Functions

- bool **checkIndex** (const QModelIndex &index) const override  
*Return true here if you want to show the overlay for the given index.*
- QWidget \* **createWidget** () override  
*Create your widget here.*
- void **setActive** (bool active) override  
*If active is true, this will call [createWidget\(\)](#), initialize the widget for use, and setup connections for the virtual slots.*
- void **slotEntered** (const QModelIndex &index) override  
*Default implementation shows the widget iff the index is valid and [checkIndex](#) returns true.*
- void **updatePosition** ()
- void **visualChange** () override  
*Called when any change from the delegate occurs - when the overlay is installed, when size hints, styles or fonts change.*

## Protected Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- bool **checkIndexOnEnter** (const QModelIndex &index) const  
*Utility method called from [slotEntered](#).*
- bool **eventFilter** (QObject \*obj, QEvent \*event) override
- virtual void **hide** ()  
*Called when the widget shall be hidden (mouse cursor left index, viewport, uninstalled etc.).*
- virtual QString **notifyMultipleMessage** (const QModelIndex &, int number)
- QWidget \* **parentWidget** () const  
*Returns the widget to be used as parent for your widget created in [createWidget\(\)](#)*
- virtual void **viewportLeaveEvent** (QObject \*obj, QEvent \*event)  
*Called when a `QEvent::Leave` of the viewport is received.*
- virtual void **widgetEnterEvent** ()  
*Called when a `QEvent::Enter` resp.*
- void **widgetEnterNotifyMultiple** (const QModelIndex &index)  
*A sample implementation for above methods.*
- virtual void **widgetLeaveEvent** ()
- void **widgetLeaveNotifyMultiple** ()



## Protected Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- `QList< QModelIndex > affectedIndexes` (const QModelIndex &index) const
- bool `affectsMultiple` (const QModelIndex &index) const  
*For the context that an overlay can affect multiple items: Assuming the currently overlaid index is given.*
- int `numberOfAffectedIndexes` (const QModelIndex &index) const
- bool `viewHasMultiSelection` () const  
*Utility method.*

## Protected Attributes

- `QPersistentModelIndex m_index`

## Protected Attributes inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- bool `m_mouseButtonPressedOnWidget` = false
- `QWidget * m_widget` = nullptr

## Protected Attributes inherited from [Digikam::ItemDelegateOverlay](#)

- `QAbstractItemDelegate * m_delegate` = nullptr
- `QAbstractItemView * m_view` = nullptr

## Additional Inherited Members

## Signals inherited from [Digikam::ItemDelegateOverlay](#)

- void `hideNotification` ()
- void `requestNotification` (const QModelIndex &index, const QString &message)
- void `update` (const QModelIndex &index)

## Protected Slots inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- virtual void `slotLayoutChanged` ()
- virtual void `slotReset` ()  
*Default implementations of these three slots call `hide()`*
- virtual void `slotRowsRemoved` (const QModelIndex &parent, int start, int end)
- virtual void `slotViewportEntered` ()

## Protected Slots inherited from [Digikam::ItemDelegateOverlay](#)

### 9.748.1 Member Function Documentation

#### 9.748.1.1 `checkIndex()`

```
bool Digikam::ImportCoordinatesOverlay::checkIndex (
    const QModelIndex & index ) const [override], [protected], [virtual]
```

The default implementation returns true.

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.748.1.2 createWidget()

```
QWidget * Digikam::ImportCoordinatesOverlay::createWidget ( ) [override], [protected], [virtual]
```

When creating the object, pass [parentWidget\(\)](#) as parent widget. Ownership of the object is passed. It will be deleted in [setActive\(false\)](#).

Implements [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.748.1.3 setActive()

```
void Digikam::ImportCoordinatesOverlay::setActive (
    bool active ) [override], [protected], [virtual]
```

If active is false, this will delete the widget and disconnect all signal from model and view to this object (!)

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.748.1.4 slotEntered()

```
void Digikam::ImportCoordinatesOverlay::slotEntered (
    const QModelIndex & index ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

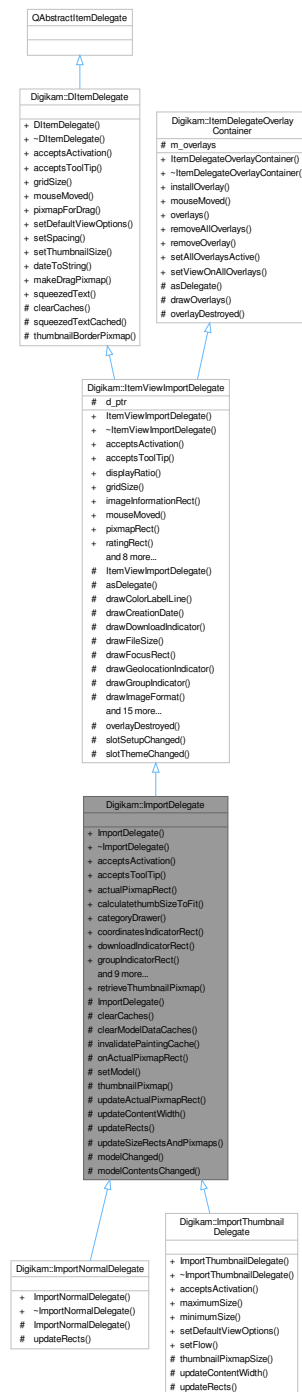
### 9.748.1.5 visualChange()

```
void Digikam::ImportCoordinatesOverlay::visualChange ( ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ItemDelegateOverlay](#).

## 9.749 Digikam::ImportDelegate Class Reference

Inheritance diagram for Digikam::ImportDelegate:



### Public Member Functions

- **ImportDelegate** (QWidget \*const parent)
- bool **acceptsActivation** (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*activationRect=nullptr) const override

- bool [acceptsToolTip](#) (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*tooltipRect=nullptr) const override  
*These methods take four parameters: The position on viewport, the rect on viewport, the index, and optionally a parameter into which, if the return value is true, a rectangle can be written for which the return value will be true as well.*
- QRect [actualPixmapRect](#) (const QModelIndex &index) const
- int [calculatethumbSizeToFit](#) (int ws)
- [ImportCategoryDrawer](#) \* [categoryDrawer](#) () const
- QRect [coordinatesIndicatorRect](#) () const
- QRect [downloadIndicatorRect](#) () const
- QRect [groupIndicatorRect](#) () const
- QRect [imageInformationRect](#) () const override  
*Returns the area where the image information is drawn, or null if empty / not supported.*
- QRect [lockIndicatorRect](#) () const
- void [paint](#) (QPainter \*painter, const QStyleOptionViewItem &option, const QModelIndex &index) const override
- QPixmap [pixmapForDrag](#) (const QStyleOptionViewItem &option, const QList< QModelIndex > &indexes) const override
- QRect [pixmapRect](#) () const override  
*Returns the area where the pixmap is drawn, or null if not supported.*
- void [setDefaultViewOptions](#) (const QStyleOptionViewItem &option) override  
*Style option with standard values to use for cached rendering.*
- void [setSpacing](#) (int spacing) override
- void [setView](#) ([ImportCategorizedView](#) \*view)
- QRect [tagsRect](#) () const

## Public Member Functions inherited from [Digikam::ItemViewImportDelegate](#)

- [ItemViewImportDelegate](#) (QWidget \*const parent)
- bool [acceptsActivation](#) (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*activationRect=nullptr) const override
- bool [acceptsToolTip](#) (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*tooltipRect=nullptr) const override  
*These methods take four parameters: The position on viewport, the rect on viewport, the index, and optionally a parameter into which, if the return value is true, a rectangle can be written for which the return value will be true as well.*
- double [displayRatio](#) () const
- QSize [gridSize](#) () const override  
*Returns the gridsize to be set by the view.*
- void [mouseMoved](#) (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index) override
- virtual QRect [ratingRect](#) () const  
*Returns the rectangle where the rating is drawn, or a null rectangle if not supported.*
- QRect [rect](#) () const
- void [setDefaultViewOptions](#) (const QStyleOptionViewItem &option) override  
*Style option with standard values to use for cached rendering.*
- void [setRatingEdited](#) (const QModelIndex &index)  
*Can be used to temporarily disable drawing of the rating.*
- void [setSpacing](#) (int spacing) override
- void [setThumbnailSize](#) (const [ThumbnailSize](#) &thumbSize) override  
*reimplemented from [DItemDelegate](#)*
- QSize [sizeHint](#) (const QStyleOptionViewItem &option, const QModelIndex &index) const override
- int [spacing](#) () const
- [ThumbnailSize](#) [thumbnailSize](#) () const

## Public Member Functions inherited from [Digikam::DItemDelegate](#)

- **DItemDelegate** (QObject \*const parent=nullptr)

## Public Member Functions inherited from [Digikam::ItemDelegateOverlayContainer](#)

- **ItemDelegateOverlayContainer** ()=default  
*This is a sample implementation for delegate management methods, to be inherited by a delegate.*
- void **installOverlay** ([ItemDelegateOverlay](#) \*overlay)
- void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)
- QList< [ItemDelegateOverlay](#) \* > **overlays** () const
- void **removeAllOverlays** ()
- void **removeOverlay** ([ItemDelegateOverlay](#) \*overlay)
- void **setAllOverlaysActive** (bool active)
- void **setViewOnAllOverlays** (QAbstractItemView \*view)

## Static Public Member Functions

- static QPixmap **retrieveThumbnailPixmap** (const QModelIndex &index, int thumbnailSize)  
*Retrieve the thumbnail pixmap in given size for the [ImportItemModel::ThumbnailRole](#) for the given index from the given index, which must adhere to [ImportThumbnailModel](#) semantics.*

## Static Public Member Functions inherited from [Digikam::DItemDelegate](#)

- static QString **dateToString** (const QDateTime &datetime)
- static QPixmap **makeDragPixmap** (const QStyleOptionViewItem &option, const QList< QModelIndex > &indexes, double displayRatio, const QPixmap &suggestedPixmap=QPixmap())
- static QString **squeezedText** (const QFontMetrics &fm, int width, const QString &text)

## Protected Slots

- void **modelChanged** ()
- void **modelContentsChanged** ()

## Protected Slots inherited from [Digikam::ItemViewImportDelegate](#)

- void **overlayDestroyed** (QObject \*o) override
- void **slotSetupChanged** ()
- void **slotThemeChanged** ()

## Protected Member Functions

- **ImportDelegate** (ImportDelegate::ImportDelegatePrivate &dd, QWidget \*const parent)
- void **clearCaches** () override
- virtual void **clearModelDataCaches** ()
  - Reimplement to clear caches based on model indexes (hash on row number etc.) Change signals are listened to this is called whenever such properties become invalid.*
- void **invalidatePaintingCache** () override
  - reimplement these in subclasses*
- bool **onActualPixmapRect** (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*actualRect) const
- void **setModel** (QAbstractItemModel \*model)
- virtual QPixmap **thumbnailPixmap** (const QModelIndex &index) const
- void **updateActualPixmapRect** (const QModelIndex &index, const QRect &rct)
- virtual void **updateContentWidth** ()
  - Reimplement this to set contentWidth.*
- virtual void **updateRects** ()=0
  - In a subclass, you need to implement this method to set up the rects for drawing.*
- void **updateSizeRectsAndPixmaps** () override

## Protected Member Functions inherited from [Digikam::ItemViewImportDelegate](#)

- **ItemViewImportDelegate** (ItemViewImportDelegatePrivate &dd, QWidget \*const parent)
- QAbstractItemDelegate \* **asDelegate** () override
  - Returns the delegate, typically, the derived class.*
- void **drawColorLabelLine** (QPainter \*p, const QRect &pixRect, int colorId) const
- void **drawCreationDate** (QPainter \*p, const QRect &dateRect, const QDateTime &date) const
- void **drawDownloadIndicator** (QPainter \*p, const QRect &r, int itemType) const
- void **drawFileSize** (QPainter \*p, const QRect &r, qlonglong bytes) const
- void **drawFocusRect** (QPainter \*p, const QStyleOptionViewItem &option, bool isSelected) const
- void **drawGeolocationIndicator** (QPainter \*p, const QRect &r) const
- void **drawGroupIndicator** (QPainter \*p, const QRect &r, int numberOfGroupedImages, bool open) const
- void **drawImageFormat** (QPainter \*p, const QRect &dimsRect, const QString &mime) const
- void **drawImageSize** (QPainter \*p, const QRect &dimsRect, const QSize &dims) const
- void **drawLockIndicator** (QPainter \*p, const QRect &r, int lockStatus) const
- void **drawMouseOverRect** (QPainter \*p, const QStyleOptionViewItem &option) const
- void **drawName** (QPainter \*p, const QRect &nameRect, const QString &name) const
- void **drawPickLabelIcon** (QPainter \*p, const QRect &r, int pickLabel) const
- void **drawRating** (QPainter \*p, const QModelIndex &index, const QRect &ratingRect, int rating, bool isSelected) const
- void **drawTags** (QPainter \*p, const QRect &r, const QString &tagsString, bool isSelected) const
- QRect **drawThumbnail** (QPainter \*p, const QRect &thumbRect, const QPixmap &background, const QPixmap &thumbnail) const
  - Use the tool methods for painting in subclasses.*
- void **prepareBackground** ()
- void **prepareFonts** ()
- void **prepareMetrics** (int maxWidth)
- void **prepareRatingPixmaps** (bool composeOverBackground=true)
- QPixmap **ratingPixmap** (int rating, bool selected) const
  - Returns the relevant pixmap from the cached rating pixmaps.*

## Protected Member Functions inherited from [Digikam::DItemDelegate](#)

- QString **squeezedTextCached** (QPainter \*const p, int width, const QString &text) const
- QPixmap **thumbnailBorderPixmap** (const QSize &pixSize, bool isGrouped=false) const

## Protected Member Functions inherited from [Digikam::ItemDelegateOverlayContainer](#)

- virtual void **drawOverlays** (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index) const
- virtual void **overlayDestroyed** (QObject \*o)

*Declare as slot in the derived class calling this method.*

## Additional Inherited Members

## Signals inherited from [Digikam::ItemViewImportDelegate](#)

- void **hideNotification** ()
- void **requestNotification** (const QModelIndex &index, const QString &message)

## Signals inherited from [Digikam::DItemDelegate](#)

- void **gridSizeChanged** (const QSize &newSize)
- void **visualChange** ()

## Protected Attributes inherited from [Digikam::ItemViewImportDelegate](#)

- ItemViewImportDelegatePrivate \*const **d\_ptr** = nullptr

## Protected Attributes inherited from [Digikam::ItemDelegateOverlayContainer](#)

- QList< [ItemDelegateOverlay](#) \* > **m\_overlays**

## 9.749.1 Member Function Documentation

### 9.749.1.1 acceptsActivation()

```
bool Digikam::ImportDelegate::acceptsActivation (
    const QPoint & pos,
    const QRect & visualRect,
    const QModelIndex & index,
    QRect * activationRect = nullptr ) const [override], [virtual]
```

Implements [Digikam::DItemDelegate](#).

### 9.749.1.2 acceptsToolTip()

```
bool Digikam::ImportDelegate::acceptsToolTip (
    const QPoint & pos,
    const QRect & visualRect,
    const QModelIndex & index,
    QRect * tooltipRect = nullptr ) const [override], [virtual]
```

Implements [Digikam::DItemDelegate](#).

### 9.749.1.3 clearCaches()

```
void Digikam::ImportDelegate::clearCaches ( ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::DItemDelegate](#).

### 9.749.1.4 imageInformationRect()

```
QRect Digikam::ImportDelegate::imageInformationRect ( ) const [override], [virtual]
```

The image information is textual or graphical information, but not the pixmap. The [ratingRect\(\)](#) will e.g. typically be contained in this area.

Reimplemented from [Digikam::ItemViewImportDelegate](#).

### 9.749.1.5 invalidatePaintingCache()

```
void Digikam::ImportDelegate::invalidatePaintingCache ( ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ItemViewImportDelegate](#).

### 9.749.1.6 pixmapForDrag()

```
QPixmap Digikam::ImportDelegate::pixmapForDrag (
    const QStyleOptionViewItem & option,
    const QList< QModelIndex > & indexes ) const [override], [virtual]
```

Implements [Digikam::DItemDelegate](#).

### 9.749.1.7 pixmapRect()

```
QRect Digikam::ImportDelegate::pixmapRect ( ) const [override], [virtual]
```

Reimplemented from [Digikam::ItemViewImportDelegate](#).



### 9.749.1.8 setDefaultViewOptions()

```
void Digikam::ImportDelegate::setDefaultViewOptions (
    const QStyleOptionViewItem & option ) [override], [virtual]
```

option.rect shall be the viewport rectangle. Call on resize, font change.

Implements [Digikam::DItemDelegate](#).

Reimplemented in [Digikam::ImportThumbnailDelegate](#).

### 9.749.1.9 setSpacing()

```
void Digikam::ImportDelegate::setSpacing (
    int spacing ) [override], [virtual]
```

Implements [Digikam::DItemDelegate](#).

### 9.749.1.10 updateContentWidth()

```
void Digikam::ImportDelegate::updateContentWidth ( ) [protected], [virtual]
```

This is the maximum width of all content rectangles, typically excluding margins on both sides.

Reimplemented in [Digikam::ImportThumbnailDelegate](#).

### 9.749.1.11 updateRects()

```
virtual void Digikam::ImportDelegate::updateRects ( ) [protected], [pure virtual]
```

The paint() method operates depending on these rects.

Implemented in [Digikam::ImportThumbnailDelegate](#), and [Digikam::ImportNormalDelegate](#).

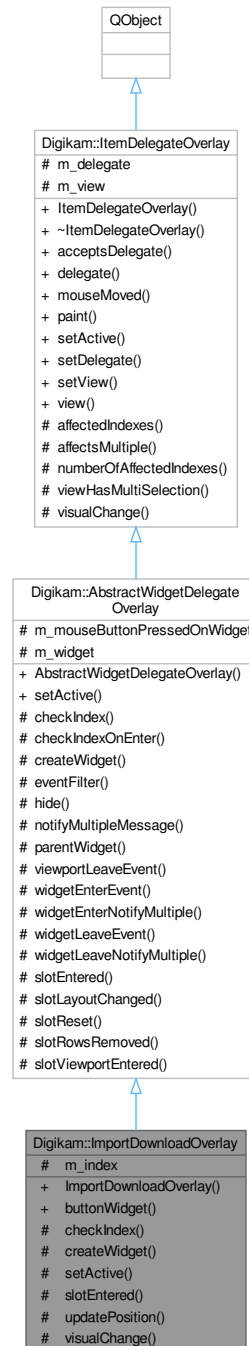
### 9.749.1.12 updateSizeRectsAndPixmapes()

```
void Digikam::ImportDelegate::updateSizeRectsAndPixmapes ( ) [override], [protected], [virtual]
```

Implements [Digikam::ItemViewImportDelegate](#).

## 9.750 Digikam::ImportDownloadOverlay Class Reference

Inheritance diagram for Digikam::ImportDownloadOverlay:



### Public Member Functions

- **ImportDownloadOverlay** (QObject \*const parent)
- **ImportOverlayWidget** \* **buttonWidget** () const

## Public Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- [AbstractWidgetDelegateOverlay](#) (QObject \*const parent)  
*This class provides functionality for using a widget in an overlay.*

## Public Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- [ItemDelegateOverlay](#) (QObject \*const parent=nullptr)
- virtual bool [acceptsDelegate](#) (QAbstractItemDelegate \*) const
- QAbstractItemDelegate \* [delegate](#) () const
- virtual void [mouseMoved](#) (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)  
*Only these two methods are implemented as virtual methods.*
- virtual void [paint](#) (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index)
- void [setDelegate](#) (QAbstractItemDelegate \*delegate)
- void [setView](#) (QAbstractItemView \*view)
- QAbstractItemView \* [view](#) () const

## Protected Member Functions

- bool [checkIndex](#) (const QModelIndex &index) const override  
*Return true here if you want to show the overlay for the given index.*
- QWidget \* [createWidget](#) () override  
*Create your widget here.*
- void [setActive](#) (bool active) override  
*If active is true, this will call [createWidget\(\)](#), initialize the widget for use, and setup connections for the virtual slots.*
- void [slotEntered](#) (const QModelIndex &index) override  
*Default implementation shows the widget iff the index is valid and [checkIndex](#) returns true.*
- void [updatePosition](#) ()
- void [visualChange](#) () override  
*Called when any change from the delegate occurs - when the overlay is installed, when size hints, styles or fonts change.*

## Protected Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- bool [checkIndexOnEnter](#) (const QModelIndex &index) const  
*Utility method called from [slotEntered](#).*
- bool [eventFilter](#) (QObject \*obj, QEvent \*event) override
- virtual void [hide](#) ()  
*Called when the widget shall be hidden (mouse cursor left index, viewport, uninstalled etc.).*
- virtual QString [notifyMultipleMessage](#) (const QModelIndex &, int number)
- QWidget \* [parentWidget](#) () const  
*Returns the widget to be used as parent for your widget created in [createWidget\(\)](#)*
- virtual void [viewportLeaveEvent](#) (QObject \*obj, QEvent \*event)  
*Called when a `QEvent::Leave` of the viewport is received.*
- virtual void [widgetEnterEvent](#) ()  
*Called when a `QEvent::Enter` resp.*
- void [widgetEnterNotifyMultiple](#) (const QModelIndex &index)  
*A sample implementation for above methods.*
- virtual void [widgetLeaveEvent](#) ()
- void [widgetLeaveNotifyMultiple](#) ()

## Protected Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- `QList< QModelIndex > affectedIndexes` (const QModelIndex &index) const
- `bool affectsMultiple` (const QModelIndex &index) const  
*For the context that an overlay can affect multiple items: Assuming the currently overlaid index is given.*
- `int numberOfAffectedIndexes` (const QModelIndex &index) const
- `bool viewHasMultiSelection` () const  
*Utility method.*

## Protected Attributes

- `QPersistentModelIndex m_index`

## Protected Attributes inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- `bool m_mouseButtonPressedOnWidget` = false
- `QWidget * m_widget` = nullptr

## Protected Attributes inherited from [Digikam::ItemDelegateOverlay](#)

- `QAbstractItemDelegate * m_delegate` = nullptr
- `QAbstractItemView * m_view` = nullptr

## Additional Inherited Members

## Signals inherited from [Digikam::ItemDelegateOverlay](#)

- `void hideNotification` ()
- `void requestNotification` (const QModelIndex &index, const QString &message)
- `void update` (const QModelIndex &index)

## Protected Slots inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- virtual `void slotLayoutChanged` ()
- virtual `void slotReset` ()  
*Default implementations of these three slots call `hide()`*
- virtual `void slotRowsRemoved` (const QModelIndex &parent, int start, int end)
- virtual `void slotViewportEntered` ()

## Protected Slots inherited from [Digikam::ItemDelegateOverlay](#)

### 9.750.1 Member Function Documentation

#### 9.750.1.1 `checkIndex()`

```
bool Digikam::ImportDownloadOverlay::checkIndex (
    const QModelIndex & index ) const [override], [protected], [virtual]
```

The default implementation returns true.

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.750.1.2 createWidget()

```
QWidget * Digikam::ImportDownloadOverlay::createWidget ( ) [override], [protected], [virtual]
```

When creating the object, pass [parentWidget\(\)](#) as parent widget. Ownership of the object is passed. It will be deleted in [setActive\(false\)](#).

Implements [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.750.1.3 setActive()

```
void Digikam::ImportDownloadOverlay::setActive (
    bool active ) [override], [protected], [virtual]
```

If active is false, this will delete the widget and disconnect all signal from model and view to this object (!)

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.750.1.4 slotEntered()

```
void Digikam::ImportDownloadOverlay::slotEntered (
    const QModelIndex & index ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

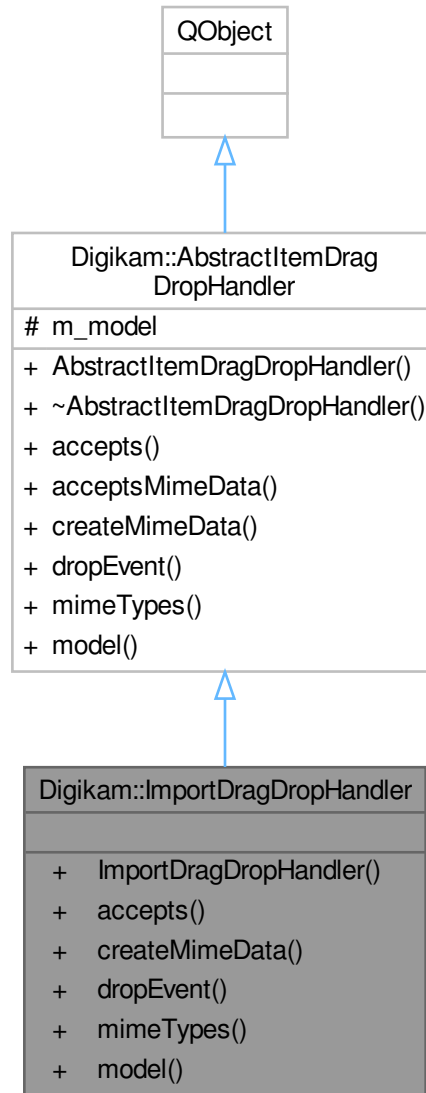
### 9.750.1.5 visualChange()

```
void Digikam::ImportDownloadOverlay::visualChange ( ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ItemDelegateOverlay](#).

## 9.751 Digikam::ImportDragDropHandler Class Reference

Inheritance diagram for Digikam::ImportDragDropHandler:



### Public Member Functions

- **ImportDragDropHandler** ([ImportItemModel](#) \*const model)
- Qt::DropAction **accepts** (const QDropEvent \*e, const QModelIndex &dropIndex) override  
*Returns if the given mime data is accepted for drop on dropIndex.*
- QMimeData \* **createMimeData** (const QList< QModelIndex > &) override  
*Create a mime data object for starting a drag from the given Albums.*
- bool **dropEvent** (QAbstractItemView \*view, const QDropEvent \*e, const QModelIndex &droppedOn) override  
*Gives the view and the occurring drop event.*

- QStringList [mimeTypes](#) () const override  
*Returns the supported mime types.*
- [ImportItemModel](#) \* **model** () const

## Public Member Functions inherited from [Digikam::AbstractItemDragDropHandler](#)

- **AbstractItemDragDropHandler** (QAbstractItemModel \*const model)
- virtual bool [acceptsMimeData](#) (const QMimeData \*data)  
*Returns if the given mime data can be handled.*
- QAbstractItemModel \* **model** () const

## Additional Inherited Members

## Protected Attributes inherited from [Digikam::AbstractItemDragDropHandler](#)

- QAbstractItemModel \* **m\_model** = nullptr

## 9.751.1 Member Function Documentation

### 9.751.1.1 [accepts\(\)](#)

```
Qt::DropAction Digikam::ImportDragDropHandler::accepts (
    const QDropEvent * e,
    const QModelIndex & dropIndex ) [override], [virtual]
```

Returns the proposed action, or Qt::IgnoreAction if not accepted.

Reimplemented from [Digikam::AbstractItemDragDropHandler](#).

### 9.751.1.2 [createMimeData\(\)](#)

```
QMimeData * Digikam::ImportDragDropHandler::createMimeData (
    const QList< QModelIndex > & ) [override], [virtual]
```

Reimplemented from [Digikam::AbstractItemDragDropHandler](#).

### 9.751.1.3 [dropEvent\(\)](#)

```
bool Digikam::ImportDragDropHandler::dropEvent (
    QAbstractItemView * view,
    const QDropEvent * e,
    const QModelIndex & droppedOn ) [override], [virtual]
```

The index is the index where the drop was dropped on. It may be invalid (dropped on decoration, viewport) Returns true if the event is to be accepted.

Reimplemented from [Digikam::AbstractItemDragDropHandler](#).

### 9.751.1.4 mimeTypees()

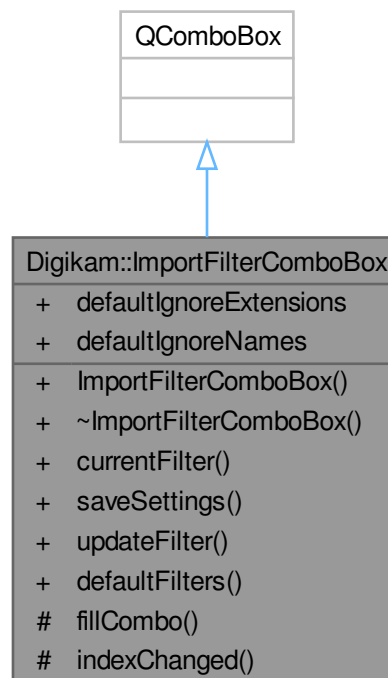
```
QStringList Digikam::ImportDragDropHandler::mimeTypees ( ) const [override], [virtual]
```

Called by the default implementation of model's [mimeTypees\(\)](#).

Reimplemented from [Digikam::AbstractItemDragDropHandler](#).

## 9.752 Digikam::ImportFilterComboBox Class Reference

Inheritance diagram for Digikam::ImportFilterComboBox:



### Signals

- void **signalFilterChanged** ([Filter](#) \*)

### Public Member Functions

- **ImportFilterComboBox** (QWidget \*const parent)
- [Filter](#) \* **currentFilter** () const
- void **saveSettings** ()
- void **updateFilter** ()



### Static Public Member Functions

- static void **defaultFilters** (FilterList \*const filters)

### Static Public Attributes

- static const QString **defaultIgnoreExtensions**
- static const QString **defaultIgnoreNames**

### Protected Slots

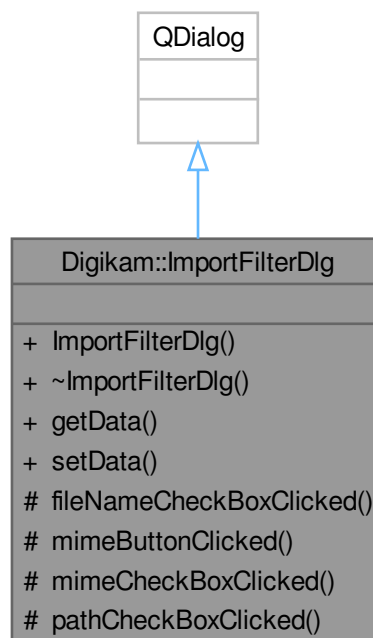
- void **indexChanged** (int index)

### Protected Member Functions

- void **fillCombo** ()

## 9.753 Digikam::ImportFilterDlg Class Reference

Inheritance diagram for Digikam::ImportFilterDlg:



**Public Member Functions**

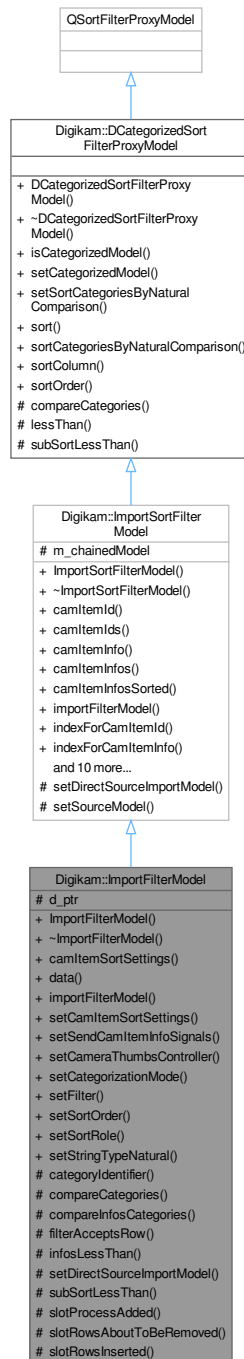
- **ImportFilterDlg** (QWidget \*const parent=nullptr)
- void **getData** (Filter \*const filter)
- void **setData** (const Filter &filter)

**Protected Slots**

- void **fileNameCheckBoxClicked** ()
- void **mimeButtonClicked** ()
- void **mimeCheckBoxClicked** ()
- void **pathCheckBoxClicked** ()

## 9.754 Digikam::ImportFilterModel Class Reference

Inheritance diagram for Digikam::ImportFilterModel:



### Public Types

- enum `ImportFilterModelRoles` {
  - `CategorizationModeRole` = `ImportItemModel::FilterModelRoles + 1` , `SortOrderRole` = `ImportItemModel::FilterModelRoles + 2` , `CategoryFormatRole` = `ImportItemModel::FilterModelRoles + 3` , `CategoryDateRole`

```
= ImportItemModel::FilterModelRoles + 4 ,
ImportFilterModelPointerRole = ImportItemModel::FilterModelRoles + 50 }
```

## Public Types inherited from [Digikam::DCategorizedSortFilterProxyModel](#)

- enum [AdditionalRoles](#) { [CategoryDisplayRole](#) = 0x17CE990A , [CategorySortRole](#) = 0x27857E60 }

## Public Slots

- void **setCameraThumbsController** ([CameraThumbsCtrl](#) \*const thumbsCtrl)
- void **setCategorizationMode** ([CamItemSortSettings::CategorizationMode](#) mode)
- void **setFilter** ([Filter](#) \*)
- void **setSortOrder** ([CamItemSortSettings::SortOrder](#) order)
- void **setSortRole** ([CamItemSortSettings::SortRole](#) role)
- void **setStringTypeNatural** (bool natural)

## Signals

- void **camItemInfosAboutToBeRemoved** (const [QList](#)< [CamItemInfo](#) > &infos)
- void **camItemInfosAdded** (const [QList](#)< [CamItemInfo](#) > &infos)

*Changes the current image filter settings and refilters.*

## Public Member Functions

- ImportFilterModel** ([QObject](#) \*const parent=nullptr)
- [CamItemSortSettings](#) **camItemSortSettings** () const
- [QVariant](#) **data** (const [QModelIndex](#) &index, int role=[Qt::DisplayRole](#)) const override
- [ImportFilterModel](#) \* **importFilterModel** () const override

*Returns this, any chained [ImportFilterModel](#), or 0.*

- void **setCamItemSortSettings** (const [CamItemSortSettings](#) &sorter)
  - void **setSendCamItemInfoSignals** (bool sendSignals)
- Enables sending [camItemInfosAdded](#) and [camItemInfosAboutToBeRemoved](#).*

## Public Member Functions inherited from [Digikam::ImportSortFilterModel](#)

- ImportSortFilterModel** ([QObject](#) \*const parent=nullptr)
  - qulonglong **camItemId** (const [QModelIndex](#) &index) const
  - [QList](#)< qulonglong > **camItemIds** (const [QList](#)< [QModelIndex](#) > &indexes) const
  - [CamItemInfo](#) **camItemInfo** (const [QModelIndex](#) &index) const
  - [QList](#)< [CamItemInfo](#) > **camItemInfos** (const [QList](#)< [QModelIndex](#) > &indexes) const
  - [QList](#)< [CamItemInfo](#) > **camItemInfosSorted** () const
- Returns a list of all camera infos, sorted according to this model.*
- [QModelIndex](#) **indexForCamItemId** (qulonglong id) const
  - [QModelIndex](#) **indexForCamItemInfo** (const [CamItemInfo](#) &info) const
  - [QModelIndex](#) **indexForPath** (const [QString](#) &filePath) const
  - [QModelIndex](#) **mapFromDirectSourceToSourceImportModel** (const [QModelIndex](#) &sourceModelIndex) const
  - [QModelIndex](#) **mapFromSourceImportModel** (const [QModelIndex](#) &importModelIndex) const
  - [QList](#)< [QModelIndex](#) > **mapListFromSource** (const [QList](#)< [QModelIndex](#) > &sourceIndexes) const
  - [QList](#)< [QModelIndex](#) > **mapListToSource** (const [QList](#)< [QModelIndex](#) > &indexes) const
  - [QModelIndex](#) **mapToSourceImportModel** (const [QModelIndex](#) &proxyIndex) const
- Convenience methods mapped to [ImportItemModel](#).*
- void **setSourceFilterModel** ([ImportSortFilterModel](#) \*const sourceModel)
  - void **setSourceImportModel** ([ImportItemModel](#) \*const sourceModel)
  - [ImportSortFilterModel](#) \* **sourceFilterModel** () const
  - [ImportItemModel](#) \* **sourceImportModel** () const

## Public Member Functions inherited from Digikam::DCategorizedSortFilterProxyModel

- **DCategorizedSortFilterProxyModel** (QObject \*const parent=nullptr)
- bool **isCategorizedModel** () const
- void **setCategorizedModel** (bool categorizedModel)  
*Enables or disables the categorization feature.*
- void **setSortCategoriesByNaturalComparison** (bool sortCategoriesByNaturalComparison)  
*Set if the sorting using CategorySortRole will use a natural comparison in the case that strings were returned.*
- void **sort** (int column, Qt::SortOrder order=Qt::AscendingOrder) override  
*Overridden from QSortFilterProxyModel.*
- bool **sortCategoriesByNaturalComparison** () const
- int **sortColumn** () const
- Qt::SortOrder **sortOrder** () const

## Protected Slots

- void **slotProcessAdded** (const QList< CamItemInfo > &)
- void **slotRowsAboutToBeRemoved** (const QModelIndex &parent, int start, int end)
- void **slotRowsInserted** (const QModelIndex &parent, int start, int end)

## Protected Member Functions

- virtual QString **categoryIdentifier** (const CamItemInfo &info) const  
*Returns a unique identifier for the category if info.*
- int **compareCategories** (const QModelIndex &left, const QModelIndex &right) const override  
*This method compares the category of the left index with the category of the right index.*
- virtual int **compareInfosCategories** (const CamItemInfo &left, const CamItemInfo &right) const  
*Reimplement to customize category sorting, Return negative if category of left < category right, Return 0 if left and right are in the same category, else return positive.*
- bool **filterAcceptsRow** (int source\_row, const QModelIndex &source\_parent) const override
- virtual bool **infosLessThan** (const CamItemInfo &left, const CamItemInfo &right) const  
*Reimplement to customize sorting.*
- void **setDirectSourceImportModel** (ImportItemModel \*const sourceModel) override  
*Reimplement if needed. Called only when model shall be set as (direct) sourceModel.*
- bool **subSortLessThan** (const QModelIndex &left, const QModelIndex &right) const override  
*This method has a similar purpose as lessThan() has on QSortFilterProxyModel.*

## Protected Member Functions inherited from Digikam::ImportSortFilterModel

- void **setSourceModel** (QAbstractItemModel \*sourceModel) override

## Protected Member Functions inherited from Digikam::DCategorizedSortFilterProxyModel

- bool **lessThan** (const QModelIndex &left, const QModelIndex &right) const override  
*Overridden from QSortFilterProxyModel.*

## Protected Attributes

- ImportFilterModelPrivate \*const **d\_ptr**

## Protected Attributes inherited from [Digikam::ImportSortFilterModel](#)

- [ImportSortFilterModel](#) \* `m_chainedModel` = nullptr

### 9.754.1 Member Enumeration Documentation

#### 9.754.1.1 ImportFilterModelRoles

enum [Digikam::ImportFilterModel::ImportFilterModelRoles](#)

##### Enumerator

<code>CategorizationModeRole</code>	Returns the current categorization mode.
<code>SortOrderRole</code>	Returns the current sort order.
<code>CategoryFormatRole</code>	Returns the format of the index which is used for category.
<code>CategoryDateRole</code>	Returns the date of the index which is used for category.
<code>ImportFilterModelPointerRole</code>	Returns true if the given camera item is a group leader, and the group is opened.

### 9.754.2 Member Function Documentation

#### 9.754.2.1 camItemInfosAdded

```
void Digikam::ImportFilterModel::camItemInfosAdded (
    const QList< CamItemInfo > & infos ) [signal]
```

Changes the current image sort settings and resorts. These signals need to be explicitly enabled with `setSendItemInfoSignals()`.

#### 9.754.2.2 categoryIdentifier()

```
QString Digikam::ImportFilterModel::categoryIdentifier (
    const CamItemInfo & info ) const [protected], [virtual]
```

The string need not be for user display.

#### 9.754.2.3 compareCategories()

```
int Digikam::ImportFilterModel::compareCategories (
    const QModelIndex & left,
    const QModelIndex & right ) const [override], [protected], [virtual]
```

Internally and if not reimplemented, this method will ask for `left` and `right` models for role `CategorySortRole`. In order to correctly sort categories, the `data()` method of the model should return a `qulonglong` (or numeric) value, or a `QString` object. `QString` objects will be sorted with `QString::localeAwareCompare` if `sortCategoriesByNaturalComparison()` is true.

**Note**

Please have present that: `QString(QChar(QChar::ObjectReplacementCharacter)) > QString(QChar(QChar::ReplacementCharacter)) > [ all possible strings ] > QString();`

This means that `QString()` will be sorted the first one, while `QString(QChar(QChar::ObjectReplacementCharacter))` and `QString(QChar(QChar::ReplacementCharacter))` will be sorted in last position.

**Warning**

Please note that `data()` method of the model should return always information of the same type. If you return a `QString` for an index, you should return always `QStrings` for all indexes for role `CategorySortRole` in order to correctly sort categories. You can't mix by returning a `QString` for one index, and a `qlonglong` for other.

**Note**

If you need a more complex layout, you will have to reimplement this method.

**Returns**

A negative value if the category of `left` should be placed before the category of `right`. 0 if `left` and `right` are on the same category, and a positive value if the category of `left` should be placed after the category of `right`.

Reimplemented from [Digikam::DCategorizedSortFilterProxyModel](#).

**9.754.2.4 importFilterModel()**

```
ImportFilterModel * Digikam::ImportFilterModel::importFilterModel ( ) const [override], [virtual]
```

Reimplemented from [Digikam::ImportSortFilterModel](#).

**9.754.2.5 infosLessThan()**

```
bool Digikam::ImportFilterModel::infosLessThan (
    const CamItemInfo & left,
    const CamItemInfo & right ) const [protected], [virtual]
```

Do not take categories into account here.

**9.754.2.6 setDirectSourceImportModel()**

```
void Digikam::ImportFilterModel::setDirectSourceImportModel (
    ImportItemModel *const sourceModel ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ImportSortFilterModel](#).

### 9.754.2.7 subSortLessThan()

```
bool Digikam::ImportFilterModel::subSortLessThan (  
    const QModelIndex & left,  
    const QModelIndex & right ) const [override], [protected], [virtual]
```

It is used for sorting items that are in the same category.

#### Returns

Returns true if the item `left` is less than the item `right` when sorting.

Reimplemented from [Digikam::DCategorizedSortFilterProxyModel](#).





- void **assignPickLabel** (const QModelIndex &index, int pickId)
- void **assignPickLabelToSelected** (int pickId)
- void **assignRating** (const QList< QModelIndex > &index, int rating)
- void **assignRatingToSelected** (int rating)
- void **assignTagToSelected** (int tagID)
- void **createGroupByTimeFromSelection** ()
- void **createGroupFromSelection** ()
- void **deleteSelected** (bool permanently=false)
- void **deleteSelectedDirectly** (bool permanently=false)
- void **removeSelectedFromGroup** ()
- void **removeTagFromSelected** (int tagID)
- void **rename** ()
- void **ungroupSelected** ()

### Public Slots inherited from [Digikam::ImportCategorizedView](#)

- void **hintAt** (const [CamItemInfo](#) &info)
 

*Does something to gain attention for info, but not changing current selection.*
- void **setCurrentInfo** (const [CamItemInfo](#) &info)
 

*Set as current item the item identified by the [CamItemInfo](#).*
- void **setCurrentUrl** (const QUrl &url)
 

*Set as current item the item identified by its file url.*
- void **setCurrentWhenAvailable** (qulonglong camItemId)
 

*Scroll the view to the given item when it becomes available.*
- void **setSelectedCamItemInfos** (const QList< [CamItemInfo](#) > &infos)
 

*Set selected items.*
- void **setSelectedUrls** (const QList< QUrl > &urlList)
 

*Set selected items identified by their file urls.*
- void **setThumbnailSize** (int size)

### Public Slots inherited from [Digikam::ItemViewCategorized](#)

- void **copy** () override
- void **cut** () override
- void **hideIndexNotification** ()
- void **paste** () override
- void **showIndexNotification** (const QModelIndex &index, const QString &message)

### Public Slots inherited from [Digikam::DCategorizedView](#)

- void **reset** () override

### Signals

- void **previewRequested** (const [CamItemInfo](#) &info, bool downloadPreview)

## Signals inherited from [Digikam::ImportCategorizedView](#)

- void [camItemInfoActivated](#) (const [CamItemInfo](#) &info)  
*Emitted when the given [CamItemInfo](#) is activated.*
- void **currentChanged** (const [CamItemInfo](#) &info)
- void [deselected](#) (const QList< [CamItemInfo](#) > &nowDeselectedInfos)  
*Emitted when items are deselected.*
- void **modelChanged** ()  
*Emitted when a new model is set.*
- void [selected](#) (const QList< [CamItemInfo](#) > &newSelectedInfos)  
*Emitted when new items are selected.*

## Signals inherited from [Digikam::ItemViewCategorized](#)

- void [clicked](#) (const QMouseEvent \*e, const QModelIndex &index)  
*For overlays: Like the respective parent class signals, but with additional info.*
- void **entered** (const QMouseEvent \*e, const QModelIndex &index)
- void [keyPressed](#) (QKeyEvent \*e)  
*Remember you may want to check if the event is accepted or ignored.*
- void [selectionChanged](#) ()  
*Emitted when any selection change occurs.*
- void **selectionCleared** ()  
*Emitted when the selection is completely cleared.*
- void **viewportClicked** (const QMouseEvent \*e)  
*While [clicked\(\)](#) is emitted with a valid index, this corresponds to clicking on empty space.*
- void **zoomInStep** ()
- void **zoomOutStep** ()

## Public Member Functions

- **ImportIconView** (QWidget \*const parent=nullptr)
- [CamItemInfo](#) **camItemInfo** (const QString &folder, const QString &file)
- [CamItemInfo](#) & **camItemInfoRef** (const QString &folder, const QString &file)
- int **fitToWidthIcons** ()
- void [setThumbnailSize](#) (const [ThumbnailSize](#) &size) override
- [ItemViewUtilities](#) \* **utilities** () const

## Public Member Functions inherited from [Digikam::ImportCategorizedView](#)

- **ImportCategorizedView** (QWidget \*const parent=nullptr)
- void [addOverlay](#) ([ItemDelegateOverlay](#) \*overlay, [ImportDelegate](#) \*delegate=nullptr)  
*Add and remove an overlay.*
- void **addSelectionOverlay** ([ImportDelegate](#) \*delegate=nullptr)
- QList< [CamItemInfo](#) > **camItemInfos** () const
- [CamItemInfo](#) **currentInfo** () const
- QUrl **currentUrl** () const
- [ImportDelegate](#) \* **delegate** () const
- QItemSelectionModel \* **getSelectionModel** () const
- [ImportFilterModel](#) \* **importFilterModel** () const  
*Returns any [ImportFilterModel](#) in chain.*

- [ImportItemModel](#) \* **importItemModel** () const
- [ImportSortFilterModel](#) \* **importSortFilterModel** () const
- [ImportThumbnailModel](#) \* **importThumbnailModel** () const
  - Returns 0 if the [ImportItemModel](#) is not an [ImportThumbnailModel](#).*
- [CamItemInfo](#) **nextInfo** (const [CamItemInfo](#) &info)
- [CamItemInfo](#) **nextInOrder** (const [CamItemInfo](#) &startingPoint, int nth)
  - Returns the n-th info after the given one.*
- [CamItemInfo](#) **previousInfo** (const [CamItemInfo](#) &info)
- void **removeOverlay** ([ItemDelegateOverlay](#) \*overlay)
- [QList](#)< [CamItemInfo](#) > **selectedCamItemInfos** () const
- [QList](#)< [CamItemInfo](#) > **selectedCamItemInfosCurrentFirst** () const
- [QList](#)< [QUrl](#) > **selectedUrls** () const
- void **setModels** ([ImportItemModel](#) \*model, [ImportSortFilterModel](#) \*filterModel)
- [ThumbnailSize](#) **thumbnailSize** () const
- void **toIndex** (const [QUrl](#) &url)
  - Selects the index as current and scrolls to it.*
- [QList](#)< [QUrl](#) > **urls** () const

## Public Member Functions inherited from [Digikam::ItemViewCategorized](#)

- **ItemViewCategorized** ([QWidget](#) \*const parent=nullptr)
- void **awayFromSelection** ()
- [DItemDelegate](#) \* **delegate** () const
- void **invertSelection** ()
- bool **isToolTipEnabled** () const
- int **numberOfSelectedIndexes** () const
- void **scrollTo** (const [QModelIndex](#) &index, [ScrollHint](#) hint=EnsureVisible) override
- void **scrollToRelaxed** (const [QModelIndex](#) &index, [ScrollHint](#) hint=EnsureVisible)
  - Like `scrollTo`, but only scrolls if the index is not visible, regardless of hint.*
- void **setInitialSelectedItem** (bool enabled)
  - Ensure a initial selected item.*
- void **setScrollCurrentToCenter** (bool enabled)
  - Scroll automatically the current index to center of the view.*
- void **setScrollStepGranularity** (int factor)
  - Determine a step size for scrolling: The larger this number, the smaller and more precise is the scrolling.*
- void **setSelectedIndexes** (const [QList](#)< [QModelIndex](#) > &indexes)
- void **setSpacing** (int spacing)
  - Sets the spacing.*
- void **setToolTipEnabled** (bool enabled)
- void **setUsePointingHandCursor** (bool useCursor)
  - Set if the `PointingHand` Cursor should be shown over the activation area.*
- void **toFirstIndex** ()
  - Selects the index as current and scrolls to it.*
- void **toIndex** (const [QModelIndex](#) &index)
- void **toLastIndex** ()
- void **toNextIndex** ()
- void **toPreviousIndex** ()

## Public Member Functions inherited from [Digikam::DCategorizedView](#)

- **DCategorizedView** (QWidget \*const parent=nullptr)
- virtual QModelIndexList [categorizedIndexesIn](#) (const QRect &rect) const  
*This method will return all indexes whose visual rect intersects `rect`.*
- virtual QModelIndex [categoryAt](#) (const QPoint &point) const  
*This method will return the first index of the category in the region of which `point` is found.*
- [DCategoryDrawer](#) \* **categoryDrawer** () const
- virtual QItemSelectionRange [categoryRange](#) (const QModelIndex &index) const  
*This method returns the range of indexes contained in the category in which `index` is sorted.*
- virtual QRect [categoryVisualRect](#) (const QModelIndex &index) const  
*This method will return the visual rect of the header of the category in which `index` is sorted.*
- QModelIndex **indexAt** (const QPoint &point) const override
- void **setCategoryDrawer** ([DCategoryDrawer](#) \*categoryDrawer)
- void [setDrawDraggedItems](#) (bool drawDraggedItems)  
*Switch on drawing of dragged items.*
- void **setGridSize** (const QSize &size)
- void **setModel** (QAbstractItemModel \*model) override
- QRect **visualRect** (const QModelIndex &index) const override

## Public Member Functions inherited from [Digikam::DragDropViewImplementation](#)

- virtual void **copy** ()
- virtual void **cut** ()
- virtual void **paste** ()

## Protected Member Functions

- void [activated](#) (const [CamItemInfo](#) &info, Qt::KeyboardModifiers modifiers) override  
*Reimplement these in a subclass.*
- void [showContextMenu](#) (QContextMenuEvent \*event) override
- void [showContextMenuOnInfo](#) (QContextMenuEvent \*event, const [CamItemInfo](#) &info) override
- void [slotSetupChanged](#) () override

## Protected Member Functions inherited from [Digikam::ImportCategorizedView](#)

- void **currentChanged** (const QModelIndex &index, const QModelIndex &previous) override
- [AbstractItemDragDropHandler](#) \* [dragDropHandler](#) () const override  
*You need to implement these three methods Returns the drag drop handler.*
- QSortFilterProxyModel \* [filterModel](#) () const override  
*reimplemented from parent class*
- void [indexActivated](#) (const QModelIndex &index, Qt::KeyboardModifiers modifiers) override
- QModelIndex [nextIndexHint](#) (const QModelIndex &indexToAnchor, const QItemSelectionRange &removed) const override  
*Assuming the given indexes would be removed (hypothetically!), return the index to be selected instead, starting from anchor.*
- void **paintEvent** (QPaintEvent \*e) override
- void **selectionChanged** (const QItemSelection &, const QItemSelection &) override
- void **setItemDelegate** ([ImportDelegate](#) \*delegate)
- void [showContextMenuOnIndex](#) (QContextMenuEvent \*event, const QModelIndex &index) override  
*Reimplement these in a subclass.*
- void **updateGeometries** () override

## Protected Member Functions inherited from [Digikam::ItemViewCategorized](#)

- void **contextMenuEvent** (QContextMenuEvent \*event) override  
*reimplemented from parent class*
- bool **decodelsCutSelection** (const QMimeData \*mimeType)
- void **encodelsCutSelection** (QMimeData \*mimeType, bool isCutSelection)
- QModelIndex **indexForCategoryAt** (const QPoint &pos) const  
*Returns an index that is representative for the category at position pos.*
- void **keyPressEvent** (QKeyEvent \*event) override
- void **leaveEvent** (QEvent \*event) override
- QModelIndex **mapIndexForDragDrop** (const QModelIndex &index) const override  
*Note: pure virtual [dragDropHandler\(\)](#) still open from [DragDropViewImplementation](#).*
- void **mouseMoveEvent** (QMouseEvent \*event) override
- void **mousePressEvent** (QMouseEvent \*event) override
- void **mouseReleaseEvent** (QMouseEvent \*event) override
- QModelIndex **moveCursor** (CursorAction cursorAction, Qt::KeyboardModifiers modifiers) override
- QPixmap **pixmapForDrag** (const QList< QModelIndex > &indexes) const override  
*Creates a pixmap for dragging the given indexes.*
- void **reset** () override
- void **resizeEvent** (QResizeEvent \*e) override
- void **rowsAboutToBeRemoved** (const QModelIndex &parent, int start, int end) override
- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- void **rowsRemoved** (const QModelIndex &parent, int start, int end) override
- void **selectionChanged** (const QItemSelection &, const QItemSelection &) override
- void **setItemDelegate** (DItemDelegate \*delegate)
- void **setToolTip** (ItemViewToolTip \*tip)
- virtual bool **showToolTip** (const QModelIndex &index, QStyleOptionViewItem &option, QHelpEvent \*e=nullptr)  
*Provides default behavior, can reimplement in a subclass.*
- void **updateDelegateSizes** ()
- void **userInteraction** ()
- bool **viewportEvent** (QEvent \*event) override
- void **wheelEvent** (QWheelEvent \*event) override

## Protected Member Functions inherited from [Digikam::DCategorizedView](#)

- void **dragLeaveEvent** (QDragLeaveEvent \*event) override
- void **dragMoveEvent** (QDragMoveEvent \*event) override
- void **dropEvent** (QDropEvent \*event) override
- void **leaveEvent** (QEvent \*event) override
- void **mouseMoveEvent** (QMouseEvent \*event) override
- void **mousePressEvent** (QMouseEvent \*event) override
- void **mouseReleaseEvent** (QMouseEvent \*event) override
- QModelIndex **moveCursor** (CursorAction cursorAction, Qt::KeyboardModifiers modifiers) override
- void **paintEvent** (QPaintEvent \*event) override
- void **resizeEvent** (QResizeEvent \*event) override
- void **setSelection** (const QRect &rect, QItemSelectionModel::SelectionFlags flags) override
- void **startDrag** (Qt::DropActions supportedActions) override

## Protected Member Functions inherited from [Digikam::DragDropViewImplementation](#)

- virtual `QAbstractItemView * asView ()=0`  
*This one is implemented by DECLARE\_VIEW\_DRAG\_DROP\_METHODS.*
- bool **decodelsCutSelection** (const `QMimeData *mimeData`)
- void **dragEnterEvent** (`QDragEnterEvent *event`)  
*Implements the relevant QAbstractItemView methods for drag and drop.*
- void **dragMoveEvent** (`QDragMoveEvent *e`)
- void **dropEvent** (`QDropEvent *e`)
- void **encodelsCutSelection** (`QMimeData *mime`, bool `isCutSelection`)
- void **startDrag** (`Qt::DropActions supportedActions`)

## Additional Inherited Members

## Protected Slots inherited from [Digikam::ImportCategorizedView](#)

- void **slotCamItemInfosAdded** ()

## Protected Slots inherited from [Digikam::ItemViewCategorized](#)

- void **layoutAboutToBeChanged** ()
- void **layoutWasChanged** ()
- void **slotActivated** (const `QModelIndex &index`)
- void **slotClicked** (const `QModelIndex &index`)
- void **slotEntered** (const `QModelIndex &index`)
- virtual void **slotThemeChanged** ()

## Protected Slots inherited from [Digikam::DCategorizedView](#)

- void **currentChanged** (const `QModelIndex &current`, const `QModelIndex &previous`) override
- void **rowsInserted** (const `QModelIndex &parent`, int `start`, int `end`) override
- virtual void **rowsInsertedArtificial** (const `QModelIndex &parent`, int `start`, int `end`)
- virtual void **slotLayoutChanged** ()
- void **updateGeometries** () override

## 9.755.1 Member Function Documentation

### 9.755.1.1 activated()

```
void Digikam::ImportIconView::activated (
    const CamItemInfo & info,
    Qt::KeyboardModifiers modifiers ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ImportCategorizedView](#).

### 9.755.1.2 setThumbnailSize()

```
void Digikam::ImportIconView::setThumbnailSize (
    const ThumbnailSize & size ) [override], [virtual]
```

Reimplemented from [Digikam::ImportCategorizedView](#).

### 9.755.1.3 showContextMenu()

```
void Digikam::ImportIconView::showContextMenu (
    QContextMenuEvent * event ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ItemViewCategorized](#).

### 9.755.1.4 showContextMenuOnInfo()

```
void Digikam::ImportIconView::showContextMenuOnInfo (
    QContextMenuEvent * event,
    const CamItemInfo & info ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ImportCategorizedView](#).

### 9.755.1.5 slotSetupChanged()

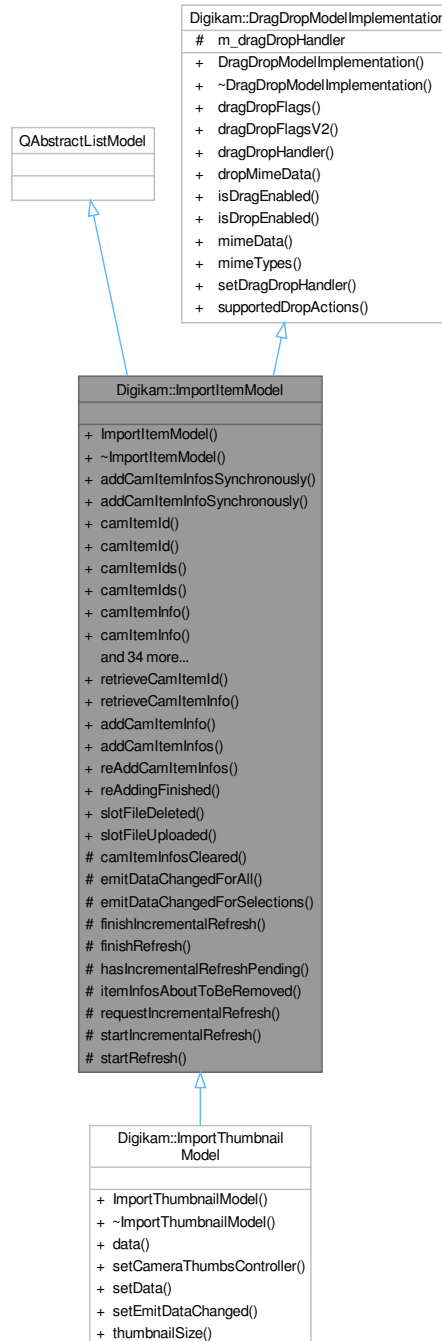
```
void Digikam::ImportIconView::slotSetupChanged ( ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ItemViewCategorized](#).



## 9.756 Digikam::ImportItemModel Class Reference

Inheritance diagram for Digikam::ImportItemModel:



### Public Types

- enum [ImportItemModelRoles](#) {
  - [ImportItemModelPointerRole](#) = Qt::UserRole , [ImportItemModelInternalId](#) = Qt::UserRole + 1 ,
  - [ThumbnailRole](#) = Qt::UserRole + 2 , [ExtraDataRole](#) = Qt::UserRole + 3 ,
  - [ExtraDataDuplicateCount](#) = Qt::UserRole + 6 , [FilterModelRoles](#) = Qt::UserRole + 100 }

## Public Slots

- void **addCamItemInfo** (const [CamItemInfo](#) &info)
- void **addCamItemInfos** (const [CamItemInfoList](#) &infos)
- void **reAddCamItemInfos** (const [CamItemInfoList](#) &infos)
- void **reAddingFinished** ()
- void **slotFileDeleted** (const QString &folder, const QString &file, bool status)
- void **slotFileUploaded** (const [CamItemInfo](#) &info)

## Signals

- void [allRefreshingFinished](#) ()  
*Signals that the model has finished currently with all scheduled refreshing, full or incremental, and all preprocessing.*
- void [itemInfosAboutToBeAdded](#) (const QList< [CamItemInfo](#) > &infos)  
*Informs that ItemInfos will be added to the model.*
- void [itemInfosAboutToBeRemoved](#) (const QList< [CamItemInfo](#) > &infos)  
*Informs that CamItemInfos will be removed from the model.*
- void [itemInfosAdded](#) (const QList< [CamItemInfo](#) > &infos)  
*Informs that ItemInfos have been added to the model.*
- void [itemInfosRemoved](#) (const QList< [CamItemInfo](#) > &infos)  
*Informs that CamItemInfos have been removed from the model.*
- void **preprocess** (const QList< [CamItemInfo](#) > &infos)  
*Connect to this signal only if you are the current preprocessor.*
- void **processAdded** (const QList< [CamItemInfo](#) > &infos)
- void [readyForIncrementalRefresh](#) ()  
*Signals that the model is right now ready to start an incremental refresh.*

## Public Member Functions

- **ImportItemModel** (QObject \*const parent=nullptr)
- void **addCamItemInfosSynchronously** (const [Digikam::CamItemInfoList](#) &infos)
- void [addCamItemInfoSynchronously](#) (const [CamItemInfo](#) &info)  
*addCamItemInfo() is asynchronous if a preprocessor is set.*
- qlonglong **camItemId** (const [QModelIndex](#) &index) const
- qlonglong **camItemId** (int row) const
- QList< qlonglong > **camItemIds** () const
- QList< qlonglong > **camItemIds** (const QList< [QModelIndex](#) > &indexes) const
- [CamItemInfo](#) **camItemInfo** (const [QModelIndex](#) &index) const  
*Returns the [CamItemInfo](#) object, reference from the underlying data pointed to by the index.*
- [CamItemInfo](#) **camItemInfo** (const [QUrl](#) &fileUrl) const
- [CamItemInfo](#) **camItemInfo** (int row) const  
*Returns the [CamItemInfo](#) object, reference from the underlying data of the given row (parent is the invalid [QModelIndex](#), column is 0).*
- [CamItemInfo](#) & **camItemInfoRef** (const [QModelIndex](#) &index) const
- [CamItemInfo](#) & **camItemInfoRef** (int row) const
- QList< [CamItemInfo](#) > **camItemInfos** () const
- [CamItemInfoList](#) **camItemInfos** (const QList< [QModelIndex](#) > &indexes) const
- QList< [CamItemInfo](#) > **camItemInfos** (const [QUrl](#) &fileUrl) const
- void **clearCamItemInfos** ()  
*Clears the CamItemInfos and resets the model.*
- [QVariant](#) **data** (const [QModelIndex](#) &index, int role) const override

- Qt::ItemFlags **flags** (const QModelIndex &index) const override
- bool **hasImage** (const CamItemInfo &info) const
- bool **hasImage** (qulonglong id) const
- QVariant **headerData** (int section, Qt::Orientation orientation, int role) const override
- QModelIndex **index** (int row, int column, const QModelIndex &parent) const override
- QList< QModelIndex > **indexesForCamItemId** (qulonglong id) const
- QList< QModelIndex > **indexesForCamItemInfo** (const CamItemInfo &info) const
- QList< QModelIndex > **indexesForUrl** (const QUrl &fileUrl) const
- QModelIndex **indexForCamItemId** (qulonglong id) const
- QModelIndex **indexForCamItemInfo** (const CamItemInfo &info) const
  - Return the index of a given CamItemInfo, if it exists in the model.*
- QModelIndex **indexForUrl** (const QUrl &fileUrl) const
  - Returns the index or CamItemInfo object from the underlying data for the given file url.*
- bool **isEmpty** () const
- bool **isRefreshing** () const
  - Returns true if this model is currently refreshing.*
- bool **keepsFileUrlCache** () const
- int **numberOfIndexesForCamItemId** (qulonglong id) const
- int **numberOfIndexesForCamItemInfo** (const CamItemInfo &info) const
- void **removeCamItemInfo** (const CamItemInfo &info)
- void **removeCamItemInfos** (const QList< CamItemInfo > &infos)
- void **removeIndex** (const QModelIndex &index)
  - Remove the given infos or indexes directly from the model.*
- void **removeIndexes** (const QList< QModelIndex > &indexes)
- int **rowCount** (const QModelIndex &parent) const override
  - QAbstractListModel implementation.*
- virtual void **setCameraThumbsController** (CameraThumbsCtrl \*const controller)
  - Used to set the camera controller, and connect with it.*
- void **setCamItemInfos** (const CamItemInfoList &infos)
  - Clears and adds infos.*
- void **setKeepsFileUrlCache** (bool keepCache)
  - If a cache is kept, lookup by file path is fast, without a cache it is O(n).*
- DECLARE\_MODEL\_DRAG\_DROP\_METHODS void **setSendRemovalSignals** (bool send)
  - DragDrop methods.*
- QList< CamItemInfo > **uniqueCamItemInfos** () const

## Public Member Functions inherited from Digikam::DragDropModelImplementation

- **DragDropModelImplementation** ()=default
  - A class providing a sample implementation for a QAbstractItemModel redirecting drag-and-drop support to a handler.*
- virtual Qt::ItemFlags **dragDropFlags** (const QModelIndex &index) const
  - Call from your flags() method, adding the relevant drag drop flags.*
- Qt::ItemFlags **dragDropFlagsV2** (const QModelIndex &index) const
  - This is an alternative approach to dragDropFlags().*
- **AbstractItemDragDropHandler** \* **dragDropHandler** () const
- bool **dropMimeData** (const QMimeData \*, Qt::DropAction, int, int, const QModelIndex &)
- virtual bool **isDragEnabled** (const QModelIndex &index) const
- virtual bool **isDropEnabled** (const QModelIndex &index) const
- QMimeData \* **mimeData** (const QModelIndexList &indexes) const
- QStringList **mimeTypes** () const
- void **setDragDropHandler** (**AbstractItemDragDropHandler** \*handler)
  - Set a drag drop handler.*
- Qt::DropActions **supportedDropActions** () const
  - Implements the relevant QAbstractItemModel methods for drag and drop.*

## Static Public Member Functions

- static qlonglong **retrieveCamItemId** (const QModelIndex &index)
- static [CamItemInfo](#) **retrieveCamItemInfo** (const QModelIndex &index)

Retrieve the [CamItemInfo](#) object from the `data()` function of the given index. The index may be from a `QSortFilterProxyModel` as long as an [ImportItemModel](#) is at the end.

## Protected Member Functions

- virtual void **camItemInfosCleared** ()  
Called when the internal storage is cleared.
- void **emitDataChangedForAll** ()
- void **emitDataChangedForSelections** (const QItemSelection &selection)
- void **finishIncrementalRefresh** ()
- void **finishRefresh** ()
- bool **hasIncrementalRefreshPending** () const
- virtual void **itemInfosAboutToBeRemoved** (int, int)  
Called before `rowsAboutToBeRemoved`.
- void [requestIncrementalRefresh](#) ()  
As soon as the model is ready to start an incremental refresh, the signal [readyForIncrementalRefresh\(\)](#) will be emitted.
- void [startIncrementalRefresh](#) ()  
Starts an incremental refresh operation.
- void [startRefresh](#) ()  
Subclasses that add `CamItemInfos` in batches shall call [startRefresh\(\)](#) when they start sending batches and `finishRefresh()` when they have finished.

## Additional Inherited Members

## Protected Attributes inherited from [Digikam::DragDropModelImplementation](#)

- [AbstractItemDragDropHandler](#) \* `m_dragDropHandler` = nullptr

## 9.756.1 Member Enumeration Documentation

### 9.756.1.1 [ImportItemModelRoles](#)

```
enum Digikam::ImportItemModel::ImportItemModelRoles
```

#### Enumerator

<code>ImportItemModelPointerRole</code>	An <code>ImportItemModel*</code> pointer to this model.
<code>ThumbnailRole</code>	Returns a thumbnail pixmap. May be implemented by subclasses. Returns either a valid pixmap or a null <code>QVariant</code> .
<code>ExtraDataRole</code>	Return (optional) <code>extraData</code> field.
<code>ExtraDataDuplicateCount</code>	Returns the number of duplicate indexes for the same image id.

## 9.756.2 Member Function Documentation

### 9.756.2.1 addCamItemInfoSynchronously()

```
void Digikam::ImportItemModel::addCamItemInfoSynchronously (
    const CamItemInfo & info )
```

This method first adds the info, synchronously. Only afterwards, the preprocessor will have the opportunity to process it. This method also bypasses any incremental updates.

### 9.756.2.2 allRefreshingFinished

```
void Digikam::ImportItemModel::allRefreshingFinished ( ) [signal]
```

The model is in polished, clean situation right now.

### 9.756.2.3 camItemInfo() [1/2]

```
CamItemInfo Digikam::ImportItemModel::camItemInfo (
    const QModelIndex & index ) const
```

For `camItemInfo` and `camItemInfoRef` If the index is not valid they will return a null [CamItemInfo](#), and 0 respectively, `camItemInfoRef` must not be called with an invalid index as it will crash.

### 9.756.2.4 camItemInfo() [2/2]

```
CamItemInfo Digikam::ImportItemModel::camItemInfo (
    int row ) const
```

Note that `camItemInfoRef` must not be called with an invalid index as it will crash.

### 9.756.2.5 indexForUrl()

```
QModelIndex Digikam::ImportItemModel::indexForUrl (
    const QUrl & fileUrl ) const
```

In case of multiple occurrences of the same file, the simpler overrides returns any one found first, use the `QList` methods to retrieve all occurrences.

### 9.756.2.6 isRefreshing()

```
bool Digikam::ImportItemModel::isRefreshing ( ) const
```

For a preprocessor this means that, although the preprocessor may currently have processed all it got, more batches are to be expected.

### 9.756.2.7 itemInfosAboutToBeAdded

```
void Digikam::ImportItemModel::itemInfosAboutToBeAdded (
    const QList< CamItemInfo > & infos ) [signal]
```

This signal is sent before the model data is changed and views are informed.

### 9.756.2.8 itemInfosAboutToBeRemoved

```
void Digikam::ImportItemModel::itemInfosAboutToBeRemoved (
    const QList< CamItemInfo > & infos ) [signal]
```

This signal is sent before the model data is changed and views are informed. Note: You need to explicitly enable sending of this signal. It is not sent in [clearCamItemInfos\(\)](#).

### 9.756.2.9 itemInfosAdded

```
void Digikam::ImportItemModel::itemInfosAdded (
    const QList< CamItemInfo > & infos ) [signal]
```

This signal is sent after the model data is changed and views are informed.

### 9.756.2.10 itemInfosRemoved

```
void Digikam::ImportItemModel::itemInfosRemoved (
    const QList< CamItemInfo > & infos ) [signal]
```

This signal is sent after the model data is changed and views are informed. Note: You need to explicitly enable sending of this signal. It is not sent in [clearCamItemInfos\(\)](#).

### 9.756.2.11 readyForIncrementalRefresh

```
void Digikam::ImportItemModel::readyForIncrementalRefresh ( ) [signal]
```

This is guaranteed only for the scope of emitting this signal.

### 9.756.2.12 requestIncrementalRefresh()

```
void Digikam::ImportItemModel::requestIncrementalRefresh ( ) [protected]
```

The signal will be emitted inline if the model is ready right now.

### 9.756.2.13 setCameraThumbsController()

```
void Digikam::ImportItemModel::setCameraThumbsController (
    CameraThumbsCtrl *const controller ) [virtual]
```

Reimplemented in [Digikam::ImportThumbnailModel](#).

#### 9.756.2.14 setKeepsFileUrlCache()

```
void DigiKam::ImportItemModel::setKeepsFileUrlCache (
    bool keepCache )
```

Default is false.

#### 9.756.2.15 setSendRemovalSignals()

```
void DigiKam::ImportItemModel::setSendRemovalSignals (
    bool send )
```

Enable sending of `itemInfosAboutToBeRemoved` and `itemsInfosRemoved` signals. Default: false

#### 9.756.2.16 startIncrementalRefresh()

```
void DigiKam::ImportItemModel::startIncrementalRefresh ( ) [protected]
```

You shall only call this method from a slot connected to [readyForIncrementalRefresh\(\)](#). To initiate an incremental refresh, call [requestIncrementalRefresh\(\)](#).

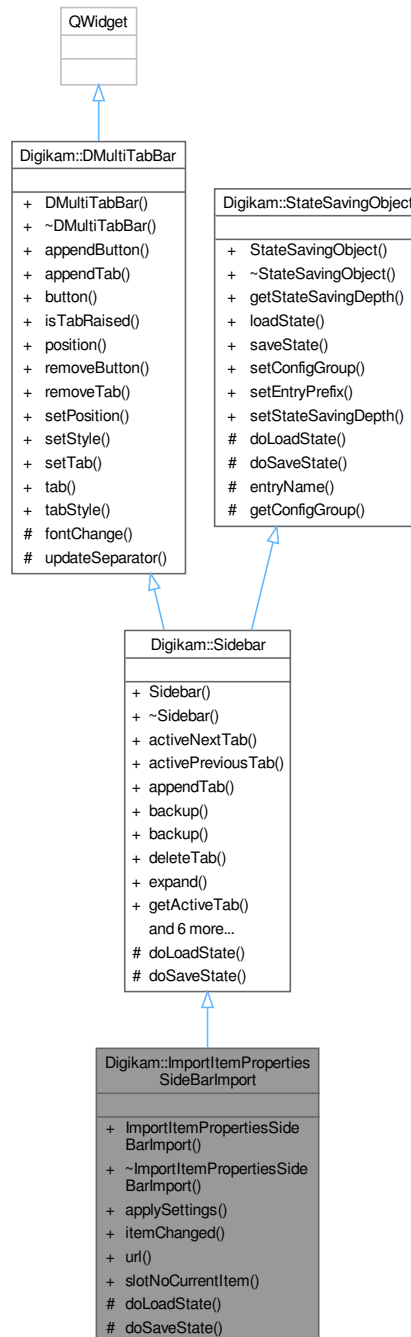
#### 9.756.2.17 startRefresh()

```
void DigiKam::ImportItemModel::startRefresh ( ) [protected]
```

No incremental refreshes will be started while listing. A [clearCamItemInfos\(\)](#) always stops listing, calling `finishRefresh()` is then not necessary.

## 9.757 Digikam::ImportItemPropertiesSideBarImport Class Reference

Inheritance diagram for Digikam::ImportItemPropertiesSideBarImport:



### Public Slots

- virtual void `slotNoCurrentItem ()`



## Signals

- void **signalFirstItem** ()
- void **signalLastItem** ()
- void **signalNextItem** ()
- void **signalPrevItem** ()

## Signals inherited from [Digikam::Sidebar](#)

- void **signalChangedTab** (QWidget \*w)  
*Is emitted, when another tab is activated.*
- void **signalViewChanged** ()  
*Is emitted, when tab is shrink or expanded.*

## Public Member Functions

- **ImportItemPropertiesSideBarImport** (QWidget \*const parent, [SidebarSplitter](#) \*const splitter, Qt::Edge side=Qt::LeftEdge, bool mimimizedDefault=false)
- void **applySettings** ()
- void **itemChanged** (const [CamItemInfo](#) &itemInfo, const [DMetadata](#) &meta)
- [QUrl](#) **url** () const

## Public Member Functions inherited from [Digikam::Sidebar](#)

- [Sidebar](#) (QWidget \*const parent, [SidebarSplitter](#) \*const sp, Qt::Edge side=Qt::LeftEdge, bool minimized↔ Default=false)  
*Creates a new sidebar.*
- void **activeNextTab** ()  
*Activates a next tab from current one.*
- void **activePreviousTab** ()  
*Activates a previous tab from current one.*
- void **appendTab** (QWidget \*const w, const [QIcon](#) &pic, const [QString](#) &title)  
*Appends a new tab to the sidebar.*
- void **backup** ()  
*Hide sidebar and backup minimized state.*
- void **backup** (const [QList](#)< [QWidget](#) \* > &thirdWidgetsToBackup, [QList](#)< int > \*const sizes)  
*Hide sidebar and backup minimized state.*
- void **deleteTab** (QWidget \*const w)  
*Deletes a tab from the tabbar.*
- void **expand** ()  
*Redisplays the whole sidebar.*
- [QWidget](#) \* **getActiveTab** () const  
*Returns the currently activated tab, or 0 if no tab is active.*
- bool **isExpanded** () const  
*Return the visible status of current sidebar tab.*
- void **restore** ()  
*Show sidebar and restore minimized state.*
- void **restore** (const [QList](#)< [QWidget](#) \* > &thirdWidgetsToRestore, const [QList](#)< int > &sizes)  
*Show sidebar and restore minimized state.*
- void **setActiveTab** (QWidget \*const w)  
*Activates a tab.*
- void **shrink** ()  
*Hides the sidebar (display only the activation buttons)*
- [SidebarSplitter](#) \* **splitter** () const

## Public Member Functions inherited from [Digikam::DMultiTabBar](#)

- **DMultiTabBar** (Qt::Edge pos, QWidget \*const parent=nullptr)
- void [appendButton](#) (const QIcon &pic, int id=-1, QMenu \*const popup=nullptr, const QString &not\_used\_↔ yet=QString())  
*append a new button to the button area.*
- void [appendTab](#) (const QIcon &pic, int id=-1, const QString &text=QString())  
*append a new tab to the tab area.*
- [DMultiTabBarButton](#) \* **button** (int id) const  
*get a pointer to a button within the button area identified by its ID*
- bool **isTabRaised** (int id) const  
*return the state of a tab, identified by its ID*
- Qt::Edge [position](#) () const  
*get the tabbar position.*
- void **removeButton** (int id)  
*remove a button with the given ID*
- void **removeTab** (int id)  
*remove a tab with a given ID*
- void [setPosition](#) (Qt::Edge pos)  
*set the real position of the widget.*
- void **setStyle** ([TextStyle](#) style)  
*set the display style of the tabs*
- void [setTab](#) (int id, bool state)  
*set a tab to "raised"*
- [DMultiTabBarTab](#) \* **tab** (int id) const  
*get a pointer to a tab within the tab area, identified by its ID*
- [TextStyle](#) [tabStyle](#) () const  
*get the display style of the tabs*

## Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (QObject \*const host)  
*Constructor.*
- virtual ~**StateSavingObject** ()  
*Destructor.*
- [StateSavingDepth](#) [getStateSavingDepth](#) () const  
*Returns the depth used for state saving or loading.*
- void **loadState** ()  
*Invokes loading the class' state.*
- void **saveState** ()  
*Invokes saving the class' state.*
- virtual void [setConfigGroup](#) (const KConfigGroup &group)  
*Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void [setEntryPrefix](#) (const QString &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void [setStateSavingDepth](#) (const [StateSavingDepth](#) depth)  
*Sets the depth used for state saving or loading.*

### Protected Member Functions

- void [doLoadState](#) () override  
*load the last view state from disk - called by [StateSavingObject::loadState\(\)](#)*
- void [doSaveState](#) () override  
*save the view state to disk - called by [StateSavingObject::saveState\(\)](#)*

### Protected Member Functions inherited from [Digikam::Sidebar](#)

- void [doLoadState](#) () override  
*Load the last view state from disk - called by [StateSavingObject::loadState\(\)](#)*
- void [doSaveState](#) () override  
*Save the view state to disk - called by [StateSavingObject::saveState\(\)](#)*

### Protected Member Functions inherited from [Digikam::DMultiTabBar](#)

- virtual void [fontChange](#) (const QFont &)
- void [updateSeparator](#) ()

### Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- QString [entryName](#) (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup [getConfigGroup](#) () const  
*Returns the config group that must be used for state saving and loading.*

### Additional Inherited Members

### Public Types inherited from [Digikam::DMultiTabBar](#)

- enum [TextStyle](#) { [ActiveIconText](#) = 0 , [AllIconsText](#) = 2 }
- The list of available styles for [DMultiTabBar](#).*

### Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }
- This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

## 9.757.1 Member Function Documentation

### 9.757.1.1 [applySettings\(\)](#)

```
void Digikam::ImportItemPropertiesSideBarImport::applySettings ( )
```

**9.757.1.2 doLoadState()**

```
void Digikam::ImportItemPropertiesSideBarImport::doLoadState ( ) [override], [protected],  
[virtual]
```

Implements [Digikam::StateSavingObject](#).

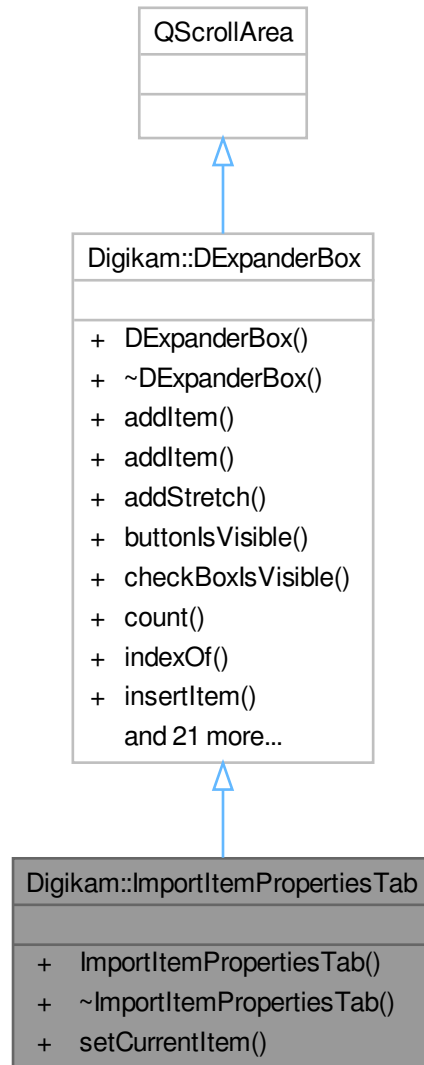
**9.757.1.3 doSaveState()**

```
void Digikam::ImportItemPropertiesSideBarImport::doSaveState ( ) [override], [protected],  
[virtual]
```

Implements [Digikam::StateSavingObject](#).

## 9.758 Digikam::ImportItemPropertiesTab Class Reference

Inheritance diagram for Digikam::ImportItemPropertiesTab:



### Public Member Functions

- **ImportItemPropertiesTab** (`QWidget *const parent`)
- void **setCurrentItem** (`const CamItemInfo &itemInfo=CamItemInfo()`, `DMetadata *const meta=nullptr`)

### Public Member Functions inherited from [Digikam::DExpanderBox](#)

- **DExpanderBox** (`QWidget *const parent=nullptr`)

- void **addItem** (QWidget \*const w, const QIcon &icon, const QString &txt, const QString &objName, bool expandBydefault)
- *Add [DLabelExpander](#) item at end of box layout with these settings : 'w' : the widget hosted by [DLabelExpander](#).*
- void **addItem** (QWidget \*const w, const QString &txt, const QString &objName, bool expandBydefault)
- void **addStretch** ()
- bool **buttonIsVisible** (int index) const
- bool **checkboxIsVisible** (int index) const
- int **count** () const
- int **indexOf** ([DLabelExpander](#) \*const widget) const
- void **insertItem** (int index, QWidget \*const w, const QIcon &icon, const QString &txt, const QString &objName, bool expandBydefault)
- *Insert [DLabelExpander](#) item at box layout index with these settings : 'w' : the widget hosted by [DLabelExpander](#).*
- void **insertItem** (int index, QWidget \*const w, const QString &txt, const QString &objName, bool expandBydefault)
- void **insertStretch** (int index)
- bool **isChecked** (int index) const
- bool **isItemEnabled** (int index) const
- bool **isItemExpanded** (int index) const
- QIcon **itemIcon** (int index) const
- QString **itemText** (int index) const
- QString **itemToolTip** (int index) const
- virtual void **readSettings** (KConfigGroup &group)
- void **removeItem** (int index)
- void **setButtonIcon** (int index, const QIcon &icon)
- void **setButtonVisible** (int index, bool b)
- void **setCheckBoxVisible** (int index, bool b)
- void **setChecked** (int index, bool b)
- void **setItemEnabled** (int index, bool enabled)
- void **setItemExpanded** (int index, bool b)
- void **setItemIcon** (int index, const QIcon &icon)
- void **setItemText** (int index, const QString &txt)
- void **setItemToolTip** (int index, const QString &tip)
- [DLabelExpander](#) \* **widget** (int index) const
- virtual void **writeSettings** (KConfigGroup &group)

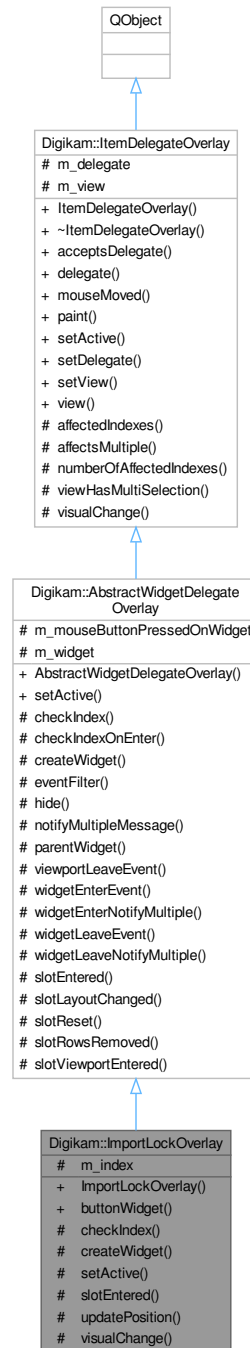
### Additional Inherited Members

### Signals inherited from [Digikam::DExpanderBox](#)

- void **signalItemButtonPressed** (int index)
- void **signalItemExpanded** (int index, bool b)
- void **signalItemToggled** (int index, bool b)

## 9.759 Digikam::ImportLockOverlay Class Reference

Inheritance diagram for Digikam::ImportLockOverlay:



### Public Member Functions

- **ImportLockOverlay** (QObject \*const parent)
- **ImportOverlayWidget** \* **buttonWidget** () const

## Public Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- [AbstractWidgetDelegateOverlay](#) (QObject \*const parent)  
*This class provides functionality for using a widget in an overlay.*

## Public Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- [ItemDelegateOverlay](#) (QObject \*const parent=nullptr)
- virtual bool [acceptsDelegate](#) (QAbstractItemDelegate \*) const
- QAbstractItemDelegate \* [delegate](#) () const
- virtual void [mouseMoved](#) (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)  
*Only these two methods are implemented as virtual methods.*
- virtual void [paint](#) (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index)
- void [setDelegate](#) (QAbstractItemDelegate \*delegate)
- void [setView](#) (QAbstractItemView \*view)
- QAbstractItemView \* [view](#) () const

## Protected Member Functions

- bool [checkIndex](#) (const QModelIndex &index) const override  
*Return true here if you want to show the overlay for the given index.*
- QWidget \* [createWidget](#) () override  
*Create your widget here.*
- void [setActive](#) (bool active) override  
*If active is true, this will call [createWidget\(\)](#), initialize the widget for use, and setup connections for the virtual slots.*
- void [slotEntered](#) (const QModelIndex &index) override  
*Default implementation shows the widget iff the index is valid and [checkIndex](#) returns true.*
- void [updatePosition](#) ()
- void [visualChange](#) () override  
*Called when any change from the delegate occurs - when the overlay is installed, when size hints, styles or fonts change.*

## Protected Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- bool [checkIndexOnEnter](#) (const QModelIndex &index) const  
*Utility method called from [slotEntered](#).*
- bool [eventFilter](#) (QObject \*obj, QEvent \*event) override
- virtual void [hide](#) ()  
*Called when the widget shall be hidden (mouse cursor left index, viewport, uninstalled etc.).*
- virtual QString [notifyMultipleMessage](#) (const QModelIndex &, int number)
- QWidget \* [parentWidget](#) () const  
*Returns the widget to be used as parent for your widget created in [createWidget\(\)](#)*
- virtual void [viewportLeaveEvent](#) (QObject \*obj, QEvent \*event)  
*Called when a `QEvent::Leave` of the viewport is received.*
- virtual void [widgetEnterEvent](#) ()  
*Called when a `QEvent::Enter` resp.*
- void [widgetEnterNotifyMultiple](#) (const QModelIndex &index)  
*A sample implementation for above methods.*
- virtual void [widgetLeaveEvent](#) ()
- void [widgetLeaveNotifyMultiple](#) ()



## Protected Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- `QList< QModelIndex > affectedIndexes` (const QModelIndex &index) const
- `bool affectsMultiple` (const QModelIndex &index) const  
*For the context that an overlay can affect multiple items: Assuming the currently overlaid index is given.*
- `int numberOfAffectedIndexes` (const QModelIndex &index) const
- `bool viewHasMultiSelection` () const  
*Utility method.*

## Protected Attributes

- `QPersistentModelIndex m_index`

## Protected Attributes inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- `bool m_mouseButtonPressedOnWidget` = false
- `QWidget * m_widget` = nullptr

## Protected Attributes inherited from [Digikam::ItemDelegateOverlay](#)

- `QAbstractItemDelegate * m_delegate` = nullptr
- `QAbstractItemView * m_view` = nullptr

## Additional Inherited Members

## Signals inherited from [Digikam::ItemDelegateOverlay](#)

- `void hideNotification` ()
- `void requestNotification` (const QModelIndex &index, const QString &message)
- `void update` (const QModelIndex &index)

## Protected Slots inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- `virtual void slotLayoutChanged` ()
- `virtual void slotReset` ()  
*Default implementations of these three slots call `hide()`*
- `virtual void slotRowsRemoved` (const QModelIndex &parent, int start, int end)
- `virtual void slotViewportEntered` ()

## Protected Slots inherited from [Digikam::ItemDelegateOverlay](#)

### 9.759.1 Member Function Documentation

#### 9.759.1.1 `checkIndex()`

```
bool Digikam::ImportLockOverlay::checkIndex (
    const QModelIndex & index ) const [override], [protected], [virtual]
```

The default implementation returns true.

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.759.1.2 createWidget()

```
QWidget * Digikam::ImportLockOverlay::createWidget ( ) [override], [protected], [virtual]
```

When creating the object, pass [parentWidget\(\)](#) as parent widget. Ownership of the object is passed. It will be deleted in [setActive\(false\)](#).

Implements [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.759.1.3 setActive()

```
void Digikam::ImportLockOverlay::setActive (
    bool active ) [override], [protected], [virtual]
```

If active is false, this will delete the widget and disconnect all signal from model and view to this object (!)

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.759.1.4 slotEntered()

```
void Digikam::ImportLockOverlay::slotEntered (
    const QModelIndex & index ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.759.1.5 visualChange()

```
void Digikam::ImportLockOverlay::visualChange ( ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ItemDelegateOverlay](#).

## 9.760 Digikam::ImportNormalDelegate Class Reference

Inheritance diagram for Digikam::ImportNormalDelegate:



### Public Member Functions

- **ImportNormalDelegate** ([ImportCategorizedView](#) \*const parent)

## Public Member Functions inherited from [Digikam::ImportDelegate](#)

- **ImportDelegate** (QWidget \*const parent)
- bool [acceptsActivation](#) (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*activationRect=nullptr) const override
- bool [acceptsToolTip](#) (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*tooltipRect=nullptr) const override
 

*These methods take four parameters: The position on viewport, the rect on viewport, the index, and optionally a parameter into which, if the return value is true, a rectangle can be written for which the return value will be true as well.*
- QRect **actualPixmapRect** (const QModelIndex &index) const
- int **calculatethumbSizeToFit** (int ws)
- [ImportCategoryDrawer](#) \* **categoryDrawer** () const
- QRect **coordinatesIndicatorRect** () const
- QRect **downloadIndicatorRect** () const
- QRect **groupIndicatorRect** () const
- QRect [imageInformationRect](#) () const override
 

*Returns the area where the image information is drawn, or null if empty / not supported.*
- QRect **lockIndicatorRect** () const
- void **paint** (QPainter \*painter, const QStyleOptionViewItem &option, const QModelIndex &index) const override
- QPixmap [pixmapForDrag](#) (const QStyleOptionViewItem &option, const QList< QModelIndex > &indexes) const override
- QRect [pixmapRect](#) () const override
 

*Returns the area where the pixmap is drawn, or null if not supported.*
- void [setDefaultViewOptions](#) (const QStyleOptionViewItem &option) override
 

*Style option with standard values to use for cached rendering.*
- void [setSpacing](#) (int spacing) override
- void **setView** ([ImportCategorizedView](#) \*view)
- QRect **tagsRect** () const

## Public Member Functions inherited from [Digikam::ItemViewImportDelegate](#)

- **ItemViewImportDelegate** (QWidget \*const parent)
- bool [acceptsActivation](#) (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*activationRect=nullptr) const override
- bool [acceptsToolTip](#) (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*tooltipRect=nullptr) const override
 

*These methods take four parameters: The position on viewport, the rect on viewport, the index, and optionally a parameter into which, if the return value is true, a rectangle can be written for which the return value will be true as well.*
- double **displayRatio** () const
- QSize [gridSize](#) () const override
 

*Returns the gridsize to be set by the view.*
- void [mouseMoved](#) (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index) override
- virtual QRect **ratingRect** () const
 

*Returns the rectangle where the rating is drawn, or a null rectangle if not supported.*
- QRect **rect** () const
- void [setDefaultViewOptions](#) (const QStyleOptionViewItem &option) override
 

*Style option with standard values to use for cached rendering.*
- void [setRatingEdited](#) (const QModelIndex &index)
 

*Can be used to temporarily disable drawing of the rating.*
- void [setSpacing](#) (int spacing) override

- void [setThumbnailSize](#) (const [ThumbnailSize](#) &thumbSize) override  
*reimplemented from [DItemDelegate](#)*
- QSize **sizeHint** (const [QStyleOptionViewItem](#) &option, const [QModelIndex](#) &index) const override
- int **spacing** () const
- [ThumbnailSize](#) **thumbnailSize** () const

## Public Member Functions inherited from [Digikam::DItemDelegate](#)

- [DItemDelegate](#) ([QObject](#) \*const parent=nullptr)

## Public Member Functions inherited from [Digikam::ItemDelegateOverlayContainer](#)

- [ItemDelegateOverlayContainer](#) ()=default  
*This is a sample implementation for delegate management methods, to be inherited by a delegate.*
- void **installOverlay** ([ItemDelegateOverlay](#) \*overlay)
- void **mouseMoved** ([QMouseEvent](#) \*e, const [QRect](#) &visualRect, const [QModelIndex](#) &index)
- [QList](#)< [ItemDelegateOverlay](#) \* > **overlays** () const
- void **removeAllOverlays** ()
- void **removeOverlay** ([ItemDelegateOverlay](#) \*overlay)
- void **setAllOverlaysActive** (bool active)
- void **setViewOnAllOverlays** ([QAbstractItemView](#) \*view)

## Protected Member Functions

- [ImportNormalDelegate](#) ([ImportNormalDelegatePrivate](#) &dd, [ImportCategorizedView](#) \*const parent)
- void [updateRects](#) () override  
*In a subclass, you need to implement this method to set up the rects for drawing.*

## Protected Member Functions inherited from [Digikam::ImportDelegate](#)

- [ImportDelegate](#) ([ImportDelegate::ImportDelegatePrivate](#) &dd, [QWidget](#) \*const parent)
- void [clearCaches](#) () override
- virtual void **clearModelDataCaches** ()  
*Reimplement to clear caches based on model indexes (hash on row number etc.) Change signals are listened to this is called whenever such properties become invalid.*
- void [invalidatePaintingCache](#) () override  
*reimplement these in subclasses*
- bool **onActualPixmapRect** (const [QPoint](#) &pos, const [QRect](#) &visualRect, const [QModelIndex](#) &index, [QRect](#) \*actualRect) const
- void **setModel** ([QAbstractItemModel](#) \*model)
- virtual [QPixmap](#) **thumbnailPixmap** (const [QModelIndex](#) &index) const
- void **updateActualPixmapRect** (const [QModelIndex](#) &index, const [QRect](#) &rect)
- virtual void [updateContentWidth](#) ()  
*Reimplement this to set contentWidth.*
- void [updateSizeRectsAndPxmmaps](#) () override

## Protected Member Functions inherited from [Digikam::ItemViewImportDelegate](#)

- **ItemViewImportDelegate** (ItemViewImportDelegatePrivate &dd, QWidget \*const parent)
- QAbstractItemDelegate \* **asDelegate** () override
 

*Returns the delegate, typically, the derived class.*
- void **drawColorLabelLine** (QPainter \*p, const QRect &pixRect, int colorId) const
- void **drawCreationDate** (QPainter \*p, const QRect &dateRect, const QDateTime &date) const
- void **drawDownloadIndicator** (QPainter \*p, const QRect &r, int itemType) const
- void **drawFileSize** (QPainter \*p, const QRect &r, qlonglong bytes) const
- void **drawFocusRect** (QPainter \*p, const QStyleOptionViewItem &option, bool isSelected) const
- void **drawGeolocationIndicator** (QPainter \*p, const QRect &r) const
- void **drawGroupIndicator** (QPainter \*p, const QRect &r, int numberOfGroupedImages, bool open) const
- void **drawImageFormat** (QPainter \*p, const QRect &dimsRect, const QString &mime) const
- void **drawImageSize** (QPainter \*p, const QRect &dimsRect, const QSize &dims) const
- void **drawLockIndicator** (QPainter \*p, const QRect &r, int lockStatus) const
- void **drawMouseOverRect** (QPainter \*p, const QStyleOptionViewItem &option) const
- void **drawName** (QPainter \*p, const QRect &nameRect, const QString &name) const
- void **drawPickLabelIcon** (QPainter \*p, const QRect &r, int pickLabel) const
- void **drawRating** (QPainter \*p, const QModelIndex &index, const QRect &ratingRect, int rating, bool isSelected) const
- void **drawTags** (QPainter \*p, const QRect &r, const QString &tagsString, bool isSelected) const
- QRect **drawThumbnail** (QPainter \*p, const QRect &thumbRect, const QPixmap &background, const QPixmap &thumbnail) const
 

*Use the tool methods for painting in subclasses.*
- void **prepareBackground** ()
- void **prepareFonts** ()
- void **prepareMetrics** (int maxWidth)
- void **prepareRatingPixmap** (bool composeOverBackground=true)
- QPixmap **ratingPixmap** (int rating, bool selected) const
 

*Returns the relevant pixmap from the cached rating pixmaps.*

## Protected Member Functions inherited from [Digikam::DItemDelegate](#)

- QString **squeezedTextCached** (QPainter \*const p, int width, const QString &text) const
- QPixmap **thumbnailBorderPixmap** (const QSize &pixSize, bool isGrouped=false) const

## Protected Member Functions inherited from [Digikam::ItemDelegateOverlayContainer](#)

- virtual void **drawOverlays** (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index) const
- virtual void **overlayDestroyed** (QObject \*o)
 

*Declare as slot in the derived class calling this method.*

## Additional Inherited Members

## Signals inherited from [Digikam::ItemViewImportDelegate](#)

- void **hideNotification** ()
- void **requestNotification** (const QModelIndex &index, const QString &message)

## Signals inherited from [Digikam::DItemDelegate](#)

- void **gridSizeChanged** (const QSize &newSize)
- void **visualChange** ()

## Static Public Member Functions inherited from [Digikam::ImportDelegate](#)

- static QPixmap **retrieveThumbnailPixmap** (const QModelIndex &index, int thumbnailSize)  
*Retrieve the thumbnail pixmap in given size for the [ImportItemModel::ThumbnailRole](#) for the given index from the given index, which must adhere to [ImportThumbnailModel](#) semantics.*

## Static Public Member Functions inherited from [Digikam::DItemDelegate](#)

- static QString **dateToString** (const QDateTime &datetime)
- static QPixmap **makeDragPixmap** (const QStyleOptionViewItem &option, const QList< QModelIndex > &indexes, double displayRatio, const QPixmap &suggestedPixmap=QPixmap())
- static QString **squeezedText** (const QFontMetrics &fm, int width, const QString &text)

## Protected Slots inherited from [Digikam::ImportDelegate](#)

- void **modelChanged** ()
- void **modelContentsChanged** ()

## Protected Slots inherited from [Digikam::ItemViewImportDelegate](#)

- void **overlayDestroyed** (QObject \*o) override
- void **slotSetupChanged** ()
- void **slotThemeChanged** ()

## Protected Attributes inherited from [Digikam::ItemViewImportDelegate](#)

- ItemViewImportDelegatePrivate \*const **d\_ptr** = nullptr

## Protected Attributes inherited from [Digikam::ItemDelegateOverlayContainer](#)

- QList< [ItemDelegateOverlay](#) \* > **m\_overlays**

## 9.760.1 Member Function Documentation

### 9.760.1.1 updateRects()

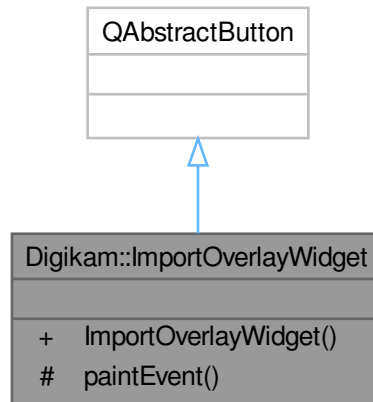
```
void Digikam::ImportNormalDelegate::updateRects ( ) [override], [protected], [virtual]
```

The paint() method operates depending on these rects.

Implements [Digikam::ImportDelegate](#).

## 9.761 Digikam::ImportOverlayWidget Class Reference

Inheritance diagram for Digikam::ImportOverlayWidget:



### Public Member Functions

- **ImportOverlayWidget** (`QWidget *const parent=nullptr`)

### Protected Member Functions

- void **paintEvent** (`QPaintEvent *`) override



## 9.762 Digikam::ImportPreviewView Class Reference

Inheritance diagram for Digikam::ImportPreviewView:



### Public Types

- enum **Mode** { **IconViewPreview** }

## Signals

- void **signalAssignColorLabel** (int)
- void **signalAssignPickLabel** (int)
- void **signalAssignRating** (int)
- void **signalDeleteItem** ()
- void **signalEscapePreview** ()
- void **signalNextItem** ()
- void **signalPreviewLoaded** (bool success)
- void **signalPrevItem** ()

## Signals inherited from [Digikam::GraphicsDImgView](#)

- void **activated** ()
- void **contentsMoved** (bool panningFinished)
- void **contentsMoving** (int, int)
- void **leftButtonClicked** ()
- void **leftButtonDoubleClicked** ()
- void **resized** ()
- void **rightButtonClicked** ()
- void **toNextImage** ()
- void **toPreviousImage** ()
- void **viewportRectChanged** (const QRectF &viewportRect)

## Public Member Functions

- **ImportPreviewView** (QWidget \*const parent, Mode mode=IconViewPreview)
- **CamItemInfo** **getCamItemInfo** () const
- void **reload** ()
- void **setCamItemInfo** (const [CamItemInfo](#) &info=[CamItemInfo](#)(), const [CamItemInfo](#) &previous=[CamItemInfo](#)(), const [CamItemInfo](#) &next=[CamItemInfo](#)())
- void **setCamItemPath** (const QString &path=QString())
- void **setPreviousNextPaths** (const QString &previous, const QString &next)
- void **showContextMenu** (const [CamItemInfo](#) &info, QGraphicsSceneContextMenuEvent \*event)

## Public Member Functions inherited from [Digikam::GraphicsDImgView](#)

- **GraphicsDImgView** (QWidget \*const parent=nullptr)
- int **contentsX** () const
- int **contentsY** () const
- void **drawText** (QPainter \*p, const QRectF &rect, const QString &text)
- void **fitToWindow** ()
- [GraphicsDImgItem](#) \* **item** () const  
*Return the instance of item set by [setItem\(\)](#).*
- [SinglePhotoPreviewLayout](#) \* **layout** () const
- [DImgPreviewItem](#) \* **previewItem** () const  
*Return a cast of item instance of item set by [setItem\(\)](#) as [DImgPreviewItem](#) Note: if you store a [GraphicsDImgItem](#) object using [setItem\(\)](#), this method will return 0.*
- void **scrollPointOnPoint** (const QPointF &scenePos, const QPoint &viewportPos)  
*Scrolls the view such that scenePos (in scene coordinates) is displayed on the viewport at viewportPos (in viewport coordinates).*
- void **setContentPos** (int x, int y)
- void **setItem** ([GraphicsDImgItem](#) \*const item)  
*Store internal instance of item as [GraphicsDImgItem](#).*
- void **toggleFullScreen** (bool set)
- QRect **visibleArea** () const

### Protected Member Functions

- bool [acceptsMouseClicked](#) (QMouseEvent \*e) override
- void [enterEvent](#) (QEnterEvent \*) override
- void [leaveEvent](#) (QEvent \*e) override
- void [showEvent](#) (QShowEvent \*e) override

### Protected Member Functions inherited from [Digikam::GraphicsDImgView](#)

- void [continuePanning](#) (const QPoint &pos)
- void [drawForeground](#) (QPainter \*painter, const QRectF &rect) override
- void [finishPanning](#) ()
- void [installPanIcon](#) ()
- void [mouseDoubleClickEvent](#) (QMouseEvent \*) override
- void [mouseMoveEvent](#) (QMouseEvent \*) override
- void [mousePressEvent](#) (QMouseEvent \*) override
- void [mouseReleaseEvent](#) (QMouseEvent \*) override
- void [resizeEvent](#) (QResizeEvent \*) override
- void [scrollContentsBy](#) (int dx, int dy) override
- void [setScaleFitToWindow](#) (bool value)
- void [setShowText](#) (bool value)
- void [startPanning](#) (const QPoint &pos)
- void [wheelEvent](#) (QWheelEvent \*) override

### Additional Inherited Members

### Protected Slots inherited from [Digikam::GraphicsDImgView](#)

- void [slotContentsMoved](#) ()
- void [slotCornerButtonPressed](#) ()
- void [slotPanIconHidden](#) ()
- virtual void [slotPanIconSelectionMoved](#) (const QRect &, bool)

## 9.762.1 Member Function Documentation

### 9.762.1.1 [acceptsMouseClicked\(\)](#)

```
bool Digikam::ImportPreviewView::acceptsMouseClicked (
    QMouseEvent * e ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::GraphicsDImgView](#).

## 9.763 Digikam::ImportRatingOverlay Class Reference

Inheritance diagram for Digikam::ImportRatingOverlay:



### Signals

- void **ratingEdited** (const QList< QModelIndex > &indexes, int rating)

## Signals inherited from [Digikam::ItemDelegateOverlay](#)

- void **hideNotification** ()
- void **requestNotification** (const QModelIndex &index, const QString &message)
- void **update** (const QModelIndex &index)

## Public Member Functions

- **ImportRatingOverlay** (QObject \*const parent)
- [RatingWidget](#) \* **ratingWidget** () const

## Public Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- [AbstractWidgetDelegateOverlay](#) (QObject \*const parent)  
*This class provides functionality for using a widget in an overlay.*

## Public Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- **ItemDelegateOverlay** (QObject \*const parent=nullptr)
- virtual bool **acceptsDelegate** (QAbstractItemDelegate \*) const
- QAbstractItemDelegate \* **delegate** () const
- virtual void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)  
*Only these two methods are implemented as virtual methods.*
- virtual void **paint** (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index)
- void **setDelegate** (QAbstractItemDelegate \*delegate)
- void **setView** (QAbstractItemView \*view)
- QAbstractItemView \* **view** () const

## Protected Slots

- void **slotDataChanged** (const QModelIndex &, const QModelIndex &)
- void **slotRatingChanged** (int)

## Protected Slots inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- virtual void **slotLayoutChanged** ()
- virtual void **slotReset** ()  
*Default implementations of these three slots call [hide\(\)](#)*
- virtual void **slotRowsRemoved** (const QModelIndex &parent, int start, int end)
- virtual void **slotViewportEntered** ()

## Protected Slots inherited from [Digikam::ItemDelegateOverlay](#)

### Protected Member Functions

- QWidget \* [createWidget](#) () override  
*Create your widget here.*
- void [hide](#) () override  
*Called when the widget shall be hidden (mouse cursor left index, viewport, uninstalled etc.).*
- void [setActive](#) (bool) override  
*If active is true, this will call [createWidget\(\)](#), initialize the widget for use, and setup connections for the virtual slots.*
- void [slotEntered](#) (const QModelIndex &index) override  
*Default implementation shows the widget iff the index is valid and [checkIndex](#) returns true.*
- void [updatePosition](#) ()
- void [updateRating](#) ()
- void [visualChange](#) () override  
*Called when any change from the delegate occurs - when the overlay is installed, when size hints, styles or fonts change.*
- void [widgetEnterEvent](#) () override  
*Called when a [QEvent::Enter](#) resp.*
- void [widgetLeaveEvent](#) () override

### Protected Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- virtual bool [checkIndex](#) (const QModelIndex &index) const  
*Return true here if you want to show the overlay for the given index.*
- bool [checkIndexOnEnter](#) (const QModelIndex &index) const  
*Utility method called from [slotEntered](#).*
- bool [eventFilter](#) (QObject \*obj, QEvent \*event) override
- virtual QString [notifyMultipleMessage](#) (const QModelIndex &, int number)
- QWidget \* [parentWidget](#) () const  
*Returns the widget to be used as parent for your widget created in [createWidget\(\)](#)*
- virtual void [viewportLeaveEvent](#) (QObject \*obj, QEvent \*event)  
*Called when a [QEvent::Leave](#) of the viewport is received.*
- void [widgetEnterNotifyMultiple](#) (const QModelIndex &index)  
*A sample implementation for above methods.*
- void [widgetLeaveNotifyMultiple](#) ()

### Protected Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- QList< QModelIndex > [affectedIndexes](#) (const QModelIndex &index) const
- bool [affectsMultiple](#) (const QModelIndex &index) const  
*For the context that an overlay can affect multiple items: Assuming the currently overlaid index is given.*
- int [numberOfAffectedIndexes](#) (const QModelIndex &index) const
- bool [viewHasMultiSelection](#) () const  
*Utility method.*

### Protected Attributes

- QPersistentModelIndex [m\\_index](#)

## Protected Attributes inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- bool `m_mouseButtonPressedOnWidget` = false
- `QWidget * m_widget` = nullptr

## Protected Attributes inherited from [Digikam::ItemDelegateOverlay](#)

- `QAbstractItemDelegate * m_delegate` = nullptr
- `QAbstractItemView * m_view` = nullptr

## 9.763.1 Member Function Documentation

### 9.763.1.1 `createWidget()`

```
QWidget * Digikam::ImportRatingOverlay::createWidget ( ) [override], [protected], [virtual]
```

When creating the object, pass [parentWidget\(\)](#) as parent widget. Ownership of the object is passed. It will be deleted in [setActive\(false\)](#).

Implements [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.763.1.2 `hide()`

```
void Digikam::ImportRatingOverlay::hide ( ) [override], [protected], [virtual]
```

Default implementation [hide\(\)](#)s `m_widget`.

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.763.1.3 `setActive()`

```
void Digikam::ImportRatingOverlay::setActive (
    bool active ) [override], [protected], [virtual]
```

If active is false, this will delete the widget and disconnect all signal from model and view to this object (!)

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.763.1.4 `slotEntered()`

```
void Digikam::ImportRatingOverlay::slotEntered (
    const QModelIndex & index ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.763.1.5 visualChange()

```
void Digikam::ImportRatingOverlay::visualChange ( ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ItemDelegateOverlay](#).

### 9.763.1.6 widgetEnterEvent()

```
void Digikam::ImportRatingOverlay::widgetEnterEvent ( ) [override], [protected], [virtual]
```

QEvent::Leave event for the widget is received. The default implementation does nothing.

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.763.1.7 widgetLeaveEvent()

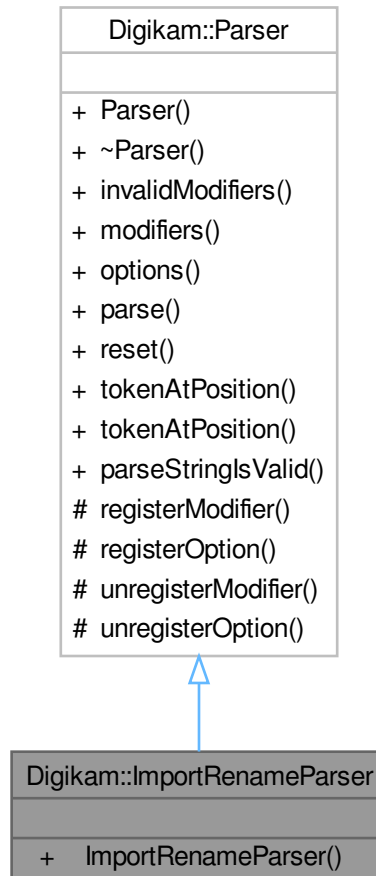
```
void Digikam::ImportRatingOverlay::widgetLeaveEvent ( ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).



## 9.764 Digikam::ImportRenameParser Class Reference

Inheritance diagram for Digikam::ImportRenameParser:



### Additional Inherited Members

### Public Member Functions inherited from [Digikam::Parser](#)

- `ParseResults` `invalidModifiers` (`ParseSettings` &settings)
- `RulesList` `modifiers` () const
- `RulesList` `options` () const
- `QString` `parse` (`ParseSettings` &settings)
- void `reset` ()
- bool `tokenAtPosition` (`ParseSettings` &settings, int pos)
- bool `tokenAtPosition` (`ParseSettings` &settings, int pos, int &start, int &length)

### Static Public Member Functions inherited from [Digikam::Parser](#)

- static bool `parseStringsValid` (const `QString` &str)  
*check if the given parse string is valid*

## Protected Member Functions inherited from [Digikam::Parser](#)

- void **registerModifier** ([Rule](#) \*modifier)
- void **registerOption** ([Rule](#) \*option)
- void **unregisterModifier** (const [Rule](#) \*modifier)
- void **unregisterOption** (const [Rule](#) \*option)

## 9.765 Digikam::ImportRotateOverlay Class Reference

Inheritance diagram for Digikam::ImportRotateOverlay:



## Signals

- void **signalRotate** (const QList< QModelIndex > &indexes)

## Signals inherited from [Digikam::ItemDelegateOverlay](#)

- void **hideNotification** ()
- void **requestNotification** (const QModelIndex &index, const QString &message)
- void **update** (const QModelIndex &index)

## Public Member Functions

- **ImportRotateOverlay** (ImportRotateOverlayDirection dir, QObject \*const parent)
- ImportRotateOverlayDirection **direction** () const
- bool **isLeft** () const
- bool **isRight** () const
- void **setActive** (bool active) override  
*Will call [createButton\(\)](#).*

## Public Member Functions inherited from [Digikam::HoverButtonDelegateOverlay](#)

- **HoverButtonDelegateOverlay** (QObject \*const parent)
- [ItemViewHoverButton](#) \* **button** () const

## Public Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- [AbstractWidgetDelegateOverlay](#) (QObject \*const parent)  
*This class provides functionality for using a widget in an overlay.*

## Public Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- **ItemDelegateOverlay** (QObject \*const parent=nullptr)
- virtual bool **acceptsDelegate** (QAbstractItemDelegate \*) const
- QAbstractItemDelegate \* **delegate** () const
- virtual void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)  
*Only these two methods are implemented as virtual methods.*
- virtual void **paint** (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index)
- void **setDelegate** (QAbstractItemDelegate \*delegate)
- void **setView** (QAbstractItemView \*view)
- QAbstractItemView \* **view** () const

## Static Public Member Functions

- static [ImportRotateOverlay](#) \* **left** (QObject \*const parent)
- static [ImportRotateOverlay](#) \* **right** (QObject \*const parent)

### Protected Member Functions

- bool `checkIndex` (const QModelIndex &index) const override  
*Return true here if you want to show the overlay for the given index.*
- `ItemViewHoverButton` \* `createButton` () override  
*Create your widget here.*
- void `updateButton` (const QModelIndex &index) override  
*Called when a new index is entered.*
- void `widgetEnterEvent` () override  
*Called when a QEvent::Enter resp.*
- void `widgetLeaveEvent` () override

### Protected Member Functions inherited from [Digikam::HoverButtonDelegateOverlay](#)

- QWidget \* `createWidget` () override  
*Create your widget here.*
- void `visualChange` () override  
*Called when any change from the delegate occurs - when the overlay is installed, when size hints, styles or fonts change.*

### Protected Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- bool `checkIndexOnEnter` (const QModelIndex &index) const  
*Utility method called from slotEntered.*
- bool `eventFilter` (QObject \*obj, QEvent \*event) override
- virtual void `hide` ()  
*Called when the widget shall be hidden (mouse cursor left index, viewport, uninstalled etc.).*
- virtual QString `notifyMultipleMessage` (const QModelIndex &, int number)
- QWidget \* `parentWidget` () const  
*Returns the widget to be used as parent for your widget created in `createWidget()`*
- virtual void `viewportLeaveEvent` (QObject \*obj, QEvent \*event)  
*Called when a QEvent::Leave of the viewport is received.*
- void `widgetEnterNotifyMultiple` (const QModelIndex &index)  
*A sample implementation for above methods.*
- void `widgetLeaveNotifyMultiple` ()

### Protected Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- QList< QModelIndex > `affectedIndexes` (const QModelIndex &index) const
- bool `affectsMultiple` (const QModelIndex &index) const  
*For the context that an overlay can affect multiple items: Assuming the currently overlayed index is given.*
- int `numberOfAffectedIndexes` (const QModelIndex &index) const
- bool `viewHasMultiSelection` () const  
*Utility method.*

### Additional Inherited Members

### Protected Slots inherited from [Digikam::HoverButtonDelegateOverlay](#)

- void `slotEntered` (const QModelIndex &index) override
- void `slotReset` () override

## Protected Slots inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- virtual void [slotEntered](#) (const QModelIndex &index)  
*Default implementation shows the widget iff the index is valid and checkIndex returns true.*
- virtual void [slotLayoutChanged](#) ()
- virtual void [slotReset](#) ()  
*Default implementations of these three slots call [hide\(\)](#)*
- virtual void [slotRowsRemoved](#) (const QModelIndex &parent, int start, int end)
- virtual void [slotViewportEntered](#) ()

## Protected Slots inherited from [Digikam::ItemDelegateOverlay](#)

## Protected Attributes inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- bool [m\\_mouseButtonPressedOnWidget](#) = false
- QWidget \* [m\\_widget](#) = nullptr

## Protected Attributes inherited from [Digikam::ItemDelegateOverlay](#)

- QAbstractItemDelegate \* [m\\_delegate](#) = nullptr
- QAbstractItemView \* [m\\_view](#) = nullptr

## 9.765.1 Member Function Documentation

### 9.765.1.1 [checkIndex\(\)](#)

```
bool Digikam::ImportRotateOverlay::checkIndex (
    const QModelIndex & index ) const [override], [protected], [virtual]
```

The default implementation returns true.

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.765.1.2 [createButton\(\)](#)

```
ItemViewHoverButton * Digikam::ImportRotateOverlay::createButton ( ) [override], [protected],
[virtual]
```

Pass view() as parent.

Implements [Digikam::HoverButtonDelegateOverlay](#).

### 9.765.1.3 [setActive\(\)](#)

```
void Digikam::ImportRotateOverlay::setActive (
    bool active ) [override], [virtual]
```

Reimplemented from [Digikam::HoverButtonDelegateOverlay](#).

#### 9.765.1.4 updateButton()

```
void Digikam::ImportRotateOverlay::updateButton (
    const QModelIndex & index ) [override], [protected], [virtual]
```

Reposition your button here, adjust and store state.

Implements [Digikam::HoverButtonDelegateOverlay](#).

#### 9.765.1.5 widgetEnterEvent()

```
void Digikam::ImportRotateOverlay::widgetEnterEvent ( ) [override], [protected], [virtual]
```

QEvent::Leave event for the widget is received. The default implementation does nothing.

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

#### 9.765.1.6 widgetLeaveEvent()

```
void Digikam::ImportRotateOverlay::widgetLeaveEvent ( ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

## 9.766 Digikam::ImportRotateOverlayButton Class Reference

Inheritance diagram for Digikam::ImportRotateOverlayButton:



### Public Member Functions

- **ImportRotateOverlayButton** (ImportRotateOverlayDirection dir, QAbstractItemView \*const parentView)
- QSize [sizeHint](#) () const override

*Reimplement to match the size of your icon.*

## Public Member Functions inherited from [Digikam::ItemViewHoverButton](#)

- **ItemViewHoverButton** (QAbstractItemView \*const parentView)
- QModelIndex **index** () const
- void **initIcon** ()
- void **reset** ()
- void **setIndex** (const QModelIndex &index)
- void **setVisible** (bool visible) override

## Protected Member Functions

- QIcon **icon** () override  
*Return your icon here.*
- void **updateToolTip** () override  
*Optionally update tooltip here.*

## Protected Member Functions inherited from [Digikam::ItemViewHoverButton](#)

- void **enterEvent** (QEnterEvent \*event)
- void **leaveEvent** (QEvent \*event)
- void **paintEvent** (QPaintEvent \*event)
- void **setup** ()  
*to call in children class constructors to init signal/slot connections.*

## Protected Attributes

- ImportRotateOverlayDirection const **m\_direction**

## Protected Attributes inherited from [Digikam::ItemViewHoverButton](#)

- QTimerLine \* **m\_fadingTimeLine** = nullptr
- int **m\_fadingValue** = 0
- QIcon **m\_icon**
- QPersistentModelIndex **m\_index**
- bool **m\_isHovered** = false

## Additional Inherited Members

## Protected Slots inherited from [Digikam::ItemViewHoverButton](#)

- void **refreshIcon** ()
- void **setFadingValue** (int value)
- void **startFading** ()
- void **stopFading** ()



## 9.766.1 Member Function Documentation

### 9.766.1.1 icon()

```
QIcon Digikam::ImportRotateOverlayButton::icon ( ) [override], [protected], [virtual]
```

Will be queried again on toggle.

Implements [Digikam::ItemViewHoverButton](#).

### 9.766.1.2 sizeHint()

```
QSize Digikam::ImportRotateOverlayButton::sizeHint ( ) const [override], [virtual]
```

Implements [Digikam::ItemViewHoverButton](#).

### 9.766.1.3 updateToolTip()

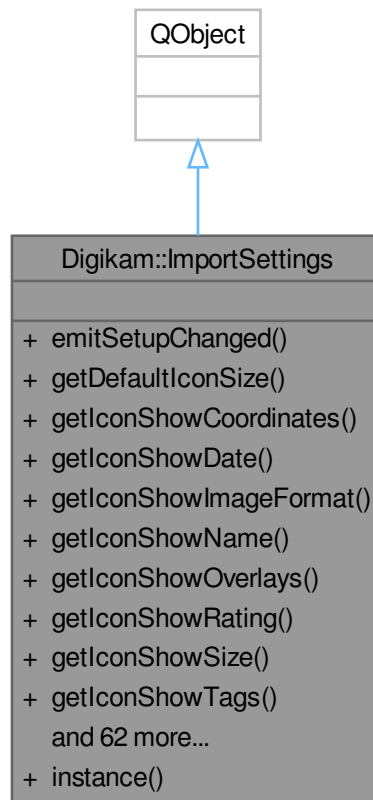
```
void Digikam::ImportRotateOverlayButton::updateToolTip ( ) [override], [protected], [virtual]
```

Will be called again on state change.

Reimplemented from [Digikam::ItemViewHoverButton](#).

## 9.767 Digikam::ImportSettings Class Reference

Inheritance diagram for Digikam::ImportSettings:



### Public Types

- enum **ItemLeftClickAction** { **ShowPreview** = 0 , **StartEditor** , **OpenDefault** }

### Signals

- void **setupChanged** ()

### Public Member Functions

- void **emitSetupChanged** ()
- int **getDefaultIconSize** () const
- bool **getIconShowCoordinates** () const
- bool **getIconShowDate** () const
- bool **getIconShowImageFormat** () const
- bool **getIconShowName** () const

- bool **getIconShowOverlays** () const
- bool **getIconShowRating** () const
- bool **getIconShowSize** () const
- bool **getIconShowTags** () const
- bool **getIconShowTitle** () const
- QFont **getIconViewFont** () const
- int **getImageSeparationMode** () const
- int **getImageSortBy** () const
- int **getImageSortOrder** () const
- int **getItemLeftClickAction** () const
- bool **getPreviewItemsWhileDownload** () const
- bool **getPreviewLoadFullImageSize** () const
- bool **getPreviewShowIcons** () const
- bool **getShowThumbbar** () const
- bool **getShowToolTips** () const
- QFont **getToolTipsFont** () const
- bool **getToolTipsShowFileDate** () const
- bool **getToolTipsShowFileName** () const
- bool **getToolTipsShowFileSize** () const
- bool **getToolTipsShowImageDim** () const
- bool **getToolTipsShowImageType** () const
- bool **getToolTipsShowLabelRating** () const
- bool **getToolTipsShowPhotoExpo** () const
- bool **getToolTipsShowPhotoFlash** () const
- bool **getToolTipsShowPhotoFocal** () const
- bool **getToolTipsShowPhotoLens** () const
- bool **getToolTipsShowPhotoMake** () const
- bool **getToolTipsShowPhotoWB** () const
- bool **getToolTipsShowTags** () const
- void **readSettings** ()
- void **saveSettings** ()
- void **setDefaultIconSize** (int val)
- void **setIconShowCoordinates** (bool val)
- void **setIconShowDate** (bool val)
- void **setIconShowImageFormat** (bool val)
- void **setIconShowName** (bool val)
- void **setIconShowOverlays** (bool val)
- void **setIconShowRating** (bool val)
- void **setIconShowSize** (bool val)
- void **setIconShowTags** (bool val)
- void **setIconShowTitle** (bool val)
- void **setIconViewFont** (const QFont &font)
- void **setImageSeparationMode** (int mode)
- void **setImageSortBy** (int sortBy)
- void **setImageSortOrder** (int order)
- void **setItemLeftClickAction** (int action)
- void **setPreviewItemsWhileDownload** (bool val)
- void **setPreviewLoadFullImageSize** (bool val)
- void **setPreviewShowIcons** (bool val)
- void **setShowThumbbar** (bool val)
- void **setShowToolTips** (bool val)
- void **setToolTipsFont** (const QFont &font)
- void **setToolTipsShowFileDate** (bool val)
- void **setToolTipsShowFileName** (bool val)
- void **setToolTipsShowFileSize** (bool val)

- void **setToolTipsShowImageDim** (bool val)
- void **setToolTipsShowImageType** (bool val)
- void **setToolTipsShowLabelRating** (bool val)
- void **setToolTipsShowPhotoExpo** (bool val)
- void **setToolTipsShowPhotoFlash** (bool val)
- void **setToolTipsShowPhotoFocal** (bool val)
- void **setToolTipsShowPhotoLens** (bool val)
- void **setToolTipsShowPhotoMake** (bool val)
- void **setToolTipsShowPhotoWB** (bool val)
- void **setToolTipsShowTags** (bool val)
- bool **showToolTipsIsValid** () const

#### Static Public Member Functions

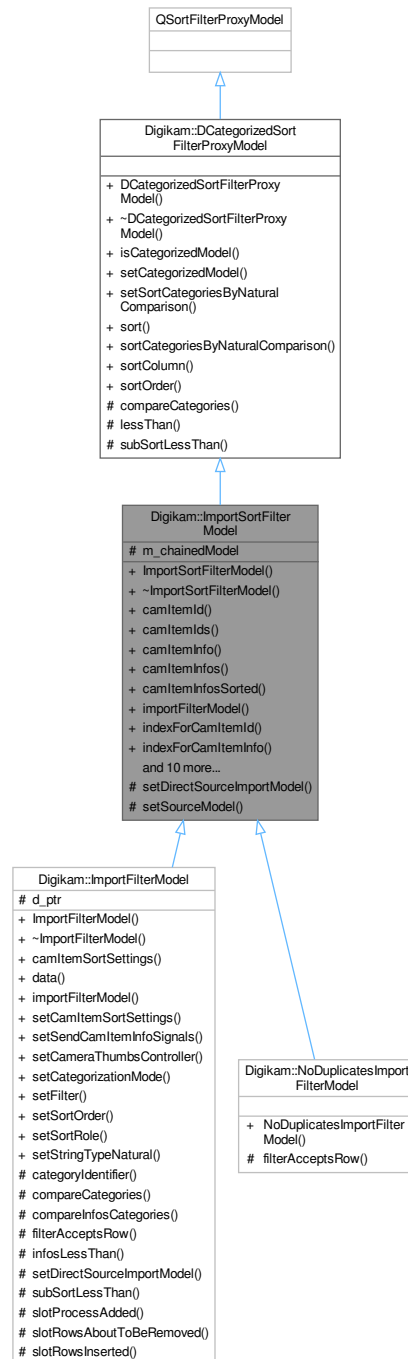
- static [ImportSettings](#) \* **instance** ()

#### Friends

- class **ImportSettingsCreator**

## 9.768 Digikam::ImportSortFilterModel Class Reference

Inheritance diagram for Digikam::ImportSortFilterModel:



### Public Member Functions

- **ImportSortFilterModel** (QObject \*const parent=nullptr)
- `qulonglong camItemId` (const QModelIndex &index) const

- `QList< qlonglong > camItemIds` (`const QList< QModelIndex > &indexes`) `const`
- `CamItemInfo camItemInfo` (`const QModelIndex &index`) `const`
- `QList< CamItemInfo > camItemInfos` (`const QList< QModelIndex > &indexes`) `const`
- `QList< CamItemInfo > camItemInfosSorted` () `const`  
*Returns a list of all camera infos, sorted according to this model.*
- virtual `ImportFilterModel * importFilterModel` () `const`  
*Returns this, any chained **ImportFilterModel**, or 0.*
- `QModelIndex indexForCamItemId` (`qlonglong id`) `const`
- `QModelIndex indexForCamItemInfo` (`const CamItemInfo &info`) `const`
- `QModelIndex indexForPath` (`const QString &filePath`) `const`
- `QModelIndex mapFromDirectSourceToSourceImportModel` (`const QModelIndex &sourceModelIndex`) `const`
- `QModelIndex mapFromSourceImportModel` (`const QModelIndex &importModelIndex`) `const`
- `QList< QModelIndex > mapListFromSource` (`const QList< QModelIndex > &sourceIndexes`) `const`
- `QList< QModelIndex > mapListToSource` (`const QList< QModelIndex > &indexes`) `const`
- `QModelIndex mapToSourceImportModel` (`const QModelIndex &proxyIndex`) `const`  
*Convenience methods mapped to **ImportItemModel**.*
- void `setSourceFilterModel` (`ImportSortFilterModel *const sourceModel`)
- void `setSourceImportModel` (`ImportItemModel *const sourceModel`)
- `ImportSortFilterModel * sourceFilterModel` () `const`
- `ImportItemModel * sourceImportModel` () `const`

## Public Member Functions inherited from [Digikam::DCategorizedSortFilterProxyModel](#)

- `DCategorizedSortFilterProxyModel` (`QObject *const parent=nullptr`)
- bool `isCategorizedModel` () `const`
- void `setCategorizedModel` (`bool categorizedModel`)  
*Enables or disables the categorization feature.*
- void `setSortCategoriesByNaturalComparison` (`bool sortCategoriesByNaturalComparison`)  
*Set if the sorting using **CategorySortRole** will use a natural comparison in the case that strings were returned.*
- void `sort` (`int column, Qt::SortOrder order=Qt::AscendingOrder`) `override`  
*Overridden from **QSortFilterProxyModel**.*
- bool `sortCategoriesByNaturalComparison` () `const`
- int `sortColumn` () `const`
- `Qt::SortOrder sortOrder` () `const`

## Protected Member Functions

- virtual void `setDirectSourceImportModel` (`ImportItemModel *const sourceModel`)  
*Reimplement if needed. Called only when model shall be set as (direct) sourceModel.*
- void `setSourceModel` (`QAbstractItemModel *sourceModel`) `override`

## Protected Member Functions inherited from [Digikam::DCategorizedSortFilterProxyModel](#)

- virtual int `compareCategories` (`const QModelIndex &left, const QModelIndex &right`) `const`  
*This method compares the category of the left index with the category of the right index.*
- bool `lessThan` (`const QModelIndex &left, const QModelIndex &right`) `const` `override`  
*Overridden from **QSortFilterProxyModel**.*
- virtual bool `subSortLessThan` (`const QModelIndex &left, const QModelIndex &right`) `const`  
*This method has a similar purpose as **lessThan()** has on **QSortFilterProxyModel**.*

## Protected Attributes

- [ImportSortFilterModel](#) \* `m_chainedModel` = nullptr

## Additional Inherited Members

## Public Types inherited from [Digikam::DCategorizedSortFilterProxyModel](#)

- enum [AdditionalRoles](#) { `CategoryDisplayRole` = 0x17CE990A , `CategorySortRole` = 0x27857E60 }

## 9.768.1 Member Function Documentation

### 9.768.1.1 `camItemInfosSorted()`

```
QList< CamItemInfo > Digikam::ImportSortFilterModel::camItemInfosSorted ( ) const
```

If you do not need a sorted list, use [ImportItemModel](#)'s `camItemInfo()` method.

### 9.768.1.2 `importFilterModel()`

```
ImportFilterModel * Digikam::ImportSortFilterModel::importFilterModel ( ) const [virtual]
```

Reimplemented in [Digikam::ImportFilterModel](#).

### 9.768.1.3 `mapToSourceImportModel()`

```
QModelIndex Digikam::ImportSortFilterModel::mapToSourceImportModel (
    const QModelIndex & proxyIndex ) const
```

Mentioned indexes returned come from the source import image model.

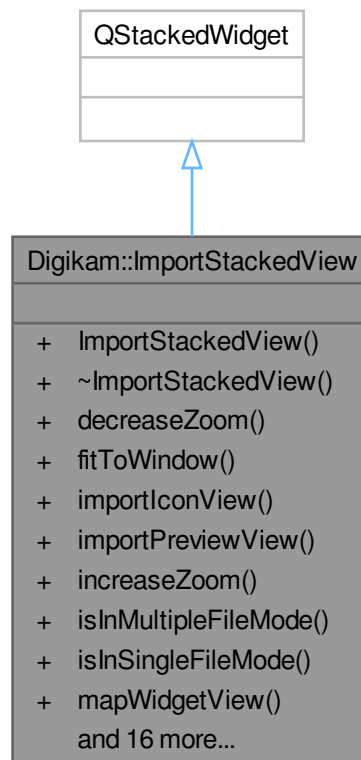
### 9.768.1.4 `setDirectSourceImportModel()`

```
void Digikam::ImportSortFilterModel::setDirectSourceImportModel (
    ImportItemModel *const sourceModel ) [protected], [virtual]
```

Reimplemented in [Digikam::ImportFilterModel](#).

## 9.769 Digikam::ImportStackedView Class Reference

Inheritance diagram for Digikam::ImportStackedView:



### Public Types

- enum `StackedViewMode` { `PreviewCameraMode` = 0 , `PreviewImageMode` , `MapWidgetMode` , `MediaPlayerMode` }

### Signals

- void `signalEscapePreview` ()
- void `signalNextItem` ()
- void `signalPrevItem` ()
- void `signalViewModeChanged` ()
- void `signalZoomFactorChanged` (double)



## Public Member Functions

- **ImportStackedView** (QWidget \*const parent=nullptr)
- void **decreaseZoom** ()
- void **fitToWindow** ()
- [ImportIconView](#) \* **importIconView** () const
- [ImportPreviewView](#) \* **importPreviewView** () const
- void **increaseZoom** ()
- bool **isInMultipleFileMode** () const
- bool **isInSingleFileMode** () const
- [MapWidgetView](#) \* **mapWidgetView** () const
- bool **maxZoom** () const
- bool **minZoom** () const
- void **previewLoaded** ()
- void **setDockArea** (QMainWindow \*)
- void **setPreviewItem** (const [CamItemInfo](#) &info=[CamItemInfo](#)(), const [CamItemInfo](#) &previous=[CamItemInfo](#)(), const [CamItemInfo](#) &next=[CamItemInfo](#)())
- void **setViewMode** (const [StackedViewMode](#) mode)
- void **setZoomFactor** (double z)
- void **setZoomFactorSnapped** (double z)
- [ImportThumbnailBar](#) \* **thumbBar** () const
- [ThumbBarDock](#) \* **thumbBarDock** () const
- void **toggleFitToWindowOr100** ()
- [StackedViewMode](#) **viewMode** () const
- double **zoomFactor** () const
- double **zoomMax** () const
- double **zoomMin** () const
- void **zoomTo100Percents** ()

## 9.769.1 Member Enumeration Documentation

### 9.769.1.1 StackedViewMode

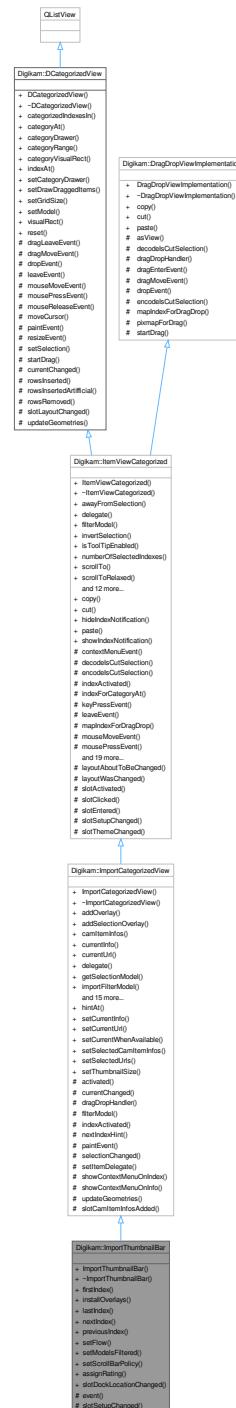
```
enum Digikam::ImportStackedView::StackedViewMode
```

#### Enumerator

PreviewCameraMode	previewing the set of items on the camera
-------------------	---

## 9.770 Digikam::ImportThumbnailBar Class Reference

Inheritance diagram for Digikam::ImportThumbnailBar:



### Public Slots

- void **assignRating** (const QList< QModelIndex > &index, int rating)
- void **slotDockLocationChanged** (Qt::DockWidgetArea area)

## Public Slots inherited from [Digikam::ImportCategorizedView](#)

- void **hintAt** (const [CamItemInfo](#) &info)
  - Does something to gain attention for info, but not changing current selection.*
- void **setCurrentInfo** (const [CamItemInfo](#) &info)
  - Set as current item the item identified by the [CamItemInfo](#).*
- void **setCurrentUrl** (const QUrl &url)
  - Set as current item the item identified by its file url.*
- void **setCurrentWhenAvailable** (qulonglong camItemId)
  - Scroll the view to the given item when it becomes available.*
- void **setSelectedCamItemInfos** (const QList< [CamItemInfo](#) > &infos)
  - Set selected items.*
- void **setSelectedUrls** (const QList< QUrl > &urlList)
  - Set selected items identified by their file urls.*
- void **setThumbnailSize** (int size)

## Public Slots inherited from [Digikam::ItemViewCategorized](#)

- void **copy** () override
- void **cut** () override
- void **hideIndexNotification** ()
- void **paste** () override
- void **showIndexNotification** (const QModelIndex &index, const QString &message)

## Public Slots inherited from [Digikam::DCategorizedView](#)

- void **reset** () override

## Public Member Functions

- **ImportThumbnailBar** (QWidget \*const parent=nullptr)
- QModelIndex **firstIndex** () const
- void **installOverlays** ()
- QModelIndex **lastIndex** () const
- QModelIndex **nextIndex** (const QModelIndex &index) const
- QModelIndex **previousIndex** (const QModelIndex &index) const
- void **setFlow** (QListView::Flow newFlow)
- void **setModelsFiltered** ([ImportItemModel](#) \*model, [ImportSortFilterModel](#) \*filterModel)
  - This installs a duplicate filter model, if the [ImportItemModel](#) may contain duplicates.*
- void **setScrollBarPolicy** (Qt::ScrollBarPolicy policy)
  - Sets the policy always for the one scroll bar which is relevant, depending on orientation.*

## Public Member Functions inherited from [Digikam::ImportCategorizedView](#)

- **ImportCategorizedView** (QWidget \*const parent=nullptr)
- void **addOverlay** (ItemDelegateOverlay \*overlay, ImportDelegate \*delegate=nullptr)
  - Add and remove an overlay.*
- void **addSelectionOverlay** (ImportDelegate \*delegate=nullptr)
- QList< [CamItemInfo](#) > **camItemInfos** () const
- [CamItemInfo](#) **currentInfo** () const
- QUrl **currentUrl** () const
- [ImportDelegate](#) \* **delegate** () const
- QItemSelectionModel \* **getSelectionModel** () const
- [ImportFilterModel](#) \* **importFilterModel** () const
  - Returns any [ImportFilterModel](#) in chain.*
- [ImportItemModel](#) \* **importItemModel** () const
- [ImportSortFilterModel](#) \* **importSortFilterModel** () const
- [ImportThumbnailModel](#) \* **importThumbnailModel** () const
  - Returns 0 if the [ImportItemModel](#) is not an [ImportThumbnailModel](#).*
- [CamItemInfo](#) **nextInfo** (const [CamItemInfo](#) &info)
- [CamItemInfo](#) **nextInOrder** (const [CamItemInfo](#) &startingPoint, int nth)
  - Returns the n-th info after the given one.*
- [CamItemInfo](#) **previousInfo** (const [CamItemInfo](#) &info)
- void **removeOverlay** (ItemDelegateOverlay \*overlay)
- QList< [CamItemInfo](#) > **selectedCamItemInfos** () const
- QList< [CamItemInfo](#) > **selectedCamItemInfosCurrentFirst** () const
- QList< QUrl > **selectedUrls** () const
- void **setModels** ([ImportItemModel](#) \*model, [ImportSortFilterModel](#) \*filterModel)
- virtual void **setThumbnailSize** (const [ThumbnailSize](#) &size)
- [ThumbnailSize](#) **thumbnailSize** () const
- void **toIndex** (const QUrl &url)
  - Selects the index as current and scrolls to it.*
- QList< QUrl > **urls** () const

## Public Member Functions inherited from [Digikam::ItemViewCategorized](#)

- **ItemViewCategorized** (QWidget \*const parent=nullptr)
- void **awayFromSelection** ()
- [DItemDelegate](#) \* **delegate** () const
- void **invertSelection** ()
- bool **isToolTipEnabled** () const
- int **numberOfSelectedIndexes** () const
- void **scrollTo** (const QModelIndex &index, ScrollHint hint=EnsureVisible) override
- void **scrollToRelaxed** (const QModelIndex &index, ScrollHint hint=EnsureVisible)
  - Like `scrollTo`, but only scrolls if the index is not visible, regardless of hint.*
- void **setInitialSelectedItem** (bool enabled)
  - Ensure a initial selected item.*
- void **setScrollCurrentToCenter** (bool enabled)
  - Scroll automatically the current index to center of the view.*
- void **setScrollStepGranularity** (int factor)
  - Determine a step size for scrolling: The larger this number, the smaller and more precise is the scrolling.*
- void **setSelectedIndexes** (const QList< QModelIndex > &indexes)
- void **setSpacing** (int spacing)
  - Sets the spacing.*

- void **setToolTipEnabled** (bool enabled)
- void **setUsePointingHandCursor** (bool useCursor)  
*Set if the PointingHand Cursor should be shown over the activation area.*
- void **toFirstIndex** ()  
*Selects the index as current and scrolls to it.*
- void **toIndex** (const QModelIndex &index)
- void **toLastIndex** ()
- void **toNextIndex** ()
- void **toPreviousIndex** ()

### Public Member Functions inherited from [Digikam::DCategorizedView](#)

- **DCategorizedView** (QWidget \*const parent=nullptr)
- virtual QModelIndexList **categorizedIndexesIn** (const QRect &rect) const  
*This method will return all indexes whose visual rect intersects rect.*
- virtual QModelIndex **categoryAt** (const QPoint &point) const  
*This method will return the first index of the category in the region of which point is found.*
- **DCategoryDrawer** \* **categoryDrawer** () const
- virtual QItemSelectionRange **categoryRange** (const QModelIndex &index) const  
*This method returns the range of indexes contained in the category in which index is sorted.*
- virtual QRect **categoryVisualRect** (const QModelIndex &index) const  
*This method will return the visual rect of the header of the category in which index is sorted.*
- QModelIndex **indexAt** (const QPoint &point) const override
- void **setCategoryDrawer** ([DCategoryDrawer](#) \*categoryDrawer)
- void **setDrawDraggedItems** (bool drawDraggedItems)  
*Switch on drawing of dragged items.*
- void **setGridSize** (const QSize &size)
- void **setModel** (QAbstractItemModel \*model) override
- QRect **visualRect** (const QModelIndex &index) const override

### Public Member Functions inherited from [Digikam::DragDropViewImplementation](#)

- virtual void **copy** ()
- virtual void **cut** ()
- virtual void **paste** ()

### Protected Member Functions

- bool **event** (QEvent \*) override
- void **slotSetupChanged** () override

## Protected Member Functions inherited from [Digikam::ImportCategorizedView](#)

- virtual void **activated** (const [CamItemInfo](#) &info, Qt::KeyboardModifiers modifiers)
  - Reimplement these in a subclass.*
- void **currentChanged** (const QModelIndex &index, const QModelIndex &previous) override
- [AbstractItemDragDropHandler](#) \* **dragDropHandler** () const override
  - You need to implement these three methods Returns the drag drop handler.*
- QSortFilterProxyModel \* **filterModel** () const override
  - reimplemented from parent class*
- void **indexActivated** (const QModelIndex &index, Qt::KeyboardModifiers modifiers) override
- QModelIndex **nextIndexHint** (const QModelIndex &indexToAnchor, const QItemSelectionRange &removed) const override
  - Assuming the given indexes would be removed (hypothetically!), return the index to be selected instead, starting from anchor.*
- void **paintEvent** (QPaintEvent \*e) override
- void **selectionChanged** (const QItemSelection &, const QItemSelection &) override
- void **setItemDelegate** ([ImportDelegate](#) \*delegate)
- void **showContextMenuOnIndex** (QContextMenuEvent \*event, const QModelIndex &index) override
  - Reimplement these in a subclass.*
- virtual void **showContextMenuOnInfo** (QContextMenuEvent \*event, const [CamItemInfo](#) &info)
- void **updateGeometries** () override

## Protected Member Functions inherited from [Digikam::ItemViewCategorized](#)

- void **contextMenuEvent** (QContextMenuEvent \*event) override
  - reimplemented from parent class*
- bool **decodelsCutSelection** (const QMimeData \*mimeData)
- void **encodelsCutSelection** (QMimeData \*mime, bool isCutSelection)
- QModelIndex **indexForCategoryAt** (const QPoint &pos) const
  - Returns an index that is representative for the category at position pos.*
- void **keyPressEvent** (QKeyEvent \*event) override
- void **leaveEvent** (QEvent \*event) override
- QModelIndex **mapIndexForDragDrop** (const QModelIndex &index) const override
  - Note: pure virtual [dragDropHandler\(\)](#) still open from [DragDropViewImplementation](#).*
- void **mouseMoveEvent** (QMouseEvent \*event) override
- void **mousePressEvent** (QMouseEvent \*event) override
- void **mouseReleaseEvent** (QMouseEvent \*event) override
- QModelIndex **moveCursor** (CursorAction cursorAction, Qt::KeyboardModifiers modifiers) override
- QPixmap **pixmapForDrag** (const QList< QModelIndex > &indexes) const override
  - Creates a pixmap for dragging the given indexes.*
- void **reset** () override
- void **resizeEvent** (QResizeEvent \*e) override
- void **rowsAboutToBeRemoved** (const QModelIndex &parent, int start, int end) override
- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- void **rowsRemoved** (const QModelIndex &parent, int start, int end) override
- void **selectionChanged** (const QItemSelection &, const QItemSelection &) override
- void **setItemDelegate** ([DItemDelegate](#) \*delegate)
- void **setToolTip** ([ItemViewToolTip](#) \*tip)
- virtual void **showContextMenu** (QContextMenuEvent \*event)
- virtual bool **showToolTip** (const QModelIndex &index, QStyleOptionViewItem &option, QHelpEvent \*e=nullptr)
  - Provides default behavior, can reimplement in a subclass.*
- void **updateDelegateSizes** ()
- void **userInteraction** ()
- bool **viewportEvent** (QEvent \*event) override
- void **wheelEvent** (QWheelEvent \*event) override

## Protected Member Functions inherited from [Digikam::DCategorizedView](#)

- void **dragLeaveEvent** (QDragLeaveEvent \*event) override
- void **dragMoveEvent** (QDragMoveEvent \*event) override
- void **dropEvent** (QDropEvent \*event) override
- void **leaveEvent** (QEvent \*event) override
- void **mouseMoveEvent** (QMouseEvent \*event) override
- void **mousePressEvent** (QMouseEvent \*event) override
- void **mouseReleaseEvent** (QMouseEvent \*event) override
- QModelIndex **moveCursor** (CursorAction cursorAction, Qt::KeyboardModifiers modifiers) override
- void **paintEvent** (QPaintEvent \*event) override
- void **resizeEvent** (QResizeEvent \*event) override
- void **setSelection** (const QRect &rect, QItemSelectionModel::SelectionFlags flags) override
- void **startDrag** (Qt::DropActions supportedActions) override

## Protected Member Functions inherited from [Digikam::DragDropViewImplementation](#)

- virtual QAbstractItemView \* **asView** ()=0  
*This one is implemented by DECLARE\_VIEW\_DRAG\_DROP\_METHODS.*
- bool **decodelsCutSelection** (const QMimeData \*mimeData)
- void **dragEnterEvent** (QDragEnterEvent \*event)  
*Implements the relevant QAbstractItemView methods for drag and drop.*
- void **dragMoveEvent** (QDragMoveEvent \*e)
- void **dropEvent** (QDropEvent \*e)
- void **encodelsCutSelection** (QMimeData \*mime, bool isCutSelection)
- void **startDrag** (Qt::DropActions supportedActions)

## Additional Inherited Members

## Signals inherited from [Digikam::ImportCategorizedView](#)

- void **camItemInfoActivated** (const CamItemInfo &info)  
*Emitted when the given CamItemInfo is activated.*
- void **currentChanged** (const CamItemInfo &info)
- void **deselected** (const QList< CamItemInfo > &nowDeselectedInfos)  
*Emitted when items are deselected.*
- void **modelChanged** ()  
*Emitted when a new model is set.*
- void **selected** (const QList< CamItemInfo > &newSelectedInfos)  
*Emitted when new items are selected.*

## Signals inherited from [Digikam::ItemViewCategorized](#)

- void **clicked** (const QMouseEvent \*e, const QModelIndex &index)  
*For overlays: Like the respective parent class signals, but with additional info.*
- void **entered** (const QMouseEvent \*e, const QModelIndex &index)
- void **keyPressed** (QKeyEvent \*e)  
*Remember you may want to check if the event is accepted or ignored.*
- void **selectionChanged** ()  
*Emitted when any selection change occurs.*
- void **selectionCleared** ()  
*Emitted when the selection is completely cleared.*
- void **viewportClicked** (const QMouseEvent \*e)  
*While clicked() is emitted with a valid index, this corresponds to clicking on empty space.*
- void **zoomInStep** ()
- void **zoomOutStep** ()

### Protected Slots inherited from [Digikam::ImportCategorizedView](#)

- void **slotCamItemInfosAdded** ()

### Protected Slots inherited from [Digikam::ItemViewCategorized](#)

- void **layoutAboutToBeChanged** ()
- void **layoutWasChanged** ()
- void **slotActivated** (const QModelIndex &index)
- void **slotClicked** (const QModelIndex &index)
- void **slotEntered** (const QModelIndex &index)
- virtual void **slotThemeChanged** ()

### Protected Slots inherited from [Digikam::DCategorizedView](#)

- void **currentChanged** (const QModelIndex &current, const QModelIndex &previous) override
- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- virtual void **rowsInsertedArtificial** (const QModelIndex &parent, int start, int end)
- virtual void **slotLayoutChanged** ()
- void **updateGeometries** () override

## 9.770.1 Member Function Documentation

### 9.770.1.1 setModelsFiltered()

```
void Digikam::ImportThumbnailBar::setModelsFiltered (
    ImportItemModel * model,
    ImportSortFilterModel * filterModel )
```

Otherwise, just use setModels().

### 9.770.1.2 slotSetupChanged()

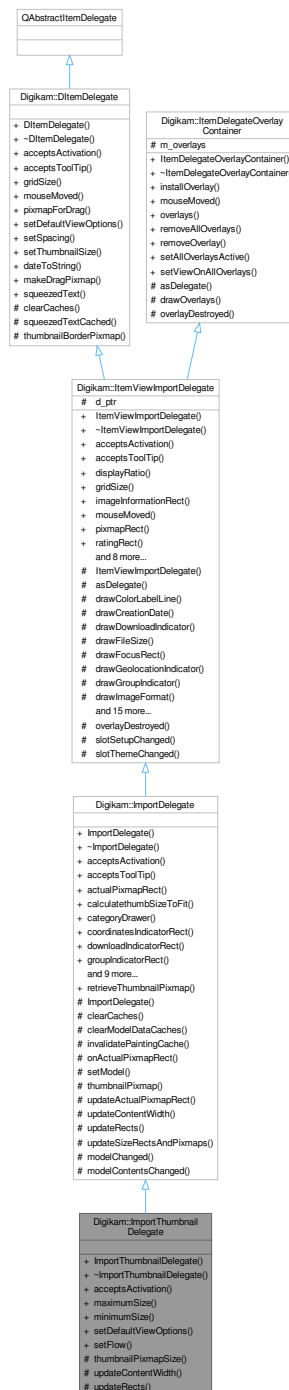
```
void Digikam::ImportThumbnailBar::slotSetupChanged ( ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ItemViewCategorized](#).



## 9.771 Digikam::ImportThumbnailDelegate Class Reference

Inheritance diagram for Digikam::ImportThumbnailDelegate:



### Public Member Functions

- **ImportThumbnailDelegate** (**ImportCategorizedView** \*const parent)
- bool **acceptsActivation** (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*activationRect) const override

- int **maximumSize** () const  
*Returns the minimum or maximum viewport size in the limiting dimension, width or height, depending on current flow.*
- int **minimumSize** () const
- void **setDefaultViewOptions** (const QStyleOptionViewItem &option) override  
*Style option with standard values to use for cached rendering.*
- void **setFlow** (QListView::Flow flow)

## Public Member Functions inherited from [Digikam::ImportDelegate](#)

- **ImportDelegate** (QWidget \*const parent)
- bool **acceptsToolTip** (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*tooltipRect=nullptr) const override  
*These methods take four parameters: The position on viewport, the rect on viewport, the index, and optionally a parameter into which, if the return value is true, a rectangle can be written for which the return value will be true as well.*
- QRect **actualPixmapRect** (const QModelIndex &index) const
- int **calculatethumbSizeToFit** (int ws)
- **ImportCategoryDrawer** \* **categoryDrawer** () const
- QRect **coordinatesIndicatorRect** () const
- QRect **downloadIndicatorRect** () const
- QRect **groupIndicatorRect** () const
- QRect **imageInformationRect** () const override  
*Returns the area where the image information is drawn, or null if empty / not supported.*
- QRect **lockIndicatorRect** () const
- void **paint** (QPainter \*painter, const QStyleOptionViewItem &option, const QModelIndex &index) const override
- QPixmap **pixmapForDrag** (const QStyleOptionViewItem &option, const QList< QModelIndex > &indexes) const override
- QRect **pixmapRect** () const override  
*Returns the area where the pixmap is drawn, or null if not supported.*
- void **setSpacing** (int spacing) override
- void **setView** (**ImportCategorizedView** \*view)
- QRect **tagsRect** () const

## Public Member Functions inherited from [Digikam::ItemViewImportDelegate](#)

- **ItemViewImportDelegate** (QWidget \*const parent)
- bool **acceptsActivation** (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*activationRect=nullptr) const override
- bool **acceptsToolTip** (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*tooltipRect=nullptr) const override  
*These methods take four parameters: The position on viewport, the rect on viewport, the index, and optionally a parameter into which, if the return value is true, a rectangle can be written for which the return value will be true as well.*
- double **displayRatio** () const
- QSize **gridSize** () const override  
*Returns the gridsize to be set by the view.*
- void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index) override
- virtual QRect **ratingRect** () const  
*Returns the rectangle where the rating is drawn, or a null rectangle if not supported.*
- QRect **rect** () const
- void **setDefaultViewOptions** (const QStyleOptionViewItem &option) override

*Style option with standard values to use for cached rendering.*

- void [setRatingEdited](#) (const QModelIndex &index)  
*Can be used to temporarily disable drawing of the rating.*
- void [setSpacing](#) (int spacing) override
- void [setThumbnailSize](#) (const [ThumbnailSize](#) &thumbSize) override  
*reimplemented from [DItemDelegate](#)*
- QSize **sizeHint** (const QStyleOptionViewItem &option, const QModelIndex &index) const override
- int **spacing** () const
- [ThumbnailSize](#) **thumbnailSize** () const

## Public Member Functions inherited from [Digikam::DItemDelegate](#)

- [DItemDelegate](#) (QObject \*const parent=nullptr)

## Public Member Functions inherited from [Digikam::ItemDelegateOverlayContainer](#)

- [ItemDelegateOverlayContainer](#) ()=default  
*This is a sample implementation for delegate management methods, to be inherited by a delegate.*
- void **installOverlay** ([ItemDelegateOverlay](#) \*overlay)
- void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)
- QList< [ItemDelegateOverlay](#) \* > **overlays** () const
- void **removeAllOverlays** ()
- void **removeOverlay** ([ItemDelegateOverlay](#) \*overlay)
- void **setAllOverlaysActive** (bool active)
- void **setViewOnAllOverlays** (QAbstractItemView \*view)

## Protected Member Functions

- int **thumbnailPixmapSize** (bool withHighlight, int size)
- void [updateContentWidth](#) () override  
*Reimplement this to set contentWidth.*
- void [updateRects](#) () override  
*In a subclass, you need to implement this method to set up the rects for drawing.*

## Protected Member Functions inherited from [Digikam::ImportDelegate](#)

- **ImportDelegate** (ImportDelegate::ImportDelegatePrivate &dd, QWidget \*const parent)
- void [clearCaches](#) () override
- virtual void **clearModelDataCaches** ()  
*Reimplement to clear caches based on model indexes (hash on row number etc.) Change signals are listened to this is called whenever such properties become invalid.*
- void [invalidatePaintingCache](#) () override  
*reimplement these in subclasses*
- bool **onActualPixmapRect** (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*actualRect) const
- void **setModel** (QAbstractItemModel \*model)
- virtual QPixmap **thumbnailPixmap** (const QModelIndex &index) const
- void **updateActualPixmapRect** (const QModelIndex &index, const QRect &rect)
- void [updateSizeRectsAndPixmaps](#) () override

## Protected Member Functions inherited from [Digikam::ItemViewImportDelegate](#)

- **ItemViewImportDelegate** (ItemViewImportDelegatePrivate &dd, QWidget \*const parent)
- QAbstractItemDelegate \* **asDelegate** () override
 

*Returns the delegate, typically, the derived class.*
- void **drawColorLabelLine** (QPainter \*p, const QRect &pixRect, int colorId) const
- void **drawCreationDate** (QPainter \*p, const QRect &dateRect, const QDateTime &date) const
- void **drawDownloadIndicator** (QPainter \*p, const QRect &r, int itemType) const
- void **drawFileSize** (QPainter \*p, const QRect &r, qlonglong bytes) const
- void **drawFocusRect** (QPainter \*p, const QStyleOptionViewItem &option, bool isSelected) const
- void **drawGeolocationIndicator** (QPainter \*p, const QRect &r) const
- void **drawGroupIndicator** (QPainter \*p, const QRect &r, int numberOfGroupedImages, bool open) const
- void **drawImageFormat** (QPainter \*p, const QRect &dimsRect, const QString &mime) const
- void **drawImageSize** (QPainter \*p, const QRect &dimsRect, const QSize &dims) const
- void **drawLockIndicator** (QPainter \*p, const QRect &r, int lockStatus) const
- void **drawMouseOverRect** (QPainter \*p, const QStyleOptionViewItem &option) const
- void **drawName** (QPainter \*p, const QRect &nameRect, const QString &name) const
- void **drawPickLabelIcon** (QPainter \*p, const QRect &r, int pickLabel) const
- void **drawRating** (QPainter \*p, const QModelIndex &index, const QRect &ratingRect, int rating, bool isSelected) const
- void **drawTags** (QPainter \*p, const QRect &r, const QString &tagsString, bool isSelected) const
- QRect **drawThumbnail** (QPainter \*p, const QRect &thumbRect, const QPixmap &background, const QPixmap &thumbnail) const
 

*Use the tool methods for painting in subclasses.*
- void **prepareBackground** ()
- void **prepareFonts** ()
- void **prepareMetrics** (int maxWidth)
- void **prepareRatingPixmap** (bool composeOverBackground=true)
- QPixmap **ratingPixmap** (int rating, bool selected) const
 

*Returns the relevant pixmap from the cached rating pixmaps.*

## Protected Member Functions inherited from [Digikam::DItemDelegate](#)

- QString **squeezedTextCached** (QPainter \*const p, int width, const QString &text) const
- QPixmap **thumbnailBorderPixmap** (const QSize &pixSize, bool isGrouped=false) const

## Protected Member Functions inherited from [Digikam::ItemDelegateOverlayContainer](#)

- virtual void **drawOverlays** (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index) const
- virtual void **overlayDestroyed** (QObject \*o)

*Declare as slot in the derived class calling this method.*

## Additional Inherited Members

## Signals inherited from [Digikam::ItemViewImportDelegate](#)

- void **hideNotification** ()
- void **requestNotification** (const QModelIndex &index, const QString &message)

## Signals inherited from [Digikam::DItemDelegate](#)

- void **gridSizeChanged** (const QSize &newSize)
- void **visualChange** ()

## Static Public Member Functions inherited from [Digikam::ImportDelegate](#)

- static QPixmap **retrieveThumbnailPixmap** (const QModelIndex &index, int thumbnailSize)  
*Retrieve the thumbnail pixmap in given size for the [ImportItemModel::ThumbnailRole](#) for the given index from the given index, which must adhere to [ImportThumbnailModel](#) semantics.*

## Static Public Member Functions inherited from [Digikam::DItemDelegate](#)

- static QString **dateToString** (const QDateTime &datetime)
- static QPixmap **makeDragPixmap** (const QStyleOptionViewItem &option, const QList< QModelIndex > &indexes, double displayRatio, const QPixmap &suggestedPixmap=QPixmap())
- static QString **squeezedText** (const QFontMetrics &fm, int width, const QString &text)

## Protected Slots inherited from [Digikam::ImportDelegate](#)

- void **modelChanged** ()
- void **modelContentsChanged** ()

## Protected Slots inherited from [Digikam::ItemViewImportDelegate](#)

- void **overlayDestroyed** (QObject \*o) override
- void **slotSetupChanged** ()
- void **slotThemeChanged** ()

## Protected Attributes inherited from [Digikam::ItemViewImportDelegate](#)

- ItemViewImportDelegatePrivate \*const **d\_ptr** = nullptr

## Protected Attributes inherited from [Digikam::ItemDelegateOverlayContainer](#)

- QList< [ItemDelegateOverlay](#) \* > **m\_overlays**

## 9.771.1 Member Function Documentation

### 9.771.1.1 acceptsActivation()

```
bool Digikam::ImportThumbnailDelegate::acceptsActivation (
    const QPoint & pos,
    const QRect & visualRect,
    const QModelIndex & index,
    QRect * activationRect ) const [override], [virtual]
```

Reimplemented from [Digikam::ImportDelegate](#).

### 9.771.1.2 setDefaultViewOptions()

```
void Digikam::ImportThumbnailDelegate::setDefaultViewOptions (
    const QStyleOptionViewItem & option ) [override], [virtual]
```

option.rect shall be the viewport rectangle. Call on resize, font change.

Reimplemented from [Digikam::ImportDelegate](#).

### 9.771.1.3 updateContentWidth()

```
void Digikam::ImportThumbnailDelegate::updateContentWidth ( ) [override], [protected], [virtual]
```

This is the maximum width of all content rectangles, typically excluding margins on both sides.

Reimplemented from [Digikam::ImportDelegate](#).

### 9.771.1.4 updateRects()

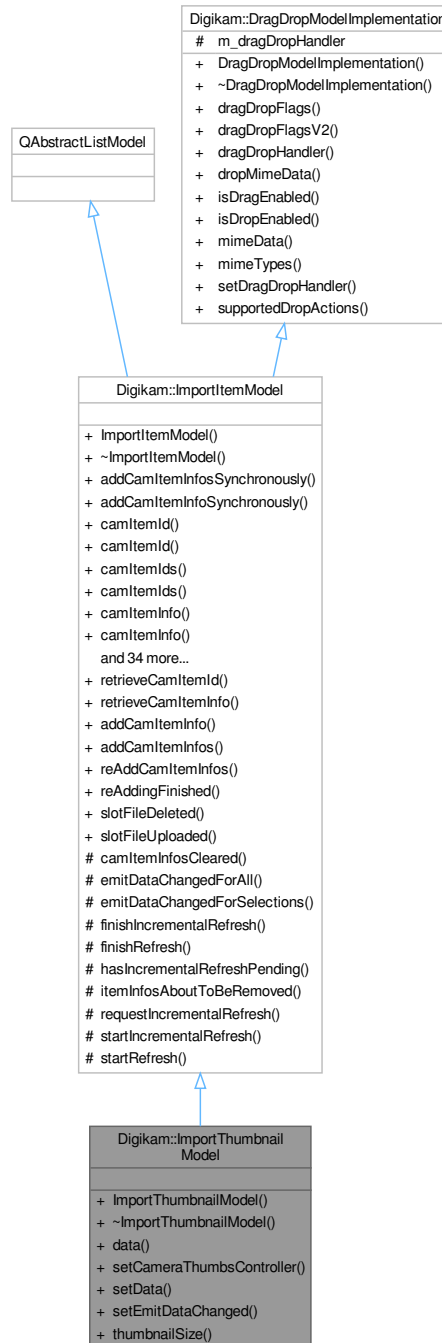
```
void Digikam::ImportThumbnailDelegate::updateRects ( ) [override], [protected], [virtual]
```

The paint() method operates depending on these rects.

Implements [Digikam::ImportDelegate](#).

## 9.772 Digikam::ImportThumbnailModel Class Reference

Inheritance diagram for Digikam::ImportThumbnailModel:



### Signals

- void **thumbnailAvailable** (const QModelIndex &index, int requestedSize)
- void **thumbnailFailed** (const QModelIndex &index, int requestedSize)

## Signals inherited from [Digikam::ImportItemModel](#)

- void [allRefreshingFinished](#) ()  
*Signals that the model has finished currently with all scheduled refreshing, full or incremental, and all preprocessing.*
- void [itemInfosAboutToBeAdded](#) (const QList< [CamItemInfo](#) > &infos)  
*Informs that ItemInfos will be added to the model.*
- void [itemInfosAboutToBeRemoved](#) (const QList< [CamItemInfo](#) > &infos)  
*Informs that CamItemInfos will be removed from the model.*
- void [itemInfosAdded](#) (const QList< [CamItemInfo](#) > &infos)  
*Informs that ItemInfos have been added to the model.*
- void [itemInfosRemoved](#) (const QList< [CamItemInfo](#) > &infos)  
*Informs that CamItemInfos have been removed from the model.*
- void [preprocess](#) (const QList< [CamItemInfo](#) > &infos)  
*Connect to this signal only if you are the current preprocessor.*
- void [processAdded](#) (const QList< [CamItemInfo](#) > &infos)
- void [readyForIncrementalRefresh](#) ()  
*Signals that the model is right now ready to start an incremental refresh.*

## Public Member Functions

- [ImportThumbnailModel](#) (QObject \*const parent)  
*This model provides thumbnail loading, it uses the Camera Controller to retrieve thumbnails for CamItemInfos.*
- QVariant [data](#) (const QModelIndex &index, int role=Qt::DisplayRole) const override  
*Handles the ThumbnailRole.*
- void [setCameraThumbsController](#) ([CameraThumbsCtrl](#) \*const thumbsCtrl) override  
*Sets the camera thumbs controller which is used to get the thumbnails for item infos.*
- bool [setData](#) (const QModelIndex &index, const QVariant &value, int role=Qt::DisplayRole) override  
*You can override the current thumbnail size by giving an integer value for ThumbnailRole.*
- void [setEmitDataChanged](#) (bool emitSignal)  
*Enable emitting dataChanged() when a thumbnail becomes available.*
- [ThumbnailSize](#) [thumbnailSize](#) () const  
*Get the thumbnail size.*

## Public Member Functions inherited from [Digikam::ImportItemModel](#)

- [ImportItemModel](#) (QObject \*const parent=nullptr)
- void [addCamItemInfosSynchronously](#) (const Digikam::CamItemInfoList &infos)
- void [addCamItemInfoSynchronously](#) (const [CamItemInfo](#) &info)  
*addCamItemInfo() is asynchronous if a preprocessor is set.*
- qlonglong [camItemId](#) (const QModelIndex &index) const
- qlonglong [camItemId](#) (int row) const
- QList< qlonglong > [camItemIds](#) () const
- QList< qlonglong > [camItemIds](#) (const QList< QModelIndex > &indexes) const
- [CamItemInfo](#) [camItemInfo](#) (const QModelIndex &index) const  
*Returns the [CamItemInfo](#) object, reference from the underlying data pointed to by the index.*
- [CamItemInfo](#) [camItemInfo](#) (const QUrl &fileUrl) const
- [CamItemInfo](#) [camItemInfo](#) (int row) const  
*Returns the [CamItemInfo](#) object, reference from the underlying data of the given row (parent is the invalid QModelIndex, column is 0).*
- [CamItemInfo](#) & [camItemInfoRef](#) (const QModelIndex &index) const
- [CamItemInfo](#) & [camItemInfoRef](#) (int row) const



- `QList< CamItemInfo > camItemInfos ()` const
- `CamItemInfoList camItemInfos (const QList< QModelIndex > &indexes)` const
- `QList< CamItemInfo > camItemInfos (const QUrl &fileUrl)` const
- `void clearCamItemInfos ()`
  - Clears the CamItemInfos and resets the model.*
- `QVariant data (const QModelIndex &index, int role)` const override
- `Qt::ItemFlags flags (const QModelIndex &index)` const override
- `bool hasImage (const CamItemInfo &info)` const
- `bool hasImage (qulonglong id)` const
- `QVariant headerData (int section, Qt::Orientation orientation, int role)` const override
- `QModelIndex index (int row, int column, const QModelIndex &parent)` const override
- `QList< QModelIndex > indexesForCamItemId (qulonglong id)` const
- `QList< QModelIndex > indexesForCamItemInfo (const CamItemInfo &info)` const
- `QList< QModelIndex > indexesForUrl (const QUrl &fileUrl)` const
- `QModelIndex indexForCamItemId (qulonglong id)` const
- `QModelIndex indexForCamItemInfo (const CamItemInfo &info)` const
  - Return the index of a given [CamItemInfo](#), if it exists in the model.*
- `QModelIndex indexForUrl (const QUrl &fileUrl)` const
  - Returns the index or [CamItemInfo](#) object from the underlying data for the given file url.*
- `bool isEmpty ()` const
- `bool isRefreshing ()` const
  - Returns true if this model is currently refreshing.*
- `bool keepsFileUrlCache ()` const
- `int numberOfIndexesForCamItemId (qulonglong id)` const
- `int numberOfIndexesForCamItemInfo (const CamItemInfo &info)` const
- `void removeCamItemInfo (const CamItemInfo &info)`
- `void removeCamItemInfos (const QList< CamItemInfo > &infos)`
- `void removeIndex (const QModelIndex &index)`
  - Remove the given infos or indexes directly from the model.*
- `void removeIndexes (const QList< QModelIndex > &indexes)`
- `int rowCount (const QModelIndex &parent)` const override
  - QAbstractListModel implementation.*
- `void setCamItemInfos (const CamItemInfoList &infos)`
  - Clears and adds infos.*
- `void setKeepsFileUrlCache (bool keepCache)`
  - If a cache is kept, lookup by file path is fast, without a cache it is O(n).*
- `DECLARE_MODEL_DRAG_DROP_METHODS void setSendRemovalSignals (bool send)`
  - DragDrop methods.*
- `QList< CamItemInfo > uniqueCamItemInfos ()` const

## Public Member Functions inherited from [Digikam::DragDropModelImplementation](#)

- `DragDropModelImplementation ()`=default
  - A class providing a sample implementation for a QAbstractItemModel redirecting drag-and-drop support to a handler.*
- `virtual Qt::ItemFlags dragDropFlags (const QModelIndex &index)` const
  - Call from your flags() method, adding the relevant drag drop flags.*
- `Qt::ItemFlags dragDropFlagsV2 (const QModelIndex &index)` const
  - This is an alternative approach to [dragDropFlags\(\)](#).*
- `AbstractItemDragDropHandler * dragDropHandler ()` const
- `bool dropMimeData (const QMimeData *, Qt::DropAction, int, int, const QModelIndex &)`
- `virtual bool isDragEnabled (const QModelIndex &index)` const
- `virtual bool isDropEnabled (const QModelIndex &index)` const

- `QMimeType * mimeType` (const `QModelIndexList &indexes`) const
- `QStringList mimeTypees` () const
- void **setDragDropHandler** (`AbstractItemDragDropHandler *handler`)  
*Set a drag drop handler.*
- `Qt::DropActions supportedDropActions` () const  
*Implements the relevant `QAbstractItemModel` methods for drag and drop.*

### Additional Inherited Members

### Public Types inherited from `Digikam::ImportItemModel`

- enum `ImportItemModelRoles` {  
`ImportItemModelPointerRole` = `Qt::UserRole` , `ImportItemModelInternalId` = `Qt::UserRole` + 1 ,  
`ThumbnailRole` = `Qt::UserRole` + 2 , `ExtraDataRole` = `Qt::UserRole` + 3 ,  
`ExtraDataDuplicateCount` = `Qt::UserRole` + 6 , `FilterModelRoles` = `Qt::UserRole` + 100 }

### Public Slots inherited from `Digikam::ImportItemModel`

- void **addCamItemInfo** (const `CamItemInfo &info`)
- void **addCamItemInfos** (const `CamItemInfoList &infos`)
- void **reAddCamItemInfos** (const `CamItemInfoList &infos`)
- void **reAddingFinished** ()
- void **slotFileDeleted** (const `QString &folder`, const `QString &file`, bool status)
- void **slotFileUploaded** (const `CamItemInfo &info`)

### Static Public Member Functions inherited from `Digikam::ImportItemModel`

- static `qulonglong retrieveCamItemId` (const `QModelIndex &index`)
- static `CamItemInfo retrieveCamItemInfo` (const `QModelIndex &index`)  
*Retrieve the `CamItemInfo` object from the `data()` function of the given index. The index may be from a `QSortFilterProxyModel` as long as an `ImportItemModel` is at the end.*

### Protected Member Functions inherited from `Digikam::ImportItemModel`

- virtual void **camItemInfosCleared** ()  
*Called when the internal storage is cleared.*
- void **emitDataChangedForAll** ()
- void **emitDataChangedForSelections** (const `QItemSelection &selection`)
- void **finishIncrementalRefresh** ()
- void **finishRefresh** ()
- bool **hasIncrementalRefreshPending** () const
- virtual void **itemInfosAboutToBeRemoved** (int, int)  
*Called before `rowsAboutToBeRemoved`.*
- void **requestIncrementalRefresh** ()  
*As soon as the model is ready to start an incremental refresh, the signal `readyForIncrementalRefresh()` will be emitted.*
- void **startIncrementalRefresh** ()  
*Starts an incremental refresh operation.*
- void **startRefresh** ()  
*Subclasses that add `CamItemInfos` in batches shall call `startRefresh()` when they start sending batches and finish `Refresh()` when they have finished.*

## Protected Attributes inherited from [Digikam::DragDropModelImplementation](#)

- [AbstractItemDragDropHandler](#) \* `m_dragDropHandler` = nullptr

### 9.772.1 Constructor & Destructor Documentation

#### 9.772.1.1 ImportThumbnailModel()

```
Digikam::ImportThumbnailModel::ImportThumbnailModel (
    QObject *const parent ) [explicit]
```

It also provides preloading of thumbnails, and caching facility. Thumbnails size can be adjusted.

### 9.772.2 Member Function Documentation

#### 9.772.2.1 data()

```
QVariant Digikam::ImportThumbnailModel::data (
    const QModelIndex & index,
    int role = Qt::DisplayRole ) const [override]
```

If the pixmap is available, returns it in the QVariant. If it still needs to be loaded, returns a null QVariant and emits `thumbnailAvailable()` as soon as it is available.

#### 9.772.2.2 setCameraThumbsController()

```
void Digikam::ImportThumbnailModel::setCameraThumbsController (
    CameraThumbsCtrl *const thumbsCtrl ) [override], [virtual]
```

Reimplemented from [Digikam::ImportItemModel](#).

#### 9.772.2.3 setData()

```
bool Digikam::ImportThumbnailModel::setData (
    const QModelIndex & index,
    const QVariant & value,
    int role = Qt::DisplayRole ) [override]
```

Set a null QVariant to use the thumbnail size set by `setThumbnailSize()` again. The index given here is ignored for this purpose.

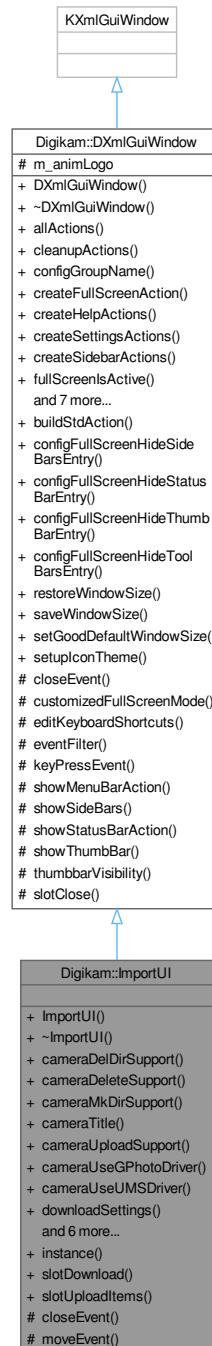
#### 9.772.2.4 setEmitDataChanged()

```
void Digikam::ImportThumbnailModel::setEmitDataChanged (
    bool emitSignal )
```

The `thumbnailAvailable()` signal will be emitted in any case. Default is true.

## 9.773 Digikam::ImportUI Class Reference

Inheritance diagram for Digikam::ImportUI:



### Public Slots

- void **slotDownload** (bool onlySelected, bool deleteAfter, [Album](#) \*pAlbum=nullptr)
- void **slotUploadItems** (const QList< QUrl > &)

## Signals

- void **signalEscapePressed** ()
- void **signalLastDestination** (const QUrl &)
- void **signalNewSelection** (bool)
- void **signalPreviewRequested** (const [CamItemInfo](#) &, bool)
- void **signalWindowHasMoved** ()

## Public Member Functions

- **ImportUI** (const QString &cameraTitle, const QString &model, const QString &port, const QString &path, int startIndex)
- bool **cameraDelDirSupport** () const
- bool **cameraDeleteSupport** () const
- bool **cameraMkDirSupport** () const
- QString **cameraTitle** () const
- bool **cameraUploadSupport** () const
- bool **cameraUseGPhotoDriver** () const
- bool **cameraUseUMSDriver** () const
- [DownloadSettings](#) **downloadSettings** () const
- void **enableZoomMinusAction** (bool val)
- void **enableZoomPlusAction** (bool val)
- [CameraThumbsCtrl](#) \* **getCameraThumbsCtrl** () const
- [DInfoInterface](#) \* **infoface** ([DPluginAction](#) \*const) override  
*Return the interface instance to access to items information.*
- bool **isBusy** () const
- bool **isClosed** () const

## Public Member Functions inherited from [Digikam::DXmlGuiWindow](#)

- **DXmlGuiWindow** (QWidget \*const parent=nullptr, Qt::WindowFlags f=Qt::WindowFlags())
- QList< QAction \* > **allActions** () const  
*Return all actions from internal collection.*
- void **cleanupActions** ()  
*Cleanup unwanted actions from action collection.*
- QString **configGroupName** () const
- void **createFullScreenAction** (const QString &name)  
*Create Full-screen action to action collection instance from managed window set through setManagedWindow().*
- void **createHelpActions** (const QString &handbookSection, bool coreOptions=true)  
*Create common actions from Help menu for all digiKam main windows.*
- void **createSettingsActions** ()  
*Create common actions to setup all digiKam main windows.*
- void **createSidebarActions** ()  
*Create common actions to handle side-bar through keyboard shortcuts.*
- bool **fullScreensIsActive** () const  
*Return true if managed window is currently in Full Screen Mode.*
- void **readFullScreenSettings** (const KConfigGroup &group)  
*Read full-screen settings from KDE config file.*
- virtual void **registerExtraPluginsActions** (QString &)
- void **registerPluginsActions** ()  
*Register all generic plugins action to this instance.*
- void **setConfigGroupName** (const QString &name)  
*Manage config group name used by window instance to get/set settings from config file.*
- void **setFullScreenOptions** (int options)  
*Set full-screen options to managed window.*
- void **unminimizeAndActivateWindow** ()

### Static Public Member Functions

- static [ImportUI](#) \* `instance` ()

### Static Public Member Functions inherited from [Digikam::DXmlGuiWindow](#)

- static QAction \* **buildStdAction** (StdActionType type, const QObject \*const recvr, const char \*const slot, QObject \*const parent)
- static QString **configFullScreenHideSideBarsEntry** ()
- static QString **configFullScreenHideStatusBarEntry** ()
- static QString **configFullScreenHideThumbBarEntry** ()
- static QString **configFullScreenHideToolBarsEntry** ()

*Shared with [FullScreenSettings](#).*

- static void **restoreWindowSize** (QWindow \*const win, const KConfigGroup &group)
- static void **saveWindowSize** (QWindow \*const win, KConfigGroup &group)
- static void **setGoodDefaultWindowSize** (QWindow \*const win)
- static void **setupIconTheme** ()

*If we have some local breeze icon resource, prefer it.*

### Protected Member Functions

- void **closeEvent** (QCloseEvent \*e) override
- void **moveEvent** (QMoveEvent \*e) override

### Protected Member Functions inherited from [Digikam::DXmlGuiWindow](#)

- void **closeEvent** (QCloseEvent \*e) override
- void **editKeyboardShortcuts** (KActionCollection \*const extraac=nullptr, const QString &actitle=QString())  
*Call this method from your main window to show keyboard shortcut config dialog with an extra action collection to configure.*
- bool **eventFilter** (QObject \*obj, QEvent \*ev) override
- void **keyPressEvent** (QKeyEvent \*e) override
- QAction \* **showMenuBarAction** () const
- QAction \* **showStatusBarAction** () const

### Additional Inherited Members

### Protected Slots inherited from [Digikam::DXmlGuiWindow](#)

- bool **slotClose** ()

### Protected Attributes inherited from [Digikam::DXmlGuiWindow](#)

- [DLogoAction](#) \* `m_animLogo` = nullptr

## 9.773.1 Member Function Documentation

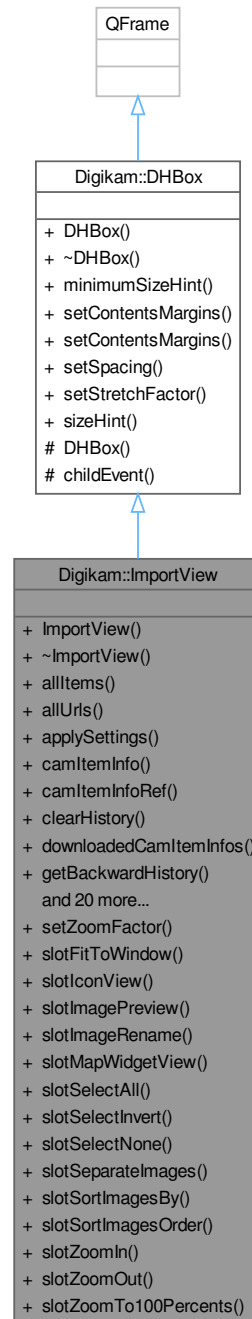
### 9.773.1.1 infoIface()

```
DInfoInterface * Digikam::ImportUI::infoIface (  
    DPluginAction * const ac ) [inline], [override], [virtual]
```

Implements [Digikam::DXmlGuiWindow](#).

## 9.774 Digikam::ImportView Class Reference

Inheritance diagram for Digikam::ImportView:



### Public Slots

- void **setZoomFactor** (double zoom)
- void **slotFitToWindow** ()



- void **slotIconView** ()
- void **slotImagePreview** ()
- void **slotImageRename** ()
- void **slotMapView** ()
- void **slotSelectAll** ()
- void **slotSelectInvert** ()
- void **slotSelectNone** ()
- void **slotSeparateImages** (int mode)
- void **slotSortImagesBy** (int sortBy)
- void **slotSortImagesOrder** (int order)
- void **slotZoomIn** ()
- View Action slots.*
- void **slotZoomOut** ()
- void **slotZoomTo100Percents** ()

### Signals

- void **signalImageSelected** (const CamItemInfoList &selectedImage, const CamItemInfoList &allImages)
- void **signalNewSelection** (bool hasSelection)
- void **signalNoCurrentItem** ()
- void **signalSelectionChanged** (int numberOfSelectedItems)
- void **signalSwitchedToIconView** ()
- void **signalSwitchedToMapView** ()
- void **signalSwitchedToPreview** ()
- void **signalThumbSizeChanged** (int)
- void **signalZoomChanged** (double)

### Public Member Functions

- **ImportView** ([Digikam::ImportUI](#) \*const ui, QWidget \*const parent)
- QList< [CamItemInfo](#) > **allItems** () const
- QList< QUrl > **allUrls** () const
- void **applySettings** ()
- [CamItemInfo](#) **camItemInfo** (const QString &folder, const QString &file) const
- [CamItemInfo](#) & **camItemInfoRef** (const QString &folder, const QString &file) const
- void **clearHistory** ()
- int **downloadedCamItemInfos** () const
- void **getBackwardHistory** (QStringList &titles)
- void **getForwardHistory** (QStringList &titles)
- bool **hasCurrentItem** () const
- bool **hasImage** (const [CamItemInfo](#) &info) const
- void **hideSideBars** ()
- [ImportFilterModel](#) \* **importFilterModel** () const
- bool **isSelected** (const QUrl &url) const
- void **refreshView** ()
- void **scrollTo** (const QString &folder, const QString &file)
- QList< [CamItemInfo](#) > **selectedCamItemInfos** () const
- QList< QUrl > **selectedUrls** () const
- void **setSelectedCamItemInfos** (const CamItemInfoList &infos) const
- void **setThumbSize** (int size)
- void **showSideBars** ()
- [ThumbnailSize](#) **thumbnailSize** () const
- void **toggleFullScreen** (bool set)
- void **toggleShowBar** (bool b)
- void **updateIconView** ()
- [ImportStackedView::StackedViewMode](#) **viewMode** () const
- double **zoomMax** () const
- double **zoomMin** () const

## Public Member Functions inherited from Digikam::DHBox

- **DHBox** (QWidget \*const parent=nullptr)
- QSize **minimumSizeHint** () const override
- void **setContentsMargins** (const QMargins &margins)
- void **setContentsMargins** (int left, int top, int right, int bottom)
- void **setSpacing** (int space)
- void **setStretchFactor** (QWidget \*const widget, int stretch)
- QSize **sizeHint** () const override

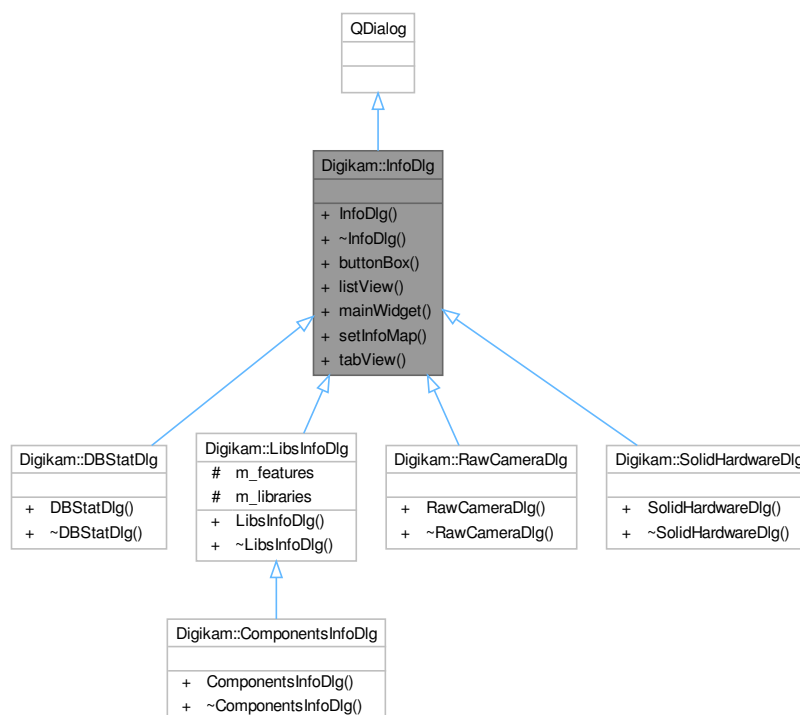
## Additional Inherited Members

## Protected Member Functions inherited from Digikam::DHBox

- **DHBox** (bool vertical, QWidget \*const parent)
- void **childEvent** (QChildEvent \*e) override

## 9.775 Digikam::InfoDlg Class Reference

Inheritance diagram for Digikam::InfoDlg:



### Public Member Functions

- **InfoDlg** (QWidget \*const parent)
- QDialogButtonBox \* **buttonBox** () const
- QTreeWidget \* **listView** () const
- QWidget \* **mainWidget** () const
- virtual void **setInfoMap** (const QMap< QString, QString > &list)
- QTabWidget \* **tabView** () const

## 9.776 Digikam::InfraredContainer Class Reference

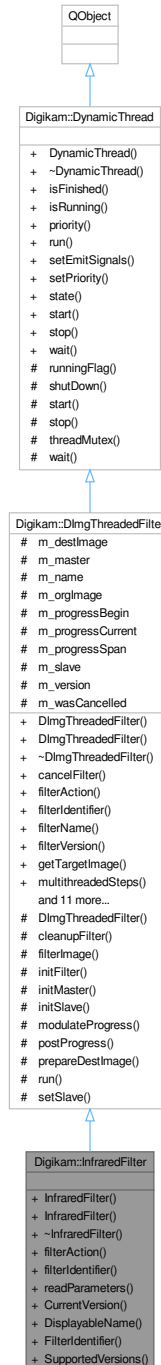
### Public Attributes

- double **blueGain** = -0.8
- double **greenGain** = 2.1
- double **redGain** = 0.4
- int **sensibility** = 200

*Sensibility: 200..2600 ISO.*

## 9.777 Digikam::InfraredFilter Class Reference

Inheritance diagram for Digikam::InfraredFilter:



### Public Member Functions

- **InfraredFilter** ([Dlmg](#) \*const orgImage, [QObject](#) \*const parent=nullptr, const [InfraredContainer](#) &settings=[InfraredContainer](#)())
- **InfraredFilter** ([QObject](#) \*const parent=nullptr)

- [FilterAction filterAction \(\)](#) override  
*Returns the action description corresponding to currently set options.*
- [QString filterIdentifier \(\)](#) const override  
*Return the identifier for this filter in the image history.*
- void [readParameters \(const FilterAction &action\)](#) override

## Public Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter \(DImg \\*const orgImage, QObject \\*const parent, const QString &name=QString\(\)\)](#)  
*Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter \(QObject \\*const parent=nullptr, const QString &name=QString\(\)\)](#)  
*Constructs a filter without argument.*
- virtual void [cancelFilter \(\)](#)  
*Cancel the threaded computation.*
- const [QString &filterName \(\)](#)
- int [filterVersion \(\)](#) const
- [DImg getTargetImage \(\)](#)
- [QList< int > multithreadedSteps \(int stop, int start=0\)](#) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool [parametersSuccessfullyRead \(\)](#) const  
*Optional: error handling for readParameters.*
- virtual [QString readParametersError \(const FilterAction &actionThatFailed\)](#) const
- void [setFilterName \(const QString &name\)](#)
- void [setFilterVersion \(int version\)](#)  
*Replaying a filter action: Set the filter version.*
- void [setOriginalImage \(const DImg &orgImage\)](#)
- void [setupAndStartDirectly \(const DImg &orgImage, DImgThreadedFilter \\*const master, int progressBegin=0, int progressEnd=100\)](#)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void [setupFilter \(const DImg &orgImage\)](#)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void [startFilter \(\)](#)  
*Start the threaded computation.*
- virtual void [startFilterDirectly \(\)](#)  
*Start computation of this filter, directly in this thread.*
- virtual [QList< int > supportedVersions \(\)](#) const

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread \(QObject \\*const parent=nullptr\)](#)  
*This class extends QRunnable, so you have to reimplement virtual void [run\(\)](#).*
- [~DynamicThread \(\)](#) override  
*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool [isFinished \(\)](#) const
- bool [isRunning \(\)](#) const
- [QThread::Priority priority \(\)](#) const
- void [setEmitSignals \(bool emitThem\)](#)
- void [setPriority \(QThread::Priority priority\)](#)  
*Sets the priority for this dynamic thread.*
- State [state \(\)](#) const

### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from Digikam::DImgThreadedFilter

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int [progress](#))  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int [progress](#))  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from Digikam::DynamicThread

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from Digikam::DImgThreadedFilter

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.777.1 Member Function Documentation

### 9.777.1.1 filterAction()

`FilterAction` Digikam::InfraredFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.777.1.2 filterIdentifier()

`QString` Digikam::InfraredFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.777.1.3 readParameters()

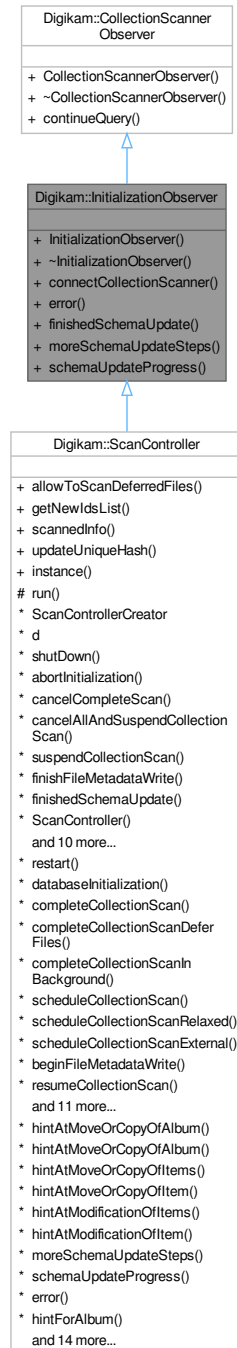
```
void Digikam::InfraredFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).



## 9.778 Digikam::InitializationObserver Class Reference

Inheritance diagram for Digikam::InitializationObserver:



### Public Types

- enum **UpdateResult** { **UpdateSuccess** , **UpdateError** , **UpdateErrorMustAbort** }

**Public Member Functions**

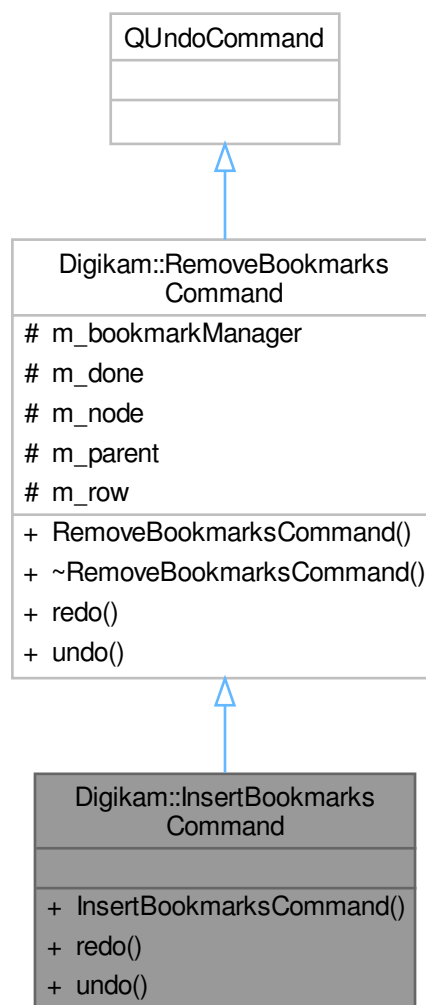
- virtual void **connectCollectionScanner** ([CollectionScanner](#) \*const scanner)=0
- virtual void **error** (const QString &errorMessage)=0
- virtual void **finishedSchemaUpdate** (UpdateResult result)=0
- virtual void **moreSchemaUpdateSteps** (int numberOfSteps)=0
- virtual void **schemaUpdateProgress** (const QString &message, int numberOfSteps=1)=0

**Public Member Functions inherited from [Digikam::CollectionScannerObserver](#)**

- virtual bool **continueQuery** ()=0

**9.779 Digikam::InsertBookmarksCommand Class Reference**

Inheritance diagram for Digikam::InsertBookmarksCommand:



**Public Member Functions**

- **InsertBookmarksCommand** ([BookmarksManager](#) \*const mngr, [BookmarkNode](#) \*const parent, [BookmarkNode](#) \*const node, int row)
- void **redo** () override
- void **undo** () override

**Public Member Functions inherited from [Digikam::RemoveBookmarksCommand](#)**

- **RemoveBookmarksCommand** ([BookmarksManager](#) \*const mngr, [BookmarkNode](#) \*const parent, int row)
- void **redo** () override
- void **undo** () override

**Additional Inherited Members****Protected Attributes inherited from [Digikam::RemoveBookmarksCommand](#)**

- [BookmarksManager](#) \* **m\_bookmarkManager** = nullptr
- bool **m\_done** = false
- [BookmarkNode](#) \* **m\_node** = nullptr
- [BookmarkNode](#) \* **m\_parent** = nullptr
- int **m\_row** = 0

**9.780 Digikam::InternalTagName Class Reference****Static Public Member Functions**

- static [QLatin1String](#) **colorLabelBlack** ()
- static [QLatin1String](#) **colorLabelBlue** ()
- static [QLatin1String](#) **colorLabelGray** ()
- static [QLatin1String](#) **colorLabelGreen** ()
- static [QLatin1String](#) **colorLabelMagenta** ()
- static [QLatin1String](#) **colorLabelNone** ()
- static [QLatin1String](#) **colorLabelOrange** ()
- static [QLatin1String](#) **colorLabelRed** ()
- static [QLatin1String](#) **colorLabelWhite** ()
- static [QLatin1String](#) **colorLabelYellow** ()
- static [QLatin1String](#) **currentVersion** ()
- static [QLatin1String](#) **intermediateVersion** ()
- static [QLatin1String](#) **needResolvingHistory** ()
- static [QLatin1String](#) **needTaggingHistoryGraph** ()
- static [QLatin1String](#) **originalVersion** ()
- static [QLatin1String](#) **pickLabelAccepted** ()
- static [QLatin1String](#) **pickLabelNone** ()
- static [QLatin1String](#) **pickLabelPending** ()
- static [QLatin1String](#) **pickLabelRejected** ()
- static [QLatin1String](#) **scannedForFaces** ()
- static [QLatin1String](#) **versionAlwaysVisible** ()

## 9.781 Digikam::InvertFilter Class Reference

Inheritance diagram for Digikam::InvertFilter:



### Public Member Functions

- **InvertFilter** ([DImg](#) \*const orgImage, [QObject](#) \*const parent=nullptr)
- **InvertFilter** ([DImgThreadedFilter](#) \*const parentFilter, const [DImg](#) &orgImage, [DImg](#) &destImage, int progressBegin=0, int progressEnd=100)

- **InvertFilter** (QObject \*const parent=nullptr)
- **FilterAction filterAction** () override  
*Returns the action description corresponding to currently set options.*
- **QString filterIdentifier** () const override  
*Return the identifier for this filter in the image history.*
- void **readParameters** (const **FilterAction** &action) override

## Public Member Functions inherited from Digikam::DImgThreadedFilter

- **DImgThreadedFilter** (DImg \*const orgImage, QObject \*const parent, const QString &name=QString())  
*Constructs a filter with all arguments (ready to use).*
- **DImgThreadedFilter** (QObject \*const parent=nullptr, const QString &name=QString())  
*Constructs a filter without argument.*
- virtual void **cancelFilter** ()  
*Cancel the threaded computation.*
- const QString & **filterName** ()
- int **filterVersion** () const
- **DImg getTargetImage** ()
- QList< int > **multithreadedSteps** (int stop, int start=0) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool **parametersSuccessfullyRead** () const  
*Optional: error handling for readParameters.*
- virtual QString **readParametersError** (const **FilterAction** &actionThatFailed) const
- void **setFilterName** (const QString &name)
- void **setFilterVersion** (int version)  
*Replaying a filter action: Set the filter version.*
- void **setOriginalImage** (const **DImg** &orgImage)
- void **setupAndStartDirectly** (const **DImg** &orgImage, **DImgThreadedFilter** \*const master, int progress←Begin=0, int progressEnd=100)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void **setupFilter** (const **DImg** &orgImage)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void **startFilter** ()  
*Start the threaded computation.*
- virtual void **startFilterDirectly** ()  
*Start computation of this filter, directly in this thread.*
- virtual QList< int > **supportedVersions** () const

## Public Member Functions inherited from Digikam::DynamicThread

- **DynamicThread** (QObject \*const parent=nullptr)  
*This class extends QRunnable, so you have to reimplement virtual void run().*
- **~DynamicThread** () override  
*The destructor calls stop() and wait(), but if you, in your destructor, delete any data that is accessed by your run() method, you must call stop() and wait() before yourself.*
- bool **isFinished** () const
- bool **isRunning** () const
- QThread::Priority **priority** () const
- void **setEmitSignals** (bool emitThem)
- void **setPriority** (QThread::Priority priority)  
*Sets the priority for this dynamic thread.*
- State **state** () const

### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from Digikam::DImgThreadedFilter

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from Digikam::DynamicThread

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from Digikam::DImgThreadedFilter

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.781.1 Member Function Documentation

### 9.781.1.1 filterAction()

`FilterAction` Digikam::InvertFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.781.1.2 filterIdentifier()

`QString` Digikam::InvertFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.781.1.3 readParameters()

`void` Digikam::InvertFilter::readParameters (   
     const `FilterAction` & action ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

## 9.782 Digikam::IOFileSettings Class Reference

### Public Attributes

- int **AVIFCompression** = 75  
*AVIF quality value.*
- bool **AVIFLossLess** = true  
*AVIF lossless compression.*
- int **HEIFCompression** = 75  
*HEIF quality value.*
- bool **HEIFLossLess** = true  
*HEIF lossless compression.*
- int **JPEG2000Compression** = 75  
*JPEG2000 quality value.*
- bool **JPEG2000LossLess** = true  
*JPEG2000 lossless compression.*
- int **JPEGCompression** = 75  
*JPEG quality value.*
- int **JPEGSubSampling** = 1  
*JPEG chroma sub-sampling value.*
- int **JXLCompression** = 75  
*JXL quality value.*
- bool **JXLLossLess** = true  
*JXL lossless compression.*
- int **PGFCompression** = 3  
*PGF quality value.*
- bool **PGFLossLess** = true



- *PGF lossless compression.*
- int **PNGCompression** = 9  
*PNG compression value.*
- [DRawDecoding](#) **rawDecodingSettings**  
*RAW File decoding options.*
- QString **rawImportToolId** = QLatin1String("org.kde.digikam.plugin.rawimport.Native")
- bool **TIFFCompression** = false  
*TIFF deflate compression.*
- bool **useRAWImport** = true  
*Use Raw Import tool to load a RAW picture.*
- int **WEBPCompression** = 75  
*WEBP quality value.*
- bool **WEBPLossLess** = true  
*WEBP lossless compression.*

## 9.782.1 Member Data Documentation

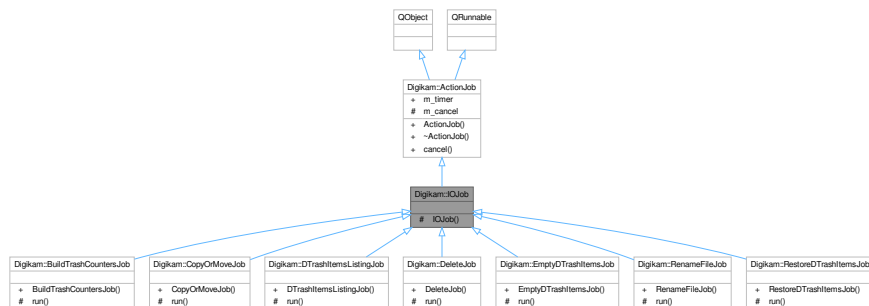
### 9.782.1.1 JPEGSubSampling

```
int Digikam::IOFileSettings::JPEGSubSampling = 1
```

Medium sub-sampling

## 9.783 Digikam::IOJob Class Reference

Inheritance diagram for Digikam::IOJob:



## Signals

- void **signalError** (const QString &errMsg)
- void **signalOneProcessed** (const QUrl &url)

## Signals inherited from [Digikam::ActionJob](#)

- void **signalDone** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job is done.*
- void **signalProgress** (int)  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job progress.*
- void **signalStarted** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager that job is started.*

## Additional Inherited Members

## Public Slots inherited from [Digikam::ActionJob](#)

- void **cancel** ()  
*Call this method to cancel job.*

## Public Member Functions inherited from [Digikam::ActionJob](#)

- **ActionJob** (QObject \*const parent=nullptr)  
*Constructor which delegate deletion of QRunnable instance to [ActionThreadBase](#), not QThreadPool.*
- **~ActionJob** () override  
*Re-implement destructor in you implementation.*

## Public Attributes inherited from [Digikam::ActionJob](#)

- QElapsedTimer **m\_timer**  
*Timer to determine the running time of the job.*

## Protected Attributes inherited from [Digikam::ActionJob](#)

- bool **m\_cancel** = false  
*You can use this boolean in your implementation to know if job must be canceled.*

## 9.784 Digikam::IOJobData Class Reference

### Public Types

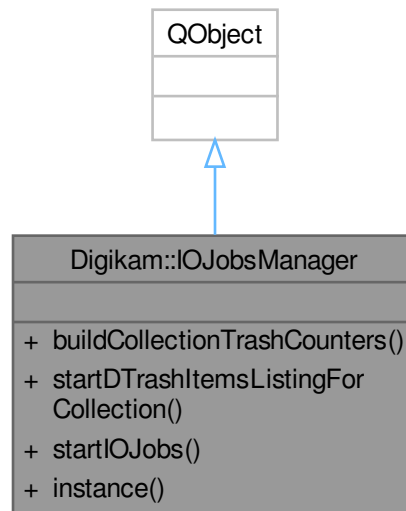
- enum **FileConflict** { **Continue** = 0 , **AutoRename** , **Overwrite** }
- enum **Operation** { **Unknown** = 0 , **CopyAlbum** , **CopyImage** , **CopyFiles** , **CopyToExt** , **MoveAlbum** , **MoveImage** , **MoveFiles** , **Restore** , **Rename** , **Delete** , **Trash** , **Empty** }

## Public Member Functions

- **IOJobData** (int operation, const DTrashItemInfoList &infos)
- **IOJobData** (int operation, const [ItemInfo](#) &info, const QString &newName, bool overwrite=false)
- **IOJobData** (int operation, const QList< [ItemInfo](#) > &infos, const QUrl &dest)
- **IOJobData** (int operation, const QList< [ItemInfo](#) > &infos, [PAlbum](#) \*const dest=nullptr)
- **IOJobData** (int operation, const QList< QUrl > &urls, const QUrl &dest)
- **IOJobData** (int operation, const QList< QUrl > &urls, [PAlbum](#) \*const dest=nullptr)
- **IOJobData** (int operation, [PAlbum](#) \*const src, [PAlbum](#) \*const dest=nullptr)
- [PAlbum](#) \* **destAlbum** () const
- QString **destName** (const QUrl &srcUrl) const
- QUrl **destUrl** (const QUrl &srcUrl=QUrl()) const
- bool **errorOrCancel** () const
- int **fileConflict** () const
- [ItemInfo](#) **findItemInfo** (const QUrl &url) const
- QUrl **getNextUrl** () const
- QString **getProgressId** () const
- QList< [ItemInfo](#) > **itemInfos** () const
- QDateTime **jobTime** () const
- int **operation** () const
- void **setDestUrl** (const QUrl &srcUrl, const QUrl &destUrl)
- void **setErrorOrCancel** (bool err)
- void **setFileConflict** (int fc)
- void **setItemInfos** (const QList< [ItemInfo](#) > &infos)
- void **setProgressId** (const QString &id)
- void **setSourceUrls** (const QList< QUrl > &urls)
- QList< QUrl > **sourceUrls** () const
- [PAlbum](#) \* **srcAlbum** () const
- QList< int > **srcAlbumIds** () const
- DTrashItemInfoList **trashItems** () const

## 9.785 Digikam::IOJobsManager Class Reference

Inheritance diagram for Digikam::IOJobsManager:



### Public Member Functions

- `IOJobsThread * buildCollectionTrashCounters ()`  
*Starts a thread for count trash items for all collections.*
- `IOJobsThread * startDTrashItemsListingForCollection (const QString &collectionPath)`  
*Starts a thread for listing items inside trash for specific collection.*
- `IOJobsThread * startIOJobs (IOJobData *const data)`  
*startIOJobs: Starts a thread to copy, move, delete or rename items*

### Static Public Member Functions

- static `IOJobsManager * instance ()`  
*instance: returns the singleton of IO Jobs Manager*

### Friends

- class `IOJobsManagerCreator`

## 9.785.1 Member Function Documentation

### 9.785.1.1 buildCollectionTrashCounters()

`IOJobsThread * Digikam::IOJobsManager::buildCollectionTrashCounters ()`

#### Returns

`IOJobsThread` pointer for signal/slot connection

### 9.785.1.2 instance()

```
IOJobsManager * Digikam::IOJobsManager::instance ( ) [static]
```

#### Returns

[IOJobsManager](#) global instance

### 9.785.1.3 startDTrashItemsListingForCollection()

```
IOJobsThread * Digikam::IOJobsManager::startDTrashItemsListingForCollection (
    const QString & collectionPath )
```

#### Parameters

<i>collectionPath</i>	the path for collection to list items for it's trash
-----------------------	--

#### Returns

[IOJobsThread](#) pointer for signal/slot connection

### 9.785.1.4 startIOJobs()

```
IOJobsThread * Digikam::IOJobsManager::startIOJobs (
    IOJobData *const data )
```

#### Parameters

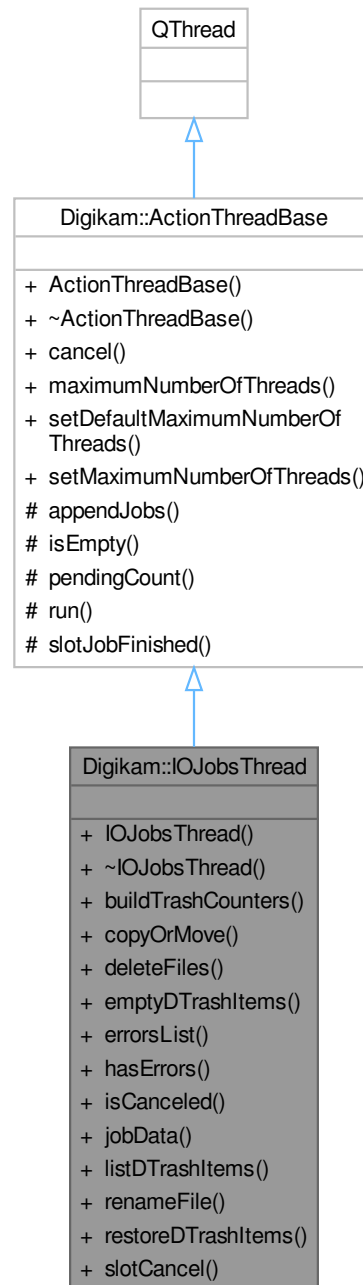
<i>data</i>	<a href="#">IOJobData</a> container with source and destination url
-------------	---

#### Returns

[IOJobsThread](#) pointer for signal/slot connection

## 9.786 Digikam::IOJobsThread Class Reference

Inheritance diagram for Digikam::IOJobsThread:



### Public Slots

- void **slotCancel** ()  
*cancels thread execution*

## Signals

- void **collectionTrashItemInfo** (const [DTrashItemInfo](#) &trashItemInfo)
- void **signalFinished** ()
- void **signalOneProcessed** (const [QUrl](#) &url)
- void **signalRenameFailed** (const [QUrl](#) &url)
- void **signalTrashCountersMap** (const [QMap](#)< [QString](#), int > &counterMap)

## Public Member Functions

- **IOJobsThread** ([QObject](#) \*const parent)
- void **buildTrashCounters** ()  
*creates a job for count trash items from all collections*
- void **copyOrMove** ([IOJobData](#) \*const data)  
*Starts a number of jobs to copy or move source files to destination.*
- void **deleteFiles** ([IOJobData](#) \*const data)  
*Starts a number of jobs to delete multiple files.*
- void **emptyDTrashItems** ([IOJobData](#) \*const data)  
*creates a job for every item to delete from collection trash*
- [QStringList](#) & **errorsList** () const
- bool **hasErrors** () const  
*hasErrors*
- bool **isCanceled** () const  
*isCanceled*
- [IOJobData](#) \* **jobData** () const
- void **listDTrashItems** (const [QString](#) &collectionPath)  
*Starts a job for listing trash items in a collection.*
- void **renameFile** ([IOJobData](#) \*const data)  
*Starts one job to rename a file to a new name.*
- void **restoreDTrashItems** ([IOJobData](#) \*const data)  
*creates a job for every item to restore back to album*

## Public Member Functions inherited from [Digikam::ActionThreadBase](#)

- **ActionThreadBase** ([QObject](#) \*const parent=nullptr)
- void **cancel** (bool isCancel=true)  
*Cancel processing of current jobs under progress.*
- int **maximumNumberOfThreads** () const  
*Return the maximum number of threads used to parallelize collection of job processing.*
- void **setDefaultMaximumNumberOfThreads** ()  
*Reset maximum number of threads used to parallelize collection of job processing to max core detected on computer.*
- void **setMaximumNumberOfThreads** (int n)  
*Adjust maximum number of threads used to parallelize collection of job processing.*

## Additional Inherited Members

## Protected Slots inherited from [Digikam::ActionThreadBase](#)

- void **slotJobFinished** ()

## Protected Member Functions inherited from [Digikam::ActionThreadBase](#)

- void [appendJobs](#) (const [ActionJobCollection](#) &jobs)  
*Append a collection of jobs to process into QThreadPool.*
- bool [isEmpty](#) () const  
*Return true if list of pending jobs to process is empty.*
- int [pendingCount](#) () const  
*Return the number of pending jobs to process.*
- void [run](#) () override  
*Main thread loop used to process jobs in todo list.*

### 9.786.1 Member Function Documentation

#### 9.786.1.1 [copyOrMove\(\)](#)

```
void Digikam::IOJobsThread::copyOrMove (
    IOJobData *const data )
```

##### Parameters

<i>data</i>	IOJobsData container
-------------	----------------------

#### 9.786.1.2 [deleteFiles\(\)](#)

```
void Digikam::IOJobsThread::deleteFiles (
    IOJobData *const data )
```

##### Parameters

<i>data</i>	IOJobsData container
-------------	----------------------

#### 9.786.1.3 [emptyDTrashItems\(\)](#)

```
void Digikam::IOJobsThread::emptyDTrashItems (
    IOJobData *const data )
```

##### Parameters

<i>data</i>	IOJobsData container
-------------	----------------------

#### 9.786.1.4 [errorsList\(\)](#)

```
QStringList & Digikam::IOJobsThread::errorsList ( ) const
```



**Returns**

the current errors list

**9.786.1.5 hasErrors()**

```
bool Digikam::IOJobsThread::hasErrors ( ) const
```

**Returns**

true if string list was not empty

**9.786.1.6 isCanceled()**

```
bool Digikam::IOJobsThread::isCanceled ( ) const
```

**Returns**

true if the thread was interrupted

**9.786.1.7 jobData()**

```
IOJobData * Digikam::IOJobsThread::jobData ( ) const
```

**Returns**

the current data job instance

**9.786.1.8 listDTrashItems()**

```
void Digikam::IOJobsThread::listDTrashItems (
    const QString & collectionPath )
```

**Parameters**

<i>collectionPath</i>	
-----------------------	--

**9.786.1.9 renameFile()**

```
void Digikam::IOJobsThread::renameFile (
    IOJobData *const data )
```

**Parameters**

<i>data</i>	IOJobsData container
-------------	----------------------

### 9.786.1.10 restoreDTrashItems()

```
void Digikam::IOJobsThread::restoreDTrashItems (
    IOJobData *const data )
```

#### Parameters

<i>data</i>	IOJobsData container
-------------	----------------------

## 9.787 Digikam::IptcCoreContactInfo Class Reference

### Public Member Functions

- bool **isEmpty** () const
- bool **isNull** () const
- void **merge** (const [IptcCoreContactInfo](#) &t)
- bool **operator==** (const [IptcCoreContactInfo](#) &t) const

### Public Attributes

- QString **address**
- QString **city**
- QString **country**
- QString **email**
- QString **phone**
- QString **postalCode**
- QString **provinceState**
- QString **webUrl**

## 9.788 Digikam::IptcCoreLocationInfo Class Reference

### Public Member Functions

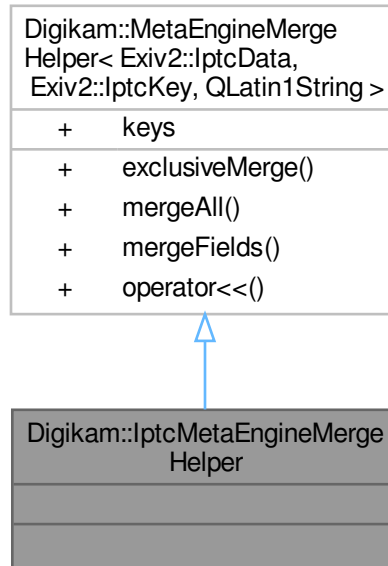
- bool **isEmpty** () const
- bool **isNull** () const
- void **merge** (const [IptcCoreLocationInfo](#) &t)
- bool **operator==** (const [IptcCoreLocationInfo](#) &t) const

### Public Attributes

- QString **city**
- QString **country**
- QString **countryCode**
- QString **location**
- QString **provinceState**

## 9.789 Digikam::IptcMetaEngineMergeHelper Class Reference

Inheritance diagram for Digikam::IptcMetaEngineMergeHelper:



### Additional Inherited Members

#### Public Member Functions inherited from

[Digikam::MetaEngineMergeHelper< Exiv2::IptcData, Exiv2::IptcKey, QLatin1String >](#)

- void [exclusiveMerge](#) (const Exiv2::IptcData &src, Exiv2::IptcData &dest)  
*Merge two (Exif,IPTC,Xmp) Data packages, the result is stored in dest.*
- void [mergeAll](#) (const Exiv2::IptcData &src, Exiv2::IptcData &dest)  
*Merge two (Exif,IPTC,Xmp) Data packages, where the result is stored in dest and fields from src take precedence over existing data from dest.*
- void [mergeFields](#) (const Exiv2::IptcData &src, Exiv2::IptcData &dest)  
*Merge two (Exif,IPTC,Xmp) Data packages, the result is stored in dest.*
- [MetaEngineMergeHelper](#) & [operator<<](#) (const QLatin1String &key)

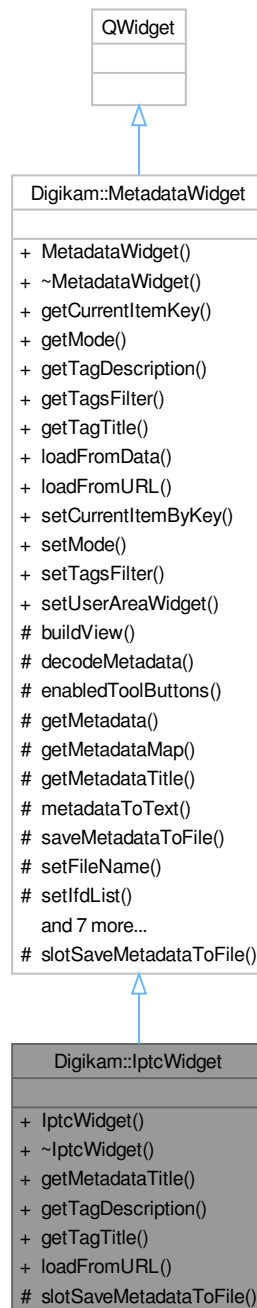
#### Public Attributes inherited from

[Digikam::MetaEngineMergeHelper< Exiv2::IptcData, Exiv2::IptcKey, QLatin1String >](#)

- QList< QLatin1String > [keys](#)

## 9.790 Digikam::IptcWidget Class Reference

Inheritance diagram for Digikam::IptcWidget:



### Public Member Functions

- **IptcWidget** (`QWidget *const parent, const QString &name=QString()`)
- `QString` [getMetadataTitle](#) () const override

- QString [getTagDescription](#) (const QString &key) override
- QString [getTagTitle](#) (const QString &key) override
- bool [loadFromURL](#) (const QUrl &url) override

### Public Member Functions inherited from [Digikam::MetadataWidget](#)

- **MetadataWidget** (QWidget \*const parent, const QString &name=QString())
- QString [getCurrentItemKey](#) () const
- int [getMode](#) () const
- QStringList [getTagsFilter](#) () const
- virtual bool [loadFromData](#) (const QString &fileName, const [DMetadata](#) &data=[DMetadata](#)())
- void [setCurrentItemByKey](#) (const QString &itemKey)
- void [setMode](#) (int mode)
- void [setTagsFilter](#) (const QStringList &list)
- void [setUserAreaWidget](#) (QWidget \*const w)

### Protected Slots

- void [slotSaveMetadataToFile](#) () override

### Protected Slots inherited from [Digikam::MetadataWidget](#)

- virtual void [slotSaveMetadataToFile](#) ()=0

### Additional Inherited Members

### Public Types inherited from [Digikam::MetadataWidget](#)

- enum [TagFilters](#) { NONE = 0 , PHOTO , CUSTOM }

### Signals inherited from [Digikam::MetadataWidget](#)

- void [signalSetupMetadataFilters](#) ()

### Protected Member Functions inherited from [Digikam::MetadataWidget](#)

- void [enabledToolButtons](#) (bool)
- [DMetadata](#) \* [getMetadata](#) () const
- const [DMetadata::MetaDatum](#) & [getMetadataMap](#) ()
- QString [metadataToText](#) () const
- QUrl [saveMetadataToFile](#) (const QString &caption, const QString &fileFilter)
- void [setFileName](#) (const QString &fileName)
- void [setIfdList](#) (const [DMetadata::MetaDatum](#) &ifds, const QStringList &keysFilter, const QStringList &tagsFilter)
- void [setIfdList](#) (const [DMetadata::MetaDatum](#) &ifds, const QStringList &tagsFilter=QStringList())
- bool [setMetadata](#) (const [DMetadata](#) &data=[DMetadata](#)())
- virtual void [setMetadataEmpty](#) ()
- void [setMetadataMap](#) (const [DMetadata::MetaDatum](#) &data=[DMetadata::MetaDatum](#)())
- void [setup](#) ()
  - *Call this method in children class constructors to init signal/slots connections.*
- bool [storeMetadataToFile](#) (const QUrl &url, const QByteArray &metaData)
- [MetadataListView](#) \* [view](#) () const

## 9.790.1 Member Function Documentation

### 9.790.1.1 `getMetadataTitle()`

```
QString Digikam::IptcWidget::getMetadataTitle ( ) const [override], [virtual]
```

Implements [Digikam::MetadataWidget](#).

### 9.790.1.2 `getTagDescription()`

```
QString Digikam::IptcWidget::getTagDescription (
    const QString & key ) [override], [virtual]
```

Reimplemented from [Digikam::MetadataWidget](#).

### 9.790.1.3 `getTagTitle()`

```
QString Digikam::IptcWidget::getTagTitle (
    const QString & key ) [override], [virtual]
```

Reimplemented from [Digikam::MetadataWidget](#).

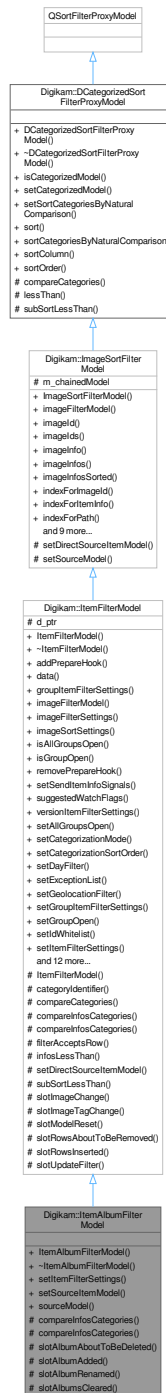
### 9.790.1.4 `loadFromURL()`

```
bool Digikam::IptcWidget::loadFromURL (
    const QUrl & url ) [override], [virtual]
```

Implements [Digikam::MetadataWidget](#).

## 9.791 Digikam::ItemAlbumFilterModel Class Reference

Inheritance diagram for Digikam::ItemAlbumFilterModel:



### Public Member Functions

- **ItemAlbumFilterModel** (QObject \*const parent=nullptr)
- void **setItemFilterSettings** (const **ItemFilterSettings** &settings) override

*Changes the current image filter settings and refilters.*

- void **setSourceItemModel** ([ItemAlbumModel](#) \*model)
- [ItemAlbumModel](#) \* **sourceModel** () const

## Public Member Functions inherited from [Digikam::ItemFilterModel](#)

- **ItemFilterModel** (QObject \*const parent=nullptr)
- void **addPrepareHook** ([ItemFilterModelPrepareHook](#) \*const hook)
 

*Add a hook to get added images for preparation tasks before they are added in the model.*
- QVariant **data** (const QModelIndex &index, int role=Qt::DisplayRole) const override
- [GroupItemFilterSettings](#) **groupItemFilterSettings** () const
- [ItemFilterModel](#) \* **imageFilterModel** () const override
 

*Returns this, any chained [ItemFilterModel](#), or 0.*
- [ItemFilterSettings](#) **imageFilterSettings** () const
- [ItemSortSettings](#) **imageSortSettings** () const
- bool **isAllGroupsOpen** () const
- bool **isGroupOpen** (qulonglong group) const
 

*group is identified by the id of its group leader*
- void **removePrepareHook** ([ItemFilterModelPrepareHook](#) \*const hook)
- void **setSendItemInfoSignals** (bool sendSignals)
 

*Enables sending imageInfosAdded and imageInfosAboutToBeRemoved.*
- [DatabaseFields::Set](#) **suggestedWatchFlags** () const
 

*Returns a set of DatabaseFields suggested to set as watch flags on the source [ItemModel](#).*
- [VersionItemFilterSettings](#) **versionItemFilterSettings** () const

## Public Member Functions inherited from [Digikam::ImageSortFilterModel](#)

- **ImageSortFilterModel** (QObject \*const parent=nullptr)
- qulonglong **imageId** (const QModelIndex &index) const
- QList< qulonglong > **imageIds** (const QList< QModelIndex > &indexes) const
- [ItemInfo](#) **imageInfo** (const QModelIndex &index) const
- QList< [ItemInfo](#) > **imageInfos** (const QList< QModelIndex > &indexes) const
- QList< [ItemInfo](#) > **imageInfosSorted** () const
 

*Returns a list of all image infos, sorted according to this model.*
- QModelIndex **indexForImageId** (qulonglong id) const
- QModelIndex **indexForItemInfo** (const [ItemInfo](#) &info) const
- QModelIndex **indexForPath** (const QString &filePath) const
- QModelIndex **mapFromDirectSourceToSourceItemModel** (const QModelIndex &sourceModel\_index) const
- QModelIndex **mapFromSourceItemModel** (const QModelIndex &imagemodel\_index) const
- QList< QModelIndex > **mapListFromSource** (const QList< QModelIndex > &sourceIndexes) const
- QList< QModelIndex > **mapListToSource** (const QList< QModelIndex > &indexes) const
 

*Convenience methods mapped to [ItemModel](#).*
- QModelIndex **mapToSourceItemModel** (const QModelIndex &index) const
- void **setSourceFilterModel** ([ImageSortFilterModel](#) \*const model)
- void **setSourceItemModel** ([ItemModel](#) \*const model)
- [ImageSortFilterModel](#) \* **sourceFilterModel** () const
- [ItemModel](#) \* **sourceItemModel** () const



## Public Member Functions inherited from Digikam::DCategorizedSortFilterProxyModel

- **DCategorizedSortFilterProxyModel** (QObject \*const parent=nullptr)
- bool **isCategorizedModel** () const
- void **setCategorizedModel** (bool categorizedModel)  
*Enables or disables the categorization feature.*
- void **setSortCategoriesByNaturalComparison** (bool sortCategoriesByNaturalComparison)  
*Set if the sorting using CategorySortRole will use a natural comparison in the case that strings were returned.*
- void **sort** (int column, Qt::SortOrder order=Qt::AscendingOrder) override  
*Overridden from QSortFilterProxyModel.*
- bool **sortCategoriesByNaturalComparison** () const
- int **sortColumn** () const
- Qt::SortOrder **sortOrder** () const

## Protected Slots

- void **slotAlbumAboutToBeDeleted** (Album \*album)
- void **slotAlbumAdded** (Album \*album)
- void **slotAlbumRenamed** (Album \*album)
- void **slotAlbumsCleared** ()

## Protected Slots inherited from Digikam::ItemFilterModel

- void **slotImageChange** (const ImageChangeset &changeset)
- void **slotImageTagChange** (const ImageTagChangeset &changeset)
- void **slotModelReset** ()
- void **slotRowsAboutToBeRemoved** (const QModelIndex &parent, int start, int end)
- void **slotRowsInserted** (const QModelIndex &parent, int start, int end)
- void **slotUpdateFilter** ()

## Protected Member Functions

- int **compareInfosCategories** (const ItemInfo &left, const ItemInfo &right) const override  
*Reimplement to customize category sorting. Return negative if category of left < category right, Return 0 if left and right are in the same category, else return positive.*
- int **compareInfosCategories** (const ItemInfo &left, const ItemInfo &right, const FaceTagsIface &leftFace, const FaceTagsIface &rightFace) const override  
*In order to be able to Categorize by Faces, it's necessary to pass in the face as well.*

## Protected Member Functions inherited from Digikam::ItemFilterModel

- **ItemFilterModel** (ItemFilterModelPrivate &dd, QObject \*const parent)
- virtual QString **categoryIdentifier** (const ItemInfo &info, const FaceTagsIface &face) const  
*Returns a unique identifier for the category if info.*
- int **compareCategories** (const QModelIndex &left, const QModelIndex &right) const override  
*This method compares the category of the left index with the category of the right index.*
- bool **filterAcceptsRow** (int source\_row, const QModelIndex &source\_parent) const override
- virtual bool **infosLessThan** (const ItemInfo &left, const ItemInfo &right) const  
*Reimplement to customize sorting.*
- void **setDirectSourceItemModel** (ItemModel \*const model) override  
*Reimplement if needed.*
- bool **subSortLessThan** (const QModelIndex &left, const QModelIndex &right) const override  
*This method has a similar purpose as lessThan() has on QSortFilterProxyModel.*

## Protected Member Functions inherited from [Digikam::ImageSortFilterModel](#)

- void [setSourceModel](#) (QAbstractItemModel \*const model) override

## Protected Member Functions inherited from [Digikam::DCategorizedSortFilterProxyModel](#)

- bool [lessThan](#) (const QModelIndex &left, const QModelIndex &right) const override  
*Overridden from [QSortFilterProxyModel](#).*

## Additional Inherited Members

## Public Types inherited from [Digikam::ItemFilterModel](#)

- enum [ItemFilterModelRoles](#) {  
[CategorizationModeRole](#) = ItemModel::FilterModelRoles + 1 , [SortOrderRole](#) = ItemModel::FilterModelRoles + 2 , [CategoryAlbumIdRole](#) = ItemModel::FilterModelRoles + 3 , [CategoryFormatRole](#) = ItemModel::FilterModelRoles + 4 ,  
[CategoryDateRole](#) = ItemModel::FilterModelRoles + 5 , [CategoryFaceRole](#) = ItemModel::FilterModelRoles + 6 , [GroupsOpenRole](#) = ItemModel::FilterModelRoles + 7 , [ItemFilterModelPointerRole](#) = ItemModel::FilterModelRoles + 50 }

## Public Types inherited from [Digikam::DCategorizedSortFilterProxyModel](#)

- enum [AdditionalRoles](#) { [CategoryDisplayRole](#) = 0x17CE990A , [CategorySortRole](#) = 0x27857E60 }

## Public Slots inherited from [Digikam::ItemFilterModel](#)

- void [setAllGroupsOpen](#) (bool open)
- void [setCategorizationMode](#) ([ItemSortSettings::CategorizationMode](#) mode)
- void [setCategorizationSortOrder](#) ([ItemSortSettings::SortOrder](#) order)
- void [setDayFilter](#) (const QList< QDateTime > &days)  
*Adjust the current [ItemFilterSettings](#).*
- void [setExceptionList](#) (const QList< qlonglong > &idlist, const QString &id)
- void [setGeolocationFilter](#) (const [ItemFilterSettings::GeolocationCondition](#) &condition)
- void [setGroupItemFilterSettings](#) (const [GroupItemFilterSettings](#) &settings)  
*Changes the current version image filter settings and refilters.*
- void [setGroupOpen](#) (qlonglong group, bool open)
- void [setIdWhitelist](#) (const QList< qlonglong > &idList, const QString &id)
- virtual void [setItemSortSettings](#) (const [ItemSortSettings](#) &settings)  
*Changes the current image sort settings and resorts.*
- void [setMimeTypeFilter](#) (int mimeTypeFilter)
- void [setRatingFilter](#) (int rating, [ItemFilterSettings::RatingCondition](#) ratingCond, bool isUnratedExcluded)
- void [setSortOrder](#) ([ItemSortSettings::SortOrder](#) order)
- void [setSortRole](#) ([ItemSortSettings::SortRole](#) role)
- void [setStringTypeNatural](#) (bool natural)
- void [setTagFilter](#) (const QList< int > &includedTags, const QList< int > &excludedTags, [ItemFilterSettings::MatchingCondition](#) matchingCond, bool showUnTagged, const QList< int > &clTagIds, const QList< int > &plTagIds)
- void [setTextFilter](#) (const [SearchTextFilterSettings](#) &settings)
- void [setUriWhitelist](#) (const QList< QUrl > &urlList, const QString &id)
- void [setVersionItemFilterSettings](#) (const [VersionItemFilterSettings](#) &settings)  
*Changes the current version image filter settings and refilters.*
- void [setVersionManagerSettings](#) (const [VersionManagerSettings](#) &settings)
- void [toggleGroupOpen](#) (qlonglong group)

## Signals inherited from [Digikam::ItemFilterModel](#)

- void **filterMatches** (bool matches)  
*Signals that the set filter matches at least one index.*
- void **filterMatchesForText** (bool matchesByText)  
*Signals that the set text filter matches at least one entry.*
- void **filterSettingsChanged** (const [ItemFilterSettings](#) &settings)  
*Emitted when the filter settings have been changed (the model may not yet have been updated)*
- void **imageInfosAboutToBeRemoved** (const QList< [ItemInfo](#) > &infos)
- void **imageInfosAdded** (const QList< [ItemInfo](#) > &infos)  
*These signals need to be explicitly enabled with [setSendItemInfoSignals\(\)](#)*

## Protected Attributes inherited from [Digikam::ItemFilterModel](#)

- [ItemFilterModelPrivate](#) \*const **d\_ptr** = nullptr

## Protected Attributes inherited from [Digikam::ImageSortFilterModel](#)

- [ImageSortFilterModel](#) \* **m\_chainedModel** = nullptr

## 9.791.1 Member Function Documentation

### 9.791.1.1 [compareInfosCategories\(\)](#) [1/2]

```
int Digikam::ItemAlbumFilterModel::compareInfosCategories (
    const ItemInfo & left,
    const ItemInfo & right ) const [override], [protected], [virtual]
```

Reimplemented from [Digikam::ItemFilterModel](#).

### 9.791.1.2 [compareInfosCategories\(\)](#) [2/2]

```
int Digikam::ItemAlbumFilterModel::compareInfosCategories (
    const ItemInfo & left,
    const ItemInfo & right,
    const FaceTagsIface & leftFace,
    const FaceTagsIface & rightFace ) const [override], [protected], [virtual]
```

One image may have multiple Faces in it, hence just the [ItemInfo](#) isn't sufficient.

Reimplemented from [Digikam::ItemFilterModel](#).

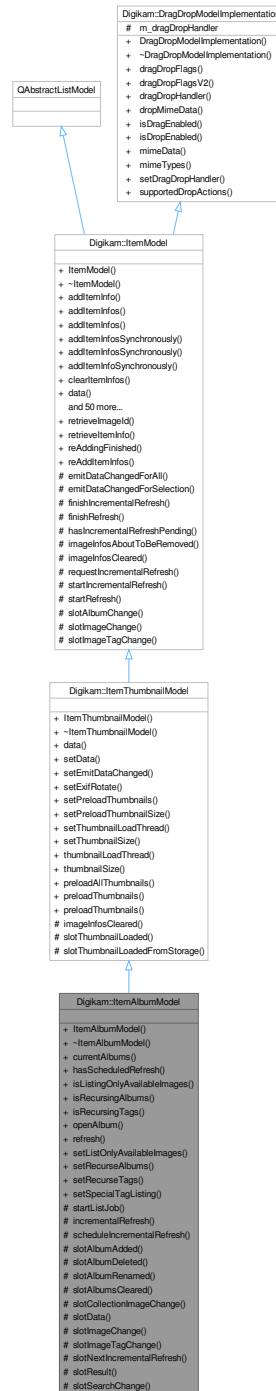
### 9.791.1.3 [setItemFilterSettings\(\)](#)

```
void Digikam::ItemAlbumFilterModel::setItemFilterSettings (
    const ItemFilterSettings & settings ) [override], [virtual]
```

Reimplemented from [Digikam::ItemFilterModel](#).

## 9.792 Digikam::ItemAlbumModel Class Reference

Inheritance diagram for Digikam::ItemAlbumModel:



### Public Slots

- void `openAlbum` (const QList< Album \* > &albums)

Call this method to populate the model with data from the given album.

- void **refresh** ()  
*Reloads the current album.*
- void **setListOnlyAvailableImages** (bool onlyAvailable)
- void **setRecurseAlbums** (bool recursiveListing)
- void **setRecurseTags** (bool recursiveListing)
- void **setSpecialTagListing** (const QString &specialListing)

### Public Slots inherited from [Digikam::ItemThumbnailModel](#)

- void **preloadAllThumbnails** ()
- void **preloadThumbnails** (const QList< [ItemInfo](#) > &)  
*Preload thumbnail for the given infos resp.*
- void **preloadThumbnails** (const QList< QModelIndex > &)

### Public Slots inherited from [Digikam::ItemModel](#)

- void **reAddingFinished** ()
- void **reAddItemInfos** (const QList< [ItemInfo](#) > &infos, const QList< QVariant > &extraValues)

### Public Member Functions

- **ItemAlbumModel** (QWidget \*const parent)
- QList< [Album](#) \* > **currentAlbums** () const
- bool **hasScheduledRefresh** () const
- bool **isListingOnlyAvailableImages** () const
- bool **isRecurringAlbums** () const
- bool **isRecurringTags** () const

### Public Member Functions inherited from [Digikam::ItemThumbnailModel](#)

- **ItemThumbnailModel** (QWidget \*const parent)  
*An [ItemModel](#) that supports thumbnail loading.*
- QVariant **data** (const QModelIndex &index, int role=Qt::DisplayRole) const override  
*Handles the ThumbnailRole.*
- bool **setData** (const QModelIndex &index, const QVariant &value, int role=Qt::DisplayRole) override  
*You can override the current thumbnail size by giving an integer value for ThumbnailRole.*
- void **setEmitDataChanged** (bool emitSignal)  
*Enable emitting dataChanged() when a thumbnail becomes available.*
- void **setExifRotate** (bool rotate)
- void **setPreloadThumbnails** (bool preload)  
*Enable preloading of thumbnails: If preloading is enabled, for every entry in the model a thumbnail generation is started.*
- void **setPreloadThumbnailSize** (const [ThumbnailSize](#) &thumbSize)  
*If you want to fix a size for preloading, do it here.*
- void **setThumbnailLoadThread** ([ThumbnailLoadThread](#) \*const thread)  
*Enable thumbnail loading and set the thread that shall be used.*
- void **setThumbnailSize** (const [ThumbnailSize](#) &thumbSize)  
*Set the thumbnail size to use.*
- [ThumbnailLoadThread](#) \* **thumbnailLoadThread** () const
- [ThumbnailSize](#) **thumbnailSize** () const

## Public Member Functions inherited from [Digikam::ItemModel](#)

- **ItemModel** (QObject \*const parent=nullptr)
- void **addItemInfo** (const [ItemInfo](#) &info)
  - Main entry point for subclasses adding image infos to the model.*
- void **addItemInfos** (const QList< [ItemInfo](#) > &infos)
- void **addItemInfos** (const QList< [ItemInfo](#) > &infos, const QList< QVariant > &extraValues)
- void **addItemInfosSynchronously** (const QList< [ItemInfo](#) > &infos)
- void **addItemInfosSynchronously** (const QList< [ItemInfo](#) > &infos, const QList< QVariant > &extraValues)
- void **addItemInfoSynchronously** (const [ItemInfo](#) &info)
  - addItemInfo() is asynchronous if a preprocessor is set.*
- void **clearItemInfos** ()
  - Clears image infos and resets model.*
- QVariant **data** (const QModelIndex &index, int role=Qt::DisplayRole) const override
- void **ensureHasGroupedImages** (const [ItemInfo](#) &groupLeader)
  - Ensure that all images grouped on the given leader are contained in the model.*
- void **ensureHasItemInfo** (const [ItemInfo](#) &info)
  - Add the given entries.*
- void **ensureHasItemInfos** (const QList< [ItemInfo](#) > &infos)
- void **ensureHasItemInfos** (const QList< [ItemInfo](#) > &infos, const QList< QVariant > &extraValues)
- Qt::ItemFlags **flags** (const QModelIndex &index) const override
- bool **hasImage** (const [ItemInfo](#) &info) const
- bool **hasImage** (const [ItemInfo](#) &info, const QVariant &extraValue) const
- bool **hasImage** (qulonglong id) const
- bool **hasImage** (qulonglong id, const QVariant &extraValue) const
- QVariant **headerData** (int section, Qt::Orientation orientation, int role=Qt::DisplayRole) const override
- qulonglong **imageId** (const QModelIndex &index) const
- qulonglong **imageId** (int row) const
- QList< qulonglong > **imageIds** () const
- QList< qulonglong > **imageIds** (const QList< QModelIndex > &indexes) const
- [ItemInfo](#) **imageInfo** (const QModelIndex &index) const
  - Returns the ItemInfo object, reference or image id from the underlying data pointed to by the index.*
- [ItemInfo](#) **imageInfo** (const QString &filePath) const
- [ItemInfo](#) **imageInfo** (int row) const
  - Returns the ItemInfo object, reference or image id from the underlying data of the given row (parent is the invalid QModelIndex, column is 0).*
- [ItemInfo](#) & **imageInfoRef** (const QModelIndex &index) const
- [ItemInfo](#) & **imageInfoRef** (int row) const
- QList< [ItemInfo](#) > **imageInfos** () const
- QList< [ItemInfo](#) > **imageInfos** (const QList< QModelIndex > &indexes) const
- QList< [ItemInfo](#) > **imageInfos** (const QString &filePath) const
- QModelIndex **index** (int row, int column=0, const QModelIndex &parent=QModelIndex()) const override
- QList< QModelIndex > **indexesForImageId** (qulonglong id) const
- QList< QModelIndex > **indexesForItemInfo** (const [ItemInfo](#) &info) const
- QList< QModelIndex > **indexesForPath** (const QString &filePath) const
- QModelIndex **indexForImageId** (qulonglong id) const
- QModelIndex **indexForImageId** (qulonglong id, const QVariant &extraValue) const
- QModelIndex **indexForItemInfo** (const [ItemInfo](#) &info) const
  - Return the index for the given ItemInfo or id, if contained in this model.*
- QModelIndex **indexForItemInfo** (const [ItemInfo](#) &info, const QVariant &extraValue) const
- QModelIndex **indexForPath** (const QString &filePath) const
  - Returns the index or ItemInfo object from the underlying data for the given file path.*
- bool **isEmpty** () const

- bool **isRefreshing** () const  
*Returns true if this model is currently refreshing.*
- int **itemCount** () const
- bool **keepsFilePathCache** () const
- int **numberOfIndexesForImageId** (qulonglong id) const
- int **numberOfIndexesForItemInfo** (const [ItemInfo](#) &info) const
- void **removeIndex** (const QModelIndex &indexes)  
*Directly remove the given indexes or infos from the model.*
- void **removeIndexes** (const QList< QModelIndex > &indexes)
- void **removeItemInfo** (const [ItemInfo](#) &info)
- void **removeItemInfos** (const QList< [ItemInfo](#) > &infos)
- void **removeItemInfos** (const QList< [ItemInfo](#) > &infos, const QList< QVariant > &extraValues)
- int **rowCount** (const QModelIndex &parent=QModelIndex()) const override
- void **setItemInfos** (const QList< [ItemInfo](#) > &infos)  
*Clears and adds the infos.*
- void **setKeepsFilePathCache** (bool keepCache)  
*If a cache is kept, lookup by file path is fast, without a cache it is O(n).*
- DECLARE\_MODEL\_DRAG\_DROP\_METHODS void **setPreprocessor** (QObject \*const processor)  
*Install an object as a preprocessor for ItemInfos added to this model.*
- void **setSendRemovalSignals** (bool send)  
*Enable sending of imageInfosAboutToBeRemoved and imageInfosRemoved signals.*
- void **setWatchFlags** (const [DatabaseFields::Set](#) &set)  
*Set a set of database fields to watch.*
- QList< [ItemInfo](#) > **uniqueItemInfos** () const
- void **unsetPreprocessor** (QObject \*const processor)

## Public Member Functions inherited from [Digikam::DragDropModelImplementation](#)

- [DragDropModelImplementation](#) ()=default  
*A class providing a sample implementation for a QAbstractItemModel redirecting drag-and-drop support to a handler.*
- virtual Qt::ItemFlags **dragDropFlags** (const QModelIndex &index) const  
*Call from your flags() method, adding the relevant drag drop flags.*
- Qt::ItemFlags **dragDropFlagsV2** (const QModelIndex &index) const  
*This is an alternative approach to dragDropFlags().*
- [AbstractItemDragDropHandler](#) \* **dragDropHandler** () const
- bool **dropMimeData** (const QMimeData \*, Qt::DropAction, int, int, const QModelIndex &)
- virtual bool **isDragEnabled** (const QModelIndex &index) const
- virtual bool **isDropEnabled** (const QModelIndex &index) const
- QMimeData \* **mimeData** (const QModelIndexList &indexes) const
- QStringList **mimeTypes** () const
- void **setDragDropHandler** ([AbstractItemDragDropHandler](#) \*handler)  
*Set a drag drop handler.*
- Qt::DropActions **supportedDropActions** () const  
*Implements the relevant QAbstractItemModel methods for drag and drop.*

### Protected Slots

- void **incrementalRefresh** ()
- void **scheduleIncrementalRefresh** ()
- void **slotAlbumAdded** ([Album](#) \*album)
- void **slotAlbumDeleted** ([Album](#) \*album)
- void **slotAlbumRenamed** ([Album](#) \*album)
- void **slotAlbumsCleared** ()
- void **slotCollectionImageChange** (const [CollectionImageChangeset](#) &changeset)
- void **slotData** (const QList< [ItemLISTERRecord](#) > &records)
- void **slotImageChange** (const [ImageChangeset](#) &changeset) override
- void **slotImageTagChange** (const [ImageTagChangeset](#) &changeset) override
- void **slotNextIncrementalRefresh** ()
- void **slotResult** ()
- void **slotSearchChange** (const [SearchChangeset](#) &changeset)

### Protected Slots inherited from [Digikam::ItemThumbnailModel](#)

- void **slotThumbnailLoaded** (const [LoadingDescription](#) &loadingDescription, const QPixmap &thumb)
- void **slotThumbnailLoadedFromStorage** (const [LoadingDescription](#) &loadingDescription, const QPixmap &thumb)

### Protected Slots inherited from [Digikam::ItemModel](#)

- virtual void **slotAlbumChange** (const [AlbumChangeset](#) &changeset)
- virtual void **slotImageChange** (const [ImageChangeset](#) &changeset)
- virtual void **slotImageTagChange** (const [ImageTagChangeset](#) &changeset)

### Protected Member Functions

- void **startListJob** (const QList< [Album](#) \* > &albums)

### Protected Member Functions inherited from [Digikam::ItemThumbnailModel](#)

- void **imageInfosCleared** () override  
*Called when the internal storage is cleared.*

### Protected Member Functions inherited from [Digikam::ItemModel](#)

- void **emitDataChangedForAll** ()
- void **emitDataChangedForSelection** (const QItemSelection &selection)
- void **finishIncrementalRefresh** ()
- void **finishRefresh** ()
- bool **hasIncrementalRefreshPending** () const
- virtual void **imageInfosAboutToBeRemoved** (int, int)  
*Called before rowsAboutToBeRemoved.*
- void **requestIncrementalRefresh** ()  
*As soon as the model is ready to start an incremental refresh, the signal [readyForIncrementalRefresh\(\)](#) will be emitted.*
- void **startIncrementalRefresh** ()  
*Starts an incremental refresh operation.*
- void **startRefresh** ()  
*Subclasses that add ItemInfos in batches shall call [startRefresh\(\)](#) when they start sending batches and [finishRefresh\(\)](#) when they have finished.*



## Additional Inherited Members

### Public Types inherited from Digikam::ItemModel

- enum [ItemModelRoles](#) {  
[ItemModelPointerRole](#) = Qt::UserRole , [ItemModelInternalId](#) = Qt::UserRole + 1 , [ThumbnailRole](#) = Qt::UserRole + 2 , [CreationDateRole](#) = Qt::UserRole + 3 ,  
[ExtraDataRole](#) = Qt::UserRole + 5 , [ExtraDataDuplicateCount](#) = Qt::UserRole + 6 , [LTLeftPanelRole](#) = Qt::UserRole + 50 , [LTRightPanelRole](#) = Qt::UserRole + 51 ,  
[SubclassRoles](#) = Qt::UserRole + 100 , [FilterModelRoles](#) = Qt::UserRole + 500 }

### Signals inherited from Digikam::ItemThumbnailModel

- void **thumbnailAvailable** (const QModelIndex &index, int requestedSize)
- void **thumbnailFailed** (const QModelIndex &index, int requestedSize)

### Signals inherited from Digikam::ItemModel

- void [allRefreshingFinished](#) ()  
*Signals that the model has finished currently with all scheduled refreshing, full or incremental, and all preprocessing.*
- void **imageChange** (const [ImageChangeset](#) &, const QItemSelection &)  
*If an [ImageChangeset](#) affected indexes of this model with changes as set in [watchFlags\(\)](#), this signal contains the changeset and the affected indexes.*
- void [imageInfosAboutToBeAdded](#) (const QList< [ItemInfo](#) > &infos)  
*Informs that ItemInfos will be added to the model.*
- void [imageInfosAboutToBeRemoved](#) (const QList< [ItemInfo](#) > &infos)  
*Informs that ItemInfos will be removed from the model.*
- void [imageInfosAdded](#) (const QList< [ItemInfo](#) > &infos)  
*Informs that ItemInfos have been added to the model.*
- void [imageInfosRemoved](#) (const QList< [ItemInfo](#) > &infos)  
*Informs that ItemInfos have been removed from the model.*
- void **imageTagChange** (const [ImageTagChangeset](#) &, const QItemSelection &)  
*If an [ImageTagChangeset](#) affected indexes of this model, this signal contains the changeset and the affected indexes.*
- void **preprocess** (const QList< [ItemInfo](#) > &infos, const QList< QVariant > &)  
*Connect to this signal only if you are the current preprocessor.*
- void **processAdded** (const QList< [ItemInfo](#) > &infos, const QList< QVariant > &)
- void [readyForIncrementalRefresh](#) ()  
*Signals that the model is right now ready to start an incremental refresh.*

### Static Public Member Functions inherited from Digikam::ItemModel

- static qlonglong **retrieveImageId** (const QModelIndex &index)
- static [ItemInfo](#) **retrieveItemInfo** (const QModelIndex &index)  
*Retrieves the imageInfo object from the data() method of the given index.*

### Protected Attributes inherited from Digikam::DragDropModelImplementation

- [AbstractItemDragDropHandler](#) \* **m\_dragDropHandler** = nullptr

## 9.792.1 Member Function Documentation

### 9.792.1.1 openAlbum

```
void Digikam::ItemAlbumModel::openAlbum (
    const QList< Album * > & albums ) [slot]
```

If called with 0, the model will be empty. Opening the same album again is a no-op. Extra safety, ensure that no null pointers are added

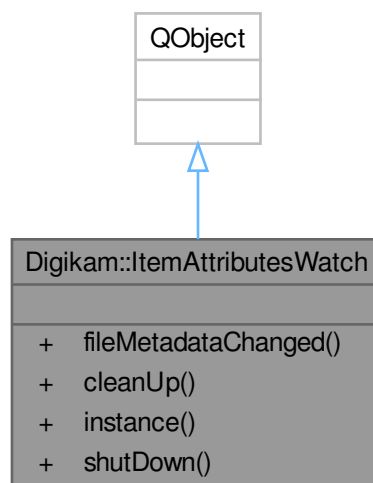
### 9.792.1.2 slotImageChange

```
void Digikam::ItemAlbumModel::slotImageChange (
    const ImageChangeset & changeset ) [override], [protected], [slot]
```

QList is designed for multiple selection, for now, only tags are supported for [SAlbum](#) it will be a list with one element

## 9.793 Digikam::ItemAttributesWatch Class Reference

Inheritance diagram for Digikam::ItemAttributesWatch:



### Signals

- void [signalFileMetadataChanged](#) (const QUrl &url)  
*Indicates that the metadata of the given file has been changed (a write operation on the file on disk).*
- void **signalImageCaptionChanged** (qulonglong imageId)
- void **signalImageDateChanged** (qulonglong imageId)
- void [signalImageRatingChanged](#) (qulonglong imageId)  
*These signals indicate that the rating, data or caption of the image with given imageId was set.*
- void [signalImagesChanged](#) (int albumId)  
*Indicates that images in the given album id may have changed their tags.*
- void [signalImageTagsChanged](#) (qulonglong imageId)  
*Indicates that tags have been assigned or removed for image with given imageId.*

## Public Member Functions

- void **fileMetadataChanged** (const QUrl &url)

## Static Public Member Functions

- static void **cleanUp** ()
- static [ItemAttributesWatch](#) \* **instance** ()
- static void **shutDown** ()

## 9.793.1 Member Function Documentation

### 9.793.1.1 signalFileMetadataChanged

```
void Digikam::ItemAttributesWatch::signalFileMetadataChanged (
    const QUrl & url ) [signal]
```

Usually, the database is updated accordingly, so then this signal is sent in combination with one or more of the above signals.

### 9.793.1.2 signalImageRatingChanged

```
void Digikam::ItemAttributesWatch::signalImageRatingChanged (
    qlonglong imageId ) [signal]
```

There is no guarantee that it actually changed.

### 9.793.1.3 signalImagesChanged

```
void Digikam::ItemAttributesWatch::signalImagesChanged (
    int albumId ) [signal]
```

This signal, the signal above, or both may be sent.

### 9.793.1.4 signalImageTagsChanged

```
void Digikam::ItemAttributesWatch::signalImageTagsChanged (
    qlonglong imageId ) [signal]
```

There is no guarantee that the tags were actually changed. This signal, the signal below, or both may be sent.

# 9.794 Digikam::ItemCategorizedView Class Reference

Inheritance diagram for Digikam::ItemCategorizedView:



### Public Slots

- void **hintAt** (const [ItemInfo](#) &info)

*Does something to gain attention for info, but not changing current selection.*

- void **openAlbum** (const QList< Album \* > &album)
- void **setCurrentInfo** (const ItemInfo &info)  
*Set as current item the item identified by the imageinfo.*
- void **setCurrentUrl** (const QUrl &url)  
*Set as current item the item identified by its file url.*
- void **setCurrentUrlWhenAvailable** (const QUrl &url)  
*Set as current item when it becomes available, the item identified by its file url.*
- void **setCurrentWhenAvailable** (qulonglong imageId)  
*Scroll the view to the given item when it becomes available.*
- void **setSelectedItemInfos** (const QList< ItemInfo > &infos)  
*Set selected items.*
- void **setSelectedUrls** (const QList< QUrl > &urlList)  
*Set selected items identified by their file urls.*
- void **setThumbnailSize** (int size)

### Public Slots inherited from Digikam::ItemViewCategorized

- void **copy** () override
- void **cut** () override
- void **hideIndexNotification** ()
- void **paste** () override
- void **showIndexNotification** (const QModelIndex &index, const QString &message)

### Public Slots inherited from Digikam::DCategorizedView

- void **reset** () override

### Signals

- void **currentChanged** (const ItemInfo &info)
- void **deselected** (const QList< ItemInfo > &nowDeselectedInfos)  
*Emitted when items are deselected. There may be other selected infos left. This signal is not emitted when the model is reset; then only selectionCleared is emitted.*
- void **imageActivated** (const ItemInfo &info)  
*Emitted when the given image is activated. Info is never null.*
- void **modelChanged** ()  
*Emitted when a new model is set.*
- void **selected** (const QList< ItemInfo > &newSelectedInfos)  
*Emitted when new items are selected. The parameter includes only the newly selected infos, there may be other already selected infos.*

### Signals inherited from Digikam::ItemViewCategorized

- void **clicked** (const QMouseEvent \*e, const QModelIndex &index)  
*For overlays: Like the respective parent class signals, but with additional info.*
- void **entered** (const QMouseEvent \*e, const QModelIndex &index)
- void **keyPressed** (QKeyEvent \*e)  
*Remember you may want to check if the event is accepted or ignored.*
- void **selectionChanged** ()  
*Emitted when any selection change occurs.*
- void **selectionCleared** ()  
*Emitted when the selection is completely cleared.*
- void **viewportClicked** (const QMouseEvent \*e)  
*While clicked() is emitted with a valid index, this corresponds to clicking on empty space.*
- void **zoomInStep** ()
- void **zoomOutStep** ()

## Public Member Functions

- **ItemCategorizedView** (QWidget \*const parent=nullptr)
- void **addOverlay** (ItemDelegateOverlay \*overlay, ItemDelegate \*delegate=nullptr)
 

*Add and remove an overlay. It will as well be removed automatically when destroyed. Unless you pass a different delegate, the current delegate will be used.*
- void **addSelectionOverlay** (ItemDelegate \*delegate=nullptr)
- Album \* **albumAt** (const QPoint &pos) const
 

*If the model is categorized by an album, returns the album of the category that contains the position.*
- ItemInfoList **allItemInfos** () const
- QList< QUrl > **allUrls** () const
- Album \* **currentAlbum** () const
- ItemInfo **currentInfo** () const
- QUrl **currentUrl** () const
- ItemDelegate \* **delegate** () const
- QItemSelectionModel \* **getSelectionModel** () const
- ItemAlbumFilterModel \* **imageAlbumFilterModel** () const
- ItemAlbumModel \* **imageAlbumModel** () const
 

*Returns 0 if the ItemModel is not an ItemAlbumModel.*
- ItemFilterModel \* **imageFilterModel** () const
 

*Returns any ItemFilterMode in chain. May not be sourceModel()*
- ItemModel \* **imageModel** () const
- ImageSortFilterModel \* **imageSortFilterModel** () const
- ItemThumbnailModel \* **imageThumbnailModel** () const
 

*Returns 0 if the ItemModel is not an ItemThumbnailModel.*
- QModelIndex **indexForInfo** (const ItemInfo &info) const
- ItemInfo **nextInfo** (const ItemInfo &info)
- ItemInfo **nextInOrder** (const ItemInfo &startingPoint, int nth)
 

*Returns the n-th info after the given one.*
- ItemInfo **previousInfo** (const ItemInfo &info)
- void **removeOverlay** (ItemDelegateOverlay \*overlay)
- ItemInfoList **selectedItemInfos** () const
- ItemInfoList **selectedItemInfosCurrentFirst** () const
- void **setModels** (ItemModel \*model, ImageSortFilterModel \*filterModel)
- virtual void **setThumbnailSize** (const ThumbnailSize &size)
- ThumbnailSize **thumbnailSize** () const
- void **toIndex** (const QUrl &url)
 

*Selects the index as current and scrolls to it.*

## Public Member Functions inherited from Digikam::ItemViewCategorized

- ItemViewCategorized (QWidget \*const parent=nullptr)
- void **awayFromSelection** ()
- DItemDelegate \* **delegate** () const
- void **invertSelection** ()
- bool **isToolTipEnabled** () const
- int **numberOfSelectedIndexes** () const
- void **scrollTo** (const QModelIndex &index, ScrollHint hint=EnsureVisible) override
- void **scrollToRelaxed** (const QModelIndex &index, ScrollHint hint=EnsureVisible)
 

*Like scrollTo, but only scrolls if the index is not visible, regardless of hint.*
- void **setInitialSelectedItem** (bool enabled)
 

*Ensure a initial selected item.*

- void **setScrollCurrentToCenter** (bool enabled)  
*Scroll automatically the current index to center of the view.*
- void **setScrollStepGranularity** (int factor)  
*Determine a step size for scrolling: The larger this number, the smaller and more precise is the scrolling.*
- void **setSelectedIndexes** (const QList< QModelIndex > &indexes)
- void **setSpacing** (int spacing)  
*Sets the spacing.*
- void **setToolTipEnabled** (bool enabled)
- void **setUsePointingHandCursor** (bool useCursor)  
*Set if the PointingHand Cursor should be shown over the activation area.*
- void **toFirstIndex** ()  
*Selects the index as current and scrolls to it.*
- void **toIndex** (const QModelIndex &index)
- void **toLastIndex** ()
- void **toNextIndex** ()
- void **toPreviousIndex** ()

## Public Member Functions inherited from [Digikam::DCategorizedView](#)

- **DCategorizedView** (QWidget \*const parent=nullptr)
- virtual QModelIndexList **categorizedIndexesIn** (const QRect &rect) const  
*This method will return all indexes whose visual rect intersects rect.*
- virtual QModelIndex **categoryAt** (const QPoint &point) const  
*This method will return the first index of the category in the region of which point is found.*
- **DCategoryDrawer \* categoryDrawer** () const
- virtual QItemSelectionRange **categoryRange** (const QModelIndex &index) const  
*This method returns the range of indexes contained in the category in which index is sorted.*
- virtual QRect **categoryVisualRect** (const QModelIndex &index) const  
*This method will return the visual rect of the header of the category in which index is sorted.*
- QModelIndex **indexAt** (const QPoint &point) const override
- void **setCategoryDrawer** (DCategoryDrawer \*categoryDrawer)
- void **setDrawDraggedItems** (bool drawDraggedItems)  
*Switch on drawing of dragged items.*
- void **setGridSize** (const QSize &size)
- void **setModel** (QAbstractItemModel \*model) override
- QRect **visualRect** (const QModelIndex &index) const override

## Public Member Functions inherited from [Digikam::DragDropViewImplementation](#)

- virtual void **copy** ()
- virtual void **cut** ()
- virtual void **paste** ()

## Protected Slots

- void **slotCurrentUriTimer** ()
- void **slotItemInfosAdded** ()

## Protected Slots inherited from [Digikam::ItemViewCategorized](#)

- void **layoutAboutToBeChanged** ()
- void **layoutWasChanged** ()
- void **slotActivated** (const QModelIndex &index)
- void **slotClicked** (const QModelIndex &index)
- void **slotEntered** (const QModelIndex &index)
- virtual void **slotSetupChanged** ()
- virtual void **slotThemeChanged** ()

## Protected Slots inherited from [Digikam::DCategorizedView](#)

- void **currentChanged** (const QModelIndex &current, const QModelIndex &previous) override
- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- virtual void **rowsInsertedArtificial** (const QModelIndex &parent, int start, int end)
- virtual void **slotLayoutChanged** ()
- void **updateGeometries** () override

## Protected Member Functions

- virtual void **activated** (const [ItemInfo](#) &info, Qt::KeyboardModifiers modifiers)
  - Reimplement these in a subclass.*
- void **currentChanged** (const QModelIndex &index, const QModelIndex &previous) override
- [AbstractItemDragDropHandler](#) \* **dragDropHandler** () const override
  - You need to implement these three methods Returns the drag drop handler.*
- [QSortFilterProxyModel](#) \* **filterModel** () const override
- [ItemInfo](#) **imageInfo** (const QModelIndex &index) const
- [ItemInfoList](#) **imageInfos** (const QList< QModelIndex > &indexes) const
- void **indexActivated** (const QModelIndex &index, Qt::KeyboardModifiers modifiers) override
- void **installDefaultModels** ()
  - install default [ItemAlbumModel](#) and filter model, ready for use*
- QModelIndex **nextIndexHint** (const QModelIndex &indexToAnchor, const [QItemSelectionRange](#) &removed) const override
  - Assuming the given indexes would be removed (hypothetically!), return the index to be selected instead, starting from anchor.*
- void **selectionChanged** (const [QItemSelection](#) &, const [QItemSelection](#) &) override
- void **setItemDelegate** ([ItemDelegate](#) \*delegate)
- void **showContextMenuOnIndex** ([QContextMenuEvent](#) \*event, const QModelIndex &index) override
  - Reimplement these in a subclass.*
- virtual void **showContextMenuOnInfo** ([QContextMenuEvent](#) \*event, const [ItemInfo](#) &info)
- void **updateGeometries** () override



## Protected Member Functions inherited from Digikam::ItemViewCategorized

- void **contextMenuEvent** (QContextMenuEvent \*event) override  
*reimplemented from parent class*
- bool **decodelsCutSelection** (const QMimeData \*mimeData)
- void **encodelsCutSelection** (QMimeData \*mime, bool isCutSelection)
- QModelIndex **indexForCategoryAt** (const QPoint &pos) const  
*Returns an index that is representative for the category at position pos.*
- void **keyPressEvent** (QKeyEvent \*event) override
- void **leaveEvent** (QEvent \*event) override
- QModelIndex **mapIndexForDragDrop** (const QModelIndex &index) const override  
*Note: pure virtual dragDropHandler() still open from DragDropViewImplementation.*
- void **mouseMoveEvent** (QMouseEvent \*event) override
- void **mousePressEvent** (QMouseEvent \*event) override
- void **mouseReleaseEvent** (QMouseEvent \*event) override
- QModelIndex **moveCursor** (CursorAction cursorAction, Qt::KeyboardModifiers modifiers) override
- QPixmap **pixmapForDrag** (const QList< QModelIndex > &indexes) const override  
*Creates a pixmap for dragging the given indexes.*
- void **reset** () override
- void **resizeEvent** (QResizeEvent \*e) override
- void **rowsAboutToBeRemoved** (const QModelIndex &parent, int start, int end) override
- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- void **rowsRemoved** (const QModelIndex &parent, int start, int end) override
- void **selectionChanged** (const QItemSelection &, const QItemSelection &) override
- void **setItemDelegate** (DItemDelegate \*delegate)
- void **setToolTip** (ItemViewToolTip \*tip)
- virtual void **showContextMenu** (QContextMenuEvent \*event)
- virtual bool **showToolTip** (const QModelIndex &index, QStyleOptionViewItem &option, QHelpEvent \*e=nullptr)  
*Provides default behavior, can reimplement in a subclass.*
- void **updateDelegateSizes** ()
- void **userInteraction** ()
- bool **viewportEvent** (QEvent \*event) override
- void **wheelEvent** (QWheelEvent \*event) override

## Protected Member Functions inherited from Digikam::DCategorizedView

- void **dragLeaveEvent** (QDragLeaveEvent \*event) override
- void **dragMoveEvent** (QDragMoveEvent \*event) override
- void **dropEvent** (QDropEvent \*event) override
- void **leaveEvent** (QEvent \*event) override
- void **mouseMoveEvent** (QMouseEvent \*event) override
- void **mousePressEvent** (QMouseEvent \*event) override
- void **mouseReleaseEvent** (QMouseEvent \*event) override
- QModelIndex **moveCursor** (CursorAction cursorAction, Qt::KeyboardModifiers modifiers) override
- void **paintEvent** (QPaintEvent \*event) override
- void **resizeEvent** (QResizeEvent \*event) override
- void **setSelection** (const QRect &rect, QItemSelectionModel::SelectionFlags flags) override
- void **startDrag** (Qt::DropActions supportedActions) override

## Protected Member Functions inherited from [Digikam::DragDropViewImplementation](#)

- virtual `QAbstractItemView * asView ()=0`  
*This one is implemented by DECLARE\_VIEW\_DRAG\_DROP\_METHODS.*
- bool **decodelsCutSelection** (const `QMimeData *mimeData`)
- void **dragEnterEvent** (`QDragEnterEvent *event`)  
*Implements the relevant QAbstractItemView methods for drag and drop.*
- void **dragMoveEvent** (`QDragMoveEvent *e`)
- void **dropEvent** (`QDropEvent *e`)
- void **encodelsCutSelection** (`QMimeData *mime`, bool `isCutSelection`)
- void **startDrag** (`Qt::DropActions supportedActions`)

### 9.794.1 Member Function Documentation

#### 9.794.1.1 activated()

```
void Digikam::ItemCategorizedView::activated (
    const ItemInfo & info,
    Qt::KeyboardModifiers modifiers ) [protected], [virtual]
```

Reimplemented in [Digikam::DigikamItemView](#).

#### 9.794.1.2 albumAt()

```
Album * Digikam::ItemCategorizedView::albumAt (
    const QPoint & pos ) const
```

If this is not applicable, return the current album. May return 0.

#### 9.794.1.3 dragDropHandler()

```
AbstractItemDragDropHandler * Digikam::ItemCategorizedView::dragDropHandler ( ) const [override],
[protected], [virtual]
```

Implements [Digikam::DragDropViewImplementation](#).

#### 9.794.1.4 filterModel()

```
QSortFilterProxyModel * Digikam::ItemCategorizedView::filterModel ( ) const [override], [protected],
[virtual]
```

Implements [Digikam::ItemViewCategorized](#).

#### 9.794.1.5 indexActivated()

```
void Digikam::ItemCategorizedView::indexActivated (
    const QModelIndex & index,
    Qt::KeyboardModifiers modifiers ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ItemViewCategorized](#).

### 9.794.1.6 nextIndexHint()

```
QModelIndex Digikam::ItemCategorizedView::nextIndexHint (
    const QModelIndex & indexToAnchor,
    const QItemSelectionRange & removed ) const [override], [protected], [virtual]
```

The default implementation returns the next remaining sibling.

Reimplemented from [Digikam::ItemViewCategorized](#).

### 9.794.1.7 nextInOrder()

```
ItemInfo Digikam::ItemCategorizedView::nextInOrder (
    const ItemInfo & startingPoint,
    int nth )
```

Specifically, return the previous info for  $nth = -1$  and the next info for  $n = 1$ . Returns a null info if either `startingPoint` or the `nth` info are not contained in the model.

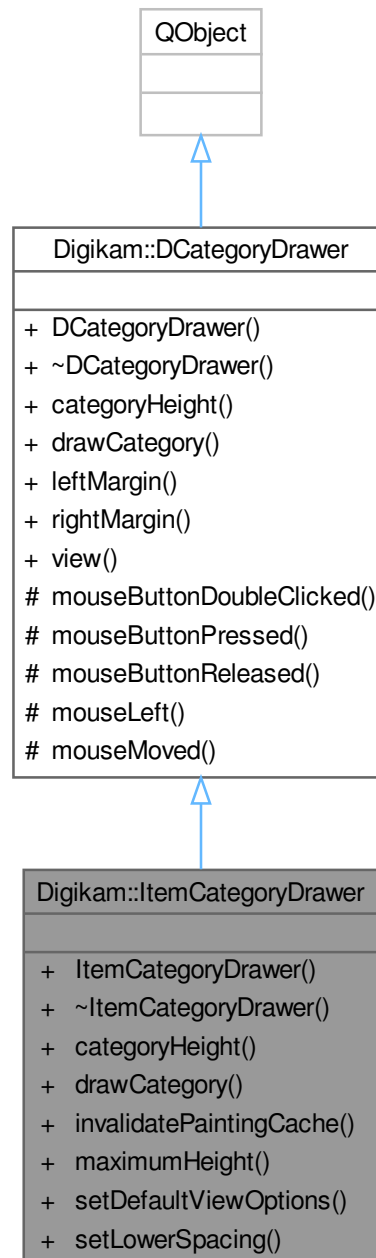
### 9.794.1.8 showContextMenuOnIndex()

```
void Digikam::ItemCategorizedView::showContextMenuOnIndex (
    QContextMenuEvent * event,
    const QModelIndex & index ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ItemViewCategorized](#).

## 9.795 Digikam::ItemCategoryDrawer Class Reference

Inheritance diagram for Digikam::ItemCategoryDrawer:



### Public Member Functions

- **ItemCategoryDrawer** ([ItemCategorizedView](#) \*const parent)
- int [categoryHeight](#) (const QModelIndex &index, const QStyleOption &option) const override

- void [drawCategory](#) (const QModelIndex &index, int sortRole, const QStyleOption &option, QPainter \*painter) const override  
*This method purpose is to draw a category represented by the given.*
- void [invalidatePaintingCache](#) ()
- virtual int [maximumHeight](#) () const
- void [setDefaultViewOptions](#) (const QStyleOptionViewItem &option)
- void [setLowerSpacing](#) (int spacing)

## Public Member Functions inherited from [Digikam::DCategoryDrawer](#)

- [DCategoryDrawer](#) ([DCategorizedView](#) \*const view)  
*Construct a category drawer for a given view.*
- virtual int [leftMargin](#) () const
- virtual int [rightMargin](#) () const
- [DCategorizedView](#) \* view () const

## Additional Inherited Members

## Signals inherited from [Digikam::DCategoryDrawer](#)

- void [actionRequested](#) (int action, const QModelIndex &index)  
*Emit this signal on your subclass implementation to notify that something happened.*
- void [collapseOrExpandClicked](#) (const QModelIndex &index)  
*This signal becomes emitted when collapse or expand has been clicked.*

## Protected Member Functions inherited from [Digikam::DCategoryDrawer](#)

- virtual void [mouseButtonDoubleClicked](#) (const QModelIndex &index, const QRect &blockRect, QMouseEvent \*event)  
*Method called when the mouse button has been double clicked.*
- virtual void [mouseButtonPressed](#) (const QModelIndex &index, const QRect &blockRect, QMouseEvent \*event)  
*Method called when the mouse button has been pressed.*
- virtual void [mouseButtonReleased](#) (const QModelIndex &index, const QRect &blockRect, QMouseEvent \*event)  
*Method called when the mouse button has been released.*
- virtual void [mouseLeft](#) (const QModelIndex &index, const QRect &blockRect)  
*Method called when the mouse button has left this block.*
- virtual void [mouseMoved](#) (const QModelIndex &index, const QRect &blockRect, QMouseEvent \*event)  
*Method called when the mouse has been moved.*

## 9.795.1 Member Function Documentation

### 9.795.1.1 [categoryHeight\(\)](#)

```
int Digikam::ItemCategoryDrawer::categoryHeight (
    const QModelIndex & index,
    const QStyleOption & option ) const [override], [virtual]
```

#### Returns

The category height for the category represented by index `index` with style options `option`.

Reimplemented from [Digikam::DCategoryDrawer](#).

### 9.795.1.2 drawCategory()

```
void Digikam::ItemCategoryDrawer::drawCategory (
    const QModelIndex & index,
    int sortRole,
    const QStyleOption & option,
    QPainter * painter ) const [override], [virtual]
```

#### Parameters

<i>index</i>	with the given
<i>sortRole</i>	sorting role
<i>option</i>	painter style options
<i>painter</i>	painter instance

#### Note

This method will be called one time per category, always with the first element in that category

Reimplemented from [Digikam::DCategoryDrawer](#).

## 9.796 Digikam::ItemChangeHint Class Reference

### Public Types

- enum [ChangeType](#) { [ItemModified](#) , [ItemRescan](#) }

An [ItemCopyMoveHint](#) describes a list of existing items that should be updated although the modification date may not have changed.

### Public Member Functions

- **ItemChangeHint** (const QList< qlonglong > &srcIds, [ChangeType](#) type=[ItemModified](#))
- [ChangeType](#) **changeType** () const
- QList< qlonglong > **ids** () const
- bool **isId** (qlonglong id) const
- bool **isModified** () const
- bool **needsRescan** () const
- [ItemChangeHint](#) & **operator**<< (const QDBusArgument &argument)
- const [ItemChangeHint](#) & **operator**>> (QDBusArgument &argument) const

### Protected Attributes

- QList< qlonglong > **m\_ids**
- [ChangeType](#) **m\_type** = [ItemModified](#)

## 9.796.1 Member Enumeration Documentation

### 9.796.1.1 ChangeType

```
enum Digikam::ItemChangeHint::ChangeType
```

## Enumerator

ItemModified	treat as if modification date changed
ItemRescan	reread metadata

## 9.797 Digikam::ItemComments Class Reference

### Public Types

- enum [LanguageChoiceBehavior](#) { [ReturnMatchingLanguageOnly](#) , [ReturnMatchingOrDefaultLanguage](#) , [ReturnMatchingDefaultOrFirstLanguage](#) }

The *ItemComments* class shall provide short-lived objects that provide read/write access to the comments stored in the database.

- enum [UniqueBehavior](#) { [UniquePerLanguage](#) , [UniquePerLanguageAndAuthor](#) }

### Public Member Functions

- **ItemComments** ()  
Create a null *ItemComments* object.
- **ItemComments** (const [CoreDbAccess](#) &access, qulonglong imageid)  
Create a *ItemComments* object for the image with the specified id.
- **ItemComments** (const [ItemComments](#) &other)
- **ItemComments** (qulonglong imageid)  
Create a *ItemComments* object for the image with the specified id.
- void **addComment** (const [QString](#) &comment, const [QString](#) &language=[QString](#)(), const [QString](#) &author=[QString](#)(), const [QDateTime](#) &date=[QDateTime](#)(), [DatabaseComment::Type](#) type=[DatabaseComment::Comment](#))  
Add a new comment to the list of normal image comments, specified with language and author.
- void **addHeadline** (const [QString](#) &headline, const [QString](#) &language=[QString](#)(), const [QString](#) &author=[QString](#)(), const [QDateTime](#) &date=[QDateTime](#)())  
Convenience method to add a comment of type *Headline*.
- void **addTitle** (const [QString](#) &title, const [QString](#) &language=[QString](#)(), const [QString](#) &author=[QString](#)(), const [QDateTime](#) &date=[QDateTime](#)())  
Convenience method to add a comment of type *Headline*.
- void **apply** ()  
Apply all changes.
- void **apply** ([CoreDbAccess](#) &access)
- [QString](#) **author** (int index) const
- void **changeAuthor** (int index, const [QString](#) &author)
- void **changeComment** (int index, const [QString](#) &comment)  
Access individual properties.
- void **changeDate** (int index, const [QDateTime](#) &date)
- void **changeLanguage** (int index, const [QString](#) &language)
- void **changeType** (int index, [DatabaseComment::Type](#) type)
- [QString](#) **comment** (int index) const
- [QString](#) **commentForLanguage** (const [QString](#) &languageCode, int \*const index=nullptr, [LanguageChoiceBehavior](#) behavior=[ReturnMatchingDefaultOrFirstLanguage](#)) const  
Returns a comment for the specified language.
- [QDateTime](#) **date** (int index) const

- QString **defaultComment** (DatabaseComment::Type **type**=DatabaseComment::Comment) const  
*This methods presents one of the comment strings of the available comment as the default value, when you just want to have one string.*
- QString **defaultComment** (int \*const index, Digikam::DatabaseComment::Type **type**=DatabaseComment::Comment) const
- bool **isNull** () const
- QString **language** (int index) const  
*RFC 3066 notation, or "x-default".*
- int **numberOfComments** () const  
*Returns the number of comments available.*
- **ItemComments** & **operator=** (const **ItemComments** &other)
- void **remove** (int index)  
*Remove the entry referred to by index.*
- void **removeAll** ()  
*Remove all entries of all types: Comments, Headlines, Titles.*
- void **removeAll** (DatabaseComment::Type **type**)  
*Remove all entries of the given type.*
- void **removeAllComments** ()  
*Convenience method: remove all entries of type Comment.*
- void **replaceComments** (const **CaptionsMap** &comments, DatabaseComment::Type **type**=DatabaseComment::Comment)  
*Replaces all existing comments with the given set of comments and associated language.*
- void **replaceFrom** (const **ItemComments** &source)  
*Replaces all entries in this object with all entries from source.*
- void **setUniqueBehavior** (**UniqueBehavior** behavior)  
*Changes the behavior to unique comments per language, see the enum above for possible values.*
- **CaptionsMap toCaptionsMap** (DatabaseComment::Type=DatabaseComment::Comment) const  
*Returns all entries of the given type in a **CaptionsMap** container.*
- DatabaseComment::Type **type** (int index) const  
*Access individual properties.*

### Protected Member Functions

- void **addCommentDirectly** (const QString &comment, const QString &language, const QString &author, DatabaseComment::Type **type**, const QDateTime &date)

### Protected Attributes

- QSharedPointer< Private > **d**

## 9.797.1 Member Enumeration Documentation

### 9.797.1.1 LanguageChoiceBehavior

enum [Digikam::ItemComments::LanguageChoiceBehavior](#)

It is a mere wrapper around the less convenient access methods in [CoreDB](#). Database results are cached, but the object will not listen to database changes from other places.

Changes are applied to the database only after calling [apply\(\)](#), which you can call any time and which will in any case be called from the destructor.



## Enumerator

ReturnMatchingLanguageOnly	Return only a comment if the language code (at least the language code, the country part may differ) is identical. Else returns a null QString.
ReturnMatchingOrDefaultLanguage	If no matching language as above is found, return the default language.
ReturnMatchingDefaultOrFirstLanguage	If no matching or default language is found, return the first comment. Returns a null string only if no comment is available.

## 9.797.1.2 UniqueBehavior

```
enum Digikam::ItemComments::UniqueBehavior
```

## Enumerator

UniquePerLanguage	Allow only one comment per language. Default setting.
UniquePerLanguageAndAuthor	Allow multiple comments per language, each with a different author.

## 9.797.2 Constructor &amp; Destructor Documentation

## 9.797.2.1 ItemComments()

```
Digikam::ItemComments::ItemComments (
    const CoreDbAccess & access,
    qulonglong imageid )
```

The existing [CoreDbAccess](#) object will be used to access the database.

## 9.797.3 Member Function Documentation

## 9.797.3.1 addComment()

```
void Digikam::ItemComments::addComment (
    const QString & comment,
    const QString & language = QString(),
    const QString & author = QString(),
    const QDateTime & date = QDateTime(),
    DatabaseComment::Type type = DatabaseComment::Comment )
```

Checking for unique comments is done as set by `setUniqueBehavior`. If you pass a null string as language, it will be translated to the language code designating the default language ("x-default"). If you just want to change the one comment of the image, call `addComment(myComment)`;

### 9.797.3.2 addHeadline()

```
void Digikam::ItemComments::addHeadline (
    const QString & headline,
    const QString & language = QString(),
    const QString & author = QString(),
    const QDateTime & date = QDateTime() )
```

Calls addComment, see above for more info.

### 9.797.3.3 addTitle()

```
void Digikam::ItemComments::addTitle (
    const QString & title,
    const QString & language = QString(),
    const QString & author = QString(),
    const QDateTime & date = QDateTime() )
```

Calls addComment, see above for more info.

### 9.797.3.4 apply()

```
void Digikam::ItemComments::apply ( )
```

Also called in destructor, so you typically do not need to call this.

### 9.797.3.5 changeComment()

```
void Digikam::ItemComments::changeComment (
    int index,
    const QString & comment )
```

Please ensure that the specified index is a valid index

### 9.797.3.6 commentForLanguage()

```
QString Digikam::ItemComments::commentForLanguage (
    const QString & languageCode,
    int *const index = nullptr,
    LanguageChoiceBehavior behavior = ReturnMatchingDefaultOrFirstLanguage ) const
```

Matching behavior can be specified. Optionally also returns the index with which you can access further information about the comment.

### 9.797.3.7 defaultComment()

```
QString Digikam::ItemComments::defaultComment (
    DatabaseComment::Type type = DatabaseComment::Comment ) const
```

Optionally also returns the index with which you can access further information about the comment.

### 9.797.3.8 replaceComments()

```
void Digikam::ItemComments::replaceComments (
    const CaptionsMap & comments,
    DatabaseComment::Type type = DatabaseComment::Comment )
```

Optionally date and author can be specified in [CaptionsMap](#) container.

### 9.797.3.9 setUniqueBehavior()

```
void Digikam::ItemComments::setUniqueBehavior (
    UniqueBehavior behavior )
```

Default value is UniquePerLanguage. Note: This is *not* a property of the database, but only of this single [ItemComments](#) object,

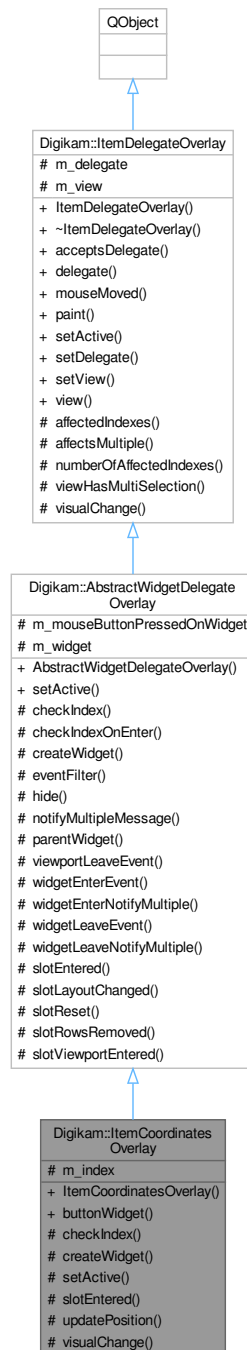
### 9.797.3.10 type()

```
DatabaseComment::Type Digikam::ItemComments::type (
    int index ) const
```

Please ensure that the specified index is a valid index

## 9.798 Digikam::ItemCoordinatesOverlay Class Reference

Inheritance diagram for Digikam::ItemCoordinatesOverlay:



### Public Member Functions

- **ItemCoordinatesOverlay** (QObject \*const parent)
- [CoordinatesOverlayWidget](#) \* **buttonWidget** () const

## Public Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- [AbstractWidgetDelegateOverlay](#) (QObject \*const parent)  
*This class provides functionality for using a widget in an overlay.*

## Public Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- [ItemDelegateOverlay](#) (QObject \*const parent=nullptr)
- virtual bool [acceptsDelegate](#) (QAbstractItemDelegate \*) const
- QAbstractItemDelegate \* [delegate](#) () const
- virtual void [mouseMoved](#) (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)  
*Only these two methods are implemented as virtual methods.*
- virtual void [paint](#) (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index)
- void [setDelegate](#) (QAbstractItemDelegate \*delegate)
- void [setView](#) (QAbstractItemView \*view)
- QAbstractItemView \* [view](#) () const

## Protected Member Functions

- bool [checkIndex](#) (const QModelIndex &index) const override  
*Return true here if you want to show the overlay for the given index.*
- QWidget \* [createWidget](#) () override  
*Create your widget here.*
- void [setActive](#) (bool active) override  
*If active is true, this will call [createWidget\(\)](#), initialize the widget for use, and setup connections for the virtual slots.*
- void [slotEntered](#) (const QModelIndex &index) override  
*Default implementation shows the widget iff the index is valid and [checkIndex](#) returns true.*
- void [updatePosition](#) ()
- void [visualChange](#) () override  
*Called when any change from the delegate occurs - when the overlay is installed, when size hints, styles or fonts change.*

## Protected Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- bool [checkIndexOnEnter](#) (const QModelIndex &index) const  
*Utility method called from [slotEntered](#).*
- bool [eventFilter](#) (QObject \*obj, QEvent \*event) override
- virtual void [hide](#) ()  
*Called when the widget shall be hidden (mouse cursor left index, viewport, uninstalled etc.).*
- virtual QString [notifyMultipleMessage](#) (const QModelIndex &, int number)
- QWidget \* [parentWidget](#) () const  
*Returns the widget to be used as parent for your widget created in [createWidget\(\)](#)*
- virtual void [viewportLeaveEvent](#) (QObject \*obj, QEvent \*event)  
*Called when a `QEvent::Leave` of the viewport is received.*
- virtual void [widgetEnterEvent](#) ()  
*Called when a `QEvent::Enter` resp.*
- void [widgetEnterNotifyMultiple](#) (const QModelIndex &index)  
*A sample implementation for above methods.*
- virtual void [widgetLeaveEvent](#) ()
- void [widgetLeaveNotifyMultiple](#) ()

## Protected Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- `QList< QModelIndex > affectedIndexes` (const QModelIndex &index) const
- `bool affectsMultiple` (const QModelIndex &index) const  
*For the context that an overlay can affect multiple items: Assuming the currently overlaid index is given.*
- `int numberOfAffectedIndexes` (const QModelIndex &index) const
- `bool viewHasMultiSelection` () const  
*Utility method.*

## Protected Attributes

- `QPersistentModelIndex m_index`

## Protected Attributes inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- `bool m_mouseButtonPressedOnWidget` = false
- `QWidget * m_widget` = nullptr

## Protected Attributes inherited from [Digikam::ItemDelegateOverlay](#)

- `QAbstractItemDelegate * m_delegate` = nullptr
- `QAbstractItemView * m_view` = nullptr

## Additional Inherited Members

## Signals inherited from [Digikam::ItemDelegateOverlay](#)

- `void hideNotification` ()
- `void requestNotification` (const QModelIndex &index, const QString &message)
- `void update` (const QModelIndex &index)

## Protected Slots inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- `virtual void slotLayoutChanged` ()
- `virtual void slotReset` ()  
*Default implementations of these three slots call `hide()`*
- `virtual void slotRowsRemoved` (const QModelIndex &parent, int start, int end)
- `virtual void slotViewportEntered` ()

## Protected Slots inherited from [Digikam::ItemDelegateOverlay](#)

### 9.798.1 Member Function Documentation

#### 9.798.1.1 `checkIndex()`

```
bool Digikam::ItemCoordinatesOverlay::checkIndex (
    const QModelIndex & index ) const [override], [protected], [virtual]
```

The default implementation returns true.

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.798.1.2 createWidget()

```
QWidget * Digikam::ItemCoordinatesOverlay::createWidget ( ) [override], [protected], [virtual]
```

When creating the object, pass [parentWidget\(\)](#) as parent widget. Ownership of the object is passed. It will be deleted in [setActive\(false\)](#).

Implements [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.798.1.3 setActive()

```
void Digikam::ItemCoordinatesOverlay::setActive (
    bool active ) [override], [protected], [virtual]
```

If active is false, this will delete the widget and disconnect all signal from model and view to this object (!)

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.798.1.4 slotEntered()

```
void Digikam::ItemCoordinatesOverlay::slotEntered (
    const QModelIndex & index ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.798.1.5 visualChange()

```
void Digikam::ItemCoordinatesOverlay::visualChange ( ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ItemDelegateOverlay](#).

## 9.799 Digikam::ItemCopyMoveHint Class Reference

### Public Member Functions

- [ItemCopyMoveHint](#) ()=default
  - An [ItemCopyMoveHint](#) describes a list of existing items that will be copied, moved or renamed to an album given by album root id and album id.*
- [ItemCopyMoveHint](#) (const QList< qlonglong > &srcIds, int dstAlbumRootId, int albumId, const QStringList &dstNames)
- int [albumIdDst](#) () const
- int [albumRootIdDst](#) () const
- [CollectionScannerHints::Album](#) [dst](#) () const
- QString [dstName](#) (qlonglong id) const
- QStringList [dstNames](#) () const
- bool [isDstAlbum](#) (int albumRootId, int albumId) const
- bool [isSrcId](#) (qlonglong id) const
- **operator const [CollectionScannerHints::Album](#) & ()** const
- [ItemCopyMoveHint](#) & **operator**<< (const QDBusArgument &argument)
- bool **operator**== (const [CollectionScannerHints::Album](#) &dst) const
- const [ItemCopyMoveHint](#) & **operator**>> (QDBusArgument &argument) const
- QList< qlonglong > [srcIds](#) () const

## Protected Attributes

- CollectionScannerHints::Album **m\_dst**
- QStringList **m\_dstNames**
- QList< qlonglong > **m\_srcIds**

## 9.799.1 Constructor & Destructor Documentation

### 9.799.1.1 ItemCopyMoveHint()

```
Digikam::ItemCopyMoveHint::ItemCopyMoveHint ( ) [default]
```

In the new album, the items will have the filenames given in dstNames.

## 9.800 Digikam::ItemCopyright Class Reference

### Public Types

- enum [ReplaceMode](#) { [ReplaceAllEntries](#) , [ReplaceLanguageEntry](#) , [AddEntryToExisting](#) }

### Public Member Functions

- **ItemCopyright** ()=default  
*Create a null [ItemCopyright](#) object.*
- **ItemCopyright** (const [ItemCopyright](#) &other)
- **ItemCopyright** (qlonglong imageid)
- [MetaEngine::AltLangMap](#) **allCopyrightNotices** ()
- [MetaEngine::AltLangMap](#) **allRightsUsageTerms** ()
- QStringList **author** () const
- QString **authorsPosition** () const
- QStringList **byLine** () const
- QString **byLineTitle** () const
- [IptcCoreContactInfo](#) **contactInfo** ()  
*Returns the creator's contact info.*
- QString **copyrightNotice** (const QString &languageCode=QString())  
*Returns the copyright notice.*
- QStringList **creator** () const  
*Returns the author/creator/byline.*
- QString **creatorJobTitle** () const  
*Returns the creator's job title.*
- QString **credit** () const
- void **fillTemplate** ([Template](#) &t)  
*Fills the information fields in template concerning copyright info (note there are other fields in the a [Template](#)).*
- QString **instructions** ()  
*Returns the instructions.*
- [ItemCopyright](#) & **operator=** (const [ItemCopyright](#) &other)
- QString **provider** () const  
*Returns the credit/provider.*
- void **removeAll** ()



*Calls all remove...() methods in this class.*

- void **removeContactInfo** ()
- void **removeCopyrightNotices** ()
- void **removeCreatorJobTitle** ()
- void **removeCreators** ()
- void **removeInstructions** ()
- void **removeProvider** ()
- void **removeRightsUsageTerms** ()
- void **removeSource** ()
- void **replaceFrom** (const [ItemCopyright](#) &source)

*Removes all entries and replaces them with the entries from source.*

- QString **rights** (const QString &languageCode=QString())
- QString **rightsUsageTerms** (const QString &languageCode=QString())

*Returns the right usage terms.*

- void **setAuthor** (const QString &author, [ReplaceMode](#) mode=[ReplaceAllEntries](#))
- void **setAuthorsPosition** (const QString &position)
- void **setByLine** (const QString &byline, [ReplaceMode](#) mode=[ReplaceAllEntries](#))
- void **setByLineTitle** (const QString &title)
- void **setContactInfo** (const [IptcCoreContactInfo](#) &info)
- void **setCopyrightNotice** (const QString &notice, const QString &languageCode=QString(), [ReplaceMode](#) mode=[ReplaceLanguageEntry](#))

*Sets the copyright notice.*

- void **setCreator** (const QString &creator, [ReplaceMode](#) mode=[ReplaceAllEntries](#))

*Sets the creator.*

- void **setCreatorJobTitle** (const QString &title)
- void **setCredit** (const QString &credit)
- void **setFromTemplate** (const [Template](#) &t)

*Sets all database copyright fields from the template.*

- void **setInstructions** (const QString &instructions)
- void **setProvider** (const QString &provider)
- void **setRights** (const QString &notice, const QString &languageCode=QString(), [ReplaceMode](#) mode=[ReplaceLanguageEntry](#))
- void **setRightsUsageTerms** (const QString &term, const QString &languageCode=QString(), [ReplaceMode](#) mode=[ReplaceLanguageEntry](#))
- void **setSource** (const QString &source)
- QString **source** ()

*Returns the source.*

## Protected Member Functions

- [CopyrightInfo](#) **copyrightInfo** (const QString &property) const
- QList< [CopyrightInfo](#) > **copyrightInfos** (const QString &property) const
- int **languageMatch** (const QList< [CopyrightInfo](#) > &infos, const QString &languageCode) const
- [MetaEngine::AltLangMap](#) **readLanguageProperties** (const QString &property)
- QString **readLanguageProperty** (const QString &property, const QString &languageCode)
- QString **readSimpleProperty** (const QString &property) const
- void **removeLanguageProperty** (const QString &property, const QString &languageCode)
- void **removeProperties** (const QString &property)
- void **setLanguageProperty** (const QString &property, const QString &value, const QString &languageCode, [ReplaceMode](#) mode)
- void **setSimpleProperty** (const QString &property, const QString &value)

## Protected Attributes

- ItemCopyrightCache \* **m\_cache** = nullptr
- qlonglong **m\_id** = 0

## Friends

- class **ItemCopyrightCache**

## 9.800.1 Member Enumeration Documentation

### 9.800.1.1 ReplaceMode

```
enum Digikam::ItemCopyright::ReplaceMode
```

#### Enumerator

ReplaceAllEntries	Remove entries for all languages and add one new entry.
ReplaceLanguageEntry	Only replace the entry with the given language.
AddEntryToExisting	No constraints on adding the entry.

## 9.800.2 Member Function Documentation

### 9.800.2.1 contactInfo()

```
IptcCoreContactInfo Digikam::ItemCopyright::contactInfo ( )
```

This is Iptc4xmpCore:CreatorContactInfo in XMP. The creator's contact information provides all necessary information to get in contact with the creator of this news object and comprises a set of sub-properties for proper addressing.

### 9.800.2.2 copyrightNotice()

```
QString Digikam::ItemCopyright::copyrightNotice (
    const QString & languageCode = QString() )
```

This is Photoshop Copyright Notice. This is IPTC Copyright Notice. This is DC Rights. This is dc:rights in XMP. Contains any necessary copyright notice for claiming the intellectual property for this news object and should identify the current owner of the copyright for the news object. Other entities like the creator of the news object may be added. Notes on usage rights should be provided in Rights usage terms. Note on language matching: You can specify a language code. If the requested language is not available, the entry with default language code is returned. If a default-language entry is not available, the first entry is returned. If you pass a null string as languageCode, the local language is returned.

### 9.800.2.3 creator()

```
QStringList Digikam::ItemCopyright::creator ( ) const
```

This is Photoshop Author. This is IPTC By-line. This is DC creator. This is dc:creator in XMP. Contains preferably the name of the person who created the content of this news object, a photographer for photos, a graphic artist for graphics, or a writer for textual news. If it is not appropriate to add the name of a person the name of a company or organization could be applied as well. Aligning with IIM notions IPTC Core intends to have only one creator for this news object despite the underlying XMP property dc:creator allows for more than one item to be included. If there are more than one item in this array the first one should be considered as the IPTC Core Creator value.

### 9.800.2.4 creatorJobTitle()

```
QString Digikam::ItemCopyright::creatorJobTitle ( ) const
```

This is Photoshop AuthorsPosition. This is IPTC By-line Title. This is photoshop:AuthorsPosition in XMP. Contains the job title of the person who created the content of this news object. As this is sort of a qualifier the Creator element has to be filled in as mandatory prerequisite for using Creator's Jobtitle.

### 9.800.2.5 fillTemplate()

```
void Digikam::ItemCopyright::fillTemplate (
    Template & t )
```

There will not be touched)

### 9.800.2.6 instructions()

```
QString Digikam::ItemCopyright::instructions ( )
```

This is Photoshop Instructions. This is IPTC Special Instruction. This is photoshop:Instructions in XMP. Any of a number of instructions from the provider or creator to the receiver of the news object which might include any of the following: embargoes (NewsMagazines OUT) and other restrictions not covered by the Rights Usage Terms field; information regarding the original means of capture (scanning notes, colorspace info) or other specific text information that the user may need for accurate reproduction; additional permissions or credits required when publishing.

### 9.800.2.7 provider()

```
QString Digikam::ItemCopyright::provider ( ) const
```

This is Photoshop Credit. This is IPTC Credit. This is photoshop:Credit in XMP Identifies the provider of the news object, who is not necessarily the owner/creator.

### 9.800.2.8 rightsUsageTerms()

```
QString Digikam::ItemCopyright::rightsUsageTerms (
    const QString & languageCode = QString() )
```

This has no equivalent in Photoshop, IPTC, or DC. This is xmpRights:UsageTerms in XMP. Language matching is done as with [copyrightNotice\(\)](#). Free text instructions on how this news object can be legally used.

### 9.800.2.9 setCopyrightNotice()

```
void Digikam::ItemCopyright::setCopyrightNotice (
    const QString & notice,
    const QString & languageCode = QString(),
    ReplaceMode mode = ReplaceLanguageEntry )
```

If you supply a null QString as language code, this is regarded as an entry for the default language ("x-default"). The ReplaceMode determines how existing entries are handled.

### 9.800.2.10 setCreator()

```
void Digikam::ItemCopyright::setCreator (
    const QString & creator,
    ReplaceMode mode = ReplaceAllEntries )
```

If you want to specify only one creator, set the replace mode to ReplaceAllEntries. If you want to add it to a list of existing entries, pass AddEntryToExisting. You shall not use ReplaceLanguageEntry for this method, creators have no language associated.

### 9.800.2.11 setFromTemplate()

```
void Digikam::ItemCopyright::setFromTemplate (
    const Template & t )
```

This does not clear any fields before.

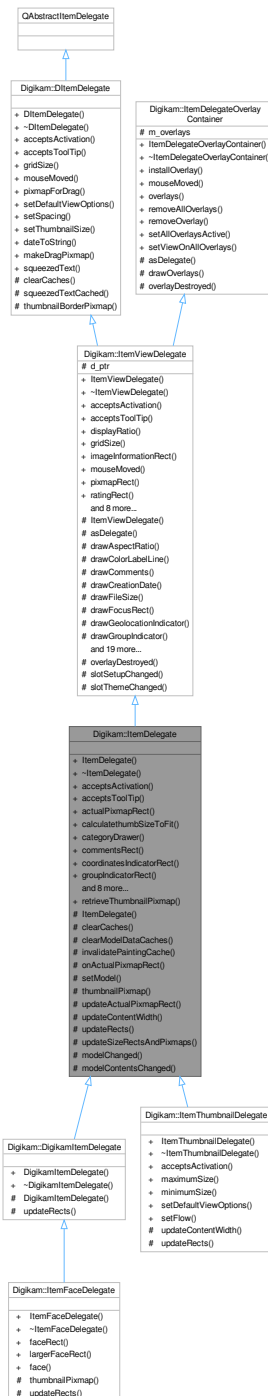
### 9.800.2.12 source()

```
QString Digikam::ItemCopyright::source ( )
```

This is Photoshop Source. This is IPTC Source. This is photoshop::Source in XMP. Identifies the original owner of the copyright for the intellectual content of the news object. This could be an agency, a member of an agency or an individual. Source could be different from Creator and from the entities in the CopyrightNotice. As the original owner can not change the content of this property should never be changed or deleted after the information is entered following the news object's initial creation.

## 9.801 Digikam::ItemDelegate Class Reference

Inheritance diagram for Digikam::ItemDelegate:



### Public Member Functions

- **ItemDelegate** (QWidget \*const parent)
- bool **acceptsActivation** (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*activationRect=nullptr) const override

- bool [acceptsToolTip](#) (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*tooltipRect=nullptr) const override  
*These methods take four parameters: The position on viewport, the rect on viewport, the index, and optionally a parameter into which, if the return value is true, a rectangle can be written for which the return value will be true as well.*
- QRect **actualPixmapRect** (const QModelIndex &index) const
- int **calculatethumbSizeToFit** (int ws)
- [ItemCategoryDrawer](#) \* **categoryDrawer** () const
- QRect **commentsRect** () const
- QRect **coordinatesIndicatorRect** () const
- QRect **groupIndicatorRect** () const
- QRect [imageInformationRect](#) () const override  
*Returns the area where the image information is drawn, or null if empty / not supported.*
- void **paint** (QPainter \*painter, const QStyleOptionViewItem &option, const QModelIndex &index) const override
- QPixmap [pixmapForDrag](#) (const QStyleOptionViewItem &option, const QList< QModelIndex > &indexes) const override
- QRect [pixmapRect](#) () const override  
*Returns the area where the pixmap is drawn, or null if not supported.*
- void [setDefaultViewOptions](#) (const QStyleOptionViewItem &option) override  
*Style option with standard values to use for cached rendering.*
- void [setSpacing](#) (int spacing) override
- void **setView** ([ItemCategorizedView](#) \*view)
- QRect **tagsRect** () const

## Public Member Functions inherited from [Digikam::ItemViewDelegate](#)

- **ItemViewDelegate** (QWidget \*const parent)
- bool [acceptsActivation](#) (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*activationRect=nullptr) const override
- bool [acceptsToolTip](#) (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*tooltipRect=nullptr) const override  
*These methods take four parameters: The position on viewport, the rect on viewport, the index, and optionally a parameter into which, if the return value is true, a rectangle can be written for which the return value will be true as well.*
- double **displayRatio** () const
- QSize [gridSize](#) () const override  
*Returns the gridsize to be set by the view.*
- void [mouseMoved](#) (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index) override
- virtual QRect **ratingRect** () const  
*Returns the rectangle where the rating is drawn, or a null rectangle if not supported.*
- QRect **rect** () const
- void [setDefaultViewOptions](#) (const QStyleOptionViewItem &option) override  
*Style option with standard values to use for cached rendering.*
- void [setRatingEdited](#) (const QModelIndex &index)  
*Can be used to temporarily disable drawing of the rating.*
- void [setSpacing](#) (int spacing) override
- void [setThumbnailSize](#) (const [ThumbnailSize](#) &thumbSize) override  
*You must set these options from the view.*
- QSize **sizeHint** (const QStyleOptionViewItem &option, const QModelIndex &index) const override
- int **spacing** () const
- [ThumbnailSize](#) **thumbnailSize** () const

## Public Member Functions inherited from [Digikam::DItemDelegate](#)

- **DItemDelegate** (QObject \*const parent=nullptr)

## Public Member Functions inherited from [Digikam::ItemDelegateOverlayContainer](#)

- **ItemDelegateOverlayContainer** ()=default  
*This is a sample implementation for delegate management methods, to be inherited by a delegate.*
- void **installOverlay** ([ItemDelegateOverlay](#) \*overlay)
- void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)
- QList< [ItemDelegateOverlay](#) \* > **overlays** () const
- void **removeAllOverlays** ()
- void **removeOverlay** ([ItemDelegateOverlay](#) \*overlay)
- void **setAllOverlaysActive** (bool active)
- void **setViewOnAllOverlays** (QAbstractItemView \*view)

## Static Public Member Functions

- static QPixmap **retrieveThumbnailPixmap** (const QModelIndex &index, int thumbnailSize)  
*Retrieve the thumbnail pixmap in given size for the [ItemModel::ThumbnailRole](#) for the given index from the given index, which must adhere to [ItemThumbnailModel](#) semantics.*

## Static Public Member Functions inherited from [Digikam::DItemDelegate](#)

- static QString **dateToString** (const QDateTime &datetime)
- static QPixmap **makeDragPixmap** (const QStyleOptionViewItem &option, const QList< QModelIndex > &indexes, double displayRatio, const QPixmap &suggestedPixmap=QPixmap())
- static QString **squeezedText** (const QFontMetrics &fm, int width, const QString &text)

## Protected Slots

- void **modelChanged** ()
- void **modelContentsChanged** ()

## Protected Slots inherited from [Digikam::ItemViewDelegate](#)

- void **overlayDestroyed** (QObject \*o) override
- void **slotSetupChanged** ()
- void **slotThemeChanged** ()

## Protected Member Functions

- **ItemDelegate** (ItemDelegate::ItemDelegatePrivate &dd, QWidget \*const parent)
- void **clearCaches** () override
- virtual void **clearModelDataCaches** ()
 

*Reimplement to clear caches based on model indexes (hash on row number etc.) Change signals are listened to this is called whenever such properties become invalid.*
- void **invalidatePaintingCache** () override
- bool **onActualPixmapRect** (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*actualRect) const
- void **setModel** (QAbstractItemModel \*model)
- virtual QPixmap **thumbnailPixmap** (const QModelIndex &index) const
- void **updateActualPixmapRect** (const QModelIndex &index, const QRect &rect)
- virtual void **updateContentWidth** ()
 

*Reimplement this to set contentWidth.*
- virtual void **updateRects** ()=0
 

*In a subclass, you need to implement this method to set up the rects for drawing.*
- void **updateSizeRectsAndPxmmaps** () override

## Protected Member Functions inherited from [Digikam::ItemViewDelegate](#)

- **ItemViewDelegate** (ItemViewDelegatePrivate &dd, QWidget \*const parent)
- QAbstractItemDelegate \* **asDelegate** () override
 

*Returns the delegate, typically, the derived class.*
- void **drawAspectRatio** (QPainter \*p, const QRect &dimsRect, const QSize &dims) const
- void **drawColorLabelLine** (QPainter \*p, const QRect &pixRect, int colorId) const
- void **drawComments** (QPainter \*p, const QRect &commentsRect, const QString &comments) const
- void **drawCreationDate** (QPainter \*p, const QRect &dateRect, const QDateTime &date) const
- void **drawFileSize** (QPainter \*p, const QRect &r, qlonglong bytes) const
- void **drawFocusRect** (QPainter \*p, const QStyleOptionViewItem &option, bool isSelected) const
- void **drawGeolocationIndicator** (QPainter \*p, const QRect &r) const
- void **drawGroupIndicator** (QPainter \*p, const QRect &r, int numberOfGroupedImages, bool open) const
- void **drawImageFormat** (QPainter \*p, const QRect &r, const QString &f, bool drawTop) const
- void **drawImageSize** (QPainter \*p, const QRect &dimsRect, const QSize &dims) const
- void **drawModificationDate** (QPainter \*p, const QRect &dateRect, const QDateTime &date) const
- void **drawMouseOverRect** (QPainter \*p, const QStyleOptionViewItem &option) const
- void **drawName** (QPainter \*p, const QRect &nameRect, const QString &name) const
- void **drawPanelSidelcon** (QPainter \*p, bool left, bool right) const
- void **drawPickLabelIcon** (QPainter \*p, const QRect &r, int pickLabel) const
- void **drawRating** (QPainter \*p, const QModelIndex &index, const QRect &ratingRect, int rating, bool isSelected) const
- void **drawSpecialInfo** (QPainter \*p, const QRect &r, const QString &text) const
- void **drawTags** (QPainter \*p, const QRect &r, const QString &tagsString, bool isSelected) const
- QRect **drawThumbnail** (QPainter \*p, const QRect &thumbRect, const QPixmap &background, const QPixmap &thumbnail, bool isGrouped) const
 

*Use the tool methods for painting in subclasses.*
- void **drawTitle** (QPainter \*p, const QRect &titleRect, const QString &title) const
- void **prepareBackground** ()
- void **prepareFonts** ()
- void **prepareMetrics** (int maxWidth)
- void **prepareRatingPxmmaps** (bool composeOverBackground=true)
- QPixmap **ratingPixmap** (int rating, bool selected) const
 

*Returns the relevant pixmap from the cached rating pxmaps.*



## Protected Member Functions inherited from [Digikam::DItemDelegate](#)

- QString **squeezedTextCached** (QPainter \*const p, int width, const QString &text) const
- QPixmap **thumbnailBorderPixmap** (const QSize &pixSize, bool isGrouped=false) const

## Protected Member Functions inherited from [Digikam::ItemDelegateOverlayContainer](#)

- virtual void **drawOverlays** (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index) const
- virtual void **overlayDestroyed** (QObject \*o)

*Declare as slot in the derived class calling this method.*

## Additional Inherited Members

## Signals inherited from [Digikam::ItemViewDelegate](#)

- void **hideNotification** ()
- void **requestNotification** (const QModelIndex &index, const QString &message)

## Signals inherited from [Digikam::DItemDelegate](#)

- void **gridSizeChanged** (const QSize &newSize)
- void **visualChange** ()

## Protected Attributes inherited from [Digikam::ItemViewDelegate](#)

- ItemViewDelegatePrivate \*const **d\_ptr** = nullptr

## Protected Attributes inherited from [Digikam::ItemDelegateOverlayContainer](#)

- QList< [ItemDelegateOverlay](#) \* > **m\_overlays**

## 9.801.1 Member Function Documentation

### 9.801.1.1 acceptsActivation()

```
bool Digikam::ItemDelegate::acceptsActivation (
    const QPoint & pos,
    const QRect & visualRect,
    const QModelIndex & index,
    QRect * activationRect = nullptr ) const [override], [virtual]
```

Implements [Digikam::DItemDelegate](#).

### 9.801.1.2 acceptsToolTip()

```
bool Digikam::ItemDelegate::acceptsToolTip (
    const QPoint & pos,
    const QRect & visualRect,
    const QModelIndex & index,
    QRect * tooltipRect = nullptr ) const [override], [virtual]
```

Implements [Digikam::DItemDelegate](#).

### 9.801.1.3 clearCaches()

```
void Digikam::ItemDelegate::clearCaches ( ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::DItemDelegate](#).

### 9.801.1.4 imageInformationRect()

```
QRect Digikam::ItemDelegate::imageInformationRect ( ) const [override], [virtual]
```

The image information is textual or graphical information, but not the pixmap. The [ratingRect\(\)](#) will e.g. typically be contained in this area.

Reimplemented from [Digikam::ItemViewDelegate](#).

### 9.801.1.5 invalidatePaintingCache()

```
void Digikam::ItemDelegate::invalidatePaintingCache ( ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ItemViewDelegate](#).

### 9.801.1.6 pixmapForDrag()

```
QPixmap Digikam::ItemDelegate::pixmapForDrag (
    const QStyleOptionViewItem & option,
    const QList< QModelIndex > & indexes ) const [override], [virtual]
```

Implements [Digikam::DItemDelegate](#).

### 9.801.1.7 pixmapRect()

```
QRect Digikam::ItemDelegate::pixmapRect ( ) const [override], [virtual]
```

Reimplemented from [Digikam::ItemViewDelegate](#).

### 9.801.1.8 setDefaultViewOptions()

```
void Digikam::ItemDelegate::setDefaultViewOptions (
    const QStyleOptionViewItem & option ) [override], [virtual]
```

option.rect shall be the viewport rectangle. Call on resize, font change.

Implements [Digikam::DItemDelegate](#).

Reimplemented in [Digikam::ItemThumbnailDelegate](#).

### 9.801.1.9 setSpacing()

```
void Digikam::ItemDelegate::setSpacing (
    int spacing ) [override], [virtual]
```

Implements [Digikam::DItemDelegate](#).

### 9.801.1.10 updateContentWidth()

```
void Digikam::ItemDelegate::updateContentWidth ( ) [protected], [virtual]
```

This is the maximum width of all content rectangles, typically excluding margins on both sides.

Reimplemented in [Digikam::ItemThumbnailDelegate](#).

### 9.801.1.11 updateRects()

```
virtual void Digikam::ItemDelegate::updateRects ( ) [protected], [pure virtual]
```

The paint() method operates depending on these rects.

Implemented in [Digikam::DigikamItemDelegate](#), [Digikam::ItemFaceDelegate](#), and [Digikam::ItemThumbnailDelegate](#).

### 9.801.1.12 updateSizeRectsAndPixmapes()

```
void Digikam::ItemDelegate::updateSizeRectsAndPixmapes ( ) [override], [protected], [virtual]
```

Implements [Digikam::ItemViewDelegate](#).



## Protected Member Functions

- `QList< QModelIndex > affectedIndexes (const QModelIndex &index) const`
- `bool affectsMultiple (const QModelIndex &index) const`

*For the context that an overlay can affect multiple items: Assuming the currently overlaid index is given.*

- `int numberOfAffectedIndexes (const QModelIndex &index) const`
- `bool viewHasMultiSelection () const`

*Utility method.*

## Protected Attributes

- `QAbstractItemDelegate * m_delegate = nullptr`
- `QAbstractItemView * m_view = nullptr`

## 9.802.1 Member Function Documentation

### 9.802.1.1 affectsMultiple()

```
bool Digikam::ItemDelegateOverlay::affectsMultiple (
    const QModelIndex & index ) const [protected]
```

Will an operation affect only the single item, or multiple? If multiple, retrieve the affected selection.

### 9.802.1.2 mouseMoved()

```
void Digikam::ItemDelegateOverlay::mouseMoved (
    QMouseEvent * e,
    const QRect & visualRect,
    const QModelIndex & index ) [virtual]
```

For all other events, connect to the view's signals. There are a few signals specifically for overlays and all `QAbstractItemView` standard signals.

### 9.802.1.3 setActive()

```
void Digikam::ItemDelegateOverlay::setActive (
    bool active ) [virtual]
```

[Setup](#) your connections to view and delegate here. You will be disconnected automatically on removal.

Reimplemented in [Digikam::FaceRejectionOverlay](#), [Digikam::ItemCoordinatesOverlay](#), [Digikam::ItemFullScreenOverlay](#), [Digikam::ItemRotateOverlay](#), [Digikam::ItemSelectionOverlay](#), [Digikam::ShowHideVersionsOverlay](#), [Digikam::ActionVersionsOverlay](#), [Digikam::AbstractWidgetDelegateOverlay](#), [Digikam::HoverButtonDelegateOverlay](#), [Digikam::PersistentWidgetDelegateOverlay](#), [ShowFoto::ShowfotoCoordinatesOverlay](#), [Digikam::ImportCoordinatesOverlay](#), [Digikam::ImportLockOverlay](#), [Digikam::ImportDownloadOverlay](#), [Digikam::ImportRotateOverlay](#), [Digikam::AssignNameOverlay](#), [Digikam::GroupIndicatorOverlay](#), [Digikam::ItemRatingOverlay](#), [Digikam::TagsLineEditOverlay](#), and [Digikam::ImportRatingOverlay](#).

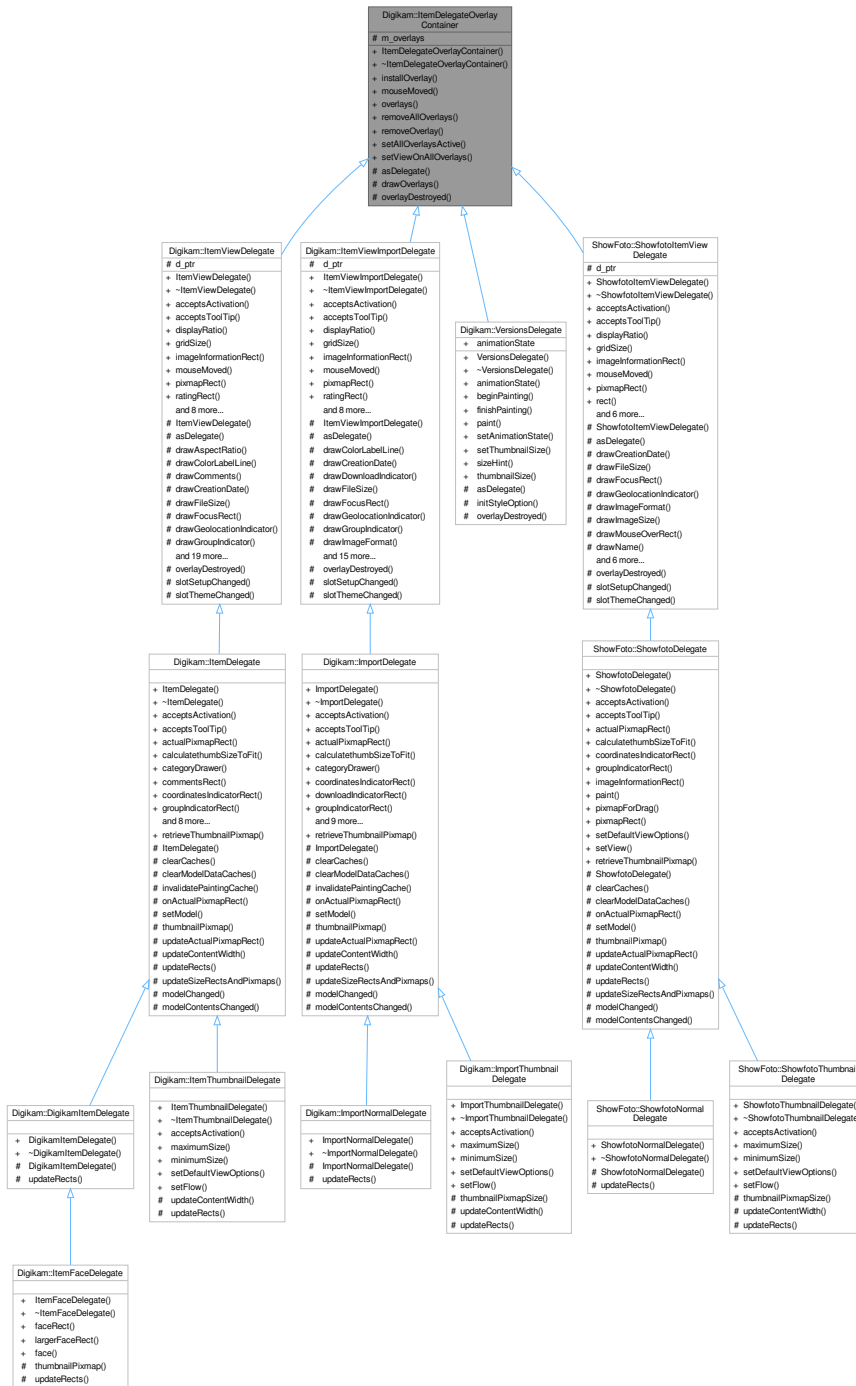
#### 9.802.1.4 visualChange

```
void Digikam::ItemDelegateOverlay::visualChange ( ) [protected], [virtual], [slot]
```

Reimplemented in [Digikam::AssignNameOverlay](#), [Digikam::GroupIndicatorOverlay](#), [Digikam::ItemCoordinatesOverlay](#), [Digikam::ItemRatingOverlay](#), [Digikam::TagsLineEditOverlay](#), [Digikam::HoverButtonDelegateOverlay](#), [ShowFoto::ShowfotoCoordinatesOverlay](#), [Digikam::ImportCoordinatesOverlay](#), [Digikam::ImportLockOverlay](#), [Digikam::ImportDownloadOverlay](#), and [Digikam::ImportRatingOverlay](#).

# 9.803 Digikam::ItemDelegateOverlayContainer Class Reference

Inheritance diagram for Digikam::ItemDelegateOverlayContainer:



## Public Member Functions

- `ItemDelegateOverlayContainer()`=default

*This is a sample implementation for delegate management methods, to be inherited by a delegate.*

- void **installOverlay** ([ItemDelegateOverlay](#) \*overlay)
- void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)
- QList< [ItemDelegateOverlay](#) \* > **overlays** () const
- void **removeAllOverlays** ()
- void **removeOverlay** ([ItemDelegateOverlay](#) \*overlay)
- void **setAllOverlaysActive** (bool active)
- void **setViewOnAllOverlays** (QAbstractItemView \*view)

### Protected Member Functions

- virtual QAbstractItemDelegate \* **asDelegate** ()=0  
*Returns the delegate, typically, the derived class.*
- virtual void **drawOverlays** (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index) const
- virtual void **overlayDestroyed** (QObject \*o)  
*Declare as slot in the derived class calling this method.*

### Protected Attributes

- QList< [ItemDelegateOverlay](#) \* > **m\_overlays**

## 9.803.1 Constructor & Destructor Documentation

### 9.803.1.1 ItemDelegateOverlayContainer()

```
Digikam::ItemDelegateOverlayContainer::ItemDelegateOverlayContainer ( ) [default]
```

Does not inherit QObject, the delegate already does.

## 9.803.2 Member Function Documentation

### 9.803.2.1 asDelegate()

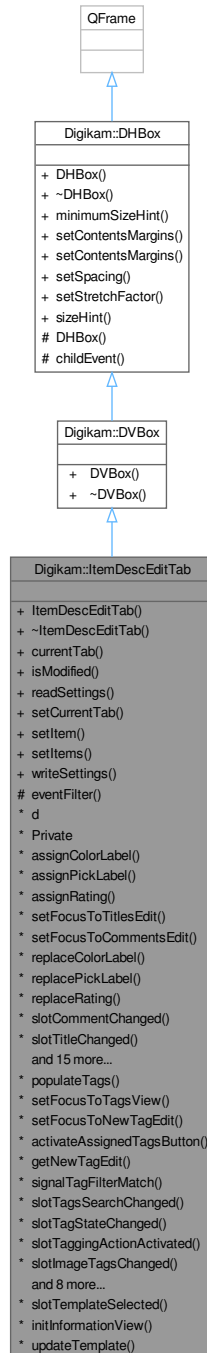
```
virtual QAbstractItemDelegate * Digikam::ItemDelegateOverlayContainer::asDelegate ( ) [protected],  
[pure virtual]
```

Implemented in [Digikam::VersionsDelegate](#), [Digikam::ItemViewDelegate](#), [ShowFoto::ShowfotoItemViewDelegate](#), and [Digikam::ItemViewImportDelegate](#).



## 9.804 Digikam::ItemDescEditTab Class Reference

Inheritance diagram for Digikam::ItemDescEditTab:



### Public Types

- enum `DescEditTab` { `DESCRIPTIONS = 0` , `TAGS` , `INFOS` }

## Signals

- void **signalAskToApplyChanges** (const QList< [ItemInfo](#) > &infos, [DisjointMetadata](#) \*hub)
- void **signalNextItem** ()
- void **signalPrevItem** ()
- void **signalProgressFinished** ()
- void **signalProgressMessageChanged** (const QString &actionDescription)
- void **signalProgressValueChanged** (float percent)
- void **signalRightSideBarBusy** (bool busy)

## Public Member Functions

- **ItemDescEditTab** (QWidget \*const parent)
- int **currentTab** () const
- bool **isModified** () const
- void **readSettings** (KConfigGroup &group)
- void **setCurrentTab** (int)
- void **setItem** (const [ItemInfo](#) &info=[ItemInfo](#)())
- void **setItems** (const [ItemInfoList](#) &infos)
- void **writeSettings** (KConfigGroup &group)

## Public Member Functions inherited from [Digikam::DVBox](#)

- **DVBox** (QWidget \*const parent=nullptr)

## Public Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (QWidget \*const parent=nullptr)
- QSize **minimumSizeHint** () const override
- void **setContentsMargins** (const QMargins &margins)
- void **setContentsMargins** (int left, int top, int right, int bottom)
- void **setSpacing** (int space)
- void **setStretchFactor** (QWidget \*const widget, int stretch)
- QSize **sizeHint** () const override

## Protected Member Functions

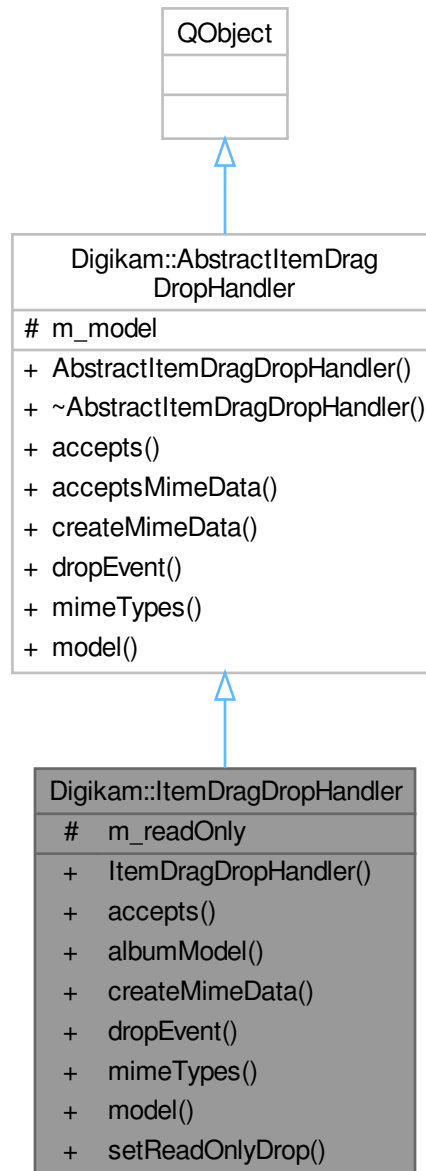
- bool **eventFilter** (QObject \*o, QEvent \*e) override

## Protected Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (bool vertical, QWidget \*const parent)
- void **childEvent** (QChildEvent \*e) override
  
- class **Private**
  
- void **assignColorLabel** (int colorId)  
*Description view methods (itemdesceditab\_descview.cpp)*
- void **assignPickLabel** (int pickId)
- void **assignRating** (int rating)
- void **setFocusToTitlesEdit** ()
- void **setFocusToCommentsEdit** ()
- void **replaceColorLabel** (int colorId)
- void **replacePickLabel** (int pickId)
- void **replaceRating** (int rating)
  
- void **populateTags** ()  
*Tags view methods (itemdesceditab\_tagview.cpp)*
- void **setFocusToTagsView** ()
- void **setFocusToNewTagEdit** ()
- void **activateAssignedTagsButton** ()
- [AddTagsLineEdit](#) \* **getNewTagEdit** () const
- void **signalTagFilterMatch** (bool)

## 9.805 Digikam::ItemDragDropHandler Class Reference

Inheritance diagram for Digikam::ItemDragDropHandler:



### Signals

- void **addToGroup** (const [ItemInfo](#) &pick, const QList< [ItemInfo](#) > &infos)
- void **assignTags** (const QList< [ItemInfo](#) > &list, const QList< int > &tagIDs)
- void **dragDropSort** (const [ItemInfo](#) &pick, const QList< [ItemInfo](#) > &infos)
- void **itemInfosDropped** (const QList< [ItemInfo](#) > &infos)
- void **urlsDropped** (const QList< QUrl > &urls)

## Public Member Functions

- **ItemDragDropHandler** ([ItemModel](#) \*const model)
- Qt::DropAction **accepts** (const QDropEvent \*e, const QModelIndex &dropIndex) override  
*Returns if the given mime data is accepted for drop on dropIndex.*
- [ItemAlbumModel](#) \* **albumModel** () const
- QMimeData \* **createMimeData** (const QList< QModelIndex > &) override  
*Create a mime data object for starting a drag from the given Albums.*
- bool **dropEvent** (QAbstractItemView \*view, const QDropEvent \*e, const QModelIndex &droppedOn) override  
*Gives the view and the occurring drop event.*
- QStringList **mimeTypes** () const override  
*Returns the supported mime types.*
- [ItemModel](#) \* **model** () const
- void **setReadOnlyDrop** (bool readOnly)  
*Enables a mode in which dropping will never start an operation which copies or moves files on disk.*

## Public Member Functions inherited from [Digikam::AbstractItemDragDropHandler](#)

- **AbstractItemDragDropHandler** (QAbstractItemModel \*const model)
- virtual bool **acceptsMimeData** (const QMimeData \*data)  
*Returns if the given mime data can be handled.*
- QAbstractItemModel \* **model** () const

## Protected Attributes

- bool **m\_readOnly** = false

## Protected Attributes inherited from [Digikam::AbstractItemDragDropHandler](#)

- QAbstractItemModel \* **m\_model** = nullptr

## 9.805.1 Member Function Documentation

### 9.805.1.1 accepts()

```
Qt::DropAction Digikam::ItemDragDropHandler::accepts (
    const QDropEvent * e,
    const QModelIndex & dropIndex ) [override], [virtual]
```

Returns the proposed action, or Qt::IgnoreAction if not accepted.

Reimplemented from [Digikam::AbstractItemDragDropHandler](#).

### 9.805.1.2 createMimeData()

```
QMimeData * Digikam::ItemDragDropHandler::createMimeData (
    const QList< QModelIndex > & ) [override], [virtual]
```

Reimplemented from [Digikam::AbstractItemDragDropHandler](#).

### 9.805.1.3 dropEvent()

```
bool Digikam::ItemDragDropHandler::dropEvent (
    QAbstractItemView * view,
    const QDropEvent * e,
    const QModelIndex & droppedOn ) [override], [virtual]
```

The index is the index where the drop was dropped on. It may be invalid (dropped on decoration, viewport) Returns true if the event is to be accepted.

Reimplemented from [Digikam::AbstractItemDragDropHandler](#).

### 9.805.1.4 mimeTypees()

```
QStringList Digikam::ItemDragDropHandler::mimeTypees ( ) const [override], [virtual]
```

Called by the default implementation of model's [mimeTypees\(\)](#).

Reimplemented from [Digikam::AbstractItemDragDropHandler](#).

### 9.805.1.5 setReadOnlyDrop()

```
void Digikam::ItemDragDropHandler::setReadOnlyDrop (
    bool readOnly )
```

Only the signals are emitted.

## 9.806 Digikam::ItemExtendedProperties Class Reference

### Public Member Functions

- **ItemExtendedProperties** ()=default  
*Create a null [ItemExtendedProperties](#) object.*
- **ItemExtendedProperties** (qulonglong imageid)
- QString [intellectualGenre](#) ()  
*Return the Intellectual Genre.*
- QString [jobId](#) ()  
*Returns the Job ID.*
- [IptcCoreLocationInfo](#) [location](#) ()  
*Return the IPTC Core Location.*
- void **removeIntellectualGenre** ()
- void **removeJobId** ()
- void **removeLocation** ()
- void **removeScene** ()
- void **removeSimilarityTo** (const qulonglong imageid)
- void **removeSubjectCode** ()
- QStringList [scene](#) ()  
*Returns the Scene.*
- void **setIntellectualGenre** (const QString &[intellectualGenre](#))
- void **setJobId** (const QString &[jobId](#))
- void **setLocation** (const [IptcCoreLocationInfo](#) &[location](#))
- void **setScene** (const QStringList &[scene](#))
- void **setSimilarityTo** (const qulonglong imageid, const double value)
- void **setSubjectCode** (const QStringList &[subjectCode](#))
- double [similarityTo](#) (const qulonglong imageid)  
*Returns the similarity.*
- QStringList [subjectCode](#) ()  
*Returns the Subject Code.*

## Protected Member Functions

- `QStringList readFakeListProperty` (const `QString` &property)
- `QString readProperty` (const `QString` &property)
- `void removeProperty` (const `QString` &property)
- `void setFakeListProperty` (const `QString` &property, const `QStringList` &value)
- `void setProperty` (const `QString` &property, const `QString` &value)

## Protected Attributes

- `qlonglong m_id = 0`

## 9.806.1 Member Function Documentation

### 9.806.1.1 intellectualGenre()

```
QString Digikam::ItemExtendedProperties::intellectualGenre ( )
```

This is Photoshop Object Attribute Reference. “ Describes the nature, intellectual or journalistic characteristic of a news object, not specifically its content. Note / Examples: Journalistic genres: actuality, interview, background, feature, summary, wrapup News category related genres: daybook, obituary, press release, transcript It is advised to use terms from a controlled vocabulary.”

### 9.806.1.2 jobId()

```
QString Digikam::ItemExtendedProperties::jobId ( )
```

This is Photoshop Transmission Reference. This is IPTC Original Transmission Reference “ Number or identifier for the purpose of improved workflow handling. This ID should be added by the creator or provider for transmission and routing purposes only and should have no significance for archiving.”

### 9.806.1.3 location()

```
IptcCoreLocationInfo Digikam::ItemExtendedProperties::location ( )
```

This includes Country, Country Code, City, Location and ProvinceState. This includes IPTC Country Name, Country Code, City, SubLocation and ProvinceState.

### 9.806.1.4 scene()

```
QStringList Digikam::ItemExtendedProperties::scene ( )
```

“ Describes the scene of a photo content. Specifies one ore more terms from the IPTC ‘Scene-NewsCodes’. Each Scene is represented as a string of 6 digits in an unordered list.”

**9.806.1.5 similarityTo()**

```
double Digikam::ItemExtendedProperties::similarityTo (
    const qlonglong imageId )
```

of the image to the given image.

**9.806.1.6 subjectCode()**

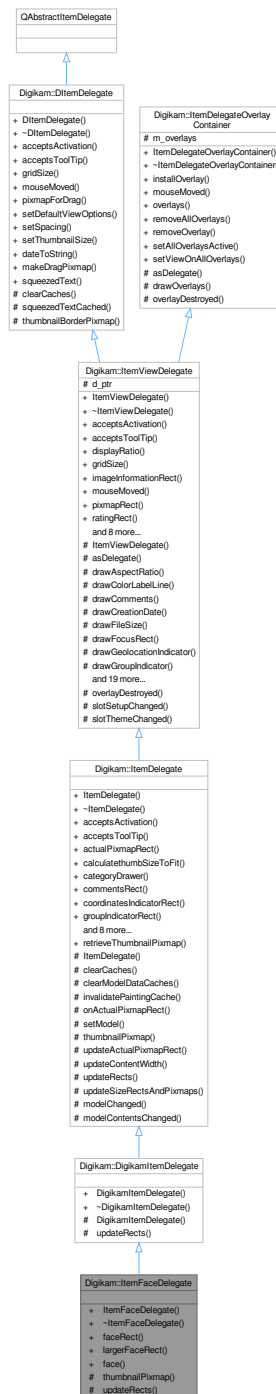
```
QStringList Digikam::ItemExtendedProperties::subjectCode ( )
```

This is IPTC Subject Reference. “ Specifies one or more Subjects from the IPTC ‘Subject-NewsCodes’ taxonomy to categorize the content. Each Subject is represented as a string of 8 digits in an unordered list. Note: Only Subjects from a controlled vocabulary should be used in this metadata element, free text has to be put into the Keyword element. More about IPTC Subject-NewsCodes at [www.newscodes.org](http://www.newscodes.org).”



## 9.807 Digikam::ItemFaceDelegate Class Reference

Inheritance diagram for Digikam::ItemFaceDelegate:



### Public Member Functions

- **ItemFaceDelegate** ([ItemCategorizedView](#) \*const parent)
- **QRect faceRect** (const QModelIndex &index) const
- **QRect largerFaceRect** (const QModelIndex &index) const

## Public Member Functions inherited from [Digikam::DigikamItemDelegate](#)

- **DigikamItemDelegate** ([ItemCategorizedView](#) \*const parent)

## Public Member Functions inherited from [Digikam::ItemDelegate](#)

- **ItemDelegate** (QWidget \*const parent)
- bool [acceptsActivation](#) (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*activationRect=nullptr) const override
- bool [acceptsToolTip](#) (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*tooltipRect=nullptr) const override
 

*These methods take four parameters: The position on viewport, the rect on viewport, the index, and optionally a parameter into which, if the return value is true, a rectangle can be written for which the return value will be true as well.*
- QRect **actualPixmapRect** (const QModelIndex &index) const
- int **calculatethumbSizeToFit** (int ws)
- [ItemCategoryDrawer](#) \* **categoryDrawer** () const
- QRect **commentsRect** () const
- QRect **coordinatesIndicatorRect** () const
- QRect **groupIndicatorRect** () const
- QRect [imageInformationRect](#) () const override
 

*Returns the area where the image information is drawn, or null if empty / not supported.*
- void **paint** (QPainter \*painter, const QStyleOptionViewItem &option, const QModelIndex &index) const override
- QPixmap [pixmapForDrag](#) (const QStyleOptionViewItem &option, const QList< QModelIndex > &indexes) const override
- QRect [pixmapRect](#) () const override
 

*Returns the area where the pixmap is drawn, or null if not supported.*
- void [setDefaultViewOptions](#) (const QStyleOptionViewItem &option) override
 

*Style option with standard values to use for cached rendering.*
- void [setSpacing](#) (int spacing) override
- void **setView** ([ItemCategorizedView](#) \*view)
- QRect **tagsRect** () const

## Public Member Functions inherited from [Digikam::ItemViewDelegate](#)

- **ItemViewDelegate** (QWidget \*const parent)
- bool [acceptsActivation](#) (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*activationRect=nullptr) const override
- bool [acceptsToolTip](#) (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*tooltipRect=nullptr) const override
 

*These methods take four parameters: The position on viewport, the rect on viewport, the index, and optionally a parameter into which, if the return value is true, a rectangle can be written for which the return value will be true as well.*
- double **displayRatio** () const
- QSize [gridSize](#) () const override
 

*Returns the gridsize to be set by the view.*
- void [mouseMoved](#) (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index) override
- virtual QRect **ratingRect** () const
 

*Returns the rectangle where the rating is drawn, or a null rectangle if not supported.*
- QRect **rect** () const
- void [setDefaultViewOptions](#) (const QStyleOptionViewItem &option) override

*Style option with standard values to use for cached rendering.*

- void [setRatingEdited](#) (const QModelIndex &index)  
*Can be used to temporarily disable drawing of the rating.*
- void [setSpacing](#) (int spacing) override
- void [setThumbnailSize](#) (const [ThumbnailSize](#) &thumbSize) override  
*You must set these options from the view.*
- QSize **sizeHint** (const QStyleOptionViewItem &option, const QModelIndex &index) const override
- int **spacing** () const
- [ThumbnailSize](#) **thumbnailSize** () const

## Public Member Functions inherited from [Digikam::DItemDelegate](#)

- [DItemDelegate](#) (QObject \*const parent=nullptr)

## Public Member Functions inherited from [Digikam::ItemDelegateOverlayContainer](#)

- [ItemDelegateOverlayContainer](#) ()=default  
*This is a sample implementation for delegate management methods, to be inherited by a delegate.*
- void **installOverlay** ([ItemDelegateOverlay](#) \*overlay)
- void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)
- QList< [ItemDelegateOverlay](#) \* > **overlays** () const
- void **removeAllOverlays** ()
- void **removeOverlay** ([ItemDelegateOverlay](#) \*overlay)
- void **setAllOverlaysActive** (bool active)
- void **setViewOnAllOverlays** (QAbstractItemView \*view)

## Static Public Member Functions

- static [FaceTagsIface](#) **face** (const QModelIndex &index)

## Static Public Member Functions inherited from [Digikam::ItemDelegate](#)

- static QPixmap **retrieveThumbnailPixmap** (const QModelIndex &index, int thumbnailSize)  
*Retrieve the thumbnail pixmap in given size for the [ItemModel::ThumbnailRole](#) for the given index from the given index, which must adhere to [ItemThumbnailModel](#) semantics.*

## Static Public Member Functions inherited from [Digikam::DItemDelegate](#)

- static QString **dateToString** (const QDateTime &datetime)
- static QPixmap **makeDragPixmap** (const QStyleOptionViewItem &option, const QList< QModelIndex > &indexes, double displayRatio, const QPixmap &suggestedPixmap=QPixmap())
- static QString **squeezedText** (const QFontMetrics &fm, int width, const QString &text)

## Protected Member Functions

- QPixmap [thumbnailPixmap](#) (const QModelIndex &index) const override
- void [updateRects](#) () override  
*In a subclass, you need to implement this method to set up the rects for drawing.*

## Protected Member Functions inherited from [Digikam::DigikamItemDelegate](#)

- **DigikamItemDelegate** (DigikamItemDelegatePrivate &dd, [ItemCategorizedView](#) \*parent)

## Protected Member Functions inherited from [Digikam::ItemDelegate](#)

- **ItemDelegate** (ItemDelegate::ItemDelegatePrivate &dd, QWidget \*const parent)
- void [clearCaches](#) () override
- virtual void **clearModelDataCaches** ()
  - Reimplement to clear caches based on model indexes (hash on row number etc.) Change signals are listened to this is called whenever such properties become invalid.*
- void [invalidatePaintingCache](#) () override
- bool **onActualPixmapRect** (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*actualRect) const
- void **setModel** (QAbstractItemModel \*model)
- void **updateActualPixmapRect** (const QModelIndex &index, const QRect &rect)
- virtual void [updateContentWidth](#) ()
  - Reimplement this to set contentWidth.*
- void [updateSizeRectsAndPxmmaps](#) () override

## Protected Member Functions inherited from [Digikam::ItemViewDelegate](#)

- **ItemViewDelegate** (ItemViewDelegatePrivate &dd, QWidget \*const parent)
- QAbstractItemDelegate \* [asDelegate](#) () override
  - Returns the delegate, typically, the derived class.*
- void **drawAspectRatio** (QPainter \*p, const QRect &dimsRect, const QSize &dims) const
- void **drawColorLabelLine** (QPainter \*p, const QRect &pixRect, int colorId) const
- void **drawComments** (QPainter \*p, const QRect &commentsRect, const QString &comments) const
- void **drawCreationDate** (QPainter \*p, const QRect &dateRect, const QDateTime &date) const
- void **drawFileSize** (QPainter \*p, const QRect &r, qlonglong bytes) const
- void **drawFocusRect** (QPainter \*p, const QStyleOptionViewItem &option, bool isSelected) const
- void **drawGeolocationIndicator** (QPainter \*p, const QRect &r) const
- void **drawGroupIndicator** (QPainter \*p, const QRect &r, int numberOfGroupedImages, bool open) const
- void **drawImageFormat** (QPainter \*p, const QRect &r, const QString &f, bool drawTop) const
- void **drawImageSize** (QPainter \*p, const QRect &dimsRect, const QSize &dims) const
- void **drawModificationDate** (QPainter \*p, const QRect &dateRect, const QDateTime &date) const
- void **drawMouseOverRect** (QPainter \*p, const QStyleOptionViewItem &option) const
- void **drawName** (QPainter \*p, const QRect &nameRect, const QString &name) const
- void **drawPanelSidelcon** (QPainter \*p, bool left, bool right) const
- void **drawPickLabelIcon** (QPainter \*p, const QRect &r, int pickLabel) const
- void **drawRating** (QPainter \*p, const QModelIndex &index, const QRect &ratingRect, int rating, bool isSelected) const
- void **drawSpecialInfo** (QPainter \*p, const QRect &r, const QString &text) const
- void **drawTags** (QPainter \*p, const QRect &r, const QString &tagsString, bool isSelected) const
- QRect **drawThumbnail** (QPainter \*p, const QRect &thumbRect, const QPixmap &background, const QPixmap &thumbnail, bool isGrouped) const
  - Use the tool methods for painting in subclasses.*
- void **drawTitle** (QPainter \*p, const QRect &titleRect, const QString &title) const
- void **prepareBackground** ()
- void **prepareFonts** ()
- void **prepareMetrics** (int maxWidth)
- void **prepareRatingPxmmaps** (bool composeOverBackground=true)
- QPixmap **ratingPixmap** (int rating, bool selected) const
  - Returns the relevant pixmap from the cached rating pxmaps.*

### Protected Member Functions inherited from [Digikam::DItemDelegate](#)

- QString **squeezedTextCached** (QPainter \*const p, int width, const QString &text) const
- QPixmap **thumbnailBorderPixmap** (const QSize &pixSize, bool isGrouped=false) const

### Protected Member Functions inherited from [Digikam::ItemDelegateOverlayContainer](#)

- virtual void **drawOverlays** (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index) const
- virtual void **overlayDestroyed** (QObject \*o)

*Declare as slot in the derived class calling this method.*

### Additional Inherited Members

### Signals inherited from [Digikam::ItemViewDelegate](#)

- void **hideNotification** ()
- void **requestNotification** (const QModelIndex &index, const QString &message)

### Signals inherited from [Digikam::DItemDelegate](#)

- void **gridSizeChanged** (const QSize &newSize)
- void **visualChange** ()

### Protected Slots inherited from [Digikam::ItemDelegate](#)

- void **modelChanged** ()
- void **modelContentsChanged** ()

### Protected Slots inherited from [Digikam::ItemViewDelegate](#)

- void **overlayDestroyed** (QObject \*o) override
- void **slotSetupChanged** ()
- void **slotThemeChanged** ()

### Protected Attributes inherited from [Digikam::ItemViewDelegate](#)

- ItemViewDelegatePrivate \*const **d\_ptr** = nullptr

### Protected Attributes inherited from [Digikam::ItemDelegateOverlayContainer](#)

- QList< [ItemDelegateOverlay](#) \* > **m\_overlays**

## 9.807.1 Member Function Documentation

### 9.807.1.1 thumbnailPixmap()

```
QPixmap Digikam::ItemFaceDelegate::thumbnailPixmap (
    const QModelIndex & index ) const [override], [protected], [virtual]
```

Reimplemented from [Digikam::ItemDelegate](#).

### 9.807.1.2 updateRects()

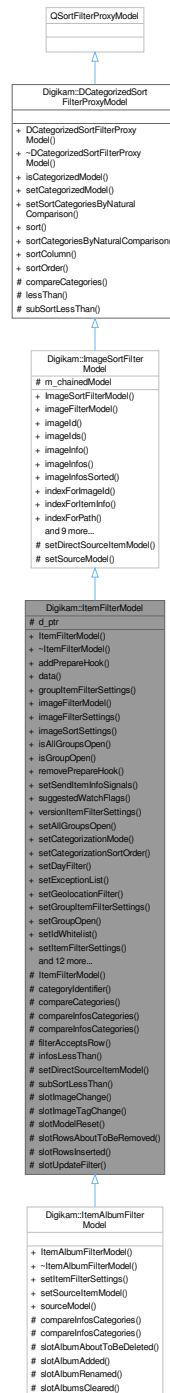
```
void Digikam::ItemFaceDelegate::updateRects ( ) [override], [protected], [virtual]
```

The paint() method operates depending on these rects.

Reimplemented from [Digikam::DigikamItemDelegate](#).

## 9.808 Digikam::ItemFilterModel Class Reference

Inheritance diagram for Digikam::ItemFilterModel:



### Public Types

- enum `ItemFilterModelRoles` {
  - `CategorizationModeRole` = `ItemModel::FilterModelRoles + 1` , `SortOrderRole` = `ItemModel::FilterModelRoles`

```
+ 2 , CategoryAlbumIdRole = ItemModel::FilterModelRoles + 3 , CategoryFormatRole = ItemModel::FilterModelRoles + 4 ,
CategoryDateRole = ItemModel::FilterModelRoles + 5 , CategoryFaceRole = ItemModel::FilterModelRoles + 6 ,
GroupsOpenRole = ItemModel::FilterModelRoles + 7 , ItemFilterModelPointerRole = ItemModel::FilterModelRoles + 50 }
```

## Public Types inherited from [Digikam::DCategorizedSortFilterProxyModel](#)

- enum [AdditionalRoles](#) { [CategoryDisplayRole](#) = 0x17CE990A , [CategorySortRole](#) = 0x27857E60 }

## Public Slots

- void [setAllGroupsOpen](#) (bool open)
- void [setCategorizationMode](#) ([ItemSortSettings::CategorizationMode](#) mode)
- void [setCategorizationSortOrder](#) ([ItemSortSettings::SortOrder](#) order)
- void [setDayFilter](#) (const QList< QDateTime > &days)
  - Adjust the current [ItemFilterSettings](#).*
- void [setExceptionList](#) (const QList< qlonglong > &idlist, const QString &id)
- void [setGeolocationFilter](#) (const [ItemFilterSettings::GeolocationCondition](#) &condition)
- void [setGroupItemFilterSettings](#) (const [GroupItemFilterSettings](#) &settings)
  - Changes the current version image filter settings and refilters.*
- void [setGroupOpen](#) (qlonglong group, bool open)
- void [setIdWhitelist](#) (const QList< qlonglong > &idList, const QString &id)
- virtual void [setItemFilterSettings](#) (const [ItemFilterSettings](#) &settings)
  - Changes the current image filter settings and refilters.*
- virtual void [setItemSortSettings](#) (const [ItemSortSettings](#) &settings)
  - Changes the current image sort settings and resorts.*
- void [setMimeTypeFilter](#) (int mimeTypeFilter)
- void [setRatingFilter](#) (int rating, [ItemFilterSettings::RatingCondition](#) ratingCond, bool isUnratedExcluded)
- void [setSortOrder](#) ([ItemSortSettings::SortOrder](#) order)
- void [setSortRole](#) ([ItemSortSettings::SortRole](#) role)
- void [setStringTypeNatural](#) (bool natural)
- void [setTagFilter](#) (const QList< int > &includedTags, const QList< int > &excludedTags, [ItemFilterSettings::MatchingCondition](#) matchingCond, bool showUnTagged, const QList< int > &clTagIds, const QList< int > &plTagIds)
- void [setTextFilter](#) (const [SearchTextFilterSettings](#) &settings)
- void [setUriWhitelist](#) (const QList< QUrl > &urlList, const QString &id)
- void [setVersionItemFilterSettings](#) (const [VersionItemFilterSettings](#) &settings)
  - Changes the current version image filter settings and refilters.*
- void [setVersionManagerSettings](#) (const [VersionManagerSettings](#) &settings)
- void [toggleGroupOpen](#) (qlonglong group)

## Signals

- void [filterMatches](#) (bool matches)
  - Signals that the set filter matches at least one index.*
- void [filterMatchesForText](#) (bool matchesByText)
  - Signals that the set text filter matches at least one entry.*
- void [filterSettingsChanged](#) (const [ItemFilterSettings](#) &settings)
  - Emitted when the filter settings have been changed (the model may not yet have been updated)*
- void [imageInfosAboutToBeRemoved](#) (const QList< [ItemInfo](#) > &infos)
- void [imageInfosAdded](#) (const QList< [ItemInfo](#) > &infos)
  - These signals need to be explicitly enabled with [setSendItemInfoSignals\(\)](#)*



## Public Member Functions

- **ItemFilterModel** (QObject \*const parent=nullptr)
- void **addPrepareHook** ([ItemFilterModelPrepareHook](#) \*const hook)
 

*Add a hook to get added images for preparation tasks before they are added in the model.*
- QVariant **data** (const QModelIndex &index, int role=Qt::DisplayRole) const override
- [GroupItemFilterSettings](#) **groupItemFilterSettings** () const
- [ItemFilterModel](#) \* **imageFilterModel** () const override
 

*Returns this, any chained [ItemFilterModel](#), or 0.*
- [ItemFilterSettings](#) **imageFilterSettings** () const
- [ItemSortSettings](#) **imageSortSettings** () const
- bool **isAllGroupsOpen** () const
- bool **isGroupOpen** (qulonglong group) const
 

*group is identified by the id of its group leader*
- void **removePrepareHook** ([ItemFilterModelPrepareHook](#) \*const hook)
- void **setSendItemInfoSignals** (bool sendSignals)
 

*Enables sending [imageInfosAdded](#) and [imageInfosAboutToBeRemoved](#).*
- [DatabaseFields::Set](#) **suggestedWatchFlags** () const
 

*Returns a set of [DatabaseFields](#) suggested to set as watch flags on the source [ItemModel](#).*
- [VersionItemFilterSettings](#) **versionItemFilterSettings** () const

## Public Member Functions inherited from [Digikam::ImageSortFilterModel](#)

- **ImageSortFilterModel** (QObject \*const parent=nullptr)
- qulonglong **imageId** (const QModelIndex &index) const
- QList< qulonglong > **imageIds** (const QList< QModelIndex > &indexes) const
- [ItemInfo](#) **imageInfo** (const QModelIndex &index) const
- QList< [ItemInfo](#) > **imageInfos** (const QList< QModelIndex > &indexes) const
- QList< [ItemInfo](#) > **imageInfosSorted** () const
 

*Returns a list of all image infos, sorted according to this model.*
- QModelIndex **indexForImageId** (qulonglong id) const
- QModelIndex **indexForItemInfo** (const [ItemInfo](#) &info) const
- QModelIndex **indexForPath** (const QString &filePath) const
- QModelIndex **mapFromDirectSourceToSourceItemModel** (const QModelIndex &sourceModel\_index) const
- QModelIndex **mapFromSourceItemModel** (const QModelIndex &imagemodel\_index) const
- QList< QModelIndex > **mapListFromSource** (const QList< QModelIndex > &sourceIndexes) const
- QList< QModelIndex > **mapListToSource** (const QList< QModelIndex > &indexes) const
 

*Convenience methods mapped to [ItemModel](#).*
- QModelIndex **mapToSourceItemModel** (const QModelIndex &index) const
- void **setSourceFilterModel** ([ImageSortFilterModel](#) \*const model)
- void **setSourceItemModel** ([ItemModel](#) \*const model)
- [ImageSortFilterModel](#) \* **sourceFilterModel** () const
- [ItemModel](#) \* **sourceItemModel** () const

## Public Member Functions inherited from [Digikam::DCategorizedSortFilterProxyModel](#)

- **DCategorizedSortFilterProxyModel** (QObject \*const parent=nullptr)
- bool **isCategorizedModel** () const
- void **setCategorizedModel** (bool categorizedModel)
 

*Enables or disables the categorization feature.*
- void **setSortCategoriesByNaturalComparison** (bool [sortCategoriesByNaturalComparison](#))
 

*Set if the sorting using CategorySortRole will use a natural comparison in the case that strings were returned.*
- void **sort** (int column, Qt::SortOrder order=Qt::AscendingOrder) override
 

*Overridden from QSortFilterProxyModel.*
- bool **sortCategoriesByNaturalComparison** () const
- int **sortColumn** () const
- Qt::SortOrder **sortOrder** () const

## Protected Slots

- void **slotImageChange** (const [ImageChangeset](#) &changeset)
- void **slotImageTagChange** (const [ImageTagChangeset](#) &changeset)
- void **slotModelReset** ()
- void **slotRowsAboutToBeRemoved** (const QModelIndex &parent, int start, int end)
- void **slotRowsInserted** (const QModelIndex &parent, int start, int end)
- void **slotUpdateFilter** ()

## Protected Member Functions

- **ItemFilterModel** (ItemFilterModelPrivate &dd, QObject \*const parent)
- virtual QString **categoryIdentifier** (const [ItemInfo](#) &info, const [FaceTagsIface](#) &face) const
 

*Returns a unique identifier for the category if info.*
- int **compareCategories** (const QModelIndex &left, const QModelIndex &right) const override
 

*This method compares the category of the left index with the category of the right index.*
- virtual int **compareInfosCategories** (const [ItemInfo](#) &left, const [ItemInfo](#) &right) const
 

*Reimplement to customize category sorting, Return negative if category of left < category right, Return 0 if left and right are in the same category, else return positive.*
- virtual int **compareInfosCategories** (const [ItemInfo](#) &left, const [ItemInfo](#) &right, const [FaceTagsIface](#) &leftFace, const [FaceTagsIface](#) &rightFace) const
 

*In order to be able to Categorize by Faces, it's necessary to pass in the face as well.*
- bool **filterAcceptsRow** (int source\_row, const QModelIndex &source\_parent) const override
- virtual bool **infosLessThan** (const [ItemInfo](#) &left, const [ItemInfo](#) &right) const
 

*Reimplement to customize sorting.*
- void **setDirectSourceItemModel** ([ItemModel](#) \*const model) override
 

*Reimplement if needed.*
- bool **subSortLessThan** (const QModelIndex &left, const QModelIndex &right) const override
 

*This method has a similar purpose as [lessThan\(\)](#) has on QSortFilterProxyModel.*

## Protected Member Functions inherited from [Digikam::ImageSortFilterModel](#)

- void **setSourceModel** (QAbstractItemModel \*const model) override

## Protected Member Functions inherited from [Digikam::DCategorizedSortFilterProxyModel](#)

- bool [lessThan](#) (const QModelIndex &left, const QModelIndex &right) const override  
*Overridden from [QSortFilterProxyModel](#).*

## Protected Attributes

- ItemFilterModelPrivate \*const [d\\_ptr](#) = nullptr

## Protected Attributes inherited from [Digikam::ImageSortFilterModel](#)

- [ImageSortFilterModel](#) \* [m\\_chainedModel](#) = nullptr

## 9.808.1 Member Enumeration Documentation

### 9.808.1.1 ItemFilterModelRoles

```
enum Digikam::ItemFilterModel::ItemFilterModelRoles
```

#### Enumerator

CategorizationModeRole	Returns the current categorization mode.
SortOrderRole	Returns the current sort order.
CategoryAlbumIdRole	Returns the number of items in the index category. Returns the id of the <a href="#">PAlbum</a> of the index which is used for category
CategoryFormatRole	Returns the format of the index which is used for category.
CategoryDateRole	Returns the date of the index which is used for category.
CategoryFaceRole	Returns the suggested name for the face in this index.
GroupsOpenRole	Returns true if the given image is a group leader, and the group is opened.

## 9.808.2 Member Function Documentation

### 9.808.2.1 categoryIdentifier()

```
QString Digikam::ItemFilterModel::categoryIdentifier (
    const ItemInfo & info,
    const FaceTagsIface & face ) const [protected], [virtual]
```

The string need not be for user display.

### 9.808.2.2 compareCategories()

```
int Digikam::ItemFilterModel::compareCategories (
    const QModelIndex & left,
    const QModelIndex & right ) const [override], [protected], [virtual]
```

Internally and if not reimplemented, this method will ask for `left` and `right` models for role `CategorySortRole`. In order to correctly sort categories, the `data()` method of the model should return a `qulonglong` (or numeric) value, or a `QString` object. `QString` objects will be sorted with `QString::localeAwareCompare` if `sortCategoriesByNaturalComparison()` is true.

**Note**

Please have present that: `QString(QChar(QChar::ObjectReplacementCharacter)) > QString(QChar(QChar::ReplacementCharacter)) > [ all possible strings ] > QString();`

This means that `QString()` will be sorted the first one, while `QString(QChar(QChar::ObjectReplacementCharacter))` and `QString(QChar(QChar::ReplacementCharacter))` will be sorted in last position.

**Warning**

Please note that `data()` method of the model should return always information of the same type. If you return a `QString` for an index, you should return always `QStrings` for all indexes for role `CategorySortRole` in order to correctly sort categories. You can't mix by returning a `QString` for one index, and a `qlonglong` for other.

**Note**

If you need a more complex layout, you will have to reimplement this method.

**Returns**

A negative value if the category of `left` should be placed before the category of `right`. 0 if `left` and `right` are on the same category, and a positive value if the category of `left` should be placed after the category of `right`.

Reimplemented from [Digikam::DCategorizedSortFilterProxyModel](#).

**9.808.2.3 compareInfosCategories() [1/2]**

```
int Digikam::ItemFilterModel::compareInfosCategories (
    const ItemInfo & left,
    const ItemInfo & right ) const [protected], [virtual]
```

Reimplemented in [Digikam::ItemAlbumFilterModel](#).

**9.808.2.4 compareInfosCategories() [2/2]**

```
int Digikam::ItemFilterModel::compareInfosCategories (
    const ItemInfo & left,
    const ItemInfo & right,
    const FaceTagsIface & leftFace,
    const FaceTagsIface & rightFace ) const [protected], [virtual]
```

One image may have multiple Faces in it, hence just the `ItemInfo` isn't sufficient.

Reimplemented in [Digikam::ItemAlbumFilterModel](#).

**9.808.2.5 data()**

```
QVariant Digikam::ItemFilterModel::data (
    const QModelIndex & index,
    int role = Qt::DisplayRole ) const [override]
```

Keeping track of the Face (if any) associated with this Model Index is important to allow categorization by Face.

### 9.808.2.6 filterMatchesForText

```
void Digikam::ItemFilterModel::filterMatchesForText (
    bool matchesByText ) [signal]
```

If no text filter is set, this signal is emitted with 'false' when [filterMatches\(\)](#) is emitted.

### 9.808.2.7 imageFilterModel()

```
ItemFilterModel * Digikam::ItemFilterModel::imageFilterModel ( ) const [override], [virtual]
```

Reimplemented from [Digikam::ImageSortFilterModel](#).

### 9.808.2.8 infosLessThan()

```
bool Digikam::ItemFilterModel::infosLessThan (
    const ItemInfo & left,
    const ItemInfo & right ) const [protected], [virtual]
```

Do not take categories into account here.

### 9.808.2.9 setDayFilter

```
void Digikam::ItemFilterModel::setDayFilter (
    const QList< QDateTime > & days ) [slot]
```

Equivalent to retrieving the current filter settings, adjusting the parameter and calling [setItemFilterSettings](#). Provided for convenience. It is encouraged to use [setItemFilterSettings](#) if you change more than one parameter at a time.

### 9.808.2.10 setDirectSourceItemModel()

```
void Digikam::ItemFilterModel::setDirectSourceItemModel (
    ItemModel *const model ) [override], [protected], [virtual]
```

Called only when model shall be set as (direct) sourceModel.

Reimplemented from [Digikam::ImageSortFilterModel](#).

### 9.808.2.11 setItemFilterSettings

```
void Digikam::ItemFilterModel::setItemFilterSettings (
    const ItemFilterSettings & settings ) [virtual], [slot]
```

Reimplemented in [Digikam::ItemAlbumFilterModel](#).

### 9.808.2.12 subSortLessThan()

```
bool Digikam::ItemFilterModel::subSortLessThan (
    const QModelIndex & left,
    const QModelIndex & right ) const [override], [protected], [virtual]
```

It is used for sorting items that are in the same category.

#### Returns

Returns true if the item `left` is less than the item `right` when sorting.

Reimplemented from [Digikam::DCategorizedSortFilterProxyModel](#).

### 9.808.2.13 suggestedWatchFlags()

```
DatabaseFields::Set Digikam::ItemFilterModel::suggestedWatchFlags ( ) const
```

The contained flags will be those that this model can sort or filter by.

## 9.809 Digikam::ItemFilterModelFilterer Class Reference

Inheritance diagram for Digikam::ItemFilterModelFilterer:



### Public Member Functions

- **ItemFilterModelFilterer** (ItemFilterModel::ItemFilterModelPrivate \*const d)
- void [process](#) (ItemFilterModelTodoPackage package) override

## Public Member Functions inherited from [Digikam::ItemFilterModelWorker](#)

- **ItemFilterModelWorker** (ItemFilterModel::ItemFilterModelPrivate \*const dd)
- bool **checkVersion** (const ItemFilterModelTodoPackage &package)

## Public Member Functions inherited from [Digikam::WorkerObject](#)

- [WorkerObject](#) ()  
*Deriving from a worker object allows you to execute your slots in a thread.*
- bool **connectAndSchedule** (const QObject \*sender, const char \*signal, const char \*method, Qt::ConnectionType type=Qt::AutoConnection) const  
*You must normally call [schedule\(\)](#) to ensure that the object is active when you send a signal with work data.*
- QThread::Priority **priority** () const
- void **setPriority** (QThread::Priority priority)  
*Sets the priority for this dynamic thread.*
- State **state** () const
- void **wait** ()

## Additional Inherited Members

## Public Types inherited from [Digikam::WorkerObject](#)

- enum [DeactivatingMode](#) { [FlushSignals](#) , [KeepSignals](#) , [PhaseOut](#) }
- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

## Public Slots inherited from [Digikam::ItemFilterModelWorker](#)

## Public Slots inherited from [Digikam::WorkerObject](#)

- void **deactivate** ([DeactivatingMode](#) mode=[FlushSignals](#))  
*Quits execution of this worker object.*
- void **schedule** ()  
*Starts execution of this worker object: The object is moved to a thread and an event loop started, so that queued signals will be received.*

## Signals inherited from [Digikam::ItemFilterModelWorker](#)

- void **discarded** (const ItemFilterModelTodoPackage &package)
- void **processed** (const ItemFilterModelTodoPackage &package)

## Signals inherited from [Digikam::WorkerObject](#)

- void **finished** ()
- void **started** ()



## Static Public Member Functions inherited from Digikam::WorkerObject

- static bool **connectAndSchedule** (const QObject \*sender, const char \*signal, const WorkerObject \*receiver, const char \*method, Qt::ConnectionType type=Qt::AutoConnection)
- static bool **disconnectAndSchedule** (const QObject \*sender, const char \*signal, const WorkerObject \*receiver, const char \*method)

## Protected Member Functions inherited from Digikam::WorkerObject

- virtual void **aboutToDeactivate** ()  
*Called from deactivate(), typically from a different thread than the worker thread, possibly the UI thread.*
- virtual void **aboutToQuitLoop** ()  
*Called from within thread's event loop to quit processing.*
- void **addRunnable** (WorkerObjectRunnable \*loop)
- bool **event** (QEvent \*e) override
- void **removeRunnable** (WorkerObjectRunnable \*loop)
- void **run** ()
- void **setEventLoop** (QEventLoop \*loop)
- void **shutDown** ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void **transitionToInactive** ()
- bool **transitionToRunning** ()

## Protected Attributes inherited from Digikam::ItemFilterModelWorker

- ItemFilterModel::ItemFilterModelPrivate \* **d** = nullptr

## 9.809.1 Member Function Documentation

### 9.809.1.1 process()

```
void Digikam::ItemFilterModelFilterer::process (
    ItemFilterModelTodoPackage package ) [override], [virtual]
```

Implements [Digikam::ItemFilterModelWorker](#).

## 9.810 Digikam::ItemFilterModelPrepareHook Class Reference

### Public Member Functions

- virtual void **prepare** (const QVector< ItemInfo > &infos)=0

## 9.811 Digikam::ItemFilterModelPreparer Class Reference

Inheritance diagram for Digikam::ItemFilterModelPreparer:



### Public Member Functions

- **ItemFilterModelPreparer** (ItemFilterModel::ItemFilterModelPrivate \*const d)
- void [process](#) (ItemFilterModelTodoPackage package) override

## Public Member Functions inherited from Digikam::ItemFilterModelWorker

- **ItemFilterModelWorker** (ItemFilterModel::ItemFilterModelPrivate \*const dd)
- bool **checkVersion** (const ItemFilterModelTodoPackage &package)

## Public Member Functions inherited from Digikam::WorkerObject

- **WorkerObject** ()
  - Deriving from a worker object allows you to execute your slots in a thread.*
- bool **connectAndSchedule** (const QObject \*sender, const char \*signal, const char \*method, Qt::ConnectionType type=Qt::AutoConnection) const
  - You must normally call [schedule\(\)](#) to ensure that the object is active when you send a signal with work data.*
- QThread::Priority **priority** () const
- void **setPriority** (QThread::Priority priority)
  - Sets the priority for this dynamic thread.*
- State **state** () const
- void **wait** ()

## Additional Inherited Members

## Public Types inherited from Digikam::WorkerObject

- enum **DeactivatingMode** { [FlushSignals](#) , [KeepSignals](#) , [PhaseOut](#) }
- enum **State** { [Inactive](#) , [Scheduled](#) , [Running](#) , [Deactivating](#) }

## Public Slots inherited from Digikam::ItemFilterModelWorker

## Public Slots inherited from Digikam::WorkerObject

- void **deactivate** ([DeactivatingMode](#) mode=[FlushSignals](#))
  - Quits execution of this worker object.*
- void **schedule** ()
  - Starts execution of this worker object: The object is moved to a thread and an event loop started, so that queued signals will be received.*

## Signals inherited from Digikam::ItemFilterModelWorker

- void **discarded** (const ItemFilterModelTodoPackage &package)
- void **processed** (const ItemFilterModelTodoPackage &package)

## Signals inherited from Digikam::WorkerObject

- void **finished** ()
- void **started** ()

## Static Public Member Functions inherited from [Digikam::WorkerObject](#)

- static bool **connectAndSchedule** (const QObject \*sender, const char \*signal, const [WorkerObject](#) \*receiver, const char \*method, Qt::ConnectionType type=Qt::AutoConnection)
- static bool **disconnectAndSchedule** (const QObject \*sender, const char \*signal, const [WorkerObject](#) \*receiver, const char \*method)

## Protected Member Functions inherited from [Digikam::WorkerObject](#)

- virtual void **aboutToDeactivate** ()  
*Called from [deactivate\(\)](#), typically from a different thread than the worker thread, possibly the UI thread.*
- virtual void **aboutToQuitLoop** ()  
*Called from within thread's event loop to quit processing.*
- void **addRunnable** (WorkerObjectRunnable \*loop)
- bool **event** (QEvent \*e) override
- void **removeRunnable** (WorkerObjectRunnable \*loop)
- void **run** ()
- void **setEventLoop** (QEventLoop \*loop)
- void **shutDown** ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void **transitionToInactive** ()
- bool **transitionToRunning** ()

## Protected Attributes inherited from [Digikam::ItemFilterModelWorker](#)

- ItemFilterModel::ItemFilterModelPrivate \* **d** = nullptr

### 9.811.1 Member Function Documentation

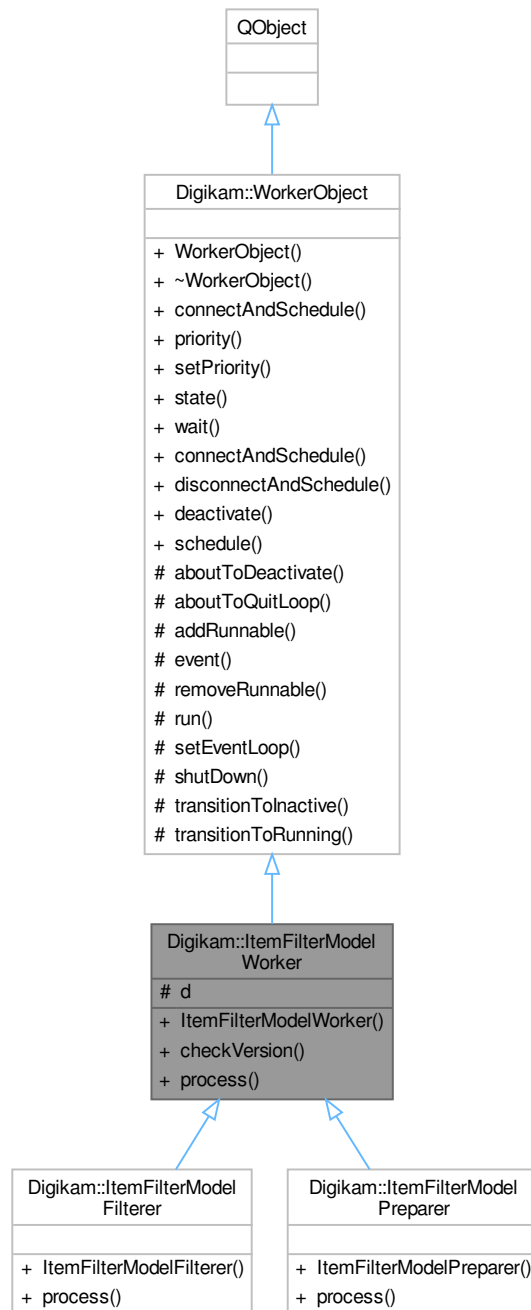
#### 9.811.1.1 process()

```
void Digikam::ItemFilterModelPreparer::process (
    ItemFilterModelTodoPackage package ) [override], [virtual]
```

Implements [Digikam::ItemFilterModelWorker](#).

## 9.812 Digikam::ItemFilterModelWorker Class Reference

Inheritance diagram for Digikam::ItemFilterModelWorker:



### Public Slots

- virtual void **process** (ItemFilterModelTodoPackage package)=0

## Public Slots inherited from [Digikam::WorkerObject](#)

- void **deactivate** ([DeactivatingMode](#) mode=[FlushSignals](#))  
*Quits execution of this worker object.*
- void **schedule** ()  
*Starts execution of this worker object: The object is moved to a thread and an event loop started, so that queued signals will be received.*

## Signals

- void **discarded** (const [ItemFilterModelTodoPackage](#) &package)
- void **processed** (const [ItemFilterModelTodoPackage](#) &package)

## Signals inherited from [Digikam::WorkerObject](#)

- void **finished** ()
- void **started** ()

## Public Member Functions

- **ItemFilterModelWorker** ([ItemFilterModel::ItemFilterModelPrivate](#) \*const dd)
- bool **checkVersion** (const [ItemFilterModelTodoPackage](#) &package)

## Public Member Functions inherited from [Digikam::WorkerObject](#)

- [WorkerObject](#) ()  
*Deriving from a worker object allows you to execute your slots in a thread.*
- bool **connectAndSchedule** (const [QObject](#) \*sender, const char \*signal, const char \*method, [Qt::](#)↔[ConnectionType](#) type=[Qt::AutoConnection](#)) const  
*You must normally call [schedule\(\)](#) to ensure that the object is active when you send a signal with work data.*
- [QThread::Priority](#) **priority** () const
- void **setPriority** ([QThread::Priority](#) priority)  
*Sets the priority for this dynamic thread.*
- State **state** () const
- void **wait** ()

## Protected Attributes

- [ItemFilterModel::ItemFilterModelPrivate](#) \* **d** = nullptr

## Additional Inherited Members

## Public Types inherited from [Digikam::WorkerObject](#)

- enum [DeactivatingMode](#) { [FlushSignals](#) , [KeepSignals](#) , [PhaseOut](#) }
- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

## Static Public Member Functions inherited from Digikam::WorkerObject

- static bool **connectAndSchedule** (const QObject \*sender, const char \*signal, const WorkerObject \*receiver, const char \*method, Qt::ConnectionType type=Qt::AutoConnection)
- static bool **disconnectAndSchedule** (const QObject \*sender, const char \*signal, const WorkerObject \*receiver, const char \*method)

## Protected Member Functions inherited from Digikam::WorkerObject

- virtual void **aboutToDeactivate** ()  
*Called from deactivate(), typically from a different thread than the worker thread, possibly the UI thread.*
- virtual void **aboutToQuitLoop** ()  
*Called from within thread's event loop to quit processing.*
- void **addRunnable** (WorkerObjectRunnable \*loop)
- bool **event** (QEvent \*e) override
- void **removeRunnable** (WorkerObjectRunnable \*loop)
- void **run** ()
- void **setEventLoop** (QEventLoop \*loop)
- void **shutDown** ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void **transitionToInactive** ()
- bool **transitionToRunning** ()

## 9.813 Digikam::ItemFilterSettings Class Reference

### Public Types

- enum **GeolocationCondition** { **GeolocationNoFilter** = 0 , **GeolocationNoCoordinates** = 1 << 1 , **GeolocationHasCoordinates** = 1 << 2 }
- Possible logical matching condition used to sort geolocation.*
- enum **MatchingCondition** { **OrCondition** , **AndCondition** }
- Possible logical matching condition used to sort tags id.*
- enum **RatingCondition** { **GreaterEqualCondition** , **EqualCondition** , **LessEqualCondition** }
- Possible conditions used to filter rating: >=, =, <=.*

### Public Member Functions

- bool **isFiltering** () const  
*Returns if images will be filtered by these criteria at all.*
- bool **isFilteringByColorLabels** () const  
*Returns if the color labels is a filter criteria.*
- bool **isFilteringByDay** () const  
*Returns if the day is a filter criteria.*
- bool **isFilteringByGeolocation** () const  
*Returns whether geolocation is a filter criteria.*
- bool **isFilteringByPickLabels** () const  
*Returns if the pick labels is a filter criteria.*
- bool **isFilteringByRating** () const

- Returns if the rating is a filter criteria.*

  - bool **isFilteringByTags** () const

*Returns if the tag is a filter criteria.*

  - bool **isFilteringByText** () const

*Returns if the text (including comment) is a filter criteria.*

  - bool **isFilteringByTypeMime** () const

*Returns if the type mime is a filter criteria.*

  - bool **matches** (const [ItemInfo](#) &info, bool \*const foundText=nullptr) const

*Returns true if the given [ItemInfo](#) matches the filter criteria.*

  - void **setAlbumNames** (const QHash< int, QString > &albumNameHash)
  - void **setDayFilter** (const QList< QDateTime > &days)
    - Date filter —
  - void **setGeolocationFilter** (const [GeolocationCondition](#) &condition)
    - Geolocation filter
  - void **setIdWhitelist** (const QList< qlonglong > &idList, const QString &id)
    - ID whitelist filter
  - void **setMimeTypeFilter** (int mimeTypeFilter)
    - Mime filter —
  - void **setRatingFilter** (int rating, [RatingCondition](#) ratingCond, bool isUnratedExcluded)
    - Rating filter —
  - void **setTagFilter** (const QList< int > &includedTags, const QList< int > &excludedTags, [MatchingCondition](#) matchingCond, bool showUnTagged, const QList< int > &clTagIds, const QList< int > &plTagIds)
    - Tags filter —
  - void **setTagNames** (const QHash< int, QString > &tagNameHash)
  - void **setTextFilter** (const [SearchTextFilterSettings](#) &settings)
    - Text filter —
  - void **setUrlWhitelist** (const QList< QUrl > &urlList, const QString &id)
    - URL whitelist filter
  - [DatabaseFields::Set](#) **watchFlags** () const
    - Change notification —

## 9.813.1 Member Function Documentation

### 9.813.1.1 matches()

```
bool Digikam::ItemFilterSettings::matches (
    const ItemInfo & info,
    bool *const foundText = nullptr ) const
```

Optionally, foundText is set to true if it matched by text search.

### 9.813.1.2 watchFlags()

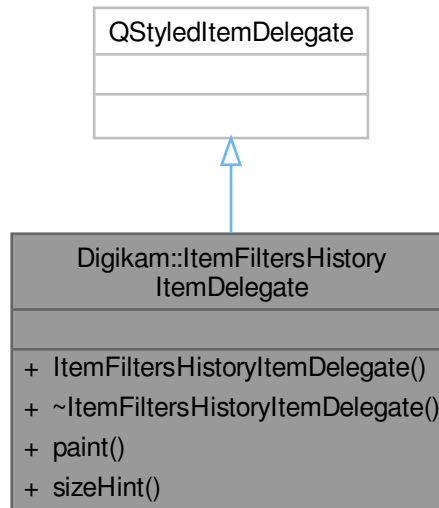
```
DatabaseFields::Set Digikam::ItemFilterSettings::watchFlags ( ) const
```

Returns database fields a change in which would affect the current filtering. To find out if an image tag change affects filtering, test [isFilteringByTags\(\)](#). The text filter will also be affected by changes in tags and album names.



## 9.814 Digikam::ItemFiltersHistoryItemDelegate Class Reference

Inheritance diagram for Digikam::ItemFiltersHistoryItemDelegate:

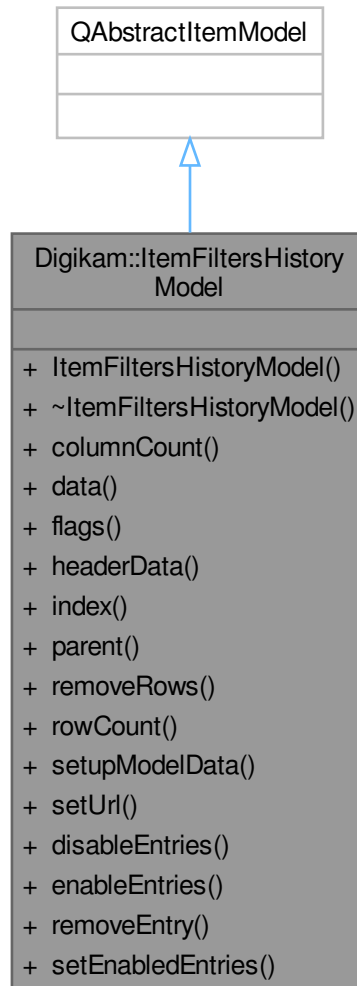


### Public Member Functions

- **ItemFiltersHistoryItemDelegate** (`QObject *const parent=nullptr`)
- void **paint** (`QPainter *painter, const QStyleOptionViewItem &option, const QModelIndex &index`) const override
- `QSize` **sizeHint** (`const QStyleOptionViewItem &option, const QModelIndex &index`) const override

## 9.815 Digikam::ItemFiltersHistoryModel Class Reference

Inheritance diagram for Digikam::ItemFiltersHistoryModel:



### Public Slots

- void **disableEntries** (int count)
- void **enableEntries** (int count)
- void **removeEntry** (const QModelIndex &index)
- void **setEnabledEntries** (int count)

### Public Member Functions

- **ItemFiltersHistoryModel** (QObject \*const parent=nullptr, const QUrl &url=QUrl())
- int **columnCount** (const QModelIndex &parent=QModelIndex()) const override

- QVariant **data** (const QModelIndex &index, int role=Qt::DisplayRole) const override
- Qt::ItemFlags **flags** (const QModelIndex &index) const override
- QVariant **headerData** (int section, Qt::Orientation orientation, int role=Qt::DisplayRole) const override
- QModelIndex **index** (int row, int column, const QModelIndex &parent=QModelIndex()) const override
- QModelIndex **parent** (const QModelIndex &index) const override
- bool **removeRows** (int row, int count, const QModelIndex &parent) override
- int **rowCount** (const QModelIndex &parent=QModelIndex()) const override
- void **setupModelData** (const QList< [DImageHistory::Entry](#) > &entries, [ItemFiltersHistoryTreeItem](#) \*parent=nullptr)
- void **setUrl** (const QUrl &url)

## 9.816 Digikam::ItemFiltersHistoryTreeItem Class Reference

### Public Member Functions

- [ItemFiltersHistoryTreeItem](#) (const QList< QVariant > &data, [ItemFiltersHistoryTreeItem](#) \*const parent=nullptr)
- [ItemFiltersHistoryTreeItem](#) (const QString &data, [ItemFiltersHistoryTreeItem](#) \*const parent=nullptr)
- void **appendChild** ([ItemFiltersHistoryTreeItem](#) \*const child)
- [ItemFiltersHistoryTreeItem](#) \* **child** (int row) const
- int **childCount** () const
- int **columnCount** () const
- QVariant **data** (int column) const
- bool **isDisabled** () const
- [ItemFiltersHistoryTreeItem](#) \* **parent** () const
- void **removeChild** (int row)
- int **row** () const
- void **setDisabled** (bool disabled) const

## 9.817 Digikam::ItemFullScreenOverlay Class Reference

Inheritance diagram for Digikam::ItemFullScreenOverlay:



### Signals

- void **signalFullscreen** (const QList< QModelIndex > &indexes)

## Signals inherited from [Digikam::ItemDelegateOverlay](#)

- void **hideNotification** ()
- void **requestNotification** (const QModelIndex &index, const QString &message)
- void **update** (const QModelIndex &index)

## Public Member Functions

- **ItemFullScreenOverlay** (QObject \*const parent)
- void **setActive** (bool active) override  
*Will call [createButton\(\)](#).*

## Public Member Functions inherited from [Digikam::HoverButtonDelegateOverlay](#)

- **HoverButtonDelegateOverlay** (QObject \*const parent)
- **ItemViewHoverButton** \* **button** () const

## Public Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- **AbstractWidgetDelegateOverlay** (QObject \*const parent)  
*This class provides functionality for using a widget in an overlay.*

## Public Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- **ItemDelegateOverlay** (QObject \*const parent=nullptr)
- virtual bool **acceptsDelegate** (QAbstractItemDelegate \*) const
- QAbstractItemDelegate \* **delegate** () const
- virtual void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)  
*Only these two methods are implemented as virtual methods.*
- virtual void **paint** (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index)
- void **setDelegate** (QAbstractItemDelegate \*delegate)
- void **setView** (QAbstractItemView \*view)
- QAbstractItemView \* **view** () const

## Static Public Member Functions

- static **ItemFullScreenOverlay** \* **instance** (QObject \*const parent)

## Protected Member Functions

- bool **checkIndex** (const QModelIndex &index) const override  
*Return true here if you want to show the overlay for the given index.*
- **ItemViewHoverButton** \* **createButton** () override  
*Create your widget here.*
- void **updateButton** (const QModelIndex &index) override  
*Called when a new index is entered.*
- void **widgetEnterEvent** () override  
*Called when a QEvent::Enter resp.*
- void **widgetLeaveEvent** () override

## Protected Member Functions inherited from [Digikam::HoverButtonDelegateOverlay](#)

- QWidget \* **createWidget** () override  
*Create your widget here.*
- void **visualChange** () override  
*Called when any change from the delegate occurs - when the overlay is installed, when size hints, styles or fonts change.*

## Protected Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- bool **checkIndexOnEnter** (const QModelIndex &index) const  
*Utility method called from slotEntered.*
- bool **eventFilter** (QObject \*obj, QEvent \*event) override
- virtual void **hide** ()  
*Called when the widget shall be hidden (mouse cursor left index, viewport, uninstalled etc.).*
- virtual QString **notifyMultipleMessage** (const QModelIndex &, int number)
- QWidget \* **parentWidget** () const  
*Returns the widget to be used as parent for your widget created in [createWidget\(\)](#)*
- virtual void **viewportLeaveEvent** (QObject \*obj, QEvent \*event)  
*Called when a QEvent::Leave of the viewport is received.*
- void **widgetEnterNotifyMultiple** (const QModelIndex &index)  
*A sample implementation for above methods.*
- void **widgetLeaveNotifyMultiple** ()

## Protected Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- QList< QModelIndex > **affectedIndexes** (const QModelIndex &index) const
- bool **affectsMultiple** (const QModelIndex &index) const  
*For the context that an overlay can affect multiple items: Assuming the currently overlaid index is given.*
- int **numberOfAffectedIndexes** (const QModelIndex &index) const
- bool **viewHasMultiSelection** () const  
*Utility method.*

## Additional Inherited Members

### Protected Slots inherited from [Digikam::HoverButtonDelegateOverlay](#)

- void **slotEntered** (const QModelIndex &index) override
- void **slotReset** () override

### Protected Slots inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- virtual void **slotEntered** (const QModelIndex &index)  
*Default implementation shows the widget iff the index is valid and checkIndex returns true.*
- virtual void **slotLayoutChanged** ()
- virtual void **slotReset** ()  
*Default implementations of these three slots call [hide\(\)](#)*
- virtual void **slotRowsRemoved** (const QModelIndex &parent, int start, int end)
- virtual void **slotViewportEntered** ()

## Protected Slots inherited from [Digikam::ItemDelegateOverlay](#)

## Protected Attributes inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- bool `m_mouseButtonPressedOnWidget` = false
- `QWidget * m_widget` = nullptr

## Protected Attributes inherited from [Digikam::ItemDelegateOverlay](#)

- `QAbstractItemDelegate * m_delegate` = nullptr
- `QAbstractItemView * m_view` = nullptr

## 9.817.1 Member Function Documentation

### 9.817.1.1 `checkIndex()`

```
bool Digikam::ItemFullScreenOverlay::checkIndex (
    const QModelIndex & index ) const [override], [protected], [virtual]
```

The default implementation returns true.

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.817.1.2 `createButton()`

```
ItemViewHoverButton * Digikam::ItemFullScreenOverlay::createButton ( ) [override], [protected], [virtual]
```

Pass `view()` as parent.

Implements [Digikam::HoverButtonDelegateOverlay](#).

### 9.817.1.3 `setActive()`

```
void Digikam::ItemFullScreenOverlay::setActive (
    bool active ) [override], [virtual]
```

Reimplemented from [Digikam::HoverButtonDelegateOverlay](#).

### 9.817.1.4 `updateButton()`

```
void Digikam::ItemFullScreenOverlay::updateButton (
    const QModelIndex & index ) [override], [protected], [virtual]
```

Reposition your button here, adjust and store state.

Implements [Digikam::HoverButtonDelegateOverlay](#).

### 9.817.1.5 widgetEnterEvent()

```
void Digikam::ItemFullScreenOverlay::widgetEnterEvent ( ) [override], [protected], [virtual]
```

QEvent::Leave event for the widget is received. The default implementation does nothing.

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.817.1.6 widgetLeaveEvent()

```
void Digikam::ItemFullScreenOverlay::widgetLeaveEvent ( ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).



## 9.818 Digikam::ItemFullScreenOverlayButton Class Reference

Inheritance diagram for Digikam::ItemFullScreenOverlayButton:



### Public Member Functions

- **ItemFullScreenOverlayButton** (QAbstractItemView \*const parentView)
- QSize [sizeHint](#) () const override

*Reimplement to match the size of your icon.*

## Public Member Functions inherited from [Digikam::ItemViewHoverButton](#)

- **ItemViewHoverButton** (QAbstractItemView \*const parentView)
- QModelIndex **index** () const
- void **initIcon** ()
- void **reset** ()
- void **setIndex** (const QModelIndex &index)
- void **setVisible** (bool visible) override

## Protected Member Functions

- QIcon **icon** () override  
*Return your icon here.*
- void **updateToolTip** () override  
*Optionally update tooltip here.*

## Protected Member Functions inherited from [Digikam::ItemViewHoverButton](#)

- void **enterEvent** (QEnterEvent \*event)
- void **leaveEvent** (QEvent \*event)
- void **paintEvent** (QPaintEvent \*event)
- void **setup** ()  
*to call in children class constructors to init signal/slot connections.*

## Additional Inherited Members

## Protected Slots inherited from [Digikam::ItemViewHoverButton](#)

- void **refreshIcon** ()
- void **setFadingValue** (int value)
- void **startFading** ()
- void **stopFading** ()

## Protected Attributes inherited from [Digikam::ItemViewHoverButton](#)

- QTimerLine \* **m\_fadingTimeLine** = nullptr
- int **m\_fadingValue** = 0
- QIcon **m\_icon**
- QPersistentModelIndex **m\_index**
- bool **m\_isHovered** = false

## 9.818.1 Member Function Documentation

### 9.818.1.1 icon()

QIcon Digikam::ItemFullScreenOverlayButton::icon ( ) [override], [protected], [virtual]

Will be queried again on toggle.

Implements [Digikam::ItemViewHoverButton](#).

### 9.818.1.2 sizeHint()

```
QSize Digikam::ItemFullScreenOverlayButton::sizeHint ( ) const [override], [virtual]
```

Implements [Digikam::ItemViewHoverButton](#).

### 9.818.1.3 updateToolTip()

```
void Digikam::ItemFullScreenOverlayButton::updateToolTip ( ) [override], [protected], [virtual]
```

Will be called again on state change.

Reimplemented from [Digikam::ItemViewHoverButton](#).

## 9.819 Digikam::ItemGPS Class Reference

Inheritance diagram for Digikam::ItemGPS:



### Public Member Functions

- **ItemGPS** (const [ItemInfo](#) &info)
- bool [loadImageData](#) () override
- QString [saveChanges](#) () override

## Public Member Functions inherited from Digikam::GPSItemContainer

- **GPSItemContainer** (const QUrl &url)
  
- bool **isDirty** () const
- QUrl **url** () const
- QDateTime **dateTime** () const
  
- void **setCoordinates** (const [GeoCoordinates](#) &newCoordinates)
- [GeoCoordinates](#) **coordinates** () const
- [GPSDataContainer](#) **gpsData** () const
- void **setGPSData** (const [GPSDataContainer](#) &container)
- void **restoreGPSData** (const [GPSDataContainer](#) &container)
 

*Restore the gps data to `container`.*
  
- void **setTagList** (const QList< QList< [TagData](#) > > &externalTagList)
 

*The tags added in reverse geocoding process are stored in each image, before they end up in external tag model.*
  
- bool **isTagListDirty** () const
  
- QList< QList< [TagData](#) > > **getTagList** () const
 

*Returns the tag list of the current image.*
  
- void **restoreRGTagList** (const QList< QList< [TagData](#) > > &tagList)
 

*Replaces the current tag list with the one contained in tagList.*
  
- void **writeTagsToXmp** (const bool writeXmpTags)
 

*Writes the current tags to XMP metadata.*
  
- void **writeLocations** (const bool writeMetaLoc)
 

*Writes the current tags to the metadata location fields.*
  
- bool **lessThan** (const [GPSItemContainer](#) \*const otherItem, const int column) const

## Additional Inherited Members

## Static Public Member Functions inherited from Digikam::GPSItemContainer

- static void **setHeaderData** ([GPSItemModel](#) \*const model)

### Static Public Attributes inherited from [Digikam::GPSItemContainer](#)

- static const int **ColumnAccuracy** = 6
- static const int **ColumnAltitude** = 5
- static const int **ColumnDateTime** = 2
- static const int **ColumnDOP** = 9
- static const int **ColumnFilename** = 1
- static const int **ColumnFixType** = 10
- static const int **ColumnGPSItemContainerCount** = 13
- static const int **ColumnLatitude** = 3
- static const int **ColumnLongitude** = 4
- static const int **ColumnNSatellites** = 11
- static const int **ColumnSpeed** = 12
- static const int **ColumnStatus** = 8
- static const int **ColumnTags** = 7
- static const int **ColumnThumbnail** = 0
- static const int **RoleCoordinates** = Qt::UserRole + 1

### Protected Member Functions inherited from [Digikam::GPSItemContainer](#)

- void **setLocationInfo** (const [TagData](#) &tagData, [IptcCoreLocationInfo](#) &locationInfo)
- QVariant **data** (const int column, const int role) const  
*these are only to be called by the [GPSItemModel](#)*
- void **setModel** ([GPSItemModel](#) \*const model)
- void **emitDataChanged** ()
- [DMetadata](#) \* **getMetadataForFile** () const
- [SaveProperties](#) **saveProperties** () const

### Protected Attributes inherited from [Digikam::GPSItemContainer](#)

- [GPSItemModel](#) \* **m\_model** = nullptr
- [QUrl](#) **m\_url**
- [QDateTime](#) **m\_dateTime**
- bool **m\_dirty** = false
- [GPSDataContainer](#) **m\_gpsData**
- [GPSDataContainer](#) **m\_savedState**
- bool **m\_tagListDirty** = false
- [QList](#)< [QList](#)< [TagData](#) > > **m\_tagList**
- [QList](#)< [QList](#)< [TagData](#) > > **m\_savedTagList**
- bool **m\_writeXmpTags** = true
- bool **m\_writeMetaLoc** = true

## 9.819.1 Member Function Documentation

### 9.819.1.1 loadImageData()

```
bool Digikam::ItemGPS::loadImageData ( ) [override], [virtual]
```

Reimplemented from [Digikam::GPSItemContainer](#).

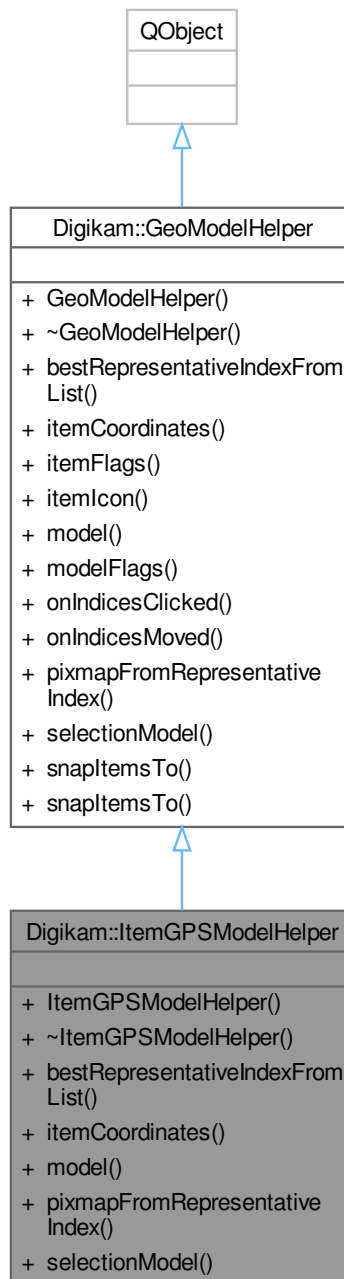
### 9.819.1.2 saveChanges()

```
QString Digikam::ItemGPS::saveChanges ( ) [override], [virtual]
```

Reimplemented from [Digikam::GPSItemContainer](#).

## 9.820 Digikam::ItemGPSModelHelper Class Reference

Inheritance diagram for Digikam::ItemGPSModelHelper:



### Public Member Functions

- **ItemGPSModelHelper** (QStandardItemModel \*const itemModel, QObject \*const parent=nullptr)
- QPersistentModelIndex [bestRepresentativeIndexFromList](#) (const QList< QPersistentModelIndex > &list, const int sortKey) override



- bool [itemCoordinates](#) (const QModelIndex &index, [GeoCoordinates](#) \*const coordinates) const override
- QAbstractItemModel \* [model](#) () const override  
*these are necessary for grouped and ungrouped models*
- QPixmap [pixmapFromRepresentativeIndex](#) (const QPersistentModelIndex &index, const QSize &size) override  
*these are used by MarkerModel for grouped models*
- QItemSelectionModel \* [selectionModel](#) () const override

## Public Member Functions inherited from [Digikam::GeoModelHelper](#)

- **GeoModelHelper** (QObject \*const parent=nullptr)
- virtual PropertyFlags **itemFlags** (const QModelIndex &index) const
- virtual bool [itemIcon](#) (const QModelIndex &index, QPoint \*const offset, QSize \*const size, QPixmap \*const pixmap, QUrl \*const url) const  
*these are necessary for ungrouped models*
- virtual PropertyFlags **modelFlags** () const
- virtual void [onIndicesClicked](#) (const QList< QPersistentModelIndex > &clickedIndices)
- virtual void **onIndicesMoved** (const QList< QPersistentModelIndex > &movedIndices, const [GeoCoordinates](#) &targetCoordinates, const QPersistentModelIndex &targetSnapIndex)
- virtual void **snapItemsTo** (const QModelIndex &targetIndex, const QList< QModelIndex > &snappedIndices)
- void **snapItemsTo** (const QModelIndex &targetIndex, const QList< QPersistentModelIndex > &snappedIndices)

## Additional Inherited Members

## Public Types inherited from [Digikam::GeoModelHelper](#)

- enum **PropertyFlag** { **FlagNull** = 0 , **FlagVisible** = 1 , **FlagMovable** = 2 , **FlagSnaps** = 4 }
- typedef QFlags< PropertyFlag > **PropertyFlags**

## Signals inherited from [Digikam::GeoModelHelper](#)

- void **signalModelChangedDrastically** ()
- void **signalThumbnailAvailableForIndex** (const QPersistentModelIndex &index, const QPixmap &pixmap)
- void **signalVisibilityChanged** ()

## 9.820.1 Member Function Documentation

### 9.820.1.1 [bestRepresentativeIndexFromList\(\)](#)

```
QPersistentModelIndex Digikam::ItemGPSModelHelper::bestRepresentativeIndexFromList (
    const QList< QPersistentModelIndex > & list,
    const int sortKey ) [override], [virtual]
```

Reimplemented from [Digikam::GeoModelHelper](#).

### 9.820.1.2 itemCoordinates()

```
bool Digikam::ItemGPSModelHelper::itemCoordinates (
    const QModelIndex & index,
    GeoCoordinates *const coordinates ) const [override], [virtual]
```

Implements [Digikam::GeoModelHelper](#).

### 9.820.1.3 model()

```
QAbstractItemModel * Digikam::ItemGPSModelHelper::model ( ) const [override], [virtual]
```

Implements [Digikam::GeoModelHelper](#).

### 9.820.1.4 pixmapFromRepresentativeIndex()

```
QPixmap Digikam::ItemGPSModelHelper::pixmapFromRepresentativeIndex (
    const QPersistentModelIndex & index,
    const QSize & size ) [override], [virtual]
```

Reimplemented from [Digikam::GeoModelHelper](#).

### 9.820.1.5 selectionModel()

```
QItemSelectionModel * Digikam::ItemGPSModelHelper::selectionModel ( ) const [override], [virtual]
```

Implements [Digikam::GeoModelHelper](#).

## 9.821 Digikam::ItemHistoryGraph Class Reference

### Public Types

- enum [HistoryLoadingFlag](#) { [LoadRelationCloud](#) = 1 << 0 , [LoadSubjectHistory](#) = 1 << 1 , [LoadLeavesHistory](#) = 1 << 2 , [LoadAll](#) = LoadRelationCloud | LoadSubjectHistory | LoadLeavesHistory }
- typedef QFlags< [HistoryLoadingFlag](#) > [HistoryLoadingMode](#)
- enum [ProcessingMode](#) { [NoProcessing](#) , [PrepareForDisplay](#) }

## Public Member Functions

- **ItemHistoryGraph** (const [ItemHistoryGraph](#) &other)
- void **addHistory** (const [DImageHistory](#) &history, const [HistoryImageId](#) &historySubject=[HistoryImageId](#)())
- void **addHistory** (const [DImageHistory](#) &history, const [ItemInfo](#) &historySubject=[ItemInfo](#)())  
*Add the given history.*
- void **addRelations** (const [QList](#)< [QPair](#)< [qulonglong](#), [qulonglong](#) > > &pairs)  
*Add images and their relations from the given pairs.*
- void **addScannedHistory** (const [DImageHistory](#) &history, [qulonglong](#) historySubjectId)  
*This is very similar to addHistory.*
- [QList](#)< [qulonglong](#) > **allImageIds** () const
- [QList](#)< [ItemInfo](#) > **allImages** () const  
*Returns image infos / ids from all vertices in this graph.*
- [QHash](#)< [ItemInfo](#), [HistoryImageId::Types](#) > **categorize** () const  
*Attempts at a categorization of all images in the graph into the types defined by [HistoryImageId](#).*
- void **clear** ()  
*Clears this graph.*
- [ItemHistoryGraphData](#) & **data** ()
- const [ItemHistoryGraphData](#) & **data** () const
- void **dropUnresolvedEntries** ()  
*Remove all vertices from the graph for which no existing [ItemInfo](#) could be found in the database.*
- bool **hasEdges** () const  
*Returns if the graph contains any edges.*
- bool **hasUnresolvedEntries** () const  
*Returns true if for any entry no [ItemInfo](#) could be located.*
- bool **isEmpty** () const
- bool **isNull** () const
- bool **isSingleVertex** () const
- [QList](#)< [ItemInfo](#) > **leafImages** () const  
*Returns image infos / ids from all leaf vertices in this graph, i.e.*
- [ItemHistoryGraph](#) & **operator=** (const [ItemHistoryGraph](#) &other)
- void **prepareForDisplay** (const [ItemInfo](#) &subject)  
*Combines [reduceEdges\(\)](#), [dropOrphans\(\)](#) and [sortForInfo\(\)](#).*
- void **reduceEdges** ()  
*Remove edges which provide only duplicate information (performs a transitive reduction).*
- [QList](#)< [QPair](#)< [qulonglong](#), [qulonglong](#) > > **relationCloud** () const  
*Returns all possible relations between images in this graph, the edges of the transitive closure.*
- [QPair](#)< [QList](#)< [qulonglong](#) >, [QList](#)< [qulonglong](#) > > **relationCloudParallel** () const
- [QList](#)< [ItemInfo](#) > **rootImages** () const  
*Returns image infos / ids from all root vertices in this graph, i.e.*
- void **sortForInfo** (const [ItemInfo](#) &subject)  
*Sort vertex information prioritizing for the given vertex.*

## Static Public Member Functions

- static [ItemHistoryGraph](#) **fromInfo** (const [ItemInfo](#) &info, [HistoryLoadingMode](#) loadingMode=[LoadAll](#), [ProcessingMode](#) processingMode=[PrepareForDisplay](#))  
*Convenience: Reads all available history for the given info from the database and returns the created graph.*

## 9.821.1 Member Enumeration Documentation

### 9.821.1.1 HistoryLoadingFlag

```
enum Digikam::ItemHistoryGraph::HistoryLoadingFlag
```

## Enumerator

LoadRelationCloud	Load the relation cloud to the graph. Will give all edges, but no further info.
LoadSubjectHistory	Will load the <a href="#">DImageHistory</a> of the given subject.
LoadLeavesHistory	Will load the <a href="#">DImageHistory</a> of all leave vertices of the graph.

## 9.821.2 Member Function Documentation

### 9.821.2.1 addHistory()

```
void Digikam::ItemHistoryGraph::addHistory (
    const DImageHistory & history,
    const ItemInfo & historySubject = ItemInfo\(\) )
```

The optionally given info or id is used as the "current" image of the history. If you read a history from a file's metadata or the database, you shall give the relevant subject.

### 9.821.2.2 addRelations()

```
void Digikam::ItemHistoryGraph::addRelations (
    const QList< QPair< qlonglong, qlonglong > > & pairs )
```

Each pair (a,b) means "a is derived from b".

### 9.821.2.3 addScannedHistory()

```
void Digikam::ItemHistoryGraph::addScannedHistory (
    const DImageHistory & history,
    qlonglong historySubjectId )
```

The only difference is that no attempt is made to retrieve an [ItemInfo](#) for the historySubjectId. Can be useful in the context of scanning

### 9.821.2.4 categorize()

```
QHash< ItemInfo, HistoryImageId::Types > Digikam::ItemHistoryGraph::categorize ( ) const
```

The type will be invalid if no decision can be made due to conflicting data.

### 9.821.2.5 fromInfo()

```
ItemHistoryGraph Digikam::ItemHistoryGraph::fromInfo (
    const ItemInfo & info,
    HistoryLoadingMode loadingMode = LoadAll,
    ProcessingMode processingMode = PrepareForDisplay ) [static]
```

Depending on mode, the graph will be preparedForDisplay(). If no history is recorded and no relations found, a single-vertex graph is returned.

### 9.821.2.6 hasEdges()

```
bool Digikam::ItemHistoryGraph::hasEdges ( ) const
```

Because loops are not allowed, this also means (!isEmpty() && !isSingleVertex()).

### 9.821.2.7 leafImages()

```
QList< ItemInfo > Digikam::ItemHistoryGraph::leafImages ( ) const
```

vertices with no subsequent history.

### 9.821.2.8 reduceEdges()

```
void Digikam::ItemHistoryGraph::reduceEdges ( )
```

Especially call this when [addRelations\(\)](#) was used.

### 9.821.2.9 relationCloud()

```
QList< QPair< qlonglong, qlonglong > > Digikam::ItemHistoryGraph::relationCloud ( ) const
```

The first variant returns (1,2),(3,4),(6,8), the second (1,3,6)(2,4,8).

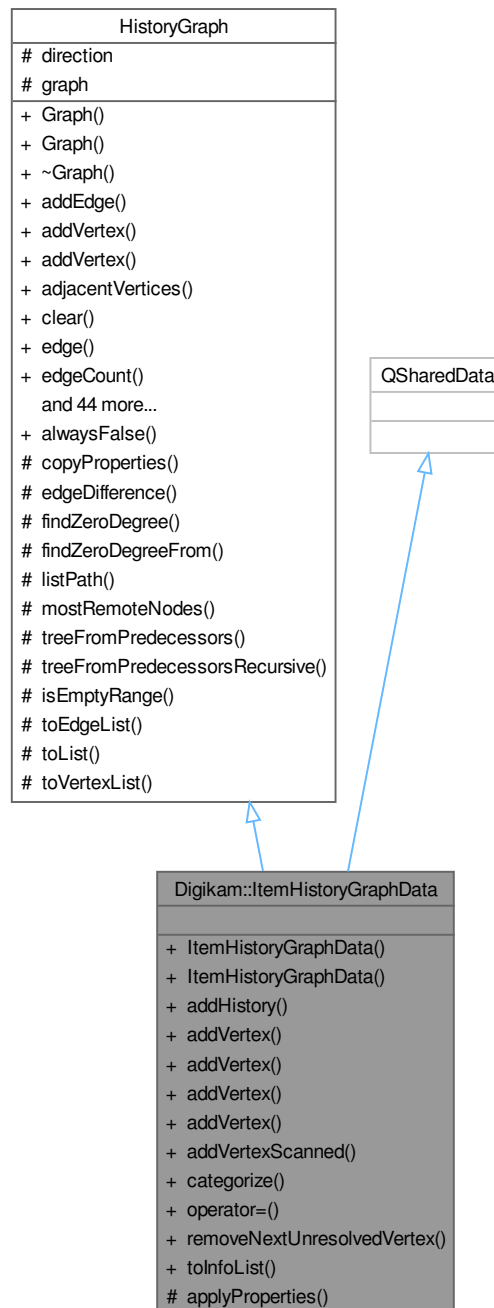
### 9.821.2.10 rootImages()

```
QList< ItemInfo > Digikam::ItemHistoryGraph::rootImages ( ) const
```

vertices with no precedent history.

## 9.822 Digikam::ItemHistoryGraphData Class Reference

Inheritance diagram for Digikam::ItemHistoryGraphData:



### Public Member Functions

- **ItemHistoryGraphData** (const [HistoryGraph](#) &g)
- void **addHistory** (const [DImageHistory](#) &givenHistory, qlonglong extraCurrent=0)

- [Vertex](#) **addVertex** (const [HistoryImageld](#) &id)
- [Vertex](#) **addVertex** (const [ItemInfo](#) &info)
- [Vertex](#) **addVertex** (const QList< [HistoryImageld](#) > &imagelds)
- [Vertex](#) **addVertex** (qlonglong id)
- [Vertex](#) **addVertexScanned** (qlonglong id)
- QHash< [Vertex](#), [HistoryImageld::Types](#) > **categorize** () const
- [ItemHistoryGraphData](#) & **operator=** (const [HistoryGraph](#) &g)
- int **removeNextUnresolvedVertex** (int begin)
- QList< [ItemInfo](#) > **toInfoList** (const QList< [Vertex](#) > &vertices) const

## Public Member Functions inherited from [Digikam::Graph](#)< [VertexProperties](#), [EdgeProperties](#) >

- [Graph](#) (const [Graph](#) &g)
- [Graph](#) ([MeaningOfDirection](#) dir=[ParentToChild](#))
- [Edge](#) **addEdge** (const [Vertex](#) &v1, const [Vertex](#) &v2)
- [Vertex](#) **addVertex** ()
- [Vertex](#) **addVertex** (const [VertexProperties](#) &properties)
- QList< [Vertex](#) > **adjacentVertices** (const [Vertex](#) &v, [AdjacencyFlags](#) flags=[AllEdges](#)) const
- void **clear** ()
- [Edge](#) **edge** (const [Vertex](#) &v1, const [Vertex](#) &v2) const
- int **edgeCount** () const
- QList< [VertexPair](#) > **edgePairs** () const
- QList< [Edge](#) > **edges** () const
- QList< [Edge](#) > **edges** (const [Vertex](#) &v, [AdjacencyFlags](#) flags=[AllEdges](#)) const
- template<class T >  
[Vertex](#) **findVertexByProperties** (const T &value) const
- const [GraphContainer](#) & **getGraph** () const  
*Accessing vertices and edges.*
- bool **hasEdge** (const [Vertex](#) &v1, const [Vertex](#) &v2) const
- bool **hasEdges** () const
- bool **hasEdges** (const [Vertex](#) &v, [AdjacencyFlags](#) flags=[AllEdges](#)) const
- int **inDegree** (const [Vertex](#) &v) const
- bool **isConnected** (const [Vertex](#) &v1, const [Vertex](#) &v2) const
- bool **isEmpty** () const
- bool **isLeaf** (const [Vertex](#) &v) const
- bool **isRoot** (const [Vertex](#) &v) const
- QList< [Vertex](#) > **leaves** () const  
*Returns all leaves, i.e.*
- QList< [Vertex](#) > **leavesFrom** (const [Vertex](#) &v) const
- QList< [Vertex](#) > **longestPathTouching** (const [Vertex](#) &v) const  
*Returns the longest path through the graph, starting from a vertex in [roots\(\)](#), ending on a vertex in [leaves\(\)](#), and passing vertex v.*
- template<typename LessThan >  
QList< [Vertex](#) > **longestPathTouching** (const [Vertex](#) &v, LessThan lessThan) const
- [MeaningOfDirection](#) **meaningOfDirection** () const
- [Graph](#) & **operator=** (const [Graph](#) &other)
- int **outDegree** (const [Vertex](#) &v) const
- [EdgeProperties](#) & **properties** (const [Edge](#) &e)
- const [EdgeProperties](#) & **properties** (const [Edge](#) &e) const
- [VertexProperties](#) & **properties** (const [Vertex](#) &v)
- const [VertexProperties](#) & **properties** (const [Vertex](#) &v) const
- [EdgeProperties](#) **properties** (const [Vertex](#) &v1, const [Vertex](#) &v2) const

- void **remove** (const [Vertex](#) &v)
- [QList](#)< [Vertex](#) > **roots** () const  
*Returns all roots, i.e.*
- [QList](#)< [Vertex](#) > **rootsOf** (const [Vertex](#) &v) const  
*Returns all roots of vertex v.*
- void **setProperty** (const [Edge](#) &e, const [EdgeProperties](#) &props)
- void **setProperty** (const [Vertex](#) &v, const [VertexProperties](#) &props)
- [QMap](#)< [Vertex](#), int > **shortestDistancesFrom** (const [Vertex](#) &v) const  
*Returns the shortest distances from [Vertex](#) to all vertices in the graph.*
- [QList](#)< [Vertex](#) > **shortestPath** (const [Vertex](#) &v1, const [Vertex](#) &v2) const  
*Returns the shortestPath between id1 and id2.*
- [Vertex](#) **source** (const [Edge](#) &e) const
- [Vertex](#) **target** (const [Edge](#) &e) const
- [QList](#)< [Vertex](#) > **topologicalSort** () const  
*Returns the vertex ids of this graph, in topological order.*
- [Graph](#) **transitiveClosure** ([GraphCopyFlags](#) flags=[CopyAllProperties](#)) const  
*Returns a copy of this graph with all edges added to form the transitive closure.*
- [Graph](#) **transitiveReduction** ([QList](#)< [Edge](#) > \*removedEdges=0, [GraphCopyFlags](#) flags=[CopyAllProperties](#)) const  
*Returns a copy of this graph, with edges removed so that the transitive reduction is formed.*
- int **vertexCount** () const
- [QList](#)< [Vertex](#) > **vertices** () const
- [QList](#)< [Vertex](#) > **verticesBreadthFirst** (const [Vertex](#) &givenRef=[Vertex](#)()) const  
*Orders all vertices of the graph in a breadth-first manner.*
- [template](#)<typename [LessThan](#) >  
[QList](#)< [Vertex](#) > **verticesDepthFirstSorted** (const [Vertex](#) &givenRef, [LessThan](#) lessThan) const  
*Orders all vertices of the graph in a depth-first manner.*
- [QList](#)< [Vertex](#) > **verticesDominatedBy** (const [Vertex](#) &v, const [Vertex](#) &root, const [QList](#)< [Vertex](#) > &presortedVertices) const  
*For a vertex v reachable from a vertex root returns all vertices dominated by v starting from root.*
- [QList](#)< [Vertex](#) > **verticesDominatedBy** (const [Vertex](#) &v, const [Vertex](#) &root, [ReturnOrder](#) order=[BreadthFirstOrder](#)) const  
*For a vertex v reachable from a vertex root, returns, in depth-first or breadth-first order, all vertices dominated by v starting from root.*
- [template](#)<typename [LessThan](#) >  
[QList](#)< [Vertex](#) > **verticesDominatedByDepthFirstSorted** (const [Vertex](#) &v, const [Vertex](#) &root, [LessThan](#) lessThan) const  
*For a vertex v reachable from a vertex root all vertices dominated by v starting from root.*

### Protected Member Functions

- void **applyProperties** ([Vertex](#) &v, const [QList](#)< [ItemInfo](#) > &infos, const [QList](#)< [HistoryImageld](#) > &ids)

### Protected Member Functions inherited from [Digikam::Graph](#)< [VertexProperties](#), [EdgeProperties](#) >

- void **copyProperties** ([Graph](#) &other, [GraphCopyFlags](#) flags, const [std::vector](#)< [vertex\\_t](#) > &copiedVertices) const  
*According to the given flags and based on the map, copies vertex and edge properties from this to the other graph.*
- [QList](#)< [Edge](#) > **edgeDifference** (const [Graph](#) &other, const [std::vector](#)< [vertex\\_t](#) > &copiedVertices) const  
*Returns a list of edges of this graph that have been removed in other.*



- `QList< Vertex > findZeroDegree` (bool inOrOut) const  
*Finds vertex ids of all vertices with zero in- our out-degree.*
- `QList< Vertex > findZeroDegreeFrom` (const Vertex &v, bool inOrOut) const
- `QList< Vertex > listPath` (const Vertex &root, const Vertex &target, const VertexVertexMap &predecessors, MeaningOfDirection dir=ParentToChild) const  
*Get a list of vertex ids for the path from root to target, using the given predecessors.*
- `QList< Vertex > mostRemoteNodes` (const VertexIntMap &distances) const  
*Get the list of vertices with the largest value in the given distance map.*
- `QList< Vertex > treeFromPredecessors` (const Vertex &v, const VertexVertexMap &predecessors) const
- void `treeFromPredecessorsRecursive` (const Vertex &v, QList< Vertex > &vertices, const VertexVertexMap &predecessors) const

### Additional Inherited Members

### Public Types inherited from Digikam::Graph< VertexProperties, EdgeProperties >

- typedef graph\_traits::adjacency\_iterator **adjacency\_iter**
- typedef std::pair< adjacency\_iter, adjacency\_iter > **adjacency\_vertex\_range\_t**
- enum **AdjacencyFlags** {  
**OutboundEdges** = 1 << 0 , **InboundEdges** = 1 << 1 , **EdgesToLeaf** = 1 << 2 , **EdgesToRoot** = 1 << 3 ,  
**AllEdges** = InboundEdges | OutboundEdges }
- typedef boost::property\_map< GraphContainer, edge\_properties\_t >::const\_type **const\_edge\_property\_↔\_map\_t**
- typedef boost::property\_map< GraphContainer, boost::vertex\_index\_t >::const\_type **const\_vertex\_index\_↔\_map\_t**
- typedef boost::property\_map< GraphContainer, vertex\_properties\_t >::const\_type **const\_vertex\_↔\_property\_map\_t**
- typedef graph\_traits::degree\_size\_type **degree\_t**
- typedef graph\_traits::edge\_iterator **edge\_iter**
- typedef boost::property\_map< GraphContainer, edge\_properties\_t >::type **edge\_property\_map\_t**
- typedef std::pair< edge\_iter, edge\_iter > **edge\_range\_t**
- typedef graph\_traits::edge\_descriptor **edge\_t**
- typedef QPair< Edge, Edge > **EdgePair**
- typedef boost::graph\_traits< GraphContainer > **graph\_traits**  
*a bunch of graph-specific typedefs that make the long boost types manageable.*
- typedef boost::adjacency\_list< boost::vecS, boost::vecS, boost::bidirectionalS, boost::property< boost\_↔::vertex\_index\_t, int, boost::property< vertex\_properties\_t, VertexProperties > >, boost::property< edge\_↔\_properties\_t, EdgeProperties > > **GraphContainer**
- enum **GraphCopyFlags** { **CopyVertexProperties** = 1 << 0 , **CopyEdgeProperties** = 1 << 1 , **CopyAll\_↔\_Properties** = CopyVertexProperties | CopyEdgeProperties }
- typedef graph\_traits::in\_edge\_iterator **in\_edge\_iter**
- typedef boost::inv\_adjacency\_iterator\_generator< GraphContainer, vertex\_t, in\_edge\_iter >::type **inv\_↔\_adjacency\_iter**
- typedef std::pair< inv\_adjacency\_iter, inv\_adjacency\_iter > **inv\_adjacency\_vertex\_range\_t**
- typedef graph\_traits::out\_edge\_iterator **out\_edge\_iter**
- typedef std::pair< out\_edge\_iter, out\_edge\_iter > **out\_edge\_range\_t**
- enum **ReturnOrder** { **BreadthFirstOrder** , **DepthFirstOrder** }
- typedef boost::property\_map< GraphContainer, boost::vertex\_index\_t >::type **vertex\_index\_map\_t**
- typedef graph\_traits::vertex\_iterator **vertex\_iter**
- typedef boost::property\_map< GraphContainer, vertex\_properties\_t >::type **vertex\_property\_map\_t**
- typedef std::pair< vertex\_iter, vertex\_iter > **vertex\_range\_t**
- typedef graph\_traits::vertex\_descriptor **vertex\_t**
- typedef QMapForAdaptors< Vertex, int > **VertexIntMap**
- typedef boost::associative\_property\_map< VertexIntMap > **VertexIntMapAdaptor**
- typedef QPair< Vertex, Vertex > **VertexPair**
- typedef QMapForAdaptors< Vertex, Vertex > **VertexVertexMap**
- typedef boost::associative\_property\_map< VertexVertexMap > **VertexVertexMapAdaptor**

### Static Public Member Functions inherited from [Digikam::Graph](#) < [VertexProperties](#), [EdgeProperties](#) >

- `template<typename T >`  
static bool **alwaysFalse** (const T &, const T &)

### Static Protected Member Functions inherited from [Digikam::Graph](#) < [VertexProperties](#), [EdgeProperties](#) >

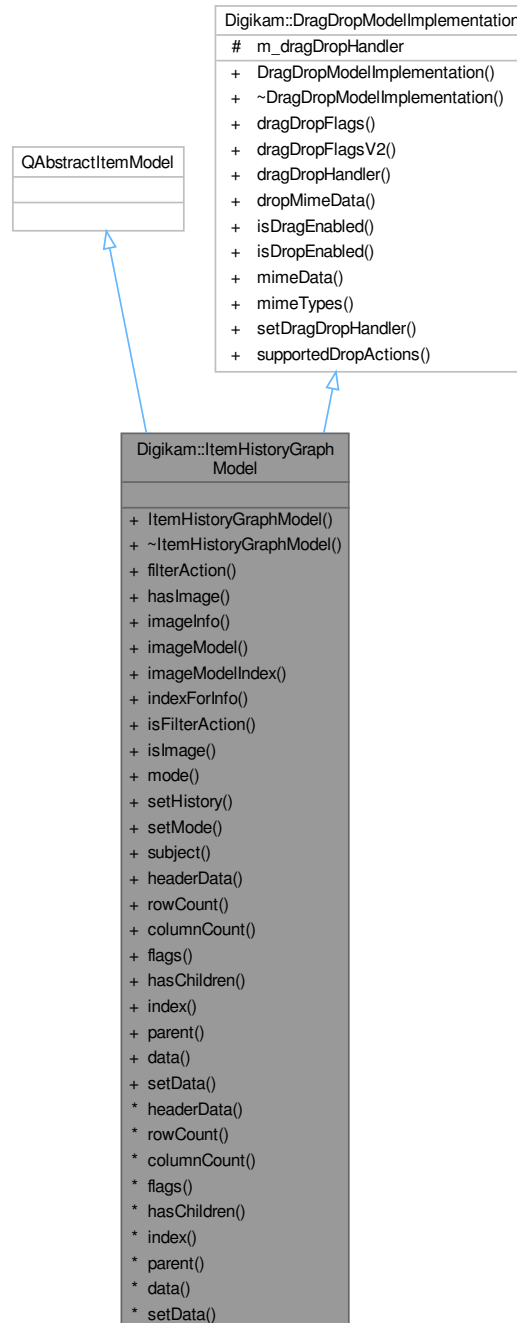
- `template<typename range_t >`  
static bool **isEmptyRange** (const range\_t &range)
- `template<typename range_t >`  
static QList< [Edge](#) > **toEdgeList** (const range\_t &range)
- `template<typename Value , typename range_t >`  
static QList< Value > **toList** (const range\_t &range)  
*Returns a list of vertex ids of vertices in the given range.*
- `template<typename range_t >`  
static QList< [Vertex](#) > **toVertexList** (const range\_t &range)

### Protected Attributes inherited from [Digikam::Graph](#) < [VertexProperties](#), [EdgeProperties](#) >

- [MeaningOfDirection](#) **direction** = [ParentToChild](#)
- GraphContainer **graph**

## 9.823 Digikam::ItemHistoryGraphModel Class Reference

Inheritance diagram for Digikam::ItemHistoryGraphModel:



### Public Types

- enum **ExtraRoles** {
  - IsImageItemRole** = Qt::UserRole + 1000 , **IsFilterActionItemRole** = Qt::UserRole + 1001 , **IsHeaderItemRole** = Qt::UserRole + 1002 , **IsCategoryItemRole** = Qt::UserRole + 1003 ,

**IsSeparatorItemRole** = Qt::UserRole + 1004 , **IsSubjectImageRole** = Qt::UserRole + 1010 , **FilterActionRole** = Qt::UserRole + 1020 }  
 • enum **Mode** { **ImagesListMode** , **ImagesTreeMode** , **CombinedTreeMode** }

## Public Member Functions

- **ItemHistoryGraphModel** (QWidget \*const parent)
  - **FilterAction filterAction** (const QModelIndex &index) const
  - bool **hasImage** (const ItemInfo &info)
  - **ItemInfo imageInfo** (const QModelIndex &index) const
  - DECLARE\_MODEL\_DRAG\_DROP\_METHODS **ItemListModel \* imageModel** () const  
*Returns an internal image model used for entries representing images.*
  - QModelIndex **imageModelIndex** (const QModelIndex &index) const  
*If the given index is represented by the internal image model, return the image model's index.*
  - QModelIndex **indexForInfo** (const ItemInfo &info) const  
*Note: There may be multiple indexes for an info.*
  - bool **isFilterAction** (const QModelIndex &index) const
  - bool **isImage** (const QModelIndex &index) const
  - Mode **mode** () const
  - void **setHistory** (const ItemInfo &subject, const ItemHistoryGraph &graph=ItemHistoryGraph())  
*Set the history subject and the history graph.*
  - void **setMode** (Mode mode)
  - **ItemInfo subject** () const
- 
- QVariant **headerData** (int section, Qt::Orientation orientation, int role=Qt::DisplayRole) const override  
*QAbstractItemModel implementation.*
  - int **rowCount** (const QModelIndex &parent=QModelIndex()) const override
  - int **columnCount** (const QModelIndex &parent=QModelIndex()) const override
  - Qt::ItemFlags **flags** (const QModelIndex &index) const override
  - bool **hasChildren** (const QModelIndex &parent=QModelIndex()) const override
  - QModelIndex **index** (int row, int column, const QModelIndex &parent=QModelIndex()) const override
  - QModelIndex **parent** (const QModelIndex &index) const override
  - QVariant **data** (const QModelIndex &index, int role=Qt::DisplayRole) const override
  - bool **setData** (const QModelIndex &index, const QVariant &value, int role) override

## Public Member Functions inherited from **Digikam::DragDropModelImplementation**

- **DragDropModelImplementation** ()=default  
*A class providing a sample implementation for a QAbstractItemModel redirecting drag-and-drop support to a handler.*
- virtual Qt::ItemFlags **dragDropFlags** (const QModelIndex &index) const  
*Call from your flags() method, adding the relevant drag drop flags.*
- Qt::ItemFlags **dragDropFlagsV2** (const QModelIndex &index) const  
*This is an alternative approach to dragDropFlags().*
- **AbstractItemDragDropHandler \* dragDropHandler** () const
- bool **dropMimeData** (const QMimeData \*, Qt::DropAction, int, int, const QModelIndex &)
- virtual bool **isDragEnabled** (const QModelIndex &index) const
- virtual bool **isDropEnabled** (const QModelIndex &index) const
- QMimeData \* **mimeData** (const QModelIndexList &indexes) const
- QStringList **mimeTypes** () const
- void **setDragDropHandler** (**AbstractItemDragDropHandler** \*handler)  
*Set a drag drop handler.*
- Qt::DropActions **supportedDropActions** () const  
*Implements the relevant QAbstractItemModel methods for drag and drop.*

## Additional Inherited Members

## Protected Attributes inherited from [Digikam::DragDropModelImplementation](#)

- [AbstractItemDragDropHandler](#) \* `m_dragDropHandler` = nullptr

## 9.823.1 Member Function Documentation

### 9.823.1.1 `imageModel()`

```
ItemListModel * Digikam::ItemHistoryGraphModel::imageModel ( ) const
```

Note: Set a thumbnail thread on this model if you need thumbnails.

### 9.823.1.2 `imageModelIndex()`

```
QModelIndex Digikam::ItemHistoryGraphModel::imageModelIndex (
    const QModelIndex & index ) const
```

Otherwise an invalid index is returned.

### 9.823.1.3 `indexForInfo()`

```
QModelIndex Digikam::ItemHistoryGraphModel::indexForInfo (
    const ItemInfo & info ) const
```

The index found first is returned.

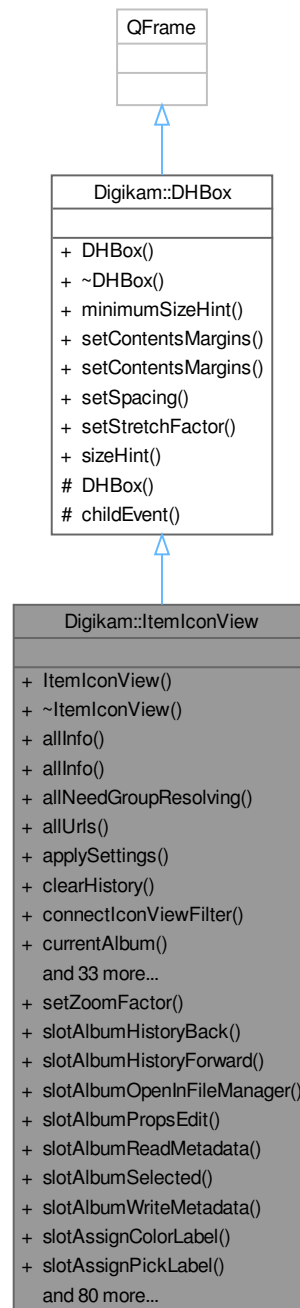
### 9.823.1.4 `setHistory()`

```
void Digikam::ItemHistoryGraphModel::setHistory (
    const ItemInfo & subject,
    const ItemHistoryGraph & graph = ItemHistoryGraph() )
```

Per default, the subject's history graph is read.

## 9.824 Digikam::ItemIconView Class Reference

Inheritance diagram for Digikam::ItemIconView:



### Public Slots

- void **setZoomFactor** (double zoom)
- void **slotAlbumHistoryBack** (int steps=1)

- void **slotAlbumHistoryForward** (int steps=1)
- void **slotAlbumOpenInFileManager** ()
- void **slotAlbumPropsEdit** ()
- void **slotAlbumReadMetadata** ()
- void **slotAlbumSelected** (const QList< Album \* > &albums)
- void **slotAlbumWriteMetadata** ()
- void **slotAssignColorLabel** (int colorId)
- void **slotAssignPickLabel** (int pickId)
- void **slotAssignRating** (int rating, bool toggle=true)
- void **slotAssignTag** ()
- void **slotAssignTag** (int tagID)
- void **slotCopySelectionTo** ()
- void **slotCreateGroupByFilenameFromSelection** ()
- void **slotCreateGroupByTimeFromSelection** ()
- void **slotCreateGroupByTimelapseFromSelection** ()
- void **slotCreateGroupFromSelection** ()
- void **slotDeleteAlbum** ()
- void **slotDeleteTag** ()
- void **slotEditor** ()
- void **slotEditTag** ()
- void **slotFileWithDefaultApplication** ()
- void **slotFitToWindow** ()
- void **slotFocusAndNextImage** ()
- void **slotGotoAlbumAndItem** (const ItemInfo &imageInfo)
- void **slotGotoDateAndItem** (const ItemInfo &imageInfo)
- void **slotGotoTagAndItem** (int tagID)
- void **slotIconView** ()
- void **slotImageAddToCurrentQueue** ()
- void **slotImageAddToExistingQueue** (int)
- void **slotImageAddToLightTable** ()
- void **slotImageAddToNewQueue** ()
- void **slotImageDelete** ()
- void **slotImageDeletePermanently** ()
- void **slotImageDeletePermanentlyDirectly** ()
- void **slotImageEdit** ()

*Tools methods (Editor, BQM, Light Table) - itemiconview\_tools.cpp.*

- void **slotImageExifOrientation** (int orientation)
- void **slotImageFindSimilar** ()
- void **slotImageLightTable** ()
- void **slotImagePaste** ()
- void **slotImagePreview** ()
- void **slotImageQualitySorter** ()

*Side-bars handling methods - itemiconview\_sidebars.cpp.*

- void **slotImageReadMetadata** ()
- void **slotImageRecognizeFaces** ()
- void **slotImageRemoveAllFaces** ()
- void **slotImageRename** ()
- void **slotImageScanForFaces** ()
- void **slotImageSeparationSortOrder** (int order)
- void **slotImageTrashDirectly** ()
- void **slotImageWriteMetadata** ()
- void **slotLeftSideBarActivate** (QWidget \*widget)
- void **slotLeftSideBarActivate** (SidebarWidget \*widget)
- void **slotLeftSideBarActivateAlbums** ()

- void **slotLeftSideBarActivateTags** ()
- void **slotLightTable** ()
- void **slotMapViewWidgetView** ()
- void **slotMoveSelectionToAlbum** ()
- void **slotNewAdvancedSearch** ()
- void **slotNewAlbum** ()
- void **slotNewDuplicatesSearch** (const QList< [PAlbum](#) \* > &albums={})
- void **slotNewDuplicatesSearch** (const QList< [TAlbum](#) \* > &albums)
- void **slotNewKeywordSearch** ()

*Search management methods - itemiconview\_search.cpp.*

- void **slotNewTag** ()
- void **slotNotificationError** (const QString &message, int type)
- void **slotOpenTagsManager** ()
- void **slotQueueMgr** ()
- void **slotRefresh** ()
- void **slotRemoveSelectedFromGroup** ()
- void **slotRemoveTag** (int tagID)
- void **slotRenameAlbum** ()
- void **slotRightSideBarActivateAssignedTags** ()
- void **slotRightSideBarActivateComments** ()
- void **slotRightSideBarActivateTitles** ()
- void **slotSelectAlbum** (const QUrl &url)
- void **slotSelectAll** ()
- void **slotSelectInvert** ()
- void **slotSelectNone** ()
- void **slotSeparateImages** (int mode)
- void **slotSetAsAlbumThumbnail** (const [ItemInfo](#) &info)
- void **slotSetCurrentUrlWhenAvailable** (const QUrl &url)
- void **slotSetCurrentWhenAvailable** (const qlonglong id)
- void **slotSortAlbums** (int role)
- void **slotSortImages** (int order)
- void **slotSortImagesOrder** (int order)
- void **slotTableView** ()
- void **slotUngroupSelected** ()
- void **slotZoomIn** ()
- void **slotZoomOut** ()
- void **slotZoomTo100Percents** ()

## Signals

- void **signalAlbumSelected** ([Album](#) \*)
- void **signalChangedTab** (QWidget \*)
- void **signalFuzzySidebarActive** (bool active)
- void **signalGotoAlbumAndItem** (const [ItemInfo](#) &)
- void **signalGotoDateAndItem** (AlbumIconItem \*)
- void **signalGotoTagAndItem** (int tagID)
- void **signalImageSelected** (const [ItemInfoList](#) &selectedImage, const [ItemInfoList](#) &allImages)
- void **signalNoCurrentItem** ()
- void **signalSelectionChanged** (int numberOfSelectedItems)
- void **signalSeparationModeChanged** (int category)
- void **signalSwitchedToIconView** ()
- void **signalSwitchedToMapView** ()
- void **signalSwitchedToPreview** ()
- void **signalSwitchedToTableView** ()
- void **signalSwitchedToTrashView** ()
- void **signalThumbSizeChanged** (int)
- void **signalTrashSelectionChanged** (const QString &text)
- void **signalZoomChanged** (double)



## Public Member Functions

- **ItemIconView** (QWidget \*const parent, [DModelFactory](#) \*const modelCollection)
- **ItemInfoList allInfo** (const bool grouping=false) const
- **ItemInfoList allInfo** (const [OperationType](#) type) const
- bool **allNeedGroupResolving** (const [OperationType](#) type) const  
*Item Group methods - itemiconview\_groups.cpp.*
- QList< QUrl > **allUrls** (bool grouping=false) const  
*Get all items in the current view.*
- void **applySettings** ()
- void **clearHistory** ()
- void **connectIconViewFilter** ([FilterStatusBar](#) \*const filter)
- **Album \* currentAlbum** () const  
*Album management methods - itemiconview\_album.cpp.*
- **ItemInfo currentInfo** () const
- QUrl **currentUrl** () const
- void **getBackwardHistory** (QStringList &titles)
- void **getForwardHistory** (QStringList &titles)
- bool **hasCurrentItem** () const
- void **hideSideBars** ()
- void **imageTransform** ([MetaEngineRotation::TransformationAction](#) transform)
- int **itemCount** () const  
*Items management methods - itemiconview\_items.cpp.*
- QList< [SidebarWidget](#) \* > **leftSidebarWidgets** () const
- void **nextLeftSideBarTab** ()
- void **nextRightSideBarTab** ()
- void **previousLeftSideBarTab** ()
- void **previousRightSideBarTab** ()
- void **refreshView** ()
- **ItemInfoList selectedInfoList** (const bool currentFirst=false, const bool grouping=false) const
- **ItemInfoList selectedInfoList** (const [OperationType](#) type, const bool currentFirst=false) const
- bool **selectedNeedGroupResolving** (const [OperationType](#) type) const
- QList< QUrl > **selectedUrls** (bool grouping=false) const  
*Get currently selected items.*
- QList< QUrl > **selectedUrls** (const [OperationType](#) type) const
- void **setAllGroupsOpen** (bool open)
- void **setRecurseAlbums** (bool recursive)
- void **setRecurseTags** (bool recursive)  
*Tags management methods - itemiconview\_tags.cpp.*
- void **setThumbSize** (int size)
- void **setToolsIconView** ([DCategorizedView](#) \*const view)  
*Views management methods - itemiconview\_views.cpp.*
- void **showSideBars** ()
- void **toggleFullScreen** (bool set)
- void **toggleLeftSidebar** ()
- void **toggleRightSidebar** ()
- void **toggleShowBar** (bool)
- void **toggleTag** (int tagID)
- [StackedView::StackedViewMode](#) **viewMode** () const
- double **zoomMax** () const
- double **zoomMin** () const  
*Zoom management methods - itemiconview\_zoom.cpp.*

## Public Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (QWidget \*const parent=nullptr)
- QSize **minimumSizeHint** () const override
- void **setContentsMargins** (const QMargins &margins)
- void **setContentsMargins** (int left, int top, int right, int bottom)
- void **setSpacing** (int space)
- void **setStretchFactor** (QWidget \*const widget, int stretch)
- QSize **sizeHint** () const override

## Additional Inherited Members

## Protected Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (bool vertical, QWidget \*const parent)
- void **childEvent** (QChildEvent \*e) override

## 9.824.1 Member Function Documentation

### 9.824.1.1 allNeedGroupResolving()

```
bool Digikam::ItemIconView::allNeedGroupResolving (
    const OperationType type ) const
```

Query whether the operation to be performed on currently selected or all all items in the currently active view should be performed on all grouped items or just the first.

### 9.824.1.2 allUrls()

```
QList< QUrl > Digikam::ItemIconView::allUrls (
    bool grouping = false ) const
```

Whether only the first or all grouped items are returned is determined as described above.

### 9.824.1.3 selectedUrls()

```
QList< QUrl > Digikam::ItemIconView::selectedUrls (
    bool grouping = false ) const
```

By default only the first images in groups are given, while all can be obtained by setting the grouping parameter to true. Given an operation, it will be determined from settings/user query whether only the first or all items in a group are returned. Ideally only the latter (giving an operation) is used.

### 9.824.1.4 slotFitToWindow

```
void Digikam::ItemIconView::slotFitToWindow ( ) [slot]
```

### 9.824.1.5 slotImageQualitySorter

```
void Digikam::ItemIconView::slotImageQualitySorter ( ) [slot]
```

Tools methods (Editor, BQM, Light Table) - itemiconview\_tools.cpp.

### 9.824.1.6 slotRemoveTag

```
void Digikam::ItemIconView::slotRemoveTag (
    int tagID ) [slot]
```

Implementation for Automatic Icon Removal of Confirmed Tags. QTimer to ensure TagRemoval is complete.

If the face just removed was the final face associated with that Tag, reset Tag Icon.

## 9.825 Digikam::ItemInfo Class Reference

The [ItemInfo](#) class contains provides access to the database for a single image.

### Public Types

- typedef [DatabaseFields::Hash](#)< QVariant > **DatabaseFieldsHashRaw**

### Public Member Functions

- **ItemInfo** ()  
*Constructor Creates a null image info.*
- **ItemInfo** (const [ItemInfo](#) &info)  
*Copy constructor.*
- **ItemInfo** (const [ItemListerRecord](#) &record)  
*Constructor.*
- **ItemInfo** (qulonglong ID)  
*Constructor.*
- **~ItemInfo** ()  
*Destructor.*
- **ItemInfo copyItem** (int dstAlbumID, const QString &dstFileName)  
*Copy database information of this item to a newly created item.*
- bool **isLocationAvailable** () const  
*Returns true if this is a valid [ItemInfo](#), and the location of the image is currently available (information freshly obtained from [CollectionManager](#))*
- bool **operator!=** (const [ItemInfo](#) &info) const
- bool **operator<** (const [ItemInfo](#) &info) const
- [ItemInfo](#) & **operator=** (const [ItemInfo](#) &info)
- bool **operator==** (const [ItemInfo](#) &info) const

### Operations with Properties

- bool **isNull** () const

- Returns if this objects contains valid data.*
- QString **name** () const
- QDateTime **dateTime** () const
- QDateTime **modDateTime** () const
- qulonglong **fileSize** () const
- QSize **dimensions** () const
- QUrl **fileUrl** () const
- Returns the `file://` url.*
- QString **filePath** () const
- Returns the file path to the image.*
- QString **relativePath** () const
- Returns the relative path part to the image.*
- qulonglong **id** () const
- int **albumId** () const
- int **albumRootId** () const
- The album root id.*
- double **aspectRatio** () const
- qulonglong **manualOrder** () const
- Returns the manual sort order.*
- DatabaseItem::Category **category** () const
- Returns the category of the item: Image, Audio, Video.*
- QString **format** () const
- Returns the image format / mimetype as a standardized string (see project/documents/DBSCHEMA.ODS).*
- bool **isVisible** () const
- Returns true if the image is marked as visible in the database.*
- bool **isRemoved** () const
- Returns true if the corresponding file was not deleted.*
- int **orientation** () const
- Returns the orientation of the image, ([MetaEngine::ImageOrientation](#), EXIF standard)*
- QString **title** () const
- QString **comment** () const
- int **faceCount** () const
- int **unconfirmedFaceCount** () const
- QMap< QString, QString > **getSuggestedNames** () const
- void **setName** (const QString &newName)
- Set the name (write it to database)*
- void **setDateTime** (const QDateTime &dateTime)
- Set the date and time (write it to database)*
- void **setModDateTime** (const QDateTime &dateTime)
- Set the modification date and time (write it to database)*
- void **setManualOrder** (qulonglong value)
- Set the manual sorting order for the item.*
- void **setOrientation** (int value)
- Set the orientation for the item.*
- void **setVisible** (bool isVisible)
- Set the visibility flag - triggers between Visible and Hidden.*
- DatabaseFieldsHashRaw **getDatabaseFieldsRaw** (const DatabaseFields::Set &requestedSet) const
- QVariant **getDatabaseFieldRaw** (const DatabaseFields::Set &requestedField) const

### Operations with Geolocation

- [ItemPosition](#) **imagePosition** () const
- Retrieve the [ItemPosition](#) object for this item.*
- double **longitudeNumber** () const
- Retrieves the coordinates and the altitude.*
- double **latitudeNumber** () const
- double **altitudeNumber** () const
- bool **hasCoordinates** () const
- bool **hasAltitude** () const

### Operations with History

- [DImageHistory](#) `imageHistory ()` const  
*Retrieves and sets the image history from the database.*
- void `setItemHistory (const DImageHistory &history)`
- bool `hasImageHistory ()` const
- QString `uuid ()` const  
*Retrieves and sets this' images UUID.*
- void `setUuid (const QString &uuid)`
- [HistoryImageId](#) `historyImageId ()` const  
*Constructs a [HistoryImageId](#) with all available information for this image.*
- bool `hasDerivedImages ()` const  
*Retrieve information about images from which this image is derived (ancestorImages) and images that have been derived from this images (derivedImages).*
- bool `hasAncestorImages ()` const
- QList< [ItemInfo](#) > `derivedImages ()` const
- QList< [ItemInfo](#) > `ancestorImages ()` const
- QList< QPair< qlonglong, qlonglong > > `relationCloud ()` const  
*Returns the cloud of all directly or indirectly related images, derived images or ancestors, in from of "a derived from b" pairs.*
- void `markDerivedFrom (const ItemInfo &ancestorImage)`  
*Add a relation to the database: This image is derived from the ancestorImage.*

### Operations with Groups

- bool `isGrouped ()` const  
*The image is grouped in the group of another (leading) image.*
- bool `hasGroupedImages ()` const  
*The image is the leading image of a group, there are other images grouped behind this one.*
- int `numberOfGroupedImages ()` const
- [ItemInfo](#) `groupImage ()` const  
*Returns the leading image of the group.*
- qlonglong `groupImageId ()` const
- QList< [ItemInfo](#) > `groupedImages ()` const  
*Returns the list of images grouped behind this image (not including this image itself) and an empty list if there is none.*
- void `addToGroup (const ItemInfo &info)`  
*Group this image behind the given image.*
- void `removeFromGroup ()`  
*This image is grouped behind another image: Remove this image from its group.*
- void `clearGroup ()`  
*This image [hasGroupedImages\(\)](#): Split up the group, remove all [groupedImages\(\)](#) from this image's group.*

### Operations with Containers

- [ImageCommonContainer](#) `imageCommonContainer ()` const  
*Retrieve information about the image, in form of numbers and user presentable strings, for certain defined fields of information (see [databaseinfocontainers.h](#))*
- [ImageMetadataContainer](#) `imageMetadataContainer ()` const
- [VideoMetadataContainer](#) `videoMetadataContainer ()` const
- [PhotoInfoContainer](#) `photoInfoContainer ()` const
- [VideoInfoContainer](#) `videoInfoContainer ()` const
- [Template](#) `metadataTemplate ()` const  
*Retrieve metadata template information about the image.*
- void `setMetadataTemplate (const Template &t)`  
*Set metadata template information (write it to database)*
- void `removeMetadataTemplate ()`  
*Remove all template info about the image from database.*
- [ItemComments](#) `imageComments (const CoreDbAccess &access)` const

- Retrieve the *ItemComments* object for this item.
- **ItemCopyright** *imageCopyright* () const  
Retrieve the *ItemCopyright* object for this item.
- **ItemExtendedProperties** *imageExtendedProperties* () const  
Retrieve the *ItemExtendedProperties* object for this item.

### Operations with Labels

- int **pickLabel** () const  
Returns the Pick Label Id (see PickLabel values in globals.h)
- int **colorLabel** () const  
Returns the Color Label Id (see ColorLabel values in globals.h)
- int **rating** () const  
Returns the rating.
- void **setPickLabel** (int value)  
Set the pick Label Id for the item (see PickLabel values from globals.h)
- void **setColorLabel** (int value)  
Set the color Label Id for the item (see ColorLabel values from globals.h)
- void **setRating** (int value)  
Set the rating for the item.

### Static Public Member Functions

- static **ItemInfo** **fromLocalFile** (const QString &path)  
Creates an *ItemInfo* object from a file url.
- static **ItemInfo** **fromLocationAlbumAndName** (int locationId, const QString &album, const QString &name)  
Create an *ItemInfo* object from the given combination, which must be cleaned and corresponding to the values in the database.
- static **ItemInfo** **fromUrl** (const QUrl &url)

### Operations with Similarity

- class **ItemInfoCache**
- class **ItemInfoList**
- double **similarityTo** (const qlonglong imageId) const
- double **currentSimilarity** () const
- qlonglong **currentReferenceImage** () const  
Returns the id of the current fuzzy search reference image.
- size\_t **hash** () const  
Return a signature for the item.
- QList< **ItemInfo** > **fromUniqueHash** (const QString &uniqueHash, qlonglong fileSize)  
Scans the database for items with the given signature.
- QString **uniqueHash** () const

### Operations with Tags

- void **setTag** (int tagId)  
Adds a tag to the item (writes it to database)
- void **addTagPaths** (const QStringList &tagPaths)  
Adds tags in the list to the item.
- void **removeTag** (int tagId)  
Remove a tag from the item (removes it from database)
- void **removeAllTags** ()  
Remove all tags from the item (removes it from database)
- **ItemTagPair** **imageTagPair** (int tagId) const  
Retrieve an *ItemTagPair* object for a single tag, or for all image/tag pairs for which properties are available (not necessarily the assigned tags)
- QList< **ItemTagPair** > **availableItemTagPairs** () const
- QList< int > **tagIds** () const

## Operations with Thumbnails

- [ThumbnailIdentifier](#) `thumbnailIdentifier ()` const  
*Fills a [ThumbnailIdentifier](#) / [ThumbnailInfo](#) from this [ItemInfo](#).*
- [ThumbnailInfo](#) `thumbnailInfo ()` const
- static [ThumbnailIdentifier](#) `thumbnailIdentifier (qulonglong id)`

### 9.825.1 Detailed Description

The properties can be read and written. Information will be cached.

#### Note

access rules for all methods in this class: [ItemInfoData](#) members shall be accessed only under [CoreDbAccess](#) lock. The id and albumId are the exception to this rule, as they are primitive and will never change during the lifetime of an object.

### 9.825.2 Constructor & Destructor Documentation

#### 9.825.2.1 ItemInfo() [1/2]

```
Digikam::ItemInfo::ItemInfo (
    qulonglong ID ) [explicit]
```

Creates an [ItemInfo](#) object without any cached data initially.

#### Parameters

<i>ID</i>	the unique ID for this image
-----------	------------------------------

#### 9.825.2.2 ItemInfo() [2/2]

```
Digikam::ItemInfo::ItemInfo (
    const ItemLISTERRecord & record ) [explicit]
```

Creates an [ItemInfo](#) object where the provided information will initially be available cached, without database access.

### 9.825.3 Member Function Documentation

#### 9.825.3.1 addTagPaths()

```
void Digikam::ItemInfo::addTagPaths (
    const QStringList & tagPaths )
```

Tags are created if they do not yet exist

### 9.825.3.2 albumId()

```
int Digikam::ItemInfo::albumId ( ) const
```

#### Returns

the id of the [PAlbum](#) to which this item belongs

### 9.825.3.3 aspectRatio()

```
double Digikam::ItemInfo::aspectRatio ( ) const
```

#### Returns

the id of the Aspect Ratio for this item

### 9.825.3.4 comment()

```
QString Digikam::ItemInfo::comment ( ) const
```

#### Returns

the default comment for this item

### 9.825.3.5 copyItem()

```
ItemInfo Digikam::ItemInfo::copyItem (
    int dstAlbumID,
    const QString & dstFileName )
```

#### Parameters

<i>dstAlbumID</i>	destination album id
<i>dstFileName</i>	new filename

#### Returns

an [ItemInfo](#) object of the new item

### 9.825.3.6 dateTime()

```
QDateTime Digikam::ItemInfo::dateTime ( ) const
```

#### Returns

the datetime of the image



### 9.825.3.7 dimensions()

```
QSize Digikam::ItemInfo::dimensions ( ) const
```

#### Returns

the dimensions of the image (valid only if dimensions have been requested)

### 9.825.3.8 faceCount()

```
int Digikam::ItemInfo::faceCount ( ) const
```

#### Returns

the number of Faces in this item.

### 9.825.3.9 fileSize()

```
qulonglong Digikam::ItemInfo::fileSize ( ) const
```

#### Returns

the filesize of the image

### 9.825.3.10 fileUrl()

```
QUrl Digikam::ItemInfo::fileUrl ( ) const
```

This is equivalent to `QUrl::fromLocalFile(filePath())`

### 9.825.3.11 getDatabaseFieldsRaw()

```
ItemInfo::DatabaseFieldsHashRaw Digikam::ItemInfo::getDatabaseFieldsRaw (
    const DatabaseFields::Set & requestedSet ) const
```

### 9.825.3.12 getSuggestedNames()

```
QMap< QString, QString > Digikam::ItemInfo::getSuggestedNames ( ) const
```

#### Returns

the map of Tag Region (in XML form) to Suggested Names for all Faces in the Image. Used to categorize images based on Face Suggestions.

### 9.825.3.13 groupImage()

```
ItemInfo Digikam::ItemInfo::groupImage ( ) const
```

Returns a null image if this image is not grouped ([isGrouped\(\)](#))

### 9.825.3.14 id()

```
qlonglong Digikam::ItemInfo::id ( ) const
```

#### Returns

the unique image id for this item

### 9.825.3.15 imageComments()

```
ItemComments Digikam::ItemInfo::imageComments (
    const CoreDbAccess & access ) const
```

This object allows full read and write access to all comments and their properties. You need to hold [CoreDbAccess](#) to ensure the validity. For simple, cached read access see [comment\(\)](#).

### 9.825.3.16 imageCopyright()

```
ItemCopyright Digikam::ItemInfo::imageCopyright ( ) const
```

This object allows full read and write access to all copyright values.

### 9.825.3.17 imageExtendedProperties()

```
ItemExtendedProperties Digikam::ItemInfo::imageExtendedProperties ( ) const
```

This object allows full read and write access to all extended properties values.

### 9.825.3.18 imageHistory()

```
DImageHistory Digikam::ItemInfo::imageHistory ( ) const
```

Note: The image history retrieved here does typically include all steps from the original to this image, but does not reference this image itself.

### 9.825.3.19 longitudeNumber()

```
double Digikam::ItemInfo::longitudeNumber ( ) const
```

Returns 0 if [hasCoordinates\(\)](#), or [hasAltitude](#) resp, is false.

### 9.825.3.20 modDateTime()

```
QDateTime Digikam::ItemInfo::modDateTime ( ) const
```

#### Returns

the modification datetime of the image

### 9.825.3.21 name()

```
QString Digikam::ItemInfo::name ( ) const
```

#### Returns

the name of the image

### 9.825.3.22 removeTag()

```
void Digikam::ItemInfo::removeTag (
    int tagID )
```

#### Parameters

<i>tagID</i>	the ID of the tag to remove
--------------	-----------------------------

### 9.825.3.23 setDateTime()

```
void Digikam::ItemInfo::setDateTime (
    const QDateTime & dateTime )
```

#### Parameters

<i>dateTime</i>	the new date and time.
-----------------	------------------------

### 9.825.3.24 setMetadataTemplate()

```
void Digikam::ItemInfo::setMetadataTemplate (
    const Template & t )
```

#### Parameters

<i>t</i>	the new template data.
----------	------------------------

### 9.825.3.25 setModDateTime()

```
void Digikam::ItemInfo::setModDateTime (
    const QDateTime & dateTime )
```

#### Parameters

<i>dateTime</i>	the new modification date and time.
-----------------	-------------------------------------

### 9.825.3.26 setName()

```
void Digikam::ItemInfo::setName (
    const QString & newName )
```

#### Parameters

<i>newName</i>	the new name.
----------------	---------------

### 9.825.3.27 setTag()

```
void Digikam::ItemInfo::setTag (
    int tagID )
```

#### Parameters

<i>tagID</i>	the ID of the tag to add
--------------	--------------------------

### 9.825.3.28 tagIds()

```
QList< int > Digikam::ItemInfo::tagIds ( ) const
```

#### Returns

a list of IDs of tags assigned to this item

#### See also

[tagNames](#)

[tagPaths](#)

[Album::id\(\)](#)

### 9.825.3.29 title()

```
QString Digikam::ItemInfo::title ( ) const
```

#### Returns

the default title for this item

**9.825.3.30 unconfirmedFaceCount()**

```
int Digikam::ItemInfo::unconfirmedFaceCount ( ) const
```

**Returns**

the number of Unconfirmed Faces in this item.

**9.825.3.31 uniqueHash()**

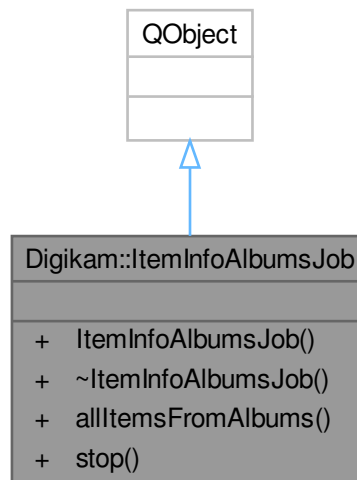
```
QString Digikam::ItemInfo::uniqueHash ( ) const
```

**Returns**

the unique hash signature as string of the image.

**9.826 Digikam::ItemInfoAlbumsJob Class Reference**

Inheritance diagram for Digikam::ItemInfoAlbumsJob:

**Signals**

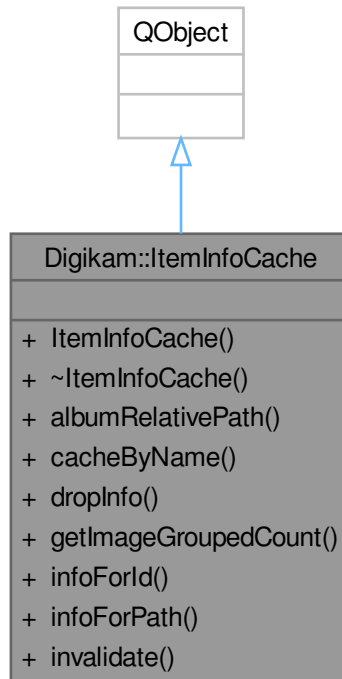
- void **signalCompleted** (const [ItemInfoList](#) &items)

**Public Member Functions**

- **ItemInfoAlbumsJob** (QObject \*const parent=nullptr)
- void **allItemsFromAlbums** (const AlbumList &albumsList)
- void **stop** ()

## 9.827 Digikam::ItemInfoCache Class Reference

Inheritance diagram for Digikam::ItemInfoCache:



### Public Member Functions

- `QString` **albumRelativePath** (int albumId)  
*Returns the cached relativePath for the given album id.*
- void **cacheByName** (const QExplicitlySharedDataPointer< [ItemInfoData](#) > &infoPtr)  
*Call this to put data in the hash by file name if you have newly created data and the name is filled.*
- void **dropInfo** (const QExplicitlySharedDataPointer< [ItemInfoData](#) > &infoPtr)  
*Call this when the data has been dereferenced, before deletion.*
- int **getImageGroupedCount** (qlonglong id)  
*Returns the cached grouped count for the given image id.*
- QExplicitlySharedDataPointer< [ItemInfoData](#) > **infoForId** (qlonglong id)  
*Return an [ItemInfoData](#) object for the given image id.*
- QExplicitlySharedDataPointer< [ItemInfoData](#) > **infoForPath** (int albumRootId, const QString &relativePath, const QString &name)  
*Return an [ItemInfoData](#) object for the given album root, relativePath and file name triple.*
- void **invalidate** ()  
*Invalidate the cache and all its cached data.*

## 9.827.1 Member Function Documentation

### 9.827.1.1 cacheByName()

```
void Digikam::ItemInfoCache::cacheByName (
    const QExplicitlySharedDataPointer< ItemInfoData > & infoPtr )
```

Call under write lock.

### 9.827.1.2 infoForId()

```
QExplicitlySharedDataPointer< ItemInfoData > Digikam::ItemInfoCache::infoForId (
    qlonglong id )
```

A new object is created, or an existing object is returned. If a new object is created, the id field will be initialized.

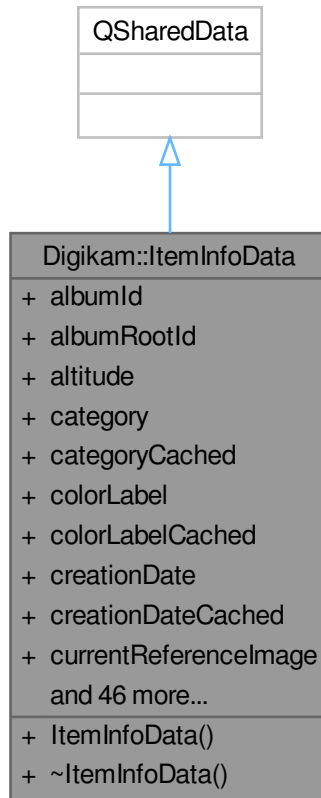
### 9.827.1.3 infoForPath()

```
QExplicitlySharedDataPointer< ItemInfoData > Digikam::ItemInfoCache::infoForPath (
    int albumRootId,
    const QString & relativePath,
    const QString & name )
```

Works if previously cached with cacheByName. Returns 0 if not found.

## 9.828 Digikam::ItemInfoData Class Reference

Inheritance diagram for Digikam::ItemInfoData:



### Public Types

- typedef `DatabaseFields::Hash` < QVariant > `DatabaseFieldsHashRaw`

### Public Attributes

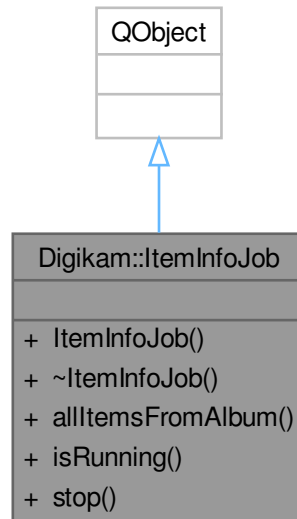
- int `albumId` = -1
- int `albumRootId` = -1
- double `altitude` = 0
- DatabaseItem::Category `category` = DatabaseItem::UndefinedCategory
- bool `categoryCached` = false
- quint8 `colorLabel` = NoColorLabel
- bool `colorLabelCached` = false
- QDateTime `creationDate`
- bool `creationDateCached` = false
- qlonglong `currentReferenceImage` = -1
- double `currentSimilarity` = 0.0



- [DatabaseFieldsHashRaw](#) **databaseFieldsHashRaw**
- QString **defaultComment**
- bool **defaultCommentCached** = false
- QString **defaultTitle**
- bool **defaultTitleCached** = false
- int **faceCount** = 0
- bool **faceCountCached** = false
- QMap< QString, QString > **faceSuggestions**
- bool **faceSuggestionsCached** = false
- qlonglong **fileSize** = 0
- bool **fileSizeCached** = false
- QString **format**
- bool **formatCached** = false
- qlonglong **groupImage** = -1
  - group leader, if the image is grouped*
- bool **groupImageCached** = false
- bool **hasAltitude** = false
- bool **hasCoordinates** = false
- bool **hasImageMetadata** = true
- bool **hasVideoMetadata** = true
- qlonglong **id** = -1
- DatabaseFields::ImageMetadataMinSizeType **imageMetadataCached** = DatabaseFields::ImageMetadata↔None
- QSize **imageSize**
- bool **imageSizeCached** = false
- bool **invalid** = false
- double **latitude** = 0
- double **longitude** = 0
- qlonglong **manualOrder** = 0
- bool **manualOrderCached** = false
- QDateTime **modificationDate**
- bool **modificationDateCached** = false
- QString **name**
- int **orientation** = 0
- bool **orientationCached** = false
- quint8 **pickLabel** = NoPickLabel
- bool **pickLabelCached** = false
- bool **positionsCached** = false
- quint8 **rating** = -1
- bool **ratingCached** = false
- QList< int > **tagIds**
- bool **tagIdsCached** = false
- int **unconfirmedFaceCount** = 0
- bool **unconfirmedFaceCountCached** = false
- QString **uniqueHash**
- bool **uniqueHashCached** = false
- DatabaseFields::VideoMetadataMinSizeType **videoMetadataCached** = DatabaseFields::VideoMetadata↔None

## 9.829 Digikam::ItemInfoJob Class Reference

Inheritance diagram for Digikam::ItemInfoJob:



### Signals

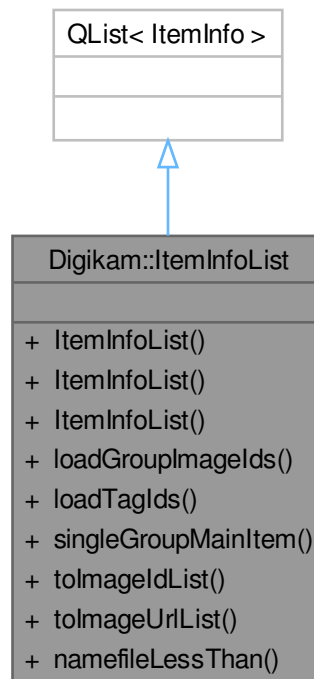
- void **signalCompleted** ()
- void **signalItemsInfo** (const [ItemInfoList](#) &items)

### Public Member Functions

- void **allItemsFromAlbum** ([Album](#) \*const album)
- bool **isRunning** () const
- void **stop** ()

## 9.830 Digikam::ItemInfoList Class Reference

Inheritance diagram for Digikam::ItemInfoList:



### Public Member Functions

- `ItemInfoList` (const QList< [ItemInfo](#) > &list)
- `ItemInfoList` (const QList< qlonglong > &idList)
- void `loadGroupImagelds` () const
- void `loadTagIds` () const
- `ItemInfo` `singleGroupMainItem` () const  
*singleGroupMainItem*
- QList< qlonglong > `toImageldList` () const
- QList< QUrl > `toImageUrlList` () const

### Static Public Member Functions

- static bool `namefileLessThan` (const [ItemInfo](#) &d1, const [ItemInfo](#) &d2)

## 9.830.1 Member Function Documentation

### 9.830.1.1 singleGroupMainItem()

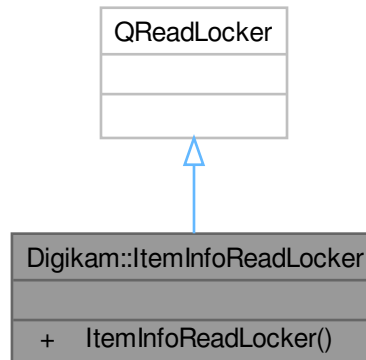
`ItemInfo` Digikam::ItemInfoList::singleGroupMainItem ( ) const

#### Returns

If the list contains of items of only one group including the main item, this main item is returned, otherwise a null [ItemInfo](#).

## 9.831 Digikam::ItemInfoReadLocker Class Reference

Inheritance diagram for Digikam::ItemInfoReadLocker:



## 9.832 Digikam::ItemInfoSet Class Reference

A container of associated [ItemInfo](#) and queue id.

### Public Member Functions

- `ItemInfoSet` (int id, const [ItemInfo](#) &inf)

### Public Attributes

- [ItemInfo](#) `info`
- int `queueId = 0`

## 9.833 Digikam::ItemInfoStatic Class Reference

### Static Public Member Functions

- static [ItemInfoCache](#) \* `cache` ()
- static void `create` ()
- static void `destroy` ()

### Public Attributes

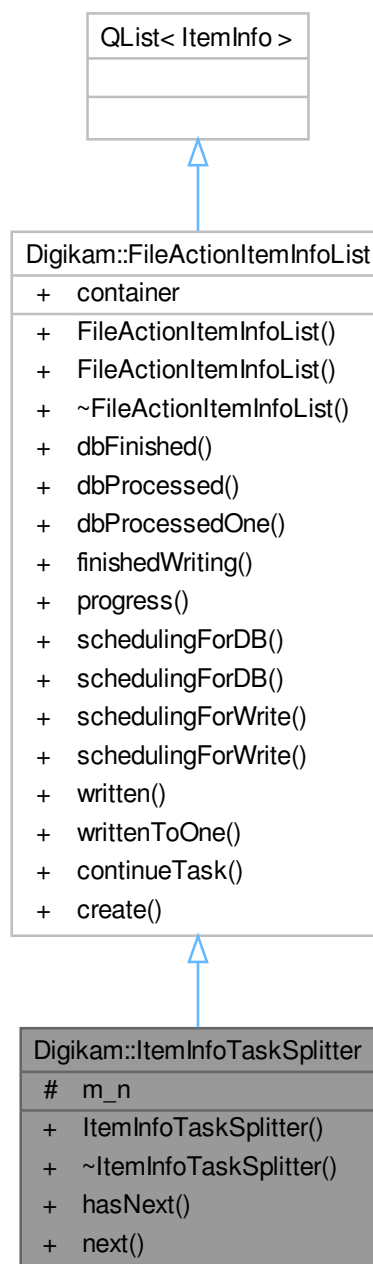
- [ItemInfoCache](#) `m_cache`
- `QReadWriteLock` `m_lock`

## Static Public Attributes

- static [ItemInfoStatic](#) \* `m_instance` = nullptr

## 9.834 Digikam::ItemInfoTaskSplitter Class Reference

Inheritance diagram for Digikam::ItemInfoTaskSplitter:



## Public Member Functions

- **ItemInfoTaskSplitter** (const [FileActionItemInfoList](#) &list)
- bool **hasNext** () const
- [FileActionItemInfoList](#) **next** ()

## Public Member Functions inherited from [Digikam::FileActionItemInfoList](#)

- **FileActionItemInfoList** (const [FileActionItemInfoList](#) &copy)
- void **dbFinished** () const
- void **dbProcessed** (int numberOfInfos) const
- void **dbProcessedOne** () const  
*db worker progress info*
- void **finishedWriting** () const
- [FileActionProgressItemContainer](#) \* **progress** () const
- void **schedulingForDB** (const QString &action, [FileActionProgressItemCreator](#) \*const creator)
- void **schedulingForDB** (int numberOfInfos, const QString &action, [FileActionProgressItemCreator](#) \*const creator)  
*before sending to db worker*
- void **schedulingForWrite** (const QString &action, [FileActionProgressItemCreator](#) \*const creator) const
- void **schedulingForWrite** (int numberOfInfos, const QString &action, [FileActionProgressItemCreator](#) \*const creator) const  
*db worker calls this before sending to file worker*
- void **written** (int numberOfInfos) const
- void **writtenToOne** () const  
*file worker calls this when finished*

## Protected Attributes

- int **m\_n** = 1

## Additional Inherited Members

## Static Public Member Functions inherited from [Digikam::FileActionItemInfoList](#)

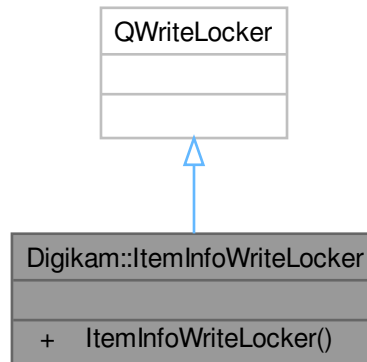
- static [FileActionItemInfoList](#) **continueTask** (const QList< [ItemInfo](#) > &list, [FileActionProgressItemContainer](#) \*const container)
- static [FileActionItemInfoList](#) **create** (const QList< [ItemInfo](#) > &list)

## Public Attributes inherited from [Digikam::FileActionItemInfoList](#)

- QExplicitlySharedDataPointer< [FileActionProgressItemContainer](#) > **container**

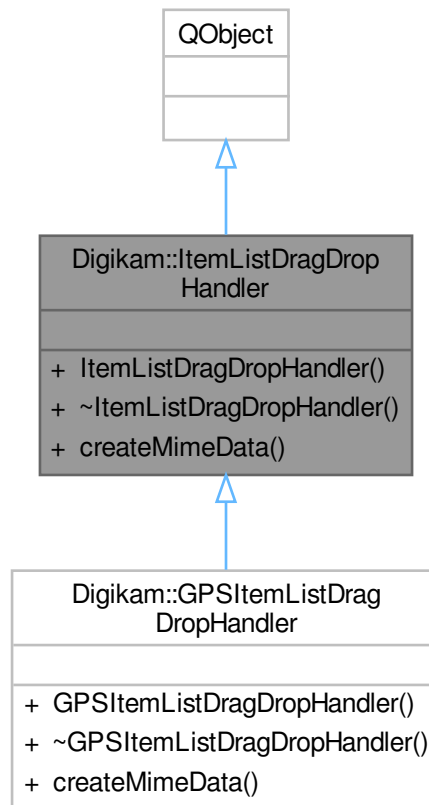
## 9.835 Digikam::ItemInfoWriteLocker Class Reference

Inheritance diagram for Digikam::ItemInfoWriteLocker:



## 9.836 Digikam::ItemListDragDropHandler Class Reference

Inheritance diagram for Digikam::ItemListDragDropHandler:



### Public Member Functions

- **ItemListDragDropHandler** (`QObject *const parent=nullptr`)
- virtual `QMimeData * createMimeData` (`const QList< QPersistentModelIndex > &modelIndices`)=0

## 9.837 Digikam::ItemLISTER Class Reference

### Public Member Functions

- void **list** (`ItemLISTERReceiver *const receiver`, `const CoreDbUrl &url`)  
*Convenience method for Album, Tag and Date URLs, not for Search URLs.*
- void **listDateRange** (`ItemLISTERReceiver *const receiver`, `const QDate &startDate`, `const QDate &endDate`)  
*List those images whose date lies in the range beginning with startDate (inclusive) and ending before endDate (exclusive).*
- void **setListOnlyAvailable** (`bool listOnlyAvailable`)



Adjust the setting if images from collections that are currently not in the state "available" will be included in the listing.

- void `setRecursive` (bool recursive)

Adjust the setting if album or tags will be listed recursively (i.e.

### Operations with TAlbum

- void `listTag` (`ItemListerReceiver` \*const receiver, const QList< int > &tagIds)  
List the images which have assigned the tags specified by tagIds Updated to support multiple tags.
- void `listImageTagPropertySearch` (`ItemListerReceiver` \*const receiver, const QString &xml)  
Execute the search specified by search XML describing a Tag Properties search.
- QString `tagSearchXml` (int tagId, const QString &type, bool includeChildTags) const

### Operations with SAlbum

- void `listSearch` (`ItemListerReceiver` \*const receiver, const QString &xml, int limit=0, qlonglong reference←  
ImageId=-1)  
Execute the search specified by search XML.
- void `listHaarSearch` (`ItemListerReceiver` \*const receiver, const QString &xml)  
Execute the search specified by search XML describing a Haar search.
- void `listAreaRange` (`ItemListerReceiver` \*const receiver, double lat1, double lat2, double lon1, double lon2)  
List the images whose coordinates are between coordinates contained in areaCoordinates(lat1, lat2, lng1, lng2).

### Operations with PAlbum

- void `listPAlbum` (`ItemListerReceiver` \*const receiver, int albumRootId, const QString &album)  
List images in the *Album* (physical album) specified by albumRoot, album.

## 9.837.1 Member Function Documentation

### 9.837.1.1 listHaarSearch()

```
void Digikam::ItemLister::listHaarSearch (
    ItemListerReceiver *const receiver,
    const QString & xml )
```

#### Parameters

<i>receiver</i>	the receiver for the searches
<i>xml</i>	SearchXml describing the query

### 9.837.1.2 listImageTagPropertySearch()

```
void Digikam::ItemLister::listImageTagPropertySearch (
    ItemListerReceiver *const receiver,
    const QString & xml )
```

Two special add-ons: Non-unique by image id; if enabled, uses the extended ImageRecord protocol to pass the property value in the record's extraValue.

## Parameters

<i>receiver</i>	the receiver for the searches
<i>xml</i>	SearchXml describing the query

**9.837.1.3 listPAlbum()**

```
void Digikam::ItemLister::listPAlbum (
    ItemListerReceiver *const receiver,
    int albumRootId,
    const QString & album )
```

The results will be fed to the specified receiver.

**9.837.1.4 listSearch()**

```
void Digikam::ItemLister::listSearch (
    ItemListerReceiver *const receiver,
    const QString & xml,
    int limit = 0,
    qlonglong referenceImageId = -1 )
```

## Parameters

<i>receiver</i>	the receiver for the searches
<i>xml</i>	SearchXml describing the query
<i>limit</i>	the limit the count of the result set. If limit = 0, then no limit is set.
<i>referenceImageId</i>	the id of a reference image in the search query.

**9.837.1.5 setListOnlyAvailable()**

```
void Digikam::ItemLister::setListOnlyAvailable (
    bool listOnlyAvailable )
```

Default: true.

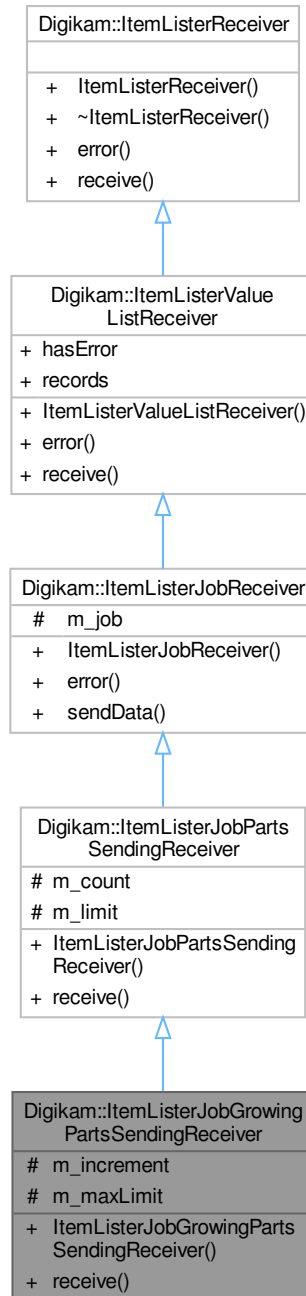
**9.837.1.6 setRecursive()**

```
void Digikam::ItemLister::setRecursive (
    bool recursive )
```

including subalbums / subtags)

## 9.838 Digikam::ItemListerJobGrowingPartsSendingReceiver Class Reference

Inheritance diagram for Digikam::ItemListerJobGrowingPartsSendingReceiver:



### Public Member Functions

- `ItemListerJobGrowingPartsSendingReceiver` (`DBJob *const job, int start, int end, int increment`)
- void `receive` (`const ItemListerRecord &record`) override

## Public Member Functions inherited from [Digikam::ItemListerJobPartsSendingReceiver](#)

- [ItemListerJobPartsSendingReceiver](#) ([DBJob](#) \*const job, int limit)
- void [receive](#) (const [ItemListerRecord](#) &record) override

## Public Member Functions inherited from [Digikam::ItemListerJobReceiver](#)

- [ItemListerJobReceiver](#) ([DBJob](#) \*const job)
- void [error](#) (const QString &errMsg) override
- void [sendData](#) ()

## Public Member Functions inherited from [Digikam::ItemListerValueListReceiver](#)

- void [error](#) (const QString &errMsg) override
- void [receive](#) (const [ItemListerRecord](#) &record) override

## Protected Attributes

- int [m\\_increment](#) = 0
- int [m\\_maxLimit](#) = 0

## Protected Attributes inherited from [Digikam::ItemListerJobPartsSendingReceiver](#)

- int [m\\_count](#) = 0
- int [m\\_limit](#) = 0

## Protected Attributes inherited from [Digikam::ItemListerJobReceiver](#)

- [DBJob](#) \*const [m\\_job](#) = nullptr

## Additional Inherited Members

## Public Attributes inherited from [Digikam::ItemListerValueListReceiver](#)

- bool [hasError](#) = false
- QList< [ItemListerRecord](#) > [records](#)

## 9.838.1 Member Function Documentation

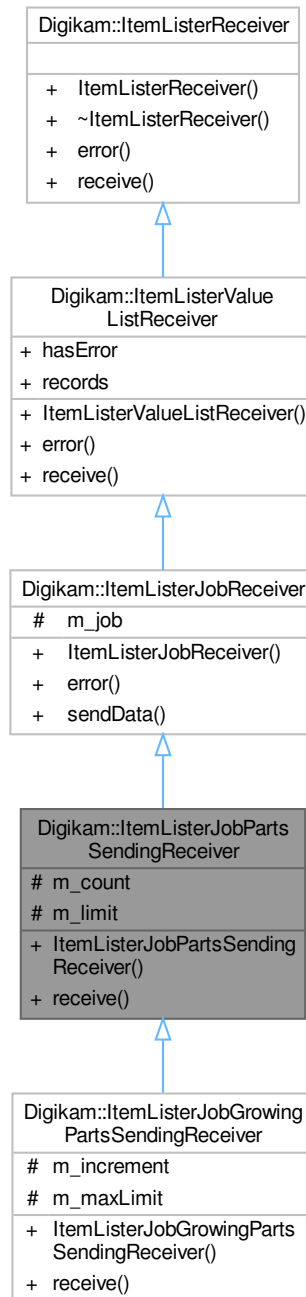
### 9.838.1.1 [receive\(\)](#)

```
void Digikam::ItemListerJobGrowingPartsSendingReceiver::receive (
    const ItemListerRecord & record ) [override], [virtual]
```

Implements [Digikam::ItemListerReceiver](#).

## 9.839 Digikam::ItemListerJobPartsSendingReceiver Class Reference

Inheritance diagram for Digikam::ItemListerJobPartsSendingReceiver:



### Public Member Functions

- `ItemListerJobPartsSendingReceiver` (`DBJob *const job`, `int limit`)
- void `receive` (`const ItemListerRecord &record`) override

## Public Member Functions inherited from [Digikam::ItemListerJobReceiver](#)

- [ItemListerJobReceiver](#) ([DBJob](#) \*const job)
- void [error](#) (const QString &errMsg) override
- void [sendData](#) ()

## Public Member Functions inherited from [Digikam::ItemListerValueListReceiver](#)

- void [error](#) (const QString &errMsg) override
- void [receive](#) (const [ItemListerRecord](#) &record) override

## Protected Attributes

- int [m\\_count](#) = 0
- int [m\\_limit](#) = 0

## Protected Attributes inherited from [Digikam::ItemListerJobReceiver](#)

- [DBJob](#) \*const [m\\_job](#) = nullptr

## Additional Inherited Members

## Public Attributes inherited from [Digikam::ItemListerValueListReceiver](#)

- bool [hasError](#) = false
- QList< [ItemListerRecord](#) > [records](#)

## 9.839.1 Member Function Documentation

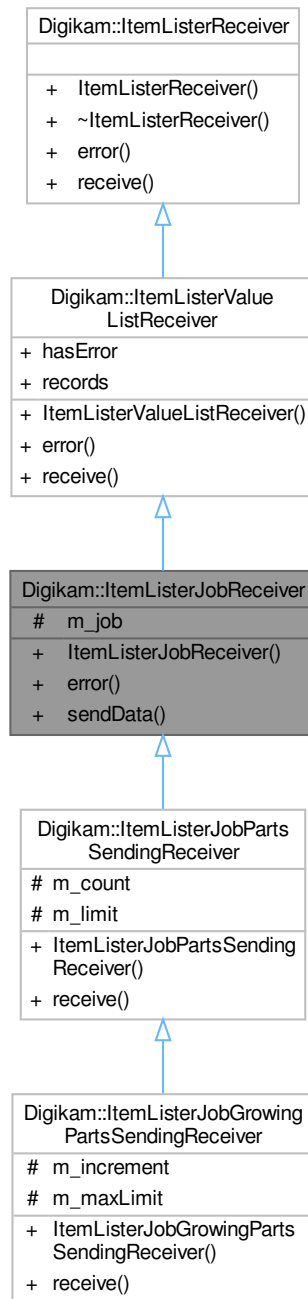
### 9.839.1.1 [receive\(\)](#)

```
void Digikam::ItemListerJobPartsSendingReceiver::receive (  
    const ItemListerRecord & record ) [override], [virtual]
```

Implements [Digikam::ItemListerReceiver](#).

## 9.840 Digikam::ItemListerJobReceiver Class Reference

Inheritance diagram for Digikam::ItemListerJobReceiver:



### Public Member Functions

- **ItemListerJobReceiver** ([DBJob](#) \*const job)
- void **error** (const QString &errMsg) override
- void **sendData** ()

## Public Member Functions inherited from [Digikam::ItemListerValueListReceiver](#)

- void [error](#) (const QString &errMsg) override
- void [receive](#) (const [ItemListerRecord](#) &record) override

## Protected Attributes

- [DBJob](#) \*const [m\\_job](#) = nullptr

## Additional Inherited Members

## Public Attributes inherited from [Digikam::ItemListerValueListReceiver](#)

- bool [hasError](#) = false
- QList< [ItemListerRecord](#) > [records](#)

## 9.840.1 Member Function Documentation

### 9.840.1.1 error()

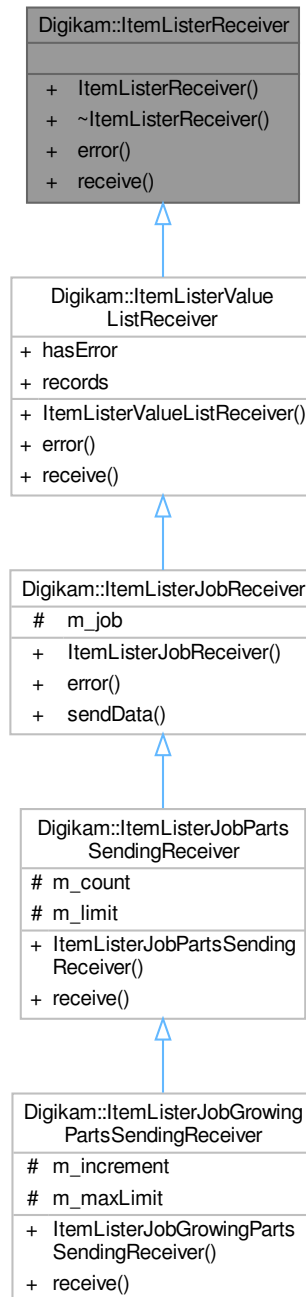
```
void Digikam::ItemListerJobReceiver::error (
    const QString & errMsg ) [override], [virtual]
```

Reimplemented from [Digikam::ItemListerReceiver](#).



## 9.841 Digikam::ItemListerReceiver Class Reference

Inheritance diagram for Digikam::ItemListerReceiver:



### Public Member Functions

- virtual void **error** (const QString &)
- virtual void **receive** (const [ItemListerRecord](#) &record)=0

## 9.842 Digikam::ItemListerRecord Class Reference

### Public Member Functions

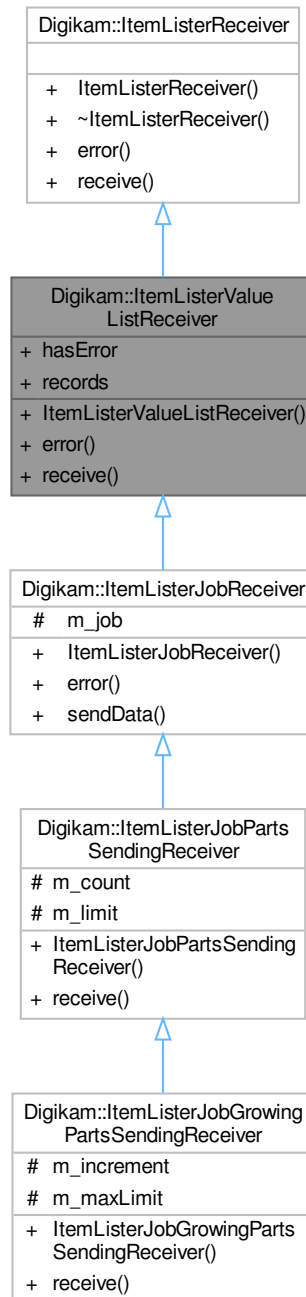
- bool **operator==** (const [ItemListerRecord](#) &record) const

### Public Attributes

- int **albumID** = -1
- int **albumRootID** = -1
- DatabaseItem::Category **category** = DatabaseItem::UndefinedCategory
- QDateTime **creationDate**
- qlonglong **currentReferenceImage** = -1
- double **currentSimilarity** = 0.0
- QList< QVariant > **extraValues**
- qlonglong **fileSize** = -1
- QString **format**
- qlonglong **imageID** = -1
- QSize **imageSize**
- QDateTime **modificationDate**
- QString **name**
- int **rating** = -1

## 9.843 Digikam::ItemListerValueListReceiver Class Reference

Inheritance diagram for Digikam::ItemListerValueListReceiver:



### Public Member Functions

- void `error` (const QString &errMsg) override
- void `receive` (const `ItemListerRecord` &record) override

## Public Attributes

- bool **hasError** = false
- QList< [ItemLISTERRecord](#) > **records**

## 9.843.1 Member Function Documentation

### 9.843.1.1 error()

```
void Digikam::ItemLISTERValueListReceiver::error (
    const QString & errMsg ) [override], [virtual]
```

Reimplemented from [Digikam::ItemLISTERReceiver](#).

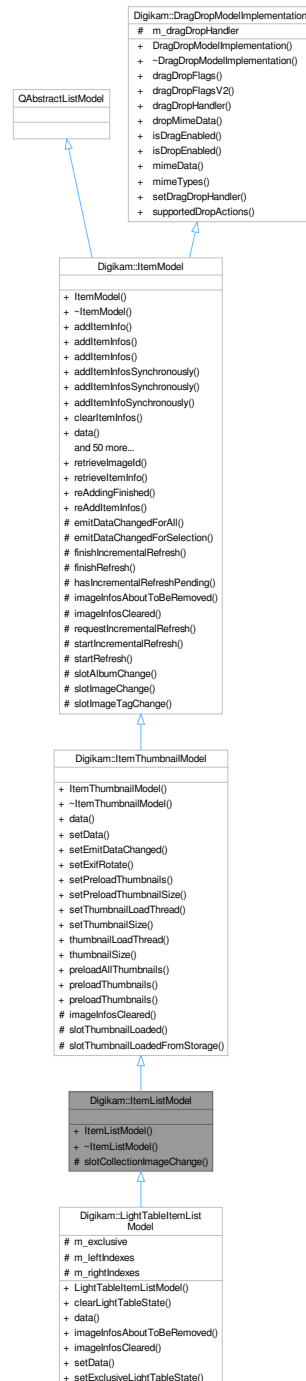
### 9.843.1.2 receive()

```
void Digikam::ItemLISTERValueListReceiver::receive (
    const ItemLISTERRecord & record ) [override], [virtual]
```

Implements [Digikam::ItemLISTERReceiver](#).

## 9.844 Digikam::ItemListModel Class Reference

Inheritance diagram for Digikam::ItemListModel:



### Signals

- void **imageInfosRemoved** (const QList< [ItemInfo](#) > &infos)

*Emitted when images are removed from the model because they are removed in the database.*

## Signals inherited from [Digikam::ItemThumbnailModel](#)

- void **thumbnailAvailable** (const QModelIndex &index, int requestedSize)
- void **thumbnailFailed** (const QModelIndex &index, int requestedSize)

## Signals inherited from [Digikam::ItemModel](#)

- void **allRefreshingFinished** ()  
*Signals that the model has finished currently with all scheduled refreshing, full or incremental, and all preprocessing.*
- void **imageChange** (const ImageChangeset &, const QItemSelection &)  
*If an [ImageChangeset](#) affected indexes of this model with changes as set in [watchFlags\(\)](#), this signal contains the [changeset](#) and the affected indexes.*
- void **imageInfosAboutToBeAdded** (const QList< [ItemInfo](#) > &infos)  
*Informs that [ItemInfos](#) will be added to the model.*
- void **imageInfosAboutToBeRemoved** (const QList< [ItemInfo](#) > &infos)  
*Informs that [ItemInfos](#) will be removed from the model.*
- void **imageInfosAdded** (const QList< [ItemInfo](#) > &infos)  
*Informs that [ItemInfos](#) have been added to the model.*
- void **imageInfosRemoved** (const QList< [ItemInfo](#) > &infos)  
*Informs that [ItemInfos](#) have been removed from the model.*
- void **imageTagChange** (const ImageTagChangeset &, const QItemSelection &)  
*If an [ImageTagChangeset](#) affected indexes of this model, this signal contains the [changeset](#) and the affected indexes.*
- void **preprocess** (const QList< [ItemInfo](#) > &infos, const QList< QVariant > &)  
*Connect to this signal only if you are the current preprocessor.*
- void **processAdded** (const QList< [ItemInfo](#) > &infos, const QList< QVariant > &)
- void **readyForIncrementalRefresh** ()  
*Signals that the model is right now ready to start an incremental refresh.*

## Public Member Functions

- **ItemListModel** (QWidget \*const parent)

## Public Member Functions inherited from [Digikam::ItemThumbnailModel](#)

- **ItemThumbnailModel** (QWidget \*const parent)  
*An [ItemModel](#) that supports thumbnail loading.*
- QVariant **data** (const QModelIndex &index, int role=Qt::DisplayRole) const override  
*Handles the [ThumbnailRole](#).*
- bool **setData** (const QModelIndex &index, const QVariant &value, int role=Qt::DisplayRole) override  
*You can override the current thumbnail size by giving an integer value for [ThumbnailRole](#).*
- void **setEmitDataChanged** (bool emitSignal)  
*Enable emitting [dataChanged\(\)](#) when a thumbnail becomes available.*
- void **setExifRotate** (bool rotate)
- void **setPreloadThumbnails** (bool preload)  
*Enable preloading of thumbnails: If preloading is enabled, for every entry in the model a thumbnail generation is started.*
- void **setPreloadThumbnailSize** (const [ThumbnailSize](#) &thumbSize)  
*If you want to fix a size for preloading, do it here.*
- void **setThumbnailLoadThread** ([ThumbnailLoadThread](#) \*const thread)  
*Enable thumbnail loading and set the thread that shall be used.*
- void **setThumbnailSize** (const [ThumbnailSize](#) &thumbSize)  
*Set the thumbnail size to use.*
- [ThumbnailLoadThread](#) \* **thumbnailLoadThread** () const
- [ThumbnailSize](#) **thumbnailSize** () const

## Public Member Functions inherited from Digikam::ItemModel

- **ItemModel** (QObject \*const parent=nullptr)
- void **addItemInfo** (const [ItemInfo](#) &info)
  - Main entry point for subclasses adding image infos to the model.*
- void **addItemInfos** (const QList< [ItemInfo](#) > &infos)
- void **addItemInfos** (const QList< [ItemInfo](#) > &infos, const QList< QVariant > &extraValues)
- void **addItemInfosSynchronously** (const QList< [ItemInfo](#) > &infos)
- void **addItemInfosSynchronously** (const QList< [ItemInfo](#) > &infos, const QList< QVariant > &extraValues)
- void **addItemInfoSynchronously** (const [ItemInfo](#) &info)
  - addItemInfo() is asynchronous if a preprocessor is set.*
- void **clearItemInfos** ()
  - Clears image infos and resets model.*
- QVariant **data** (const QModelIndex &index, int role=Qt::DisplayRole) const override
- void **ensureHasGroupedImages** (const [ItemInfo](#) &groupLeader)
  - Ensure that all images grouped on the given leader are contained in the model.*
- void **ensureHasItemInfo** (const [ItemInfo](#) &info)
  - Add the given entries.*
- void **ensureHasItemInfos** (const QList< [ItemInfo](#) > &infos)
- void **ensureHasItemInfos** (const QList< [ItemInfo](#) > &infos, const QList< QVariant > &extraValues)
- Qt::ItemFlags **flags** (const QModelIndex &index) const override
- bool **hasImage** (const [ItemInfo](#) &info) const
- bool **hasImage** (const [ItemInfo](#) &info, const QVariant &extraValue) const
- bool **hasImage** (qulonglong id) const
- bool **hasImage** (qulonglong id, const QVariant &extraValue) const
- QVariant **headerData** (int section, Qt::Orientation orientation, int role=Qt::DisplayRole) const override
- qulonglong **imageId** (const QModelIndex &index) const
- qulonglong **imageId** (int row) const
- QList< qulonglong > **imageIds** () const
- QList< qulonglong > **imageIds** (const QList< QModelIndex > &indexes) const
- [ItemInfo](#) **imageInfo** (const QModelIndex &index) const
  - Returns the ItemInfo object, reference or image id from the underlying data pointed to by the index.*
- [ItemInfo](#) **imageInfo** (const QString &filePath) const
- [ItemInfo](#) **imageInfo** (int row) const
  - Returns the ItemInfo object, reference or image id from the underlying data of the given row (parent is the invalid QModelIndex, column is 0).*
- [ItemInfo](#) & **imageInfoRef** (const QModelIndex &index) const
- [ItemInfo](#) & **imageInfoRef** (int row) const
- QList< [ItemInfo](#) > **imageInfos** () const
- QList< [ItemInfo](#) > **imageInfos** (const QList< QModelIndex > &indexes) const
- QList< [ItemInfo](#) > **imageInfos** (const QString &filePath) const
- QModelIndex **index** (int row, int column=0, const QModelIndex &parent=QModelIndex()) const override
- QList< QModelIndex > **indexesForImageId** (qulonglong id) const
- QList< QModelIndex > **indexesForItemInfo** (const [ItemInfo](#) &info) const
- QList< QModelIndex > **indexesForPath** (const QString &filePath) const
- QModelIndex **indexForImageId** (qulonglong id) const
- QModelIndex **indexForImageId** (qulonglong id, const QVariant &extraValue) const
- QModelIndex **indexForItemInfo** (const [ItemInfo](#) &info) const
  - Return the index for the given ItemInfo or id, if contained in this model.*
- QModelIndex **indexForItemInfo** (const [ItemInfo](#) &info, const QVariant &extraValue) const
- QModelIndex **indexForPath** (const QString &filePath) const
  - Returns the index or ItemInfo object from the underlying data for the given file path.*
- bool **isEmpty** () const

- bool **isRefreshing** () const  
*Returns true if this model is currently refreshing.*
- int **itemCount** () const
- bool **keepsFilePathCache** () const
- int **numberOfIndexesForImageId** (qulonglong id) const
- int **numberOfIndexesForItemInfo** (const [ItemInfo](#) &info) const
- void **removeIndex** (const QModelIndex &indexes)  
*Directly remove the given indexes or infos from the model.*
- void **removeIndexes** (const QList< QModelIndex > &indexes)
- void **removeItemInfo** (const [ItemInfo](#) &info)
- void **removeItemInfos** (const QList< [ItemInfo](#) > &infos)
- void **removeItemInfos** (const QList< [ItemInfo](#) > &infos, const QList< QVariant > &extraValues)
- int **rowCount** (const QModelIndex &parent=QModelIndex()) const override
- void **setItemInfos** (const QList< [ItemInfo](#) > &infos)  
*Clears and adds the infos.*
- void **setKeepsFilePathCache** (bool keepCache)  
*If a cache is kept, lookup by file path is fast, without a cache it is O(n).*
- DECLARE\_MODEL\_DRAG\_DROP\_METHODS void **setPreprocessor** (QObject \*const processor)  
*Install an object as a preprocessor for ItemInfos added to this model.*
- void **setSendRemovalSignals** (bool send)  
*Enable sending of imageInfosAboutToBeRemoved and imageInfosRemoved signals.*
- void **setWatchFlags** (const [DatabaseFields::Set](#) &set)  
*Set a set of database fields to watch.*
- QList< [ItemInfo](#) > **uniqueItemInfos** () const
- void **unsetPreprocessor** (QObject \*const processor)

## Public Member Functions inherited from [Digikam::DragDropModelImplementation](#)

- [DragDropModelImplementation](#) ()=default  
*A class providing a sample implementation for a QAbstractItemModel redirecting drag-and-drop support to a handler.*
- virtual Qt::ItemFlags **dragDropFlags** (const QModelIndex &index) const  
*Call from your flags() method, adding the relevant drag drop flags.*
- Qt::ItemFlags **dragDropFlagsV2** (const QModelIndex &index) const  
*This is an alternative approach to dragDropFlags().*
- [AbstractItemDragDropHandler](#) \* **dragDropHandler** () const
- bool **dropMimeData** (const QMimeData \*, Qt::DropAction, int, int, const QModelIndex &)
- virtual bool **isDragEnabled** (const QModelIndex &index) const
- virtual bool **isDropEnabled** (const QModelIndex &index) const
- QMimeData \* **mimeData** (const QModelIndexList &indexes) const
- QStringList **mimeTypes** () const
- void **setDragDropHandler** ([AbstractItemDragDropHandler](#) \*handler)  
*Set a drag drop handler.*
- Qt::DropActions **supportedDropActions** () const  
*Implements the relevant QAbstractItemModel methods for drag and drop.*

## Protected Slots

- void **slotCollectionImageChange** (const [CollectionImageChangeset](#) &changeset)



## Protected Slots inherited from [Digikam::ItemThumbnailModel](#)

- void **slotThumbnailLoaded** (const [LoadingDescription](#) &loadingDescription, const QPixmap &thumb)
- void **slotThumbnailLoadedFromStorage** (const [LoadingDescription](#) &loadingDescription, const QPixmap &thumb)

## Protected Slots inherited from [Digikam::ItemModel](#)

- virtual void **slotAlbumChange** (const [AlbumChangeset](#) &changeset)
- virtual void **slotImageChange** (const [ImageChangeset](#) &changeset)
- virtual void **slotImageTagChange** (const [ImageTagChangeset](#) &changeset)

## Additional Inherited Members

## Public Types inherited from [Digikam::ItemModel](#)

- enum [ItemModelRoles](#) {  
[ItemModelPointerRole](#) = Qt::UserRole , [ItemModelInternalId](#) = Qt::UserRole + 1 , [ThumbnailRole](#) = Qt::UserRole + 2 , [CreationDateRole](#) = Qt::UserRole + 3 ,  
[ExtraDataRole](#) = Qt::UserRole + 5 , [ExtraDataDuplicateCount](#) = Qt::UserRole + 6 , [LTLeftPanelRole](#) = Qt::UserRole + 50 , [LTRightPanelRole](#) = Qt::UserRole + 51 ,  
[SubclassRoles](#) = Qt::UserRole + 100 , [FilterModelRoles](#) = Qt::UserRole + 500 }

## Public Slots inherited from [Digikam::ItemThumbnailModel](#)

- void **preloadAllThumbnails** ()
- void **preloadThumbnails** (const QList< [ItemInfo](#) > &)  
*Preload thumbnail for the given infos resp.*
- void **preloadThumbnails** (const QList< QModelIndex > &)

## Public Slots inherited from [Digikam::ItemModel](#)

- void **reAddingFinished** ()
- void **reAddItemInfos** (const QList< [ItemInfo](#) > &infos, const QList< QVariant > &extraValues)

## Static Public Member Functions inherited from [Digikam::ItemModel](#)

- static qlonglong **retrievelmageld** (const QModelIndex &index)
- static [ItemInfo](#) **retrievelItemInfo** (const QModelIndex &index)  
*Retrieves the imageInfo object from the data() method of the given index.*

## Protected Member Functions inherited from [Digikam::ItemThumbnailModel](#)

- void **imageInfosCleared** () override  
*Called when the internal storage is cleared.*

## Protected Member Functions inherited from [Digikam::ItemModel](#)

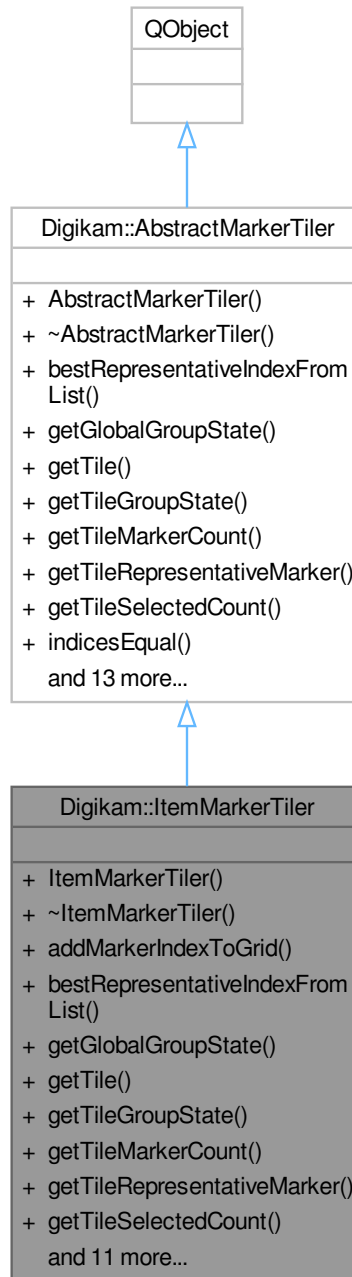
- void **emitDataChangedForAll** ()
- void **emitDataChangedForSelection** (const QItemSelection &selection)
- void **finishIncrementalRefresh** ()
- void **finishRefresh** ()
- bool **hasIncrementalRefreshPending** () const
- virtual void **imageInfosAboutToBeRemoved** (int, int)  
*Called before rowsAboutToBeRemoved.*
- void **requestIncrementalRefresh** ()  
*As soon as the model is ready to start an incremental refresh, the signal [readyForIncrementalRefresh\(\)](#) will be emitted.*
- void **startIncrementalRefresh** ()  
*Starts an incremental refresh operation.*
- void **startRefresh** ()  
*Subclasses that add ItemInfos in batches shall call [startRefresh\(\)](#) when they start sending batches and [finishRefresh\(\)](#) when they have finished.*

## Protected Attributes inherited from [Digikam::DragDropModelImplementation](#)

- [AbstractItemDragDropHandler](#) \* **m\_dragDropHandler** = nullptr

## 9.845 Digikam::ItemMarkerTiler Class Reference

Inheritance diagram for Digikam::ItemMarkerTiler:



### Public Member Functions

- **ItemMarkerTiler** ([GeoModelHelper](#) \*const modelHelper, QObject \*const parent=nullptr)
- void **addMarkerIndexToGrid** (const QPersistentModelIndex &markerIndex)

- QVariant [bestRepresentativeIndexFromList](#) (const QList< QVariant > &indices, const int sortKey) override
- GeoGroupState [getGlobalGroupState](#) () override
- Tile \* [getTile](#) (const TileIndex &tileIndex, const bool stopIfEmpty) override
- GeoGroupState [getTileGroupState](#) (const TileIndex &tileIndex) override
- int [getTileMarkerCount](#) (const TileIndex &tileIndex) override
- QVariant [getTileRepresentativeMarker](#) (const TileIndex &tileIndex, const int sortKey) override
  - *These should be implemented for thumbnail handling.*
- int [getTileSelectedCount](#) (const TileIndex &tileIndex) override
- bool [indicesEqual](#) (const QVariant &a, const QVariant &b) const override
- void [onIndicesClicked](#) (const ClickInfo &clickInfo) override
  - *These can be implemented if you want to react to actions in geolocation interface.*
- void [onIndicesMoved](#) (const TileIndex::List &tileIndicesList, const GeoCoordinates &targetCoordinates, const QPersistentModelIndex &targetSnapIndex) override
- QPixmap [pixmapFromRepresentativeIndex](#) (const QVariant &index, const QSize &size) override
- void [prepareTiles](#) (const GeoCoordinates &upperLeft, const GeoCoordinates &lowerRight, int level) override
- void [regenerateTiles](#) () override
- void [removeMarkerIndexFromGrid](#) (const QModelIndex &markerIndex, const bool ignoreSelection=false)
  - *Remove a marker from the grid.*
- void [setActive](#) (const bool state) override
- void [setMarkerGeoModelHelper](#) (GeoModelHelper \*const modelHelper)
- Tile \* [tileNew](#) () override
- TilerFlags [tilerFlags](#) () const override
  - *These have to be implemented.*

## Public Member Functions inherited from [Digikam::AbstractMarkerTiler](#)

- **AbstractMarkerTiler** (QObject \*const parent=nullptr)
- bool **indicesEqual** (const QList &a, const QList &b, const int upToLevel) const
- bool **isDirty** () const
- void **resetRootTile** ()
- Tile \* **rootTile** ()
- void **setDirty** (const bool state=true)

## Additional Inherited Members

## Public Types inherited from [Digikam::AbstractMarkerTiler](#)

- enum **TilerFlag** { **FlagNull** = 0 , **FlagMovable** = 1 }
- typedef QFlags< TilerFlag > **TilerFlags**

## Signals inherited from [Digikam::AbstractMarkerTiler](#)

- void **signalThumbnailAvailableForIndex** (const QVariant &index, const QPixmap &pixmap)
- void **signalTilesOrSelectionChanged** ()

## 9.845.1 Member Function Documentation

### 9.845.1.1 bestRepresentativeIndexFromList()

```
QVariant Digikam::ItemMarkerTiler::bestRepresentativeIndexFromList (
    const QList< QVariant > & indices,
    const int sortKey ) [override], [virtual]
```

Implements [Digikam::AbstractMarkerTiler](#).

### 9.845.1.2 getGlobalGroupState()

```
GeoGroupState Digikam::ItemMarkerTiler::getGlobalGroupState ( ) [override], [virtual]
```

Implements [Digikam::AbstractMarkerTiler](#).

### 9.845.1.3 getTile()

```
AbstractMarkerTiler::Tile * Digikam::ItemMarkerTiler::getTile (
    const TileIndex & tileIndex,
    const bool stopIfEmpty ) [override], [virtual]
```

Implements [Digikam::AbstractMarkerTiler](#).

### 9.845.1.4 getTileGroupState()

```
GeoGroupState Digikam::ItemMarkerTiler::getTileGroupState (
    const TileIndex & tileIndex ) [override], [virtual]
```

Implements [Digikam::AbstractMarkerTiler](#).

### 9.845.1.5 getTileMarkerCount()

```
int Digikam::ItemMarkerTiler::getTileMarkerCount (
    const TileIndex & tileIndex ) [override], [virtual]
```

Implements [Digikam::AbstractMarkerTiler](#).

### 9.845.1.6 getTileRepresentativeMarker()

```
QVariant Digikam::ItemMarkerTiler::getTileRepresentativeMarker (
    const TileIndex & tileIndex,
    const int sortKey ) [override], [virtual]
```

Implements [Digikam::AbstractMarkerTiler](#).

### 9.845.1.7 `getTileSelectedCount()`

```
int Digikam::ItemMarkerTiler::getTileSelectedCount (
    const TileIndex & tileIndex ) [override], [virtual]
```

Implements [Digikam::AbstractMarkerTiler](#).

### 9.845.1.8 `indicesEqual()`

```
bool Digikam::ItemMarkerTiler::indicesEqual (
    const QVariant & a,
    const QVariant & b ) const [override], [virtual]
```

Implements [Digikam::AbstractMarkerTiler](#).

### 9.845.1.9 `onIndicesClicked()`

```
void Digikam::ItemMarkerTiler::onIndicesClicked (
    const ClickInfo & clickInfo ) [override], [virtual]
```

Reimplemented from [Digikam::AbstractMarkerTiler](#).

### 9.845.1.10 `onIndicesMoved()`

```
void Digikam::ItemMarkerTiler::onIndicesMoved (
    const TileIndex::List & tileIndicesList,
    const GeoCoordinates & targetCoordinates,
    const QPersistentModelIndex & targetSnapIndex ) [override], [virtual]
```

Reimplemented from [Digikam::AbstractMarkerTiler](#).

### 9.845.1.11 `pixmapFromRepresentativeIndex()`

```
QPixmap Digikam::ItemMarkerTiler::pixmapFromRepresentativeIndex (
    const QVariant & index,
    const QSize & size ) [override], [virtual]
```

Implements [Digikam::AbstractMarkerTiler](#).

### 9.845.1.12 `prepareTiles()`

```
void Digikam::ItemMarkerTiler::prepareTiles (
    const GeoCoordinates & upperLeft,
    const GeoCoordinates & lowerRight,
    int level ) [override], [virtual]
```

Implements [Digikam::AbstractMarkerTiler](#).

**9.845.1.13 regenerateTiles()**

```
void Digikam::ItemMarkerTiler::regenerateTiles ( ) [override], [virtual]
```

Implements [Digikam::AbstractMarkerTiler](#).

**9.845.1.14 removeMarkerIndexFromGrid()**

```
void Digikam::ItemMarkerTiler::removeMarkerIndexFromGrid (
    const QModelIndex & markerIndex,
    const bool ignoreSelection = false )
```

## Parameters

<i>markerIndex</i>	The marker index to remove
<i>ignoreSelection</i>	Do not remove the marker from the count of selected items. This is only used by slotSourceModelRowsAboutToBeRemoved internally, because the selection model sends us an extra signal about the deselection.

**9.845.1.15 setActive()**

```
void Digikam::ItemMarkerTiler::setActive (
    const bool state ) [override], [virtual]
```

Implements [Digikam::AbstractMarkerTiler](#).

**9.845.1.16 tileNew()**

```
AbstractMarkerTiler::Tile * Digikam::ItemMarkerTiler::tileNew ( ) [override], [virtual]
```

Implements [Digikam::AbstractMarkerTiler](#).

**9.845.1.17 tilerFlags()**

```
AbstractMarkerTiler::TilerFlags Digikam::ItemMarkerTiler::tilerFlags ( ) const [override],
[virtual]
```

Reimplemented from [Digikam::AbstractMarkerTiler](#).

**9.846 Digikam::ItemMetadataAdjustmentHint Class Reference****Public Types**

- enum [AdjustmentStatus](#) { [AboutToEditMetadata](#) , [MetadataEditingFinished](#) , [MetadataEditingAborted](#) }

*The file's has been edited writing out information from the database, i.e., the db is already guaranteed to contain all changed information in the file's metadata.*

**Public Member Functions**

- ItemMetadataAdjustmentHint** (qulonglong id, [AdjustmentStatus](#) status, const QDateTime &modificationDateOnDisk, qulonglong fileSize)
- [AdjustmentStatus](#) **adjustmentStatus** () const
- qulonglong **fileSize** () const
- qulonglong **id** () const
- bool **isAboutToEdit** () const
- bool **isEditingFinished** () const
- bool **isEditingFinishedAborted** () const
- QDateTime **modificationDate** () const
- [ItemMetadataAdjustmentHint](#) & **operator**<< (const QDBusArgument &argument)
- const [ItemMetadataAdjustmentHint](#) & **operator**>> (QDBusArgument &argument) const



## Protected Attributes

- `qulonglong m_fileSize = 0`
- `qulonglong m_id = 0`
- `QDateTime m_modificationDate`
- `AdjustmentStatus m_status = AboutToEditMetadata`

## 9.846.1 Member Enumeration Documentation

### 9.846.1.1 AdjustmentStatus

enum `Digikam::ItemMetadataAdjustmentHint::AdjustmentStatus`

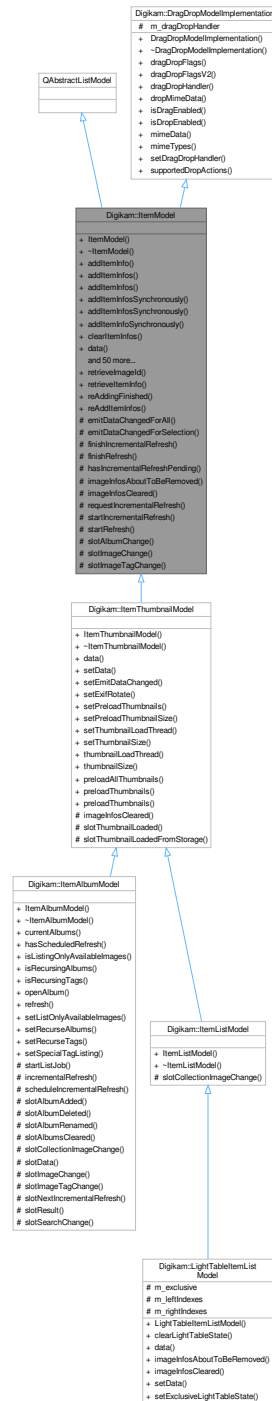
There is no need for a full rescan, optimizations are possible.

#### Enumerator

<code>AboutToEditMetadata</code>	The file is about to be edited. Suspends scanning. The Finished hint must follow.
<code>MetadataEditingFinished</code>	The file's metadata has been edited as described above.
<code>MetadataEditingAborted</code>	The file's metadata has not been edited, despite sending <code>AboutToEditMetadata</code> .

## 9.847 Digikam::ItemModel Class Reference

Inheritance diagram for Digikam::ItemModel:



### Public Types

- enum `ItemModelRoles` {
  - `ItemModelPointerRole` = Qt::UserRole , `ItemModelInternalId` = Qt::UserRole + 1 , `ThumbnailRole` = Qt::UserRole + 2 , `CreationDateRole` = Qt::UserRole + 3 ,

```

ExtraDataRole = Qt::UserRole + 5 , ExtraDataDuplicateCount = Qt::UserRole + 6 , LLeftPanelRole = Qt::UserRole + 50 , LRightPanelRole = Qt::UserRole + 51 ,
SubclassRoles = Qt::UserRole + 100 , FilterModelRoles = Qt::UserRole + 500 }

```

## Public Slots

- void **reAddingFinished** ()
- void **reAddItemInfos** (const QList< [ItemInfo](#) > &infos, const QList< QVariant > &extraValues)

## Signals

- void **allRefreshingFinished** ()  
*Signals that the model has finished currently with all scheduled refreshing, full or incremental, and all preprocessing.*
- void **imageChange** (const [ImageChangeset](#) &, const QItemSelection &)  
*If an [ImageChangeset](#) affected indexes of this model with changes as set in [watchFlags\(\)](#), this signal contains the changeset and the affected indexes.*
- void **imageInfosAboutToBeAdded** (const QList< [ItemInfo](#) > &infos)  
*Informs that ItemInfos will be added to the model.*
- void **imageInfosAboutToBeRemoved** (const QList< [ItemInfo](#) > &infos)  
*Informs that ItemInfos will be removed from the model.*
- void **imageInfosAdded** (const QList< [ItemInfo](#) > &infos)  
*Informs that ItemInfos have been added to the model.*
- void **imageInfosRemoved** (const QList< [ItemInfo](#) > &infos)  
*Informs that ItemInfos have been removed from the model.*
- void **imageTagChange** (const [ImageTagChangeset](#) &, const QItemSelection &)  
*If an [ImageTagChangeset](#) affected indexes of this model, this signal contains the changeset and the affected indexes.*
- void **preprocess** (const QList< [ItemInfo](#) > &infos, const QList< QVariant > &)  
*Connect to this signal only if you are the current preprocessor.*
- void **processAdded** (const QList< [ItemInfo](#) > &infos, const QList< QVariant > &)
- void **readyForIncrementalRefresh** ()  
*Signals that the model is right now ready to start an incremental refresh.*

## Public Member Functions

- **ItemModel** (QObject \*const parent=nullptr)
- void **addItemInfo** (const [ItemInfo](#) &info)  
*Main entry point for subclasses adding image infos to the model.*
- void **addItemInfos** (const QList< [ItemInfo](#) > &infos)
- void **addItemInfos** (const QList< [ItemInfo](#) > &infos, const QList< QVariant > &extraValues)
- void **addItemInfosSynchronously** (const QList< [ItemInfo](#) > &infos)
- void **addItemInfosSynchronously** (const QList< [ItemInfo](#) > &infos, const QList< QVariant > &extraValues)
- void **addItemInfoSynchronously** (const [ItemInfo](#) &info)  
*[addItemInfo\(\)](#) is asynchronous if a preprocessor is set.*
- void **clearItemInfos** ()  
*Clears image infos and resets model.*
- QVariant **data** (const QModelIndex &index, int role=Qt::DisplayRole) const override
- void **ensureHasGroupedImages** (const [ItemInfo](#) &groupLeader)  
*Ensure that all images grouped on the given leader are contained in the model.*
- void **ensureHasItemInfo** (const [ItemInfo](#) &info)  
*Add the given entries.*

- void **ensureHasItemInfos** (const QList< [ItemInfo](#) > &infos)
- void **ensureHasItemInfos** (const QList< [ItemInfo](#) > &infos, const QList< QVariant > &extraValues)
- Qt::ItemFlags **flags** (const QModelIndex &index) const override
- bool **hasImage** (const [ItemInfo](#) &info) const
- bool **hasImage** (const [ItemInfo](#) &info, const QVariant &extraValue) const
- bool **hasImage** (qulonglong id) const
- bool **hasImage** (qulonglong id, const QVariant &extraValue) const
- QVariant **headerData** (int section, Qt::Orientation orientation, int role=Qt::DisplayRole) const override
- qulonglong **imageId** (const QModelIndex &index) const
- qulonglong **imageId** (int row) const
- QList< qulonglong > **imageIds** () const
- QList< qulonglong > **imageIds** (const QList< QModelIndex > &indexes) const
- [ItemInfo](#) **imageInfo** (const QModelIndex &index) const
  - *Returns the [ItemInfo](#) object, reference or image id from the underlying data pointed to by the index.*
- [ItemInfo](#) **imageInfo** (const QString &filePath) const
- [ItemInfo](#) **imageInfo** (int row) const
  - *Returns the [ItemInfo](#) object, reference or image id from the underlying data of the given row (parent is the invalid QModelIndex, column is 0).*
- [ItemInfo](#) & **imageInfoRef** (const QModelIndex &index) const
- [ItemInfo](#) & **imageInfoRef** (int row) const
- QList< [ItemInfo](#) > **imageInfos** () const
- QList< [ItemInfo](#) > **imageInfos** (const QList< QModelIndex > &indexes) const
- QList< [ItemInfo](#) > **imageInfos** (const QString &filePath) const
- QModelIndex **index** (int row, int column=0, const QModelIndex &parent=QModelIndex()) const override
- QList< QModelIndex > **indexesForImageId** (qulonglong id) const
- QList< QModelIndex > **indexesForItemInfo** (const [ItemInfo](#) &info) const
- QList< QModelIndex > **indexesForPath** (const QString &filePath) const
- QModelIndex **indexForImageId** (qulonglong id) const
- QModelIndex **indexForImageId** (qulonglong id, const QVariant &extraValue) const
- QModelIndex **indexForItemInfo** (const [ItemInfo](#) &info) const
  - *Return the index for the given [ItemInfo](#) or id, if contained in this model.*
- QModelIndex **indexForItemInfo** (const [ItemInfo](#) &info, const QVariant &extraValue) const
- QModelIndex **indexForPath** (const QString &filePath) const
  - *Returns the index or [ItemInfo](#) object from the underlying data for the given file path.*
- bool **isEmpty** () const
- bool **isRefreshing** () const
  - *Returns true if this model is currently refreshing.*
- int **itemCount** () const
- bool **keepsFilePathCache** () const
- int **numberOfIndexesForImageId** (qulonglong id) const
- int **numberOfIndexesForItemInfo** (const [ItemInfo](#) &info) const
- void **removeIndex** (const QModelIndex &indexes)
  - *Directly remove the given indexes or infos from the model.*
- void **removeIndexes** (const QList< QModelIndex > &indexes)
- void **removeItemInfo** (const [ItemInfo](#) &info)
- void **removeItemInfos** (const QList< [ItemInfo](#) > &infos)
- void **removeItemInfos** (const QList< [ItemInfo](#) > &infos, const QList< QVariant > &extraValues)
- int **rowCount** (const QModelIndex &parent=QModelIndex()) const override
- void **setItemInfos** (const QList< [ItemInfo](#) > &infos)
  - *Clears and adds the infos.*
- void **setKeepsFilePathCache** (bool keepCache)
  - *If a cache is kept, lookup by file path is fast, without a cache it is O(n).*
- DECLARE\_MODEL\_DRAG\_DROP\_METHODS void **setPreprocessor** (QObject \*const processor)

- void [setSendRemovalSignals](#) (bool send)  
*Enable sending of imageInfosAboutToBeRemoved and imageInfosRemoved signals.*
- void [setWatchFlags](#) (const [DatabaseFields::Set](#) &set)  
*Set a set of database fields to watch.*
- [QList< ItemInfo > uniqueItemInfos](#) () const
- void [unsetPreprocessor](#) (QObject \*const processor)

## Public Member Functions inherited from [Digikam::DragDropModelImplementation](#)

- [DragDropModelImplementation](#) ()=default  
*A class providing a sample implementation for a QAbstractItemModel redirecting drag-and-drop support to a handler.*
- virtual [Qt::ItemFlags dragDropFlags](#) (const [QModelIndex](#) &index) const  
*Call from your flags() method, adding the relevant drag drop flags.*
- [Qt::ItemFlags dragDropFlagsV2](#) (const [QModelIndex](#) &index) const  
*This is an alternative approach to [dragDropFlags\(\)](#).*
- [AbstractItemDragDropHandler](#) \* [dragDropHandler](#) () const
- bool [dropMimeData](#) (const [QMimeData](#) \*, [Qt::DropAction](#), int, int, const [QModelIndex](#) &)
- virtual bool [isDragEnabled](#) (const [QModelIndex](#) &index) const
- virtual bool [isDropEnabled](#) (const [QModelIndex](#) &index) const
- [QMimeData](#) \* [mimeData](#) (const [QModelIndexList](#) &indexes) const
- [QStringList mimeTypes](#) () const
- void [setDragDropHandler](#) ([AbstractItemDragDropHandler](#) \*handler)  
*Set a drag drop handler.*
- [Qt::DropActions supportedDropActions](#) () const  
*Implements the relevant QAbstractItemModel methods for drag and drop.*

## Static Public Member Functions

- static [qlonglong retrieveImageld](#) (const [QModelIndex](#) &index)
- static [ItemInfo retrieveItemInfo](#) (const [QModelIndex](#) &index)  
*Retrieves the imageInfo object from the data() method of the given index.*

## Protected Slots

- virtual void [slotAlbumChange](#) (const [AlbumChangeset](#) &changeset)
- virtual void [slotImageChange](#) (const [ImageChangeset](#) &changeset)
- virtual void [slotImageTagChange](#) (const [ImageTagChangeset](#) &changeset)

## Protected Member Functions

- void **emitDataChangedForAll** ()
- void **emitDataChangedForSelection** (const QItemSelection &selection)
- void **finishIncrementalRefresh** ()
- void **finishRefresh** ()
- bool **hasIncrementalRefreshPending** () const
- virtual void **imageInfosAboutToBeRemoved** (int, int)  
*Called before rowsAboutToBeRemoved.*
- virtual void **imageInfosCleared** ()  
*Called when the internal storage is cleared.*
- void **requestIncrementalRefresh** ()  
*As soon as the model is ready to start an incremental refresh, the signal [readyForIncrementalRefresh\(\)](#) will be emitted.*
- void **startIncrementalRefresh** ()  
*Starts an incremental refresh operation.*
- void **startRefresh** ()  
*Subclasses that add ItemInfos in batches shall call [startRefresh\(\)](#) when they start sending batches and [finishRefresh\(\)](#) when they have finished.*

## Additional Inherited Members

### Protected Attributes inherited from [Digikam::DragDropModelImplementation](#)

- [AbstractItemDragDropHandler](#) \* **m\_dragDropHandler** = nullptr

## 9.847.1 Member Enumeration Documentation

### 9.847.1.1 ItemModelRoles

enum [Digikam::ItemModel::ItemModelRoles](#)

#### Enumerator

ItemModelPointerRole	An ItemModel* pointer to this model.
ThumbnailRole	Returns a thumbnail pixmap. May be implemented by subclasses. Returns either a valid pixmap or a null QVariant.
CreationDateRole	Returns a QDateTime with the creation date.
ExtraDataRole	Return (optional) extraData field.
ExtraDataDuplicateCount	Returns the number of duplicate indexes for the same image id.
LLeftPanelRole	Roles which are defined here but not implemented by <a href="#">ItemModel</a> Returns position of item in Left Light Table preview.
LRightPanelRole	Returns position of item in Right Light Table preview.
SubclassRoles	For use by subclasses.
FilterModelRoles	For use by filter models.

## 9.847.2 Member Function Documentation

### 9.847.2.1 addItemInfo()

```
void Digikam::ItemModel::addItemInfo (
    const ItemInfo & info )
```

If you list entries not unique per image id, you must add an extraValue so that every entry is unique by imageId and extraValues. Please note that these methods do not prevent addition of duplicate entries.

### 9.847.2.2 addItemInfoSynchronously()

```
void Digikam::ItemModel::addItemInfoSynchronously (
    const ItemInfo & info )
```

This method first adds the info, synchronously. Only afterwards, the preprocessor will have the opportunity to process it. This method also bypasses any incremental updates. Please note that these methods do not prevent addition of duplicate entries.

### 9.847.2.3 allRefreshingFinished

```
void Digikam::ItemModel::allRefreshingFinished ( ) [signal]
```

The model is in polished, clean situation right now.

### 9.847.2.4 ensureHasItemInfo()

```
void Digikam::ItemModel::ensureHasItemInfo (
    const ItemInfo & info )
```

Method returns immediately, the addition may happen later asynchronously. These methods prevent the addition of duplicate entries.

### 9.847.2.5 imageInfo() [1/2]

```
ItemInfo Digikam::ItemModel::imageInfo (
    const QModelIndex & index ) const
```

If the index is not valid, imageInfo will return a null [ItemInfo](#), imageId will return 0, imageInfoRef must not be called with an invalid index.

### 9.847.2.6 imageInfo() [2/2]

```
ItemInfo Digikam::ItemModel::imageInfo (
    int row ) const
```

Note that imageInfoRef will crash if index is invalid.

### 9.847.2.7 imageInfosAboutToBeAdded

```
void Digikam::ItemModel::imageInfosAboutToBeAdded (
    const QList< ItemInfo > & infos ) [signal]
```

This signal is sent before the model data is changed and views are informed.

### 9.847.2.8 imageInfosAboutToBeRemoved

```
void Digikam::ItemModel::imageInfosAboutToBeRemoved (
    const QList< ItemInfo > & infos ) [signal]
```

This signal is sent before the model data is changed and views are informed. Note: You need to explicitly enable sending of this signal. It is not sent in [clearItemInfos\(\)](#).

### 9.847.2.9 imageInfosAdded

```
void Digikam::ItemModel::imageInfosAdded (
    const QList< ItemInfo > & infos ) [signal]
```

This signal is sent after the model data is changed and views are informed.

### 9.847.2.10 imageInfosCleared()

```
virtual void Digikam::ItemModel::imageInfosCleared ( ) [inline], [protected], [virtual]
```

Reimplemented in [Digikam::ItemThumbnailModel](#).

### 9.847.2.11 imageInfosRemoved

```
void Digikam::ItemModel::imageInfosRemoved (
    const QList< ItemInfo > & infos ) [signal]
```

This signal is sent after the model data is changed and views are informed. \* Note: You need to explicitly enable sending of this signal. It is not sent in [clearItemInfos\(\)](#).

### 9.847.2.12 indexForPath()

```
QModelIndex Digikam::ItemModel::indexForPath (
    const QString & filePath ) const
```

This is fast if `keepsFilePathCache` is enabled. The file path is as returned by [ItemInfo.filePath\(\)](#). In case of multiple occurrences of the same file, the simpler variants return any one found first, use the `QList` methods to retrieve all occurrences.



### 9.847.2.13 isRefreshing()

```
bool Digikam::ItemModel::isRefreshing ( ) const
```

For a preprocessor this means that, although the preprocessor may currently have processed all it got, more batches are to be expected.

### 9.847.2.14 readyForIncrementalRefresh

```
void Digikam::ItemModel::readyForIncrementalRefresh ( ) [signal]
```

This is guaranteed only for the scope of emitting this signal.

### 9.847.2.15 requestIncrementalRefresh()

```
void Digikam::ItemModel::requestIncrementalRefresh ( ) [protected]
```

The signal will be emitted inline if the model is ready right now.

### 9.847.2.16 retrieveItemInfo()

```
ItemInfo Digikam::ItemModel::retrieveItemInfo (
    const QModelIndex & index ) [static]
```

The index may be from a QSortFilterProxyModel as long as an [ItemModel](#) is at the end.

### 9.847.2.17 setKeepsFilePathCache()

```
void Digikam::ItemModel::setKeepsFilePathCache (
    bool keepCache )
```

Default is false.

### 9.847.2.18 setPreprocessor()

```
void Digikam::ItemModel::setPreprocessor (
    QObject *const processor )
```

For every QList of ItemInfos added to `addItemInfo`, the signal `preprocess()` will be emitted. The preprocessor may process the items and shall then readd them by calling `reAddItemInfos()`. It may take some time to process. It shall discard any held infos when the `modelReset()` signal is sent. It shall call `readdFinished()` when no reset occurred and all infos on the way have been readded. This means that only after calling this method, you shall make three connections (`preprocess` -> your slot, your signal -> `reAddItemInfos`, your signal -> `reAddingFinished`) and make or already hold a connection `modelReset()` -> your slot. There is only one preprocessor at a time, a previously set object will be disconnected.

**9.847.2.19 setSendRemovalSignals()**

```
void Digikam::ItemModel::setSendRemovalSignals (
    bool send )
```

Default: false

**9.847.2.20 setWatchFlags()**

```
void Digikam::ItemModel::setWatchFlags (
    const DatabaseFields::Set & set )
```

If either of these is changed, `dataChanged()` will be emitted. Default is no flag (no signal will be emitted).

**9.847.2.21 startIncrementalRefresh()**

```
void Digikam::ItemModel::startIncrementalRefresh ( ) [protected]
```

You shall only call this method from a slot connected to [readyForIncrementalRefresh\(\)](#). To initiate an incremental refresh, call [requestIncrementalRefresh\(\)](#).

**9.847.2.22 startRefresh()**

```
void Digikam::ItemModel::startRefresh ( ) [protected]
```

No incremental refreshes will be started while listing. A [clearItemInfos\(\)](#) always stops listing, calling `finishRefresh()` is then not necessary.

**9.848 Digikam::ItemPosition Class Reference****Public Member Functions**

- **ItemPosition** ()  
*Creates a null [ItemPosition](#) object.*
- **ItemPosition** (const [CoreDbAccess](#) &access, qlonglong imageId)
- **ItemPosition** (const [ItemPosition](#) &other)
- **ItemPosition** (qlonglong imageId)  
*Creates an [ItemPosition](#) object for the given image.*
- double **accuracy** () const
- double **altitude** () const  
*The altitude in meters.*
- QString **altitudeFormatted** () const  
*Returns the altitude formatted in a user-presentable way in the form "43.45m".*
- void **apply** ()  
*Apply all changes made to this object.*
- QString **description** () const
- bool **hasAccuracy** () const
- bool **hasAltitude** () const

- bool **hasCoordinates** () const
- bool **hasOrientation** () const
- bool **hasRoll** () const
- bool **hasTilt** () const
- bool **isEmpty** () const
 

*An object is empty if no entry exists in the [ItemPosition](#) table for the referenced image, or if the object is null.*
- bool **isNull** () const
- QString **latitude** () const
 

*Returns latitude/longitude in the format as described by the XMP specification as "GPSCoordinate": A Text value in the form ?DDD,MM,SSk? or ?DDD,MM.mmk?.*
- QString **latitudeFormatted** () const
 

*Returns the latitude/longitude in a user-presentable version, in the form "30°45'55.123" East".*
- double **latitudeNumber** () const
 

*Returns latitude/longitude as a double in degrees.*
- bool **latitudeUserPresentableNumbers** (int \*degrees, int \*minutes, double \*seconds, char \*direction↵  
Reference)
 

*Returns latitude/longitude as user-presentable numbers.*
- QString **longitude** () const
- QString **longitudeFormatted** () const
- double **longitudeNumber** () const
- bool **longitudeUserPresentableNumbers** (int \*degrees, int \*minutes, double \*seconds, char \*direction↵  
Reference)
- [ItemPosition](#) & **operator=** (const [ItemPosition](#) &other)
- double **orientation** () const
- void **remove** ()
 

*Removes the whole data set for the referenced image from the database.*
- void **removeAltitude** ()
 

*Removes the altitude for the referenced image from the database.*
- double **roll** () const
- void **setAccuracy** (double accuracy)
- void **setAltitude** (double [altitude](#))
 

*Set the altitude in meters.*
- void **setDescription** (const QString &description)
- bool **setLatitude** (const QString &[latitude](#))
 

*Sets the latitude/longitude from the GPSCoordinate string as described by XMP.*
- bool **setLatitude** (double [latitudeNumber](#))
 

*Sets the latitude/longitude from a double floating point number, as described for [latitudeNumber\(\)](#) above.*
- bool **setLongitude** (const QString &longitude)
- bool **setLongitude** (double longitudeNumber)
- void **setOrientation** (double orientation)
- void **setRoll** (double roll)
- void **setTilt** (double tilt)
- double **tilt** () const

## 9.848.1 Constructor & Destructor Documentation

### 9.848.1.1 ItemPosition()

```
Digikam::ItemPosition::ItemPosition (
    qulonglong imageId ) [explicit]
```

The information is read from the database.

## 9.848.2 Member Function Documentation

### 9.848.2.1 `apply()`

```
void Digikam::ItemPosition::apply ( )
```

(Also called from destructor)

### 9.848.2.2 `isEmpty()`

```
bool Digikam::ItemPosition::isEmpty ( ) const
```

An empty object is empty even if values have been set; it becomes not empty after calling [apply\(\)](#).

### 9.848.2.3 `latitude()`

```
QString Digikam::ItemPosition::latitude ( ) const
```

This provides lossless storage.

### 9.848.2.4 `latitudeNumber()`

```
double Digikam::ItemPosition::latitudeNumber ( ) const
```

North and East have a positive sign, South and West negative. This provides high precision, with the usual floating point concerns, and possible problems finding the exact text form when converting *back* to fractions.

### 9.848.2.5 `latitudeUserPresentableNumbers()`

```
bool Digikam::ItemPosition::latitudeUserPresentableNumbers (
    int * degrees,
    int * minutes,
    double * seconds,
    char * directionReference )
```

This means that degrees and minutes are integer, the seconds fractional. Direction reference is 'N'/'S', 'E'/'W' resp. This is for the purpose of presenting to the user, there are no guarantees on precision. Returns true if the values have been changed.

### 9.848.2.6 `remove()`

```
void Digikam::ItemPosition::remove ( )
```

This object and any [ItemPosition](#) object created later will be empty.

**9.848.2.7 setLatitude()** [1/2]

```
bool Digikam::ItemPosition::setLatitude (
    const QString & latitude )
```

Returns true if the format is accepted.

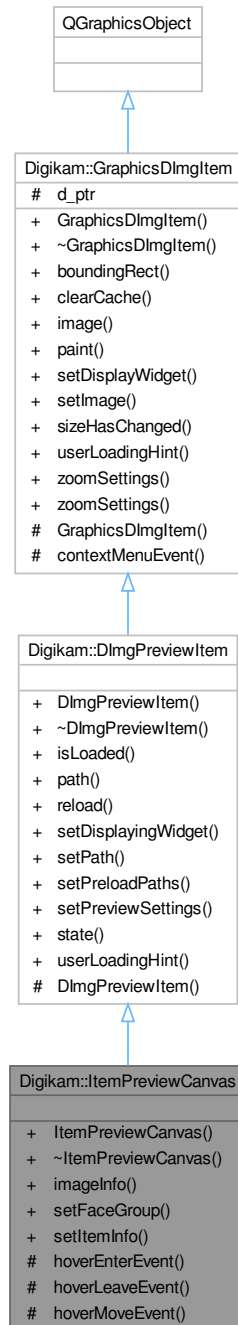
**9.848.2.8 setLatitude()** [2/2]

```
bool Digikam::ItemPosition::setLatitude (
    double latitudeNumber )
```

Returns true if the value is valid and accepted.

## 9.849 Digikam::ItemPreviewCanvas Class Reference

Inheritance diagram for Digikam::ItemPreviewCanvas:



### Public Member Functions

- [ItemInfo](#) `imageInfo` () const
- void `setFaceGroup` ([FaceGroup](#) \*const group)
- void `setItemInfo` (const [ItemInfo](#) &info)

## Public Member Functions inherited from [Digikam::DImgPreviewItem](#)

- **DImgPreviewItem** (QGraphicsItem \*const parent=nullptr)
- bool **isLoading** () const
- QString **path** () const
- void **reload** ()
- void **setDisplayingWidget** (QWidget \*const widget)
- void **setPath** (const QString &path, bool rePreview=false)
- void **setPreloadPaths** (const QStringList &pathsToPreload)
- void **setPreviewSettings** (const [PreviewSettings](#) &settings)
- State **state** () const
- QString **userLoadingHint** () const override

## Public Member Functions inherited from [Digikam::GraphicsDImgItem](#)

- **GraphicsDImgItem** (QGraphicsItem \*const parent=nullptr)
- QRectF **boundingRect** () const override
- void **clearCache** ()
- [DImg](#) **image** () const
- void **paint** (QPainter \*painter, const QStyleOptionGraphicsItem \*option, QWidget \*widget) override
- void **setDisplayWidget** (QWidget \*const widget)
- void **setImage** (const [DImg](#) &img)
  - Sets the [DImg](#) to be drawn by this item.*
- void **sizeHasChanged** ()
- [ImageZoomSettings](#) \* **zoomSettings** ()
- const [ImageZoomSettings](#) \* **zoomSettings** () const

## Protected Member Functions

- void **hoverEnterEvent** (QGraphicsSceneHoverEvent \*e) override
- void **hoverLeaveEvent** (QGraphicsSceneHoverEvent \*e) override
- void **hoverMoveEvent** (QGraphicsSceneHoverEvent \*e) override

## Protected Member Functions inherited from [Digikam::DImgPreviewItem](#)

- **DImgPreviewItem** (DImgPreviewItemPrivate &dd, QGraphicsItem \*const parent=nullptr)

## Protected Member Functions inherited from [Digikam::GraphicsDImgItem](#)

- **GraphicsDImgItem** (GraphicsDImgItemPrivate &dd, QGraphicsItem \*const parent)
- void **contextMenuEvent** (QGraphicsSceneContextMenuEvent \*e) override

## Additional Inherited Members

## Public Types inherited from [Digikam::DImgPreviewItem](#)

- enum **State** { **NoImage** , **Loading** , **ImageLoaded** , **ImageLoadingFailed** }

**Signals inherited from [Digikam::DImgPreviewItem](#)**

- void **loaded** ()
- void **loadingFailed** ()
- void **stateChanged** (int state)

**Signals inherited from [Digikam::GraphicsDImgItem](#)**

- void **imageChanged** ()
- void **imageSizeChanged** (const QSizeF &size)
- void **showContextMenu** (QGraphicsSceneContextMenuEvent \*e)

**Protected Attributes inherited from [Digikam::GraphicsDImgItem](#)**

- GraphicsDImgItemPrivate \*const **d\_ptr**



## 9.850 Digikam::ItemPreviewView Class Reference

Inheritance diagram for Digikam::ItemPreviewView:



### Public Types

- enum **Mode** { **IconViewPreview** , **LightTablePreview** }

## Signals

- void **signalAddToExistingQueue** (int)
- void **signalDeleteItem** ()
- void **signalEscapePreview** ()
- void **signalGotoAlbumAndItem** (const [ItemInfo](#) &)
- void **signalGotoDateAndItem** (const [ItemInfo](#) &)
- void **signalGotoTagAndItem** (int)
- void **signalNextItem** ()
- void **signalPopupTagsView** ()
- void **signalPreviewLoaded** (bool success)
- void **signalPrevItem** ()
- void **signalSlideShowCurrent** ()

## Signals inherited from [Digikam::GraphicsDImgView](#)

- void **activated** ()
- void **contentsMoved** (bool panningFinished)
- void **contentsMoving** (int, int)
- void **leftButtonClicked** ()
- void **leftButtonDoubleClicked** ()
- void **resized** ()
- void **rightButtonClicked** ()
- void **toNextImage** ()
- void **toPreviousImage** ()
- void **viewportRectChanged** (const [QRectF](#) &viewportRect)

## Public Member Functions

- **ItemPreviewView** ([QWidget](#) \*const parent, Mode mode=IconViewPreview, [Album](#) \*const currAlbum=nullptr)
- [ItemInfo](#) **getItemInfo** () const
- void **reload** ()
- void **setImagePath** (const [QString](#) &path=[QString](#)())
- void **setItemInfo** (const [ItemInfo](#) &info=[ItemInfo](#)(), const [ItemInfo](#) &previous=[ItemInfo](#)(), const [ItemInfo](#) &next=[ItemInfo](#)())
- void **setPreviousNextPaths** (const [QString](#) &previous, const [QString](#) &next)

## Public Member Functions inherited from [Digikam::GraphicsDImgView](#)

- **GraphicsDImgView** ([QWidget](#) \*const parent=nullptr)
- int **contentsX** () const
- int **contentsY** () const
- void **drawText** ([QPainter](#) \*p, const [QRectF](#) &rect, const [QString](#) &text)
- void **fitToWindow** ()
- [GraphicsDImgItem](#) \* **item** () const  
*Return the instance of item set by [setItem\(\)](#).*
- [SinglePhotoPreviewLayout](#) \* **layout** () const
- [DImgPreviewItem](#) \* **previewItem** () const  
*Return a cast of item instance of item set by [setItem\(\)](#) as [DImgPreviewItem](#) Note: if you store a [GraphicsDImgItem](#) object using [setItem\(\)](#), this method will return 0.*
- void **scrollPointOnPoint** (const [QPointF](#) &scenePos, const [QPoint](#) &viewportPos)  
*Scrolls the view such that scenePos (in scene coordinates is displayed on the viewport at viewportPos (in viewport coordinates).*
- void **setContentPos** (int x, int y)
- void **setItem** ([GraphicsDImgItem](#) \*const item)  
*Store internal instance of item as [GraphicsDImgItem](#).*
- void **toggleFullScreen** (bool set)
- [QRect](#) **visibleArea** () const

### Protected Member Functions

- bool [acceptsMouseClicked](#) (QMouseEvent \*e) override
- void [dragEnterEvent](#) (QDragEnterEvent \*e) override
- void [dragMoveEvent](#) (QDragMoveEvent \*e) override
- void [dropEvent](#) (QDropEvent \*e) override
- void [enterEvent](#) (QEnterEvent \*) override
- void [leaveEvent](#) (QEvent \*e) override
- void [mousePressEvent](#) (QMouseEvent \*e) override
- void [showEvent](#) (QShowEvent \*e) override

### Protected Member Functions inherited from [Digikam::GraphicsDImgView](#)

- void [continuePanning](#) (const QPoint &pos)
- void [drawForeground](#) (QPainter \*painter, const QRectF &rect) override
- void [finishPanning](#) ()
- void [installPanIcon](#) ()
- void [mouseDoubleClickEvent](#) (QMouseEvent \*) override
- void [mouseMoveEvent](#) (QMouseEvent \*) override
- void [mousePressEvent](#) (QMouseEvent \*) override
- void [mouseReleaseEvent](#) (QMouseEvent \*) override
- void [resizeEvent](#) (QResizeEvent \*) override
- void [scrollContentsBy](#) (int dx, int dy) override
- void [setScaleFitToWindow](#) (bool value)
- void [setShowText](#) (bool value)
- void [startPanning](#) (const QPoint &pos)
- void [wheelEvent](#) (QWheelEvent \*) override

### Additional Inherited Members

### Protected Slots inherited from [Digikam::GraphicsDImgView](#)

- void [slotContentsMoved](#) ()
- void [slotCornerButtonPressed](#) ()
- void [slotPanIconHidden](#) ()
- virtual void [slotPanIconSelectionMoved](#) (const QRect &, bool)

## 9.850.1 Member Function Documentation

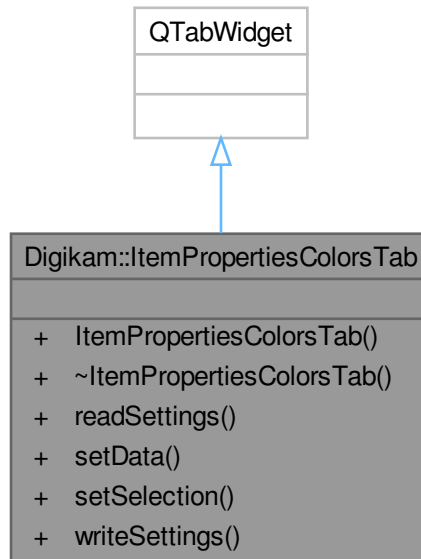
### 9.850.1.1 [acceptsMouseClicked\(\)](#)

```
bool Digikam::ItemPreviewView::acceptsMouseClicked (
    QMouseEvent * e ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::GraphicsDImgView](#).

## 9.851 Digikam::ItemPropertiesColorsTab Class Reference

Inheritance diagram for Digikam::ItemPropertiesColorsTab:

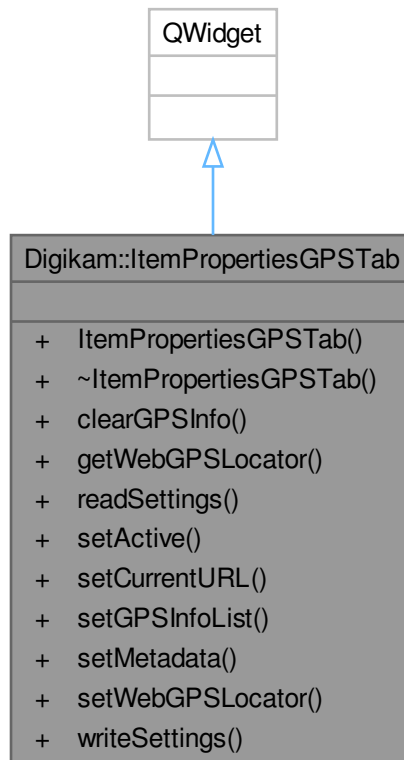


### Public Member Functions

- **ItemPropertiesColorsTab** (`QWidget *const parent`)
- void **readSettings** (`const KConfigGroup &group`)
- void **setData** (`const QUrl &url=QUrl()`, `const QRect &selectionArea=QRect()`, `DImg *const img=nullptr`)
- void **setSelection** (`const QRect &selectionArea`)
- void **writeSettings** (`KConfigGroup &group`)

## 9.852 Digikam::ItemPropertiesGPSTab Class Reference

Inheritance diagram for Digikam::ItemPropertiesGPSTab:



### Public Types

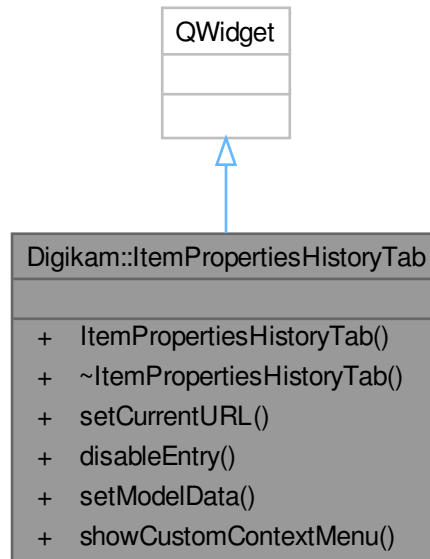
- enum `WebGPSLocator` {  
**MapQuest** = 0, **GoogleMaps**, **BingMaps**, **OpenStreetMap**,  
**LocalizeMaps** }

### Public Member Functions

- `ItemPropertiesGPSTab` (`QWidget *const parent`)
- void `clearGPSInfo` ()
- int `getWebGPSLocator` () const
- void `readSettings` (const `KConfigGroup &group`)
- void `setActive` (const bool state)
- void `setCurrentURL` (const `QUrl &url=QUrl()`)
- void `setGPSInfoList` (const `GPSItemInfo::List &list`)
- void `setMetadata` (`DMetadata *const meta`, const `QUrl &url`)
- void `setWebGPSLocator` (int locator)
- void `writeSettings` (`KConfigGroup &group`)

## 9.853 Digikam::ItemPropertiesHistoryTab Class Reference

Inheritance diagram for Digikam::ItemPropertiesHistoryTab:



### Public Slots

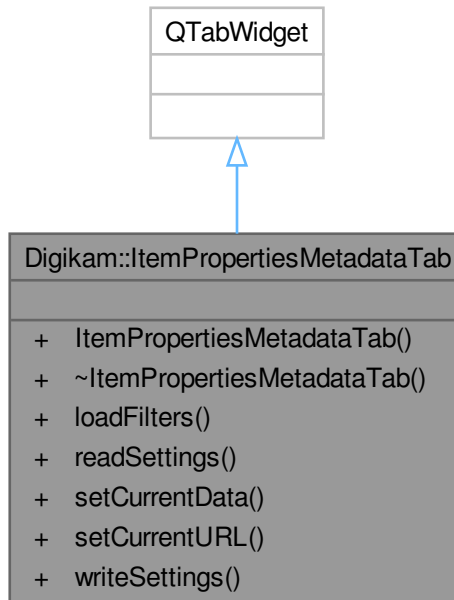
- void **disableEntry** (bool disable)
- void **setModelData** (const QList< [DImageHistory::Entry](#) > &entries)
- void **showCustomContextMenu** (const QPoint &position)

### Public Member Functions

- **ItemPropertiesHistoryTab** (QWidget \*const parent)
- void **setCurrentURL** (const QUrl &url=QUrl())

## 9.854 Digikam::ItemPropertiesMetadataTab Class Reference

Inheritance diagram for Digikam::ItemPropertiesMetadataTab:



### Signals

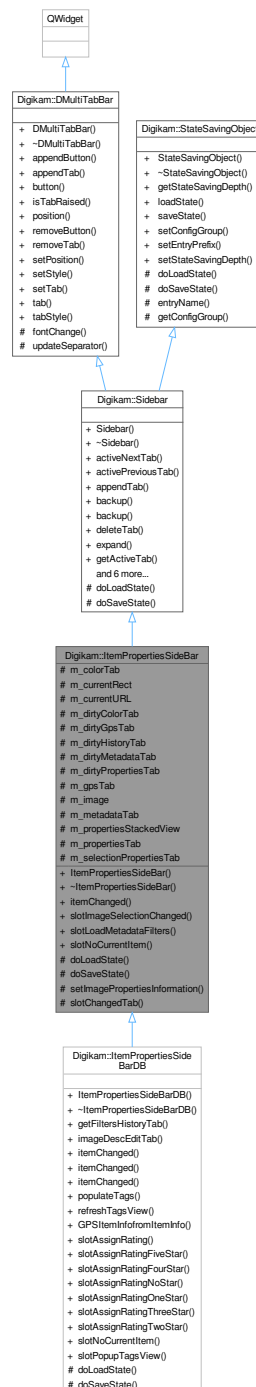
- void **signalSetupExifTool** ()
- void **signalSetupMetadataFilters** (int)

### Public Member Functions

- **ItemPropertiesMetadataTab** (QWidget \*const parent)
- void **loadFilters** ()
- void **readSettings** (const KConfigGroup &group)
- void **setCurrentData** ([DMetadata](#) \*const metadata=nullptr, const QUrl &url=QUrl())
- void **setCurrentURL** (const QUrl &url=QUrl())
- void **writeSettings** (KConfigGroup &group)

## 9.855 Digikam::ItemPropertiesSideBar Class Reference

Inheritance diagram for Digikam::ItemPropertiesSideBar:



### Public Slots

- void **slotImageSelectionChanged** (const QRect &rect)
- void **slotLoadMetadataFilters** ()
- virtual void **slotNoCurrentItem** ()



## Signals

- void **signalSetupExifTool** ()
- void **signalSetupMetadataFilters** (int)

## Signals inherited from [Digikam::Sidebar](#)

- void **signalChangedTab** (QWidget \*w)  
*Is emitted, when another tab is activated.*
- void **signalViewChanged** ()  
*Is emitted, when tab is shrink or expanded.*

## Public Member Functions

- **ItemPropertiesSideBar** (QWidget \*const parent, [SidebarSplitter](#) \*const splitter, Qt::Edge side=Qt::LeftEdge, bool mimimizedDefault=false)
- virtual void **itemChanged** (const QUrl &url, const QRect &rect=QRect(), [DImg](#) \*const img=nullptr)

## Public Member Functions inherited from [Digikam::Sidebar](#)

- [Sidebar](#) (QWidget \*const parent, [SidebarSplitter](#) \*const sp, Qt::Edge side=Qt::LeftEdge, bool minimizedDefault=false)  
*Creates a new sidebar.*
- void [activeNextTab](#) ()  
*Activates a next tab from current one.*
- void [activePreviousTab](#) ()  
*Activates a previous tab from current one.*
- void [appendTab](#) (QWidget \*const w, const QIcon &pic, const QString &title)  
*Appends a new tab to the sidebar.*
- void **backup** ()  
*Hide sidebar and backup minimized state.*
- void [backup](#) (const QList< QWidget \* > &thirdWidgetsToBackup, QList< int > \*const sizes)  
*Hide sidebar and backup minimized state.*
- void **deleteTab** (QWidget \*const w)  
*Deletes a tab from the tabbar.*
- void **expand** ()  
*Redisplays the whole sidebar.*
- QWidget \* **getActiveTab** () const  
*Returns the currently activated tab, or 0 if no tab is active.*
- bool **isExpanded** () const  
*Return the visible status of current sidebar tab.*
- void **restore** ()  
*Show sidebar and restore minimized state.*
- void [restore](#) (const QList< QWidget \* > &thirdWidgetsToRestore, const QList< int > &sizes)  
*Show sidebar and restore minimized state.*
- void **setActiveTab** (QWidget \*const w)  
*Activates a tab.*
- void **shrink** ()  
*Hides the sidebar (display only the activation buttons)*
- [SidebarSplitter](#) \* **splitter** () const

## Public Member Functions inherited from [Digikam::DMultiTabBar](#)

- **DMultiTabBar** (Qt::Edge pos, QWidget \*const parent=nullptr)
- void [appendButton](#) (const QIcon &pic, int id=-1, QMenu \*const popup=nullptr, const QString &not\_used\_↔ yet=QString())  
*append a new button to the button area.*
- void [appendTab](#) (const QIcon &pic, int id=-1, const QString &text=QString())  
*append a new tab to the tab area.*
- [DMultiTabBarButton](#) \* **button** (int id) const  
*get a pointer to a button within the button area identified by its ID*
- bool **isTabRaised** (int id) const  
*return the state of a tab, identified by its ID*
- Qt::Edge [position](#) () const  
*get the tabbar position.*
- void **removeButton** (int id)  
*remove a button with the given ID*
- void **removeTab** (int id)  
*remove a tab with a given ID*
- void [setPosition](#) (Qt::Edge pos)  
*set the real position of the widget.*
- void **setStyle** ([TextStyle](#) style)  
*set the display style of the tabs*
- void [setTab](#) (int id, bool state)  
*set a tab to "raised"*
- [DMultiTabBarTab](#) \* **tab** (int id) const  
*get a pointer to a tab within the tab area, identified by its ID*
- [TextStyle](#) [tabStyle](#) () const  
*get the display style of the tabs*

## Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (QObject \*const host)  
*Constructor.*
- virtual **~StateSavingObject** ()  
*Destructor.*
- [StateSavingDepth](#) [getStateSavingDepth](#) () const  
*Returns the depth used for state saving or loading.*
- void **loadState** ()  
*Invokes loading the class' state.*
- void **saveState** ()  
*Invokes saving the class' state.*
- virtual void [setConfigGroup](#) (const KConfigGroup &group)  
*Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void [setEntryPrefix](#) (const QString &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void [setStateSavingDepth](#) (const [StateSavingDepth](#) depth)  
*Sets the depth used for state saving or loading.*

## Protected Slots

- virtual void **slotChangedTab** (QWidget \*tab)

### Protected Member Functions

- void [doLoadState](#) () override  
*load the last view state from disk - called by [StateSavingObject::loadState\(\)](#)*
- void [doSaveState](#) () override  
*save the view state to disk - called by [StateSavingObject::saveState\(\)](#)*
- virtual void **setImagePropertiesInformation** (const QUrl &url)

### Protected Member Functions inherited from [Digikam::Sidebar](#)

- void [doLoadState](#) () override  
*Load the last view state from disk - called by [StateSavingObject::loadState\(\)](#)*
- void [doSaveState](#) () override  
*Save the view state to disk - called by [StateSavingObject::saveState\(\)](#)*

### Protected Member Functions inherited from [Digikam::DMultiTabBar](#)

- virtual void **fontChange** (const QFont &)
- void **updateSeparator** ()

### Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- QString [entryName](#) (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup [getConfigGroup](#) () const  
*Returns the config group that must be used for state saving and loading.*

### Protected Attributes

- [ItemPropertiesColorsTab](#) \* **m\_colorTab** = nullptr
- QRect **m\_currentRect**
- QUrl **m\_currentURL**
- bool **m\_dirtyColorTab** = false
- bool **m\_dirtyGpsTab** = false
- bool **m\_dirtyHistoryTab** = false
- bool **m\_dirtyMetadataTab** = false
- bool **m\_dirtyPropertiesTab** = false
- [ItemPropertiesGPSTab](#) \* **m\_gpsTab** = nullptr
- [DImg](#) \* **m\_image** = nullptr
- [ItemPropertiesMetadataTab](#) \* **m\_metadataTab** = nullptr
- QStackedWidget \* **m\_propertiesStackedView** = nullptr
- [ItemPropertiesTab](#) \* **m\_propertiesTab** = nullptr
- [ItemSelectionPropertiesTab](#) \* **m\_selectionPropertiesTab** = nullptr

### Additional Inherited Members

### Public Types inherited from [Digikam::DMultiTabBar](#)

- enum [TextStyle](#) { [ActiveIconText](#) = 0 , [AllIconsText](#) = 2 }  
*The list of available styles for [DMultiTabBar](#).*

## Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }

*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

### 9.855.1 Member Function Documentation

#### 9.855.1.1 [doLoadState\(\)](#)

```
void Digikam::ItemPropertiesSideBar::doLoadState ( ) [override], [protected], [virtual]
```

Implements [Digikam::StateSavingObject](#).

Reimplemented in [Digikam::ItemPropertiesSideBarDB](#).

#### 9.855.1.2 [doSaveState\(\)](#)

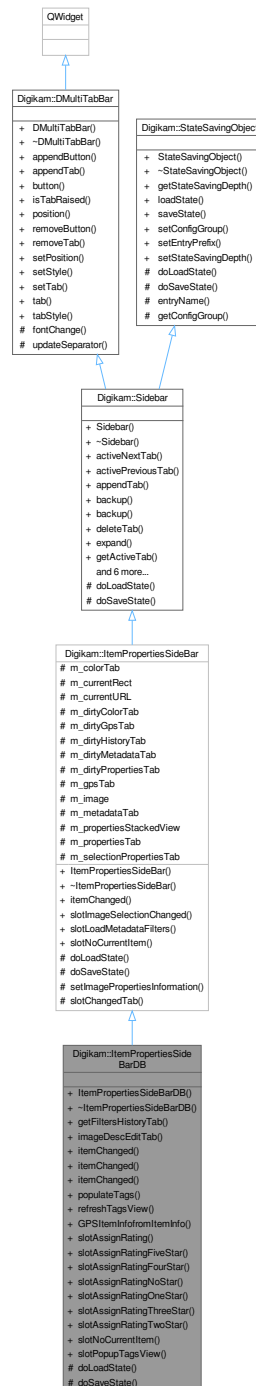
```
void Digikam::ItemPropertiesSideBar::doSaveState ( ) [override], [protected], [virtual]
```

Implements [Digikam::StateSavingObject](#).

Reimplemented in [Digikam::ItemPropertiesSideBarDB](#).

## 9.856 Digikam::ItemPropertiesSideBarDB Class Reference

Inheritance diagram for Digikam::ItemPropertiesSideBarDB:



### Public Slots

- void **slotAssignRating** (int rating)
- void **slotAssignRatingFiveStar** ()

- void **slotAssignRatingFourStar** ()
- void **slotAssignRatingNoStar** ()
- void **slotAssignRatingOneStar** ()
- void **slotAssignRatingThreeStar** ()
- void **slotAssignRatingTwoStar** ()
- void **slotNoCurrentItem** () override
- void **slotPopupTagsView** ()

### Public Slots inherited from [Digikam::ItemPropertiesSideBar](#)

- void **slotImageSelectionChanged** (const QRect &rect)
- void **slotLoadMetadataFilters** ()
- virtual void **slotNoCurrentItem** ()

### Signals

- void **signalFirstItem** ()
- void **signalLastItem** ()
- void **signalNextItem** ()
- void **signalPrevItem** ()
- void **signalRightSideBarBusy** (bool busy)

### Signals inherited from [Digikam::ItemPropertiesSideBar](#)

- void **signalSetupExifTool** ()
- void **signalSetupMetadataFilters** (int)

### Signals inherited from [Digikam::Sidebar](#)

- void **signalChangedTab** (QWidget \*w)  
*Is emitted, when another tab is activated.*
- void **signalViewChanged** ()  
*Is emitted, when tab is shrink or expanded.*

### Public Member Functions

- **ItemPropertiesSideBarDB** (QWidget \*const parent, [SidebarSplitter](#) \*const splitter, Qt::Edge side=Qt::Left↔ Edge, bool mimimizedDefault=false)
- **ItemPropertiesVersionsTab** \* **getFiltersHistoryTab** () const  
*This is for image editor to be able to update the filter list in sidebar.*
- **ItemDescEditTab** \* **imageDescEditTab** () const
- virtual void **itemChanged** (const [ItemInfo](#) &info, const QRect &rect=QRect(), [DImg](#) \*const img=nullptr, const [DImageHistory](#) &history=[DImageHistory](#)())
- virtual void **itemChanged** (const [ItemInfoList](#) &infos, const [ItemInfoList](#) &allInfos)
- void **itemChanged** (const QUrl &url, const QRect &rect=QRect(), [DImg](#) \*const img=nullptr) override
- void **populateTags** ()
- void **refreshTagsView** ()

## Public Member Functions inherited from Digikam::ItemPropertiesSideBar

- **ItemPropertiesSideBar** (QWidget \*const parent, [SidebarSplitter](#) \*const splitter, Qt::Edge side=Qt::LeftEdge, bool mimimizedDefault=false)

## Public Member Functions inherited from Digikam::Sidebar

- **Sidebar** (QWidget \*const parent, [SidebarSplitter](#) \*const sp, Qt::Edge side=Qt::LeftEdge, bool minimized←Default=false)
  - Creates a new sidebar.*
- void **activeNextTab** ()
  - Activates a next tab from current one.*
- void **activePreviousTab** ()
  - Activates a previous tab from current one.*
- void **appendTab** (QWidget \*const w, const QIcon &pic, const QString &title)
  - Appends a new tab to the sidebar.*
- void **backup** ()
  - Hide sidebar and backup minimized state.*
- void **backup** (const QList< QWidget \* > &thirdWidgetsToBackup, QList< int > \*const sizes)
  - Hide sidebar and backup minimized state.*
- void **deleteTab** (QWidget \*const w)
  - Deletes a tab from the tabbar.*
- void **expand** ()
  - Redisplays the whole sidebar.*
- QWidget \* **getActiveTab** () const
  - Returns the currently activated tab, or 0 if no tab is active.*
- bool **isExpanded** () const
  - Return the visible status of current sidebar tab.*
- void **restore** ()
  - Show sidebar and restore minimized state.*
- void **restore** (const QList< QWidget \* > &thirdWidgetsToRestore, const QList< int > &sizes)
  - Show sidebar and restore minimized state.*
- void **setActiveTab** (QWidget \*const w)
  - Activates a tab.*
- void **shrink** ()
  - Hides the sidebar (display only the activation buttons)*
- [SidebarSplitter](#) \* **splitter** () const

## Public Member Functions inherited from Digikam::DMultiTabBar

- **DMultiTabBar** (Qt::Edge pos, QWidget \*const parent=nullptr)
- void **appendButton** (const QIcon &pic, int id=-1, QMenu \*const popup=nullptr, const QString &not\_used\_←yet=QString())
  - append a new button to the button area.*
- void **appendTab** (const QIcon &pic, int id=-1, const QString &text=QString())
  - append a new tab to the tab area.*
- [DMultiTabBarButton](#) \* **button** (int id) const
  - get a pointer to a button within the button area identified by its ID*
- bool **isTabRaised** (int id) const
  - return the state of a tab, identified by its ID*

- Qt::Edge [position](#) () const  
*get the tabbar position.*
- void **removeButton** (int id)  
*remove a button with the given ID*
- void **removeTab** (int id)  
*remove a tab with a given ID*
- void [setPosition](#) (Qt::Edge pos)  
*set the real position of the widget.*
- void **setStyle** ([TextStyle](#) style)  
*set the display style of the tabs*
- void [setTab](#) (int id, bool state)  
*set a tab to "raised"*
- [DMultiTabBarTab](#) \* **tab** (int id) const  
*get a pointer to a tab within the tab area, identified by its ID*
- [TextStyle](#) **tabStyle** () const  
*get the display style of the tabs*

## Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (QObject \*const host)  
*Constructor.*
- virtual **~StateSavingObject** ()  
*Destructor.*
- [StateSavingDepth](#) [getStateSavingDepth](#) () const  
*Returns the depth used for state saving or loading.*
- void **loadState** ()  
*Invokes loading the class' state.*
- void **saveState** ()  
*Invokes saving the class' state.*
- virtual void [setConfigGroup](#) (const KConfigGroup &group)  
*Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void [setEntryPrefix](#) (const QString &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void [setStateSavingDepth](#) (const [StateSavingDepth](#) depth)  
*Sets the depth used for state saving or loading.*

## Static Public Member Functions

- static bool **GPSItemInfofromItemInfo** (const [ItemInfo](#) &, [GPSItemInfo](#) \*const)

## Protected Member Functions

- void [doLoadState](#) () override  
*load the last view state from disk - called by [StateSavingObject::loadState\(\)](#)*
- void [doSaveState](#) () override  
*save the view state to disk - called by [StateSavingObject::saveState\(\)](#)*



## Protected Member Functions inherited from Digikam::Sidebar

- void `doLoadState` () override  
*Load the last view state from disk - called by `StateSavingObject::loadState()`*
- void `doSaveState` () override  
*Save the view state to disk - called by `StateSavingObject::saveState()`*

## Protected Member Functions inherited from Digikam::DMultiTabBar

- virtual void `fontChange` (const QFont &)
- void `updateSeparator` ()

## Protected Member Functions inherited from Digikam::StateSavingObject

- QString `entryName` (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup `getConfigGroup` () const  
*Returns the config group that must be used for state saving and loading.*

## Additional Inherited Members

## Public Types inherited from Digikam::DMultiTabBar

- enum `TextStyle` { `ActiveIconText` = 0 , `AllIconsText` = 2 }
- The list of available styles for `DMultiTabBar`.*

## Public Types inherited from Digikam::StateSavingObject

- enum `StateSavingDepth` { `INSTANCE` , `DIRECT_CHILDREN` , `RECURSIVE` }
- This enum defines the "depth" of the `StateSavingObject::loadState()` and `StateSavingObject::saveState()` methods.*

## Protected Slots inherited from Digikam::ItemPropertiesSideBar

- virtual void `slotChangedTab` (QWidget \*tab)

## Protected Attributes inherited from Digikam::ItemPropertiesSideBar

- `ItemPropertiesColorsTab` \* `m_colorTab` = nullptr
- QRect `m_currentRect`
- QUrl `m_currentURL`
- bool `m_dirtyColorTab` = false
- bool `m_dirtyGpsTab` = false
- bool `m_dirtyHistoryTab` = false
- bool `m_dirtyMetadataTab` = false
- bool `m_dirtyPropertiesTab` = false
- `ItemPropertiesGPSTab` \* `m_gpsTab` = nullptr
- DImg \* `m_image` = nullptr
- `ItemPropertiesMetadataTab` \* `m_metadataTab` = nullptr
- QStackedWidget \* `m_propertiesStackedView` = nullptr
- `ItemPropertiesTab` \* `m_propertiesTab` = nullptr
- `ItemSelectionPropertiesTab` \* `m_selectionPropertiesTab` = nullptr

## 9.856.1 Member Function Documentation

### 9.856.1.1 doLoadState()

```
void Digikam::ItemPropertiesSideBarDB::doLoadState ( ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ItemPropertiesSideBar](#).

### 9.856.1.2 doSaveState()

```
void Digikam::ItemPropertiesSideBarDB::doSaveState ( ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ItemPropertiesSideBar](#).

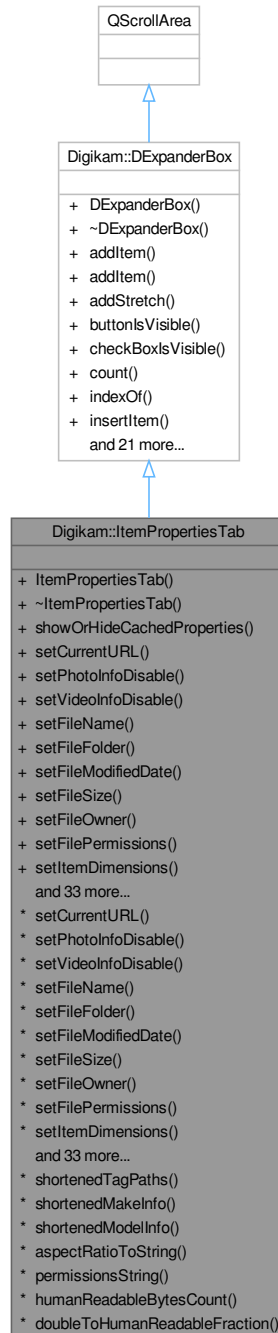
### 9.856.1.3 itemChanged()

```
void Digikam::ItemPropertiesSideBarDB::itemChanged (
    const QUrl & url,
    const QRect & rect = QRect(),
    DImg *const img = nullptr ) [override], [virtual]
```

Reimplemented from [Digikam::ItemPropertiesSideBar](#).

## 9.857 Digikam::ItemPropertiesTab Class Reference

Inheritance diagram for Digikam::ItemPropertiesTab:



### Public Types

- enum **Section** {  
**FileProperties** = 0 , **ImageProperties** , **PhotoProperties** , **VideoProperties** ,  
**digiKamProperties** , **TagsProperties** , **LocationProperties** , **RightProperties** }

## Public Member Functions

- **ItemPropertiesTab** (QWidget \*const parent)
- void **showOrHideCachedProperties** ()
  
- void **setCurrentURL** (const QUrl &url=QUrl())  
*Setter methods (itempropiestab\_setters.cpp)*
- void **setPhotoInfoDisable** (const bool b)
- void **setVideoInfoDisable** (const bool b)
- void **setFileName** (const QString &str)
- void **setFileFolder** (const QString &str)
- void **setFileModifiedDate** (const QString &str)
- void **setFileSize** (const QString &str)
- void **setFileOwner** (const QString &str)
- void **setFilePermissions** (const QString &str)
- void **setItemDimensions** (const QString &str)
- void **setImageRatio** (const QString &str)
- void **setImageMime** (const QString &str)
- void **setImageBitDepth** (const QString &str)
- void **setImageColorMode** (const QString &str)
- void **setHasSidecar** (const QString &str)
- void **setHasGPSInfo** (const QString &str)
- void **setVersionnedInfo** (const QString &str)
- void **setGroupedInfo** (const QString &str)
- void **setPhotoMake** (const QString &str)
- void **setPhotoModel** (const QString &str)
- void **setPhotoDateTime** (const QString &str)
- void **setPhotoLens** (const QString &str)
- void **setPhotoAperture** (const QString &str)
- void **setPhotoFocalLength** (const QString &str)
- void **setPhotoExposureTime** (const QString &str)
- void **setPhotoSensitivity** (const QString &str)
- void **setPhotoExposureMode** (const QString &str)
- void **setPhotoFlash** (const QString &str)
- void **setPhotoWhiteBalance** (const QString &str)
- void **setVideoAspectRatio** (const QString &str)
- void **setVideoAudioBitRate** (const QString &str)
- void **setVideoAudioChannelType** (const QString &str)
- void **setVideoAudioCodec** (const QString &str)
- void **setVideoDuration** (const QString &str)
- void **setVideoFrameRate** (const QString &str)
- void **setVideoVideoCodec** (const QString &str)
- void **setTitle** (const QString &str)
- void **setCaption** (const QString &str)
- void **setPickLabel** (int pickId)
- void **setColorLabel** (int colorId)
- void **setRating** (int rating)
- void **setTags** (const QStringList &tagPaths, const QStringList &tagNames=QStringList(), const QStringList &peopleTagPaths=QStringList(), const QStringList &peopleTagNames=QStringList())
- void **setTemplate** (const [Template](#) &t)

## Public Member Functions inherited from Digikam::DExpanderBox

- **DExpanderBox** (QWidget \*const parent=nullptr)
  - void **addItem** (QWidget \*const w, const QIcon &icon, const QString &txt, const QString &objName, bool expandBydefault)
    - Add [DLabelExpander](#) item at end of box layout with these settings : 'w' : the widget hosted by [DLabelExpander](#).*
  - void **addItem** (QWidget \*const w, const QString &txt, const QString &objName, bool expandBydefault)
  - void **addStretch** ()
  - bool **buttonIsVisible** (int index) const
  - bool **checkboxIsVisible** (int index) const
  - int **count** () const
  - int **indexOf** ([DLabelExpander](#) \*const widget) const
  - void **insertItem** (int index, QWidget \*const w, const QIcon &icon, const QString &txt, const QString &objName, bool expandBydefault)
    - Insert [DLabelExpander](#) item at box layout index with these settings : 'w' : the widget hosted by [DLabelExpander](#).*
  - void **insertItem** (int index, QWidget \*const w, const QString &txt, const QString &objName, bool expandBydefault)
  - void **insertStretch** (int index)
  - bool **isChecked** (int index) const
  - bool **isItemEnabled** (int index) const
  - bool **isItemExpanded** (int index) const
  - QIcon **itemIcon** (int index) const
  - QString **itemText** (int index) const
  - QString **itemToolTip** (int index) const
  - virtual void **readSettings** (KConfigGroup &group)
  - void **removeItem** (int index)
  - void **setButtonIcon** (int index, const QIcon &icon)
  - void **setButtonVisible** (int index, bool b)
  - void **setCheckBoxVisible** (int index, bool b)
  - void **setChecked** (int index, bool b)
  - void **setItemEnabled** (int index, bool enabled)
  - void **setItemExpanded** (int index, bool b)
  - void **setItemIcon** (int index, const QIcon &icon)
  - void **setItemText** (int index, const QString &txt)
  - void **setItemToolTip** (int index, const QString &tip)
  - [DLabelExpander](#) \* **widget** (int index) const
  - virtual void **writeSettings** (KConfigGroup &group)
- 
- static QStringList **shortenedTagPaths** (const QStringList &tagPaths, QList< QVariant > \*identifiers=nullptr)
    - Helper methods (itempropertistab\_helpers.cpp)*
  - static void **shortenedMakeInfo** (QString &make)
    - This methods shortens make an model camera info to prevent bloating GUI See bug #265231 for details.*
  - static void **shortenedModelInfo** (QString &model)
  - static bool **aspectRatioToString** (int width, int height, QString &arString)
    - Write a string with aspect ratio information formatted.*
  - static QString **permissionsString** (const QFile::Info &fi)
    - Return file permissions string.*
  - static QString **humanReadableBytesCount** (qint64 bytes, bool si=false)
    - Return an human readable string of file size in 'bytes'.*

## Additional Inherited Members

### Signals inherited from [Digikam::DExpanderBox](#)

- void **signalItemButtonPressed** (int index)
- void **signalItemExpanded** (int index, bool b)
- void **signalItemToggled** (int index, bool b)

## 9.857.1 Member Function Documentation

### 9.857.1.1 `humanReadableBytesCount()`

```
QString Digikam::ItemPropertiesTab::humanReadableBytesCount (
    qint64 bytes,
    bool si = false ) [static]
```

If 'si' is true, a decade of bytes is interpreted on base of 1000 byte, else 1024.

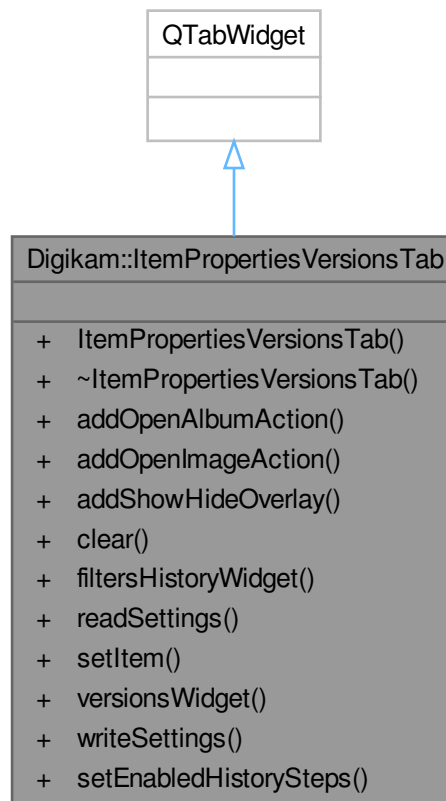
### 9.857.1.2 `shortenedTagPaths()`

```
QStringList Digikam::ItemPropertiesTab::shortenedTagPaths (
    const QStringList & tagPaths,
    QList< QVariant > * identifiers = nullptr ) [static]
```

Shortens the tag paths by sorting and then cutting identical paths from the second and following paths (only the first item gives the full path). If you want to retain information about which tag path is sorted where, you can optionally give a QVariant list. This list shall contain an identifier for the tag path at the same index and will be resorted as the returned list.

## 9.858 Digikam::ItemPropertiesVersionsTab Class Reference

Inheritance diagram for Digikam::ItemPropertiesVersionsTab:



### Public Slots

- void **setEnabledHistorySteps** (int count)

### Signals

- void **actionTriggered** (const [ItemInfo](#) &info)
- void **imageSelected** (const [ItemInfo](#) &info)

### Public Member Functions

- **ItemPropertiesVersionsTab** (QWidget \*const parent)
- void **addOpenAlbumAction** (const [ItemModel](#) \*referenceModel)
- void **addOpenImageAction** ()
- void **addShowHideOverlay** ()
- void **clear** ()
- [FiltersHistoryWidget](#) \* **filtersHistoryWidget** () const
- void **readSettings** (KConfigGroup &group)
- void **setItem** (const [ItemInfo](#) &info, const [DImageHistory](#) &history)
- [VersionsWidget](#) \* **versionsWidget** () const
- void **writeSettings** (KConfigGroup &group)

## 9.859 Digikam::ItemQueryBuilder Class Reference

### Public Member Functions

- QString **buildQuery** (const QString &q, QList< QVariant > \*boundValues, [ItemQueryPostHooks](#) \*const hooks) const
- QString **buildQueryFromUrl** (const QUrl &url, QList< QVariant > \*boundValues) const
- QString **buildQueryFromXml** (const QString &xml, QList< QVariant > \*boundValues, [ItemQueryPostHooks](#) \*const hooks) const
- QString **convertFromUrlToXml** (const QUrl &url) const
- void **setImageTagPropertiesJoined** (bool isJoined)

*Use for special queries where ImageTagProperties table is JOIN'ed.*

### Static Public Member Functions

- static void **addNoEffectContent** (QString &sql, SearchXml::Operator op)
- static void **addSqlOperator** (QString &sql, SearchXml::Operator op, bool isFirst)
- static void **addSqlRelation** (QString &sql, SearchXml::Relation rel)

### Protected Member Functions

- bool **buildField** (QString &sql, [SearchXmlCachingReader](#) &reader, const QString &name, QList< QVariant > \*boundValues, [ItemQueryPostHooks](#) \*const hooks) const
- void **buildGroup** (QString &sql, [SearchXmlCachingReader](#) &reader, QList< QVariant > \*boundValues, [ItemQueryPostHooks](#) \*const hooks) const
- QString **possibleDate** (const QString &str, bool &exact) const

### Protected Attributes

- bool **m\_imageTagPropertiesJoined** = false
- QString **m\_longMonths** [12]
- QString **m\_shortMonths** [12]

## 9.859.1 Member Function Documentation

### 9.859.1.1 setImageTagPropertiesJoined()

```
void Digikam::ItemQueryBuilder::setImageTagPropertiesJoined (
    bool isJoined )
```

(Default: false)

## 9.860 Digikam::ItemQueryPostHook Class Reference

### Public Member Functions

- **ItemQueryPostHook** ()=default  
*This is the single hook, ItemQueryPostHookS is the container.*
- virtual bool **checkPosition** (double, double)



## 9.861 Digikam::ItemQueryPostHooks Class Reference

### Public Member Functions

- void `addHook` (`ItemQueryPostHook *const hook`)  
*Called by `ItemQueryBuilder`.*
- bool `checkPosition` (double `latitudeNumber`, double `longitudeNumber`)  
*Call this method after passing the object to `buildQuery` and executing the statement.*

### Protected Attributes

- `QList< ItemQueryPostHook * > m_postHooks`

### 9.861.1 Member Function Documentation

#### 9.861.1.1 `addHook()`

```
void Digikam::ItemQueryPostHooks::addHook (  
    ItemQueryPostHook *const hook )
```

Ownership of the object is passed.

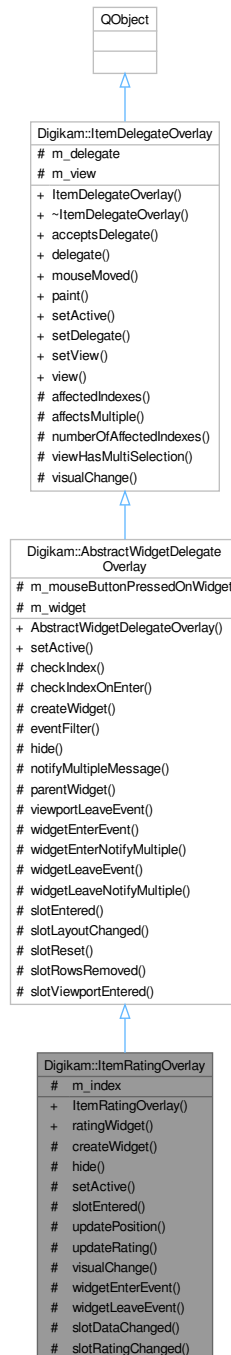
#### 9.861.1.2 `checkPosition()`

```
bool Digikam::ItemQueryPostHooks::checkPosition (  
    double latitudeNumber,  
    double longitudeNumber )
```

Returns true if the search is matched.

## 9.862 Digikam::ItemRatingOverlay Class Reference

Inheritance diagram for Digikam::ItemRatingOverlay:



### Signals

- void **ratingEdited** (const QList< QModelIndex > &indexes, int rating)

## Signals inherited from [Digikam::ItemDelegateOverlay](#)

- void **hideNotification** ()
- void **requestNotification** (const QModelIndex &index, const QString &message)
- void **update** (const QModelIndex &index)

## Public Member Functions

- **ItemRatingOverlay** (QObject \*const parent)
- [RatingWidget](#) \* **ratingWidget** () const

## Public Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- [AbstractWidgetDelegateOverlay](#) (QObject \*const parent)  
*This class provides functionality for using a widget in an overlay.*

## Public Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- **ItemDelegateOverlay** (QObject \*const parent=nullptr)
- virtual bool **acceptsDelegate** (QAbstractItemDelegate \*) const
- QAbstractItemDelegate \* **delegate** () const
- virtual void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)  
*Only these two methods are implemented as virtual methods.*
- virtual void **paint** (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index)
- void **setDelegate** (QAbstractItemDelegate \*delegate)
- void **setView** (QAbstractItemView \*view)
- QAbstractItemView \* **view** () const

## Protected Slots

- void **slotDataChanged** (const QModelIndex &, const QModelIndex &)
- void **slotRatingChanged** (int)

## Protected Slots inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- virtual void **slotLayoutChanged** ()
- virtual void **slotReset** ()  
*Default implementations of these three slots call `hide()`*
- virtual void **slotRowsRemoved** (const QModelIndex &parent, int start, int end)
- virtual void **slotViewportEntered** ()

## Protected Slots inherited from [Digikam::ItemDelegateOverlay](#)

### Protected Member Functions

- QWidget \* [createWidget](#) () override  
*Create your widget here.*
- void [hide](#) () override  
*Called when the widget shall be hidden (mouse cursor left index, viewport, uninstalled etc.).*
- void [setActive](#) (bool) override  
*If active is true, this will call [createWidget\(\)](#), initialize the widget for use, and setup connections for the virtual slots.*
- void [slotEntered](#) (const QModelIndex &index) override  
*Default implementation shows the widget iff the index is valid and [checkIndex](#) returns true.*
- void [updatePosition](#) ()
- void [updateRating](#) ()
- void [visualChange](#) () override  
*Called when any change from the delegate occurs - when the overlay is installed, when size hints, styles or fonts change.*
- void [widgetEnterEvent](#) () override  
*Called when a [QEvent::Enter](#) resp.*
- void [widgetLeaveEvent](#) () override

### Protected Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- virtual bool [checkIndex](#) (const QModelIndex &index) const  
*Return true here if you want to show the overlay for the given index.*
- bool [checkIndexOnEnter](#) (const QModelIndex &index) const  
*Utility method called from [slotEntered](#).*
- bool [eventFilter](#) (QObject \*obj, QEvent \*event) override
- virtual QString [notifyMultipleMessage](#) (const QModelIndex &, int number)
- QWidget \* [parentWidget](#) () const  
*Returns the widget to be used as parent for your widget created in [createWidget\(\)](#)*
- virtual void [viewportLeaveEvent](#) (QObject \*obj, QEvent \*event)  
*Called when a [QEvent::Leave](#) of the viewport is received.*
- void [widgetEnterNotifyMultiple](#) (const QModelIndex &index)  
*A sample implementation for above methods.*
- void [widgetLeaveNotifyMultiple](#) ()

### Protected Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- QList< QModelIndex > [affectedIndexes](#) (const QModelIndex &index) const
- bool [affectsMultiple](#) (const QModelIndex &index) const  
*For the context that an overlay can affect multiple items: Assuming the currently overlaid index is given.*
- int [numberOfAffectedIndexes](#) (const QModelIndex &index) const
- bool [viewHasMultiSelection](#) () const  
*Utility method.*

### Protected Attributes

- QPersistentModelIndex [m\\_index](#)

## Protected Attributes inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- bool `m_mouseButtonPressedOnWidget` = false
- `QWidget * m_widget` = nullptr

## Protected Attributes inherited from [Digikam::ItemDelegateOverlay](#)

- `QAbstractItemDelegate * m_delegate` = nullptr
- `QAbstractItemView * m_view` = nullptr

## 9.862.1 Member Function Documentation

### 9.862.1.1 `createWidget()`

```
QWidget * Digikam::ItemRatingOverlay::createWidget ( ) [override], [protected], [virtual]
```

When creating the object, pass [parentWidget\(\)](#) as parent widget. Ownership of the object is passed. It will be deleted in [setActive\(false\)](#).

Implements [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.862.1.2 `hide()`

```
void Digikam::ItemRatingOverlay::hide ( ) [override], [protected], [virtual]
```

Default implementation [hide\(\)](#)s `m_widget`.

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.862.1.3 `setActive()`

```
void Digikam::ItemRatingOverlay::setActive (
    bool active ) [override], [protected], [virtual]
```

If active is false, this will delete the widget and disconnect all signal from model and view to this object (!)

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.862.1.4 `slotEntered()`

```
void Digikam::ItemRatingOverlay::slotEntered (
    const QModelIndex & index ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

**9.862.1.5 visualChange()**

```
void Digikam::ItemRatingOverlay::visualChange ( ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ItemDelegateOverlay](#).

**9.862.1.6 widgetEnterEvent()**

```
void Digikam::ItemRatingOverlay::widgetEnterEvent ( ) [override], [protected], [virtual]
```

QEvent::Leave event for the widget is received. The default implementation does nothing.

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

**9.862.1.7 widgetLeaveEvent()**

```
void Digikam::ItemRatingOverlay::widgetLeaveEvent ( ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

## 9.863 Digikam::ItemRotateOverlay Class Reference

Inheritance diagram for Digikam::ItemRotateOverlay:



### Signals

- void **signalRotate** (const QList< QModelIndex > &indexes)

## Signals inherited from [Digikam::ItemDelegateOverlay](#)

- void **hideNotification** ()
- void **requestNotification** (const QModelIndex &index, const QString &message)
- void **update** (const QModelIndex &index)

## Public Member Functions

- **ItemRotateOverlay** (ItemRotateOverlayDirection dir, QObject \*const parent)
- ItemRotateOverlayDirection **direction** () const
- bool **isLeft** () const
- bool **isRight** () const
- void **setActive** (bool active) override  
*Will call [createButton\(\)](#).*

## Public Member Functions inherited from [Digikam::HoverButtonDelegateOverlay](#)

- **HoverButtonDelegateOverlay** (QObject \*const parent)
- [ItemViewHoverButton](#) \* **button** () const

## Public Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- [AbstractWidgetDelegateOverlay](#) (QObject \*const parent)  
*This class provides functionality for using a widget in an overlay.*

## Public Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- **ItemDelegateOverlay** (QObject \*const parent=nullptr)
- virtual bool **acceptsDelegate** (QAbstractItemDelegate \*) const
- QAbstractItemDelegate \* **delegate** () const
- virtual void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)  
*Only these two methods are implemented as virtual methods.*
- virtual void **paint** (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index)
- void **setDelegate** (QAbstractItemDelegate \*delegate)
- void **setView** (QAbstractItemView \*view)
- QAbstractItemView \* **view** () const

## Static Public Member Functions

- static [ItemRotateOverlay](#) \* **left** (QObject \*const parent)
- static [ItemRotateOverlay](#) \* **right** (QObject \*const parent)

## Protected Member Functions

- bool **checkIndex** (const QModelIndex &index) const override  
*Return true here if you want to show the overlay for the given index.*
- [ItemViewHoverButton](#) \* **createButton** () override  
*Create your widget here.*
- void **updateButton** (const QModelIndex &index) override  
*Called when a new index is entered.*
- void **widgetEnterEvent** () override  
*Called when a QEvent::Enter resp.*
- void **widgetLeaveEvent** () override



## Protected Member Functions inherited from [Digikam::HoverButtonDelegateOverlay](#)

- QWidget \* **createWidget** () override  
*Create your widget here.*
- void **visualChange** () override  
*Called when any change from the delegate occurs - when the overlay is installed, when size hints, styles or fonts change.*

## Protected Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- bool **checkIndexOnEnter** (const QModelIndex &index) const  
*Utility method called from slotEntered.*
- bool **eventFilter** (QObject \*obj, QEvent \*event) override
- virtual void **hide** ()  
*Called when the widget shall be hidden (mouse cursor left index, viewport, uninstalled etc.).*
- virtual QString **notifyMultipleMessage** (const QModelIndex &, int number)
- QWidget \* **parentWidget** () const  
*Returns the widget to be used as parent for your widget created in [createWidget\(\)](#)*
- virtual void **viewportLeaveEvent** (QObject \*obj, QEvent \*event)  
*Called when a QEvent::Leave of the viewport is received.*
- void **widgetEnterNotifyMultiple** (const QModelIndex &index)  
*A sample implementation for above methods.*
- void **widgetLeaveNotifyMultiple** ()

## Protected Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- QList< QModelIndex > **affectedIndexes** (const QModelIndex &index) const
- bool **affectsMultiple** (const QModelIndex &index) const  
*For the context that an overlay can affect multiple items: Assuming the currently overlaid index is given.*
- int **numberOfAffectedIndexes** (const QModelIndex &index) const
- bool **viewHasMultiSelection** () const  
*Utility method.*

## Additional Inherited Members

### Protected Slots inherited from [Digikam::HoverButtonDelegateOverlay](#)

- void **slotEntered** (const QModelIndex &index) override
- void **slotReset** () override

### Protected Slots inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- virtual void **slotEntered** (const QModelIndex &index)  
*Default implementation shows the widget iff the index is valid and checkIndex returns true.*
- virtual void **slotLayoutChanged** ()
- virtual void **slotReset** ()  
*Default implementations of these three slots call [hide\(\)](#)*
- virtual void **slotRowsRemoved** (const QModelIndex &parent, int start, int end)
- virtual void **slotViewportEntered** ()

## Protected Slots inherited from [Digikam::ItemDelegateOverlay](#)

## Protected Attributes inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- bool `m_mouseButtonPressedOnWidget` = false
- `QWidget * m_widget` = nullptr

## Protected Attributes inherited from [Digikam::ItemDelegateOverlay](#)

- `QAbstractItemDelegate * m_delegate` = nullptr
- `QAbstractItemView * m_view` = nullptr

## 9.863.1 Member Function Documentation

### 9.863.1.1 `checkIndex()`

```
bool Digikam::ItemRotateOverlay::checkIndex (
    const QModelIndex & index ) const [override], [protected], [virtual]
```

The default implementation returns true.

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.863.1.2 `createButton()`

```
ItemViewHoverButton * Digikam::ItemRotateOverlay::createButton ( ) [override], [protected],
[virtual]
```

Pass `view()` as parent.

Implements [Digikam::HoverButtonDelegateOverlay](#).

### 9.863.1.3 `setActive()`

```
void Digikam::ItemRotateOverlay::setActive (
    bool active ) [override], [virtual]
```

Reimplemented from [Digikam::HoverButtonDelegateOverlay](#).

### 9.863.1.4 `updateButton()`

```
void Digikam::ItemRotateOverlay::updateButton (
    const QModelIndex & index ) [override], [protected], [virtual]
```

Reposition your button here, adjust and store state.

Implements [Digikam::HoverButtonDelegateOverlay](#).

### 9.863.1.5 widgetEnterEvent()

```
void Digikam::ItemRotateOverlay::widgetEnterEvent ( ) [override], [protected], [virtual]
```

QEvent::Leave event for the widget is received. The default implementation does nothing.

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.863.1.6 widgetLeaveEvent()

```
void Digikam::ItemRotateOverlay::widgetLeaveEvent ( ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

## 9.864 Digikam::ItemRotateOverlayButton Class Reference

Inheritance diagram for Digikam::ItemRotateOverlayButton:



### Public Member Functions

- **ItemRotateOverlayButton** (ItemRotateOverlayDirection dir, QAbstractItemView \*const parentView)
- QSize [sizeHint](#) () const override

*Reimplement to match the size of your icon.*

## Public Member Functions inherited from [Digikam::ItemViewHoverButton](#)

- **ItemViewHoverButton** (QAbstractItemView \*const parentView)
- QModelIndex **index** () const
- void **initIcon** ()
- void **reset** ()
- void **setIndex** (const QModelIndex &index)
- void **setVisible** (bool visible) override

## Protected Member Functions

- QIcon **icon** () override  
*Return your icon here.*
- void **updateToolTip** () override  
*Optionally update tooltip here.*

## Protected Member Functions inherited from [Digikam::ItemViewHoverButton](#)

- void **enterEvent** (QEnterEvent \*event)
- void **leaveEvent** (QEvent \*event)
- void **paintEvent** (QPaintEvent \*event)
- void **setup** ()  
*to call in children class constructors to init signal/slot connections.*

## Protected Attributes

- ItemRotateOverlayDirection const **m\_direction**

## Protected Attributes inherited from [Digikam::ItemViewHoverButton](#)

- QTimerLine \* **m\_fadingTimeLine** = nullptr
- int **m\_fadingValue** = 0
- QIcon **m\_icon**
- QPersistentModelIndex **m\_index**
- bool **m\_isHovered** = false

## Additional Inherited Members

## Protected Slots inherited from [Digikam::ItemViewHoverButton](#)

- void **refreshIcon** ()
- void **setFadingValue** (int value)
- void **startFading** ()
- void **stopFading** ()

## 9.864.1 Member Function Documentation

### 9.864.1.1 icon()

`QIcon Digikam::ItemRotateOverlayButton::icon ( ) [override], [protected], [virtual]`

Will be queried again on toggle.

Implements [Digikam::ItemViewHoverButton](#).

### 9.864.1.2 sizeHint()

`QSize Digikam::ItemRotateOverlayButton::sizeHint ( ) const [override], [virtual]`

Implements [Digikam::ItemViewHoverButton](#).

### 9.864.1.3 updateToolTip()

`void Digikam::ItemRotateOverlayButton::updateToolTip ( ) [override], [protected], [virtual]`

Will be called again on state change.

Reimplemented from [Digikam::ItemViewHoverButton](#).

## 9.865 Digikam::ItemScanInfo Class Reference

### Public Member Functions

- `bool isNull ( ) const`

### Public Attributes

- `int albumID = 0`
- `DatabaselItem::Category category = DatabaselItem::UndefinedCategory`
- `qulonglong fileSize = 0`
- `qulonglong id = 0`
- `QString itemName`
- `QDateTime modificationDate`
- `DatabaselItem::Status status = DatabaselItem::UndefinedStatus`
- `QString uniqueHash`

## 9.866 Digikam::ItemScanner Class Reference

### Public Types

- `enum ScanMode { NewScan , ModifiedScan , Rescan , CleanScan }`

## Public Member Functions

- [ItemScanner](#) (const QFileInfo &info)  
*Construct an [ItemScanner](#) from an existing QFileInfo object.*
- [ItemScanner](#) (const QFileInfo &info, const [ItemScanInfo](#) &Iteminfo)  
*Construct an [ItemScanner](#) object from an existing QFileInfo and [ItemScanInfo](#) object.*
- [ItemScanner](#) (qulonglong imageid)  
*Construct an [ItemScanner](#) for an image in the database.*
- const [ItemScanInfo](#) & [itemScanInfo](#) () const  
*Provides access to the information retrieved by scanning.*
- void [loadFromDisk](#) ()  
*Loads data from disk (metadata, image file properties).*
- void [setCategory](#) (DatabaseItem::Category category)  
*Inform the scanner about the category of the file.*

## Static Public Member Functions

- static QString [formatToString](#) (const QString &format)  
*Helper method to translate enum values to user presentable strings.*

## Operations on History Metadata

- bool [hasHistoryToResolve](#) () const  
*Returns true if this file has been marked as needing history resolution at a later stage.*
- void [scanBalooInfo](#) ()  
*scanBalooInfo - retrieve tags, comments and rating from Baloo Desktop service.*
- static bool [resolveImageHistory](#) (qulonglong id, QList< qulonglong > \*needTaggingIds=nullptr)  
*Resolves the image history of the image id by filling the ImageRelations table for all contained referred images.*
- static bool [resolveImageHistory](#) (qulonglong imageid, const QString &historyXml, QList< qulonglong > \*needTaggingIds=nullptr)
- static void [tagItemHistoryGraph](#) (qulonglong id)  
*Takes the history graph reachable from the given image, and assigns versioning tags to all entries based on history image types and graph structure.*
- static [DImageHistory](#) [resolvedImageHistory](#) (const [DImageHistory](#) &history, bool mustBeAvailable=false)  
*All referred images of the given history will be resolved.*
- static bool [sameReferredImage](#) (const [HistoryImageId](#) &id1, const [HistoryImageId](#) &id2)  
*Determines if the two ids refer to the same image.*
- static QList< qulonglong > [resolveHistoryImageId](#) (const [HistoryImageId](#) &historyId)  
*Returns all image ids fulfilling the given image id.*
- void [scanImageHistory](#) ()
- void [commitImageHistory](#) ()
- void [scanImageHistoryIfModified](#) ()
- QString [uniqueHash](#) () const

## Operations with Database

- void **newFile** (int albumId)
 

*Call this when you want [ItemScanner](#) to add a new file to the database and read all information into the database.*
- void **newFileFullScan** (int albumId)
 

*Call this when you want [ItemScanner](#) to add a new file to the database and read all information into the database.*
- void **rescan** ()
 

*Call this to take an existing image in the database, but re-read all information from the file into the database, possibly overwriting information there.*
- void **cleanScan** ()
 

*This is the same as [rescan\(\)](#) but the database metadata will be cleaned up if the corresponding metadata write option is enabled.*
- void **commit** ()
 

*Commits the scanned information to the database.*
- qlonglong **id** () const
 

*Returns the image id of the scanned file, if (yet) available.*
- void **copiedFrom** (int albumId, qlonglong srcId)
 

*Similar to [newFile](#).*
- static void **sortByProximity** (QList< [ItemInfo](#) > &infos, const [ItemInfo](#) &subject)
 

*Sort a list of infos by proximity to the given subject.*
- bool **copyFromSource** (qlonglong src)
- void **commitCopyImageAttributes** ()
- void **cleanDatabaseMetadata** ()
- void **prepareAddImage** (int albumId)
- bool **commitAddImage** ()

## Operations on File Metadata

- void **fileModified** ()
 

*Call this when you have detected that a file in the database has been modified on disk.*
- static void **fillCommonContainer** (qlonglong imageid, [ImageCommonContainer](#) \*const container)
 

*Returns File-metadata container with user-presentable information.*
- static QDateTime **creationDateFromFilesystem** (const QFileInfo &info)
 

*Returns a suitable creation date from file system information.*
- void **prepareUpdateImage** ()
- void **commitUpdateImage** ()
- bool **scanFromIdenticalFile** ()
- void **scanFile** (ScanMode mode)
- void **scanItemInformation** ()
- void **commitItemInformation** ()

## Operations on Photo Metadata

- static QString **iptcCorePropertyName** (MetadataInfo::Field field)
 

*Helper method to return official property name by which IPTC core properties are stored in the database ([ItemCopyright](#) and [ImageProperties](#) table).*
- static MetadataFields **allImageMetadataFields** ()
- QString **detectImageFormat** () const
- void **scanImageMetadata** ()
- void **commitImageMetadata** ()
- void **scanItemPosition** ()
- void **commitItemPosition** ()



- void **scanItemComments** ()
- void **commitItemComments** ()
- void **scanItemCopyright** ()
- void **commitItemCopyright** ()
- void **scanIPTCCore** ()
- void **commitIPTCCore** ()
- void **scanTags** ()
- void **commitTags** ()
- void **scanFaces** ()
- void **commitFaces** ()
- bool **checkRatingFromMetadata** (const QVariant &ratingFromMetadata) const
- void **checkCreationDateFromMetadata** (QVariant &dateFromMetadata) const

### Operations on Video Metadata

- static void **fillVideoMetadataContainer** (qulonglong imageid, [VideoMetadataContainer](#) \*const container)  
*Returns Video container with user-presentable information.*
- void **scanVideoInformation** ()
- void **scanVideoMetadata** ()
- void **commitVideoMetadata** ()
- QString **detectVideoFormat** () const
- QString **detectAudioFormat** () const
- static MetadataFields **allVideoMetadataFields** ()

## 9.866.1 Constructor & Destructor Documentation

### 9.866.1.1 ItemScanner() [1/3]

```
Digikam::ItemScanner::ItemScanner (
    const QFileInfo & info,
    const ItemScanInfo & Iteminfo )
```

This constructor shall be used with [fileModified\(\)](#) or [fullScan\(\)](#).

### 9.866.1.2 ItemScanner() [2/3]

```
Digikam::ItemScanner::ItemScanner (
    const QFileInfo & info ) [explicit]
```

Use this constructor if you intend to call [newFile\(\)](#).

### 9.866.1.3 ItemScanner() [3/3]

```
Digikam::ItemScanner::ItemScanner (
    qulonglong imageid ) [explicit]
```

File info, Scan info and the category will be retrieved from the database.

## 9.866.2 Member Function Documentation

### 9.866.2.1 commit()

```
void Digikam::ItemScanner::commit ( )
```

You must call this after scanning was done for any changes to take effect. Only this method will perform write operations to the database.

### 9.866.2.2 copiedFrom()

```
void Digikam::ItemScanner::copiedFrom (
    int albumId,
    qlonglong srcId )
```

Call this when you want [ItemScanner](#) to add a new file to the database which is a copy of another file, copying attributes from the src and rescanning other attributes as appropriate. Give the id of the album of the new file, and the id of the src file.

### 9.866.2.3 creationDateFromFilesystem()

```
QDateTime Digikam::ItemScanner::creationDateFromFilesystem (
    const QFileInfo & info ) [static]
```

Use this as a fallback if metadata is not available.

### 9.866.2.4 fileModified()

```
void Digikam::ItemScanner::fileModified ( )
```

Only two groups of fields will be updated in the database:

- filesystem specific properties (those that signaled you that the file has been modified because their state on disk differed from the state in the database)
- image specific properties, for which a difference in the database independent from the actual file does not make sense (width/height, bit depth, color model)

### 9.866.2.5 fillCommonContainer()

```
void Digikam::ItemScanner::fillCommonContainer (
    qlonglong imageid,
    ImageCommonContainer *const container ) [static]
```

These methods provide the reverse service: Not writing into the db, but reading from the db.

### 9.866.2.6 fillVideoMetadataContainer()

```
void Digikam::ItemScanner::fillVideoMetadataContainer (
    qlonglong imageid,
    VideoMetadataContainer *const container ) [static]
```

These methods provide the reverse service: Not writing into the db, but reading from the db.

### 9.866.2.7 iptcCorePropertyName()

```
QString Digikam::ItemScanner::iptcCorePropertyName (
    MetadataInfo::Field field ) [static]
```

Allowed arguments: All MetadataInfo::Fields starting with "IptcCore..."

### 9.866.2.8 itemScanInfo()

```
const ItemScanInfo & Digikam::ItemScanner::itemScanInfo ( ) const
```

The validity depends on the previously executed scan.

### 9.866.2.9 loadFromDisk()

```
void Digikam::ItemScanner::loadFromDisk ( )
```

This method is called from any of the main entry points above. You can call it before if you want to control the time when it is executed. Calling it a second time with data already loaded will do nothing.

### 9.866.2.10 newFileFullScan()

```
void Digikam::ItemScanner::newFileFullScan (
    int albumId )
```

This variant will not use the unique hash to establish identify with an existing entry, but read all information newly from the file.

### 9.866.2.11 resolvedImageHistory()

```
DImageHistory Digikam::ItemScanner::resolvedImageHistory (
    const DImageHistory & history,
    bool mustBeAvailable = false ) [static]
```

In the returned history, the actions are the same, while each referred image actually exists in the collection (if `mustBeAvailable` is true, it is even in a currently available collection). That means the number of referred images may be less or greater than initially. Note that this history may have peculiar properties, like multiple Original or Current entries (if the source entry resolves to multiple collection images), so this history is only for internal use, not for storage.

### 9.866.2.12 resolveImageHistory()

```
bool Digikam::ItemScanner::resolveImageHistory (
    qlonglong id,
    QList< qlonglong > * needTaggingIds = nullptr ) [static]
```

If needTaggingIds is given, all ids marked for needing tagging of the history graph are added.

### 9.866.2.13 sameReferredImage()

```
bool Digikam::ItemScanner::sameReferredImage (
    const HistoryImageId & id1,
    const HistoryImageId & id2 ) [static]
```

Does not check if such a referred image really exists.

### 9.866.2.14 setCategory()

```
void Digikam::ItemScanner::setCategory (
    DatabaseItem::Category category )
```

Required at least for [newFile\(\)](#) calls, recommended for calls with the first constructor above as well.

### 9.866.2.15 sortByProximity()

```
void Digikam::ItemScanner::sortByProximity (
    QList< ItemInfo > & infos,
    const ItemInfo & subject ) [static]
```

Infos are near if they are e.g. in the same album. They are not near if they are e.g. in different collections.

## 9.867 Digikam::ItemSelectionOverlay Class Reference

Inheritance diagram for Digikam::ItemSelectionOverlay:



### Public Member Functions

- **ItemSelectionOverlay** (QObject \*const parent)
- void [setActive](#) (bool active) override  
*Will call [createButton](#)().*

## Public Member Functions inherited from [Digikam::HoverButtonDelegateOverlay](#)

- [HoverButtonDelegateOverlay](#) (QObject \*const parent)
- [ItemViewHoverButton](#) \* **button** () const

## Public Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- [AbstractWidgetDelegateOverlay](#) (QObject \*const parent)  
*This class provides functionality for using a widget in an overlay.*

## Public Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- [ItemDelegateOverlay](#) (QObject \*const parent=nullptr)
- virtual bool **acceptsDelegate** (QAbstractItemDelegate \*) const
- QAbstractItemDelegate \* **delegate** () const
- virtual void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)  
*Only these two methods are implemented as virtual methods.*
- virtual void **paint** (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index)
- void **setDelegate** (QAbstractItemDelegate \*delegate)
- void **setView** (QAbstractItemView \*view)
- QAbstractItemView \* **view** () const

## Protected Slots

- void **slotClicked** (bool checked)
- void **slotSelectionChanged** (const QListSelection &, const QListSelection &)

## Protected Slots inherited from [Digikam::HoverButtonDelegateOverlay](#)

- void **slotEntered** (const QModelIndex &index) override
- void **slotReset** () override

## Protected Slots inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- virtual void **slotEntered** (const QModelIndex &index)  
*Default implementation shows the widget iff the index is valid and checkIndex returns true.*
- virtual void **slotLayoutChanged** ()
- virtual void **slotReset** ()  
*Default implementations of these three slots call `hide()`*
- virtual void **slotRowsRemoved** (const QModelIndex &parent, int start, int end)
- virtual void **slotViewportEntered** ()

## Protected Slots inherited from [Digikam::ItemDelegateOverlay](#)

### Protected Member Functions

- [ItemViewHoverButton](#) \* **createButton** () override  
*Create your widget here.*
- void **updateButton** (const QModelIndex &index) override  
*Called when a new index is entered.*

## Protected Member Functions inherited from Digikam::HoverButtonDelegateOverlay

- QWidget \* **createWidget** () override  
*Create your widget here.*
- void **visualChange** () override  
*Called when any change from the delegate occurs - when the overlay is installed, when size hints, styles or fonts change.*

## Protected Member Functions inherited from Digikam::AbstractWidgetDelegateOverlay

- virtual bool **checkIndex** (const QModelIndex &index) const  
*Return true here if you want to show the overlay for the given index.*
- bool **checkIndexOnEnter** (const QModelIndex &index) const  
*Utility method called from slotEntered.*
- bool **eventFilter** (QObject \*obj, QEvent \*event) override
- virtual void **hide** ()  
*Called when the widget shall be hidden (mouse cursor left index, viewport, uninstalled etc.).*
- virtual QString **notifyMultipleMessage** (const QModelIndex &, int number)
- QWidget \* **parentWidget** () const  
*Returns the widget to be used as parent for your widget created in [createWidget\(\)](#)*
- virtual void **viewportLeaveEvent** (QObject \*obj, QEvent \*event)  
*Called when a QEvent::Leave of the viewport is received.*
- virtual void **widgetEnterEvent** ()  
*Called when a QEvent::Enter resp.*
- void **widgetEnterNotifyMultiple** (const QModelIndex &index)  
*A sample implementation for above methods.*
- virtual void **widgetLeaveEvent** ()
- void **widgetLeaveNotifyMultiple** ()

## Protected Member Functions inherited from Digikam::ItemDelegateOverlay

- QList< QModelIndex > **affectedIndexes** (const QModelIndex &index) const
- bool **affectsMultiple** (const QModelIndex &index) const  
*For the context that an overlay can affect multiple items: Assuming the currently overlaid index is given.*
- int **numberOfAffectedIndexes** (const QModelIndex &index) const
- bool **viewHasMultiSelection** () const  
*Utility method.*

## Additional Inherited Members

## Signals inherited from Digikam::ItemDelegateOverlay

- void **hideNotification** ()
- void **requestNotification** (const QModelIndex &index, const QString &message)
- void **update** (const QModelIndex &index)

## Protected Attributes inherited from Digikam::AbstractWidgetDelegateOverlay

- bool **m\_mouseButtonPressedOnWidget** = false
- QWidget \* **m\_widget** = nullptr

## Protected Attributes inherited from [Digikam::ItemDelegateOverlay](#)

- `QAbstractItemDelegate * m_delegate = nullptr`
- `QAbstractItemView * m_view = nullptr`

### 9.867.1 Member Function Documentation

#### 9.867.1.1 `createButton()`

```
ItemViewHoverButton * Digikam::ItemSelectionOverlay::createButton ( ) [override], [protected], [virtual]
```

Pass `view()` as parent.

Implements [Digikam::HoverButtonDelegateOverlay](#).

#### 9.867.1.2 `setActive()`

```
void Digikam::ItemSelectionOverlay::setActive ( bool active ) [override], [virtual]
```

Reimplemented from [Digikam::HoverButtonDelegateOverlay](#).

#### 9.867.1.3 `updateButton()`

```
void Digikam::ItemSelectionOverlay::updateButton ( const QModelIndex & index ) [override], [protected], [virtual]
```

Reposition your button here, adjust and store state.

Implements [Digikam::HoverButtonDelegateOverlay](#).



## 9.868 Digikam::ItemSelectionOverlayButton Class Reference

Inheritance diagram for Digikam::ItemSelectionOverlayButton:



### Public Member Functions

- **ItemSelectionOverlayButton** (QAbstractItemView \*const parentView)
- QSize [sizeHint](#) () const override

*Reimplement to match the size of your icon.*

## Public Member Functions inherited from [Digikam::ItemViewHoverButton](#)

- **ItemViewHoverButton** (QAbstractItemView \*const parentView)
- QModelIndex **index** () const
- void **initIcon** ()
- void **reset** ()
- void **setIndex** (const QModelIndex &index)
- void **setVisible** (bool visible) override

## Protected Member Functions

- QIcon **icon** () override  
*Return your icon here.*
- void **updateToolTip** () override  
*Optionally update tooltip here.*

## Protected Member Functions inherited from [Digikam::ItemViewHoverButton](#)

- void **enterEvent** (QEnterEvent \*event)
- void **leaveEvent** (QEvent \*event)
- void **paintEvent** (QPaintEvent \*event)
- void **setup** ()  
*to call in children class constructors to init signal/slot connections.*

## Additional Inherited Members

## Protected Slots inherited from [Digikam::ItemViewHoverButton](#)

- void **refreshIcon** ()
- void **setFadingValue** (int value)
- void **startFading** ()
- void **stopFading** ()

## Protected Attributes inherited from [Digikam::ItemViewHoverButton](#)

- QTimerLine \* **m\_fadingTimeLine** = nullptr
- int **m\_fadingValue** = 0
- QIcon **m\_icon**
- QPersistentModelIndex **m\_index**
- bool **m\_isHovered** = false

## 9.868.1 Member Function Documentation

### 9.868.1.1 icon()

QIcon Digikam::ItemSelectionOverlayButton::icon ( ) [override], [protected], [virtual]

Will be queried again on toggle.

Implements [Digikam::ItemViewHoverButton](#).

### 9.868.1.2 sizeHint()

```
QSize Digikam::ItemSelectionOverlayButton::sizeHint ( ) const [override], [virtual]
```

Implements [Digikam::ItemViewHoverButton](#).

### 9.868.1.3 updateToolTip()

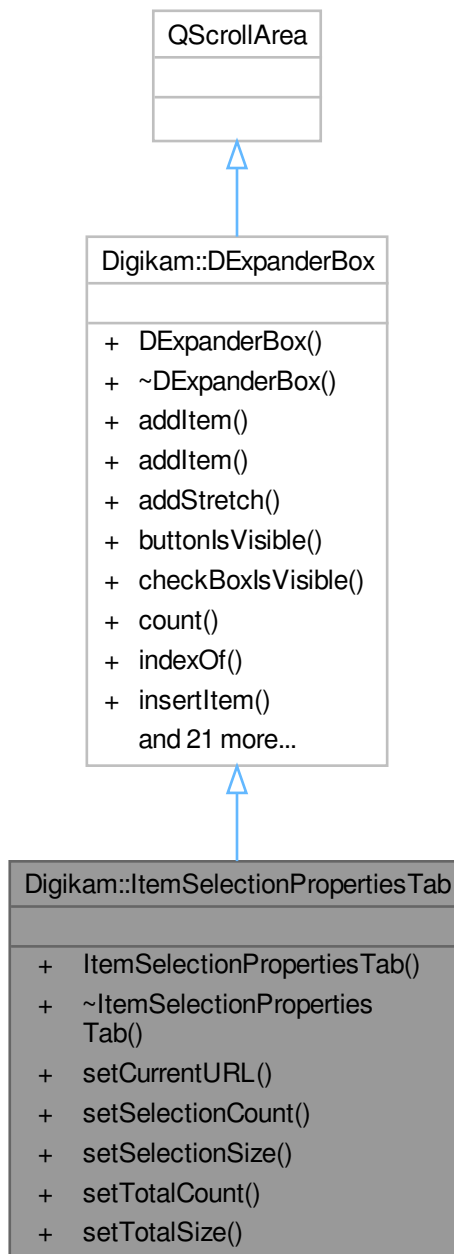
```
void Digikam::ItemSelectionOverlayButton::updateToolTip ( ) [override], [protected], [virtual]
```

Will be called again on state change.

Reimplemented from [Digikam::ItemViewHoverButton](#).

## 9.869 Digikam::ItemSelectionPropertiesTab Class Reference

Inheritance diagram for Digikam::ItemSelectionPropertiesTab:



### Public Member Functions

- **ItemSelectionPropertiesTab** (`QWidget *const parent`)
- void **setCurrentURL** (`const QUrl &url=QUrl()`)

- void **setSelectionCount** (const QString &str)
- void **setSelectionSize** (const QString &str)
- void **setTotalCount** (const QString &str)
- void **setTotalSize** (const QString &str)

## Public Member Functions inherited from Digikam::DExpanderBox

- **DExpanderBox** (QWidget \*const parent=nullptr)
- void **addItem** (QWidget \*const w, const QIcon &icon, const QString &txt, const QString &objName, bool expandBydefault)
 

*Add [DLabelExpander](#) item at end of box layout with these settings : 'w' : the widget hosted by [DLabelExpander](#).*
- void **addItem** (QWidget \*const w, const QString &txt, const QString &objName, bool expandBydefault)
- void **addStretch** ()
- bool **buttonIsVisible** (int index) const
- bool **checkboxIsVisible** (int index) const
- int **count** () const
- int **indexOf** ([DLabelExpander](#) \*const widget) const
- void **insertItem** (int index, QWidget \*const w, const QIcon &icon, const QString &txt, const QString &objName, bool expandBydefault)
 

*Insert [DLabelExpander](#) item at box layout index with these settings : 'w' : the widget hosted by [DLabelExpander](#).*
- void **insertItem** (int index, QWidget \*const w, const QString &txt, const QString &objName, bool expandBydefault)
- void **insertStretch** (int index)
- bool **isChecked** (int index) const
- bool **isItemEnabled** (int index) const
- bool **isItemExpanded** (int index) const
- QIcon **itemIcon** (int index) const
- QString **itemText** (int index) const
- QString **itemToolTip** (int index) const
- virtual void **readSettings** (KConfigGroup &group)
- void **removeItem** (int index)
- void **setButtonIcon** (int index, const QIcon &icon)
- void **setButtonVisible** (int index, bool b)
- void **setCheckBoxVisible** (int index, bool b)
- void **setChecked** (int index, bool b)
- void **setItemEnabled** (int index, bool enabled)
- void **setItemExpanded** (int index, bool b)
- void **setItemIcon** (int index, const QIcon &icon)
- void **setItemText** (int index, const QString &txt)
- void **setItemToolTip** (int index, const QString &tip)
- [DLabelExpander](#) \* **widget** (int index) const
- virtual void **writeSettings** (KConfigGroup &group)

## Additional Inherited Members

## Signals inherited from Digikam::DExpanderBox

- void **signalItemButtonPressed** (int index)
- void **signalItemExpanded** (int index, bool b)
- void **signalItemToggled** (int index, bool b)

## 9.870 Digikam::ItemShortInfo Class Reference

### Public Member Functions

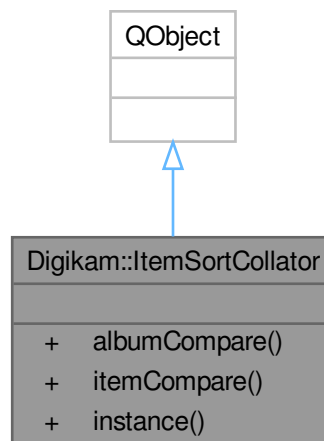
- bool **isNull** () const

### Public Attributes

- QString **album**
- int **albumID** = 0
- int **albumRootID** = 0
- qlonglong **id** = 0
- QString **itemName**

## 9.871 Digikam::ItemSortCollator Class Reference

Inheritance diagram for Digikam::ItemSortCollator:



### Public Member Functions

- int **albumCompare** (const QString &a, const QString &b, Qt::CaseSensitivity caseSensitive, bool natural) const
- int **itemCompare** (const QString &a, const QString &b, Qt::CaseSensitivity caseSensitive, bool natural) const

### Static Public Member Functions

- static `ItemSortCollator` \* **instance** ()  
*Global instance of internal item sort collator.*

## Friends

- class `ItemSortCollatorCreator`

## 9.871.1 Member Function Documentation

### 9.871.1.1 instance()

```
ItemSortCollator * Digikam::ItemSortCollator::instance ( ) [static]
```

All accessor methods are thread-safe.

## 9.872 Digikam::ItemSortSettings Class Reference

### Public Types

- enum `CategorizationMode` { `NoCategories` , `OneCategory` , `CategoryByAlbum` , `CategoryByFormat` , `CategoryByMonth` , `CategoryByFaces` }
- enum `SortOrder` { `AscendingOrder` = Qt::AscendingOrder , `DescendingOrder` = Qt::DescendingOrder , `DefaultOrder` }
- enum `SortRole` { `SortByFileName` , `SortByFilePath` , `SortByCreationDate` , `SortByModificationDate` , `SortByFileSize` , `SortByRating` , `SortByImageSize` , `SortByAspectRatio` , `SortByFaces` , `SortBySimilarity` , `SortByManualOrderAndName` , `SortByManualOrderAndDate` }

### Public Member Functions

- int `compare` (const `ItemInfo` &left, const `ItemInfo` &right) const  
*Compares the ItemInfos left and right.*
- int `compare` (const `ItemInfo` &left, const `ItemInfo` &right, `SortRole` sortRole) const  
— *Image Sorting* —
- int `compareCategories` (const `ItemInfo` &left, const `ItemInfo` &right, const `FaceTagsIface` &leftFace, const `FaceTagsIface` &rightFace) const  
*Compares the categories of left and right.*
- bool `isCategorized` () const
- bool `lessThan` (const `ItemInfo` &left, const `ItemInfo` &right) const  
*Returns true if left is less than right.*
- bool `lessThan` (const `QVariant` &left, const `QVariant` &right) const  
*Returns true if left QVariant is less than right.*
- bool `operator==` (const `ItemSortSettings` &other) const
- void `setCategorizationMode` (`CategorizationMode` mode)  
— *Categories* —
- void `setCategorizationSortOrder` (`SortOrder` order)
- void `setSortOrder` (`SortOrder` order)
- void `setSortRole` (`SortRole` role)
- void `setStringTypeNatural` (bool natural)
- `DatabaseFields::Set` `watchFlags` () const  
— *Change notification* —

## Static Public Member Functions

- `template<typename T >`  
`static int compareByOrder (const T &a, const T &b, Qt::SortOrder sortOrder)`
- `static int compareByOrder (int compareResult, Qt::SortOrder sortOrder)`  
*Takes a typical result from a compare method (0 is equal, -1 is less than, 1 is greater than) and applies the given sort order to it.*
- `template<typename T >`  
`static int compareValue (const T &a, const T &b)`  
*Returns the usual compare result of -1, 0, or 1 for lessThan, equals and greaterThan.*
- `static Qt::SortOrder defaultSortOrderForCategorizationMode (CategorizationMode mode)`
- `static Qt::SortOrder defaultSortOrderForSortRole (SortRole role)`
- `template<typename T >`  
`static bool lessThanByOrder (const T &a, const T &b, Qt::SortOrder sortOrder)`  
— Utilities —
- `static int naturalCompare (const QString &a, const QString &b, Qt::SortOrder sortOrder, Qt::CaseSensitivity caseSensitive=Qt::CaseSensitive, bool natural=true)`  
*Compares the two string by natural comparison and adheres to given sort order.*

## Public Attributes

- `Qt::CaseSensitivity categorizationCaseSensitivity = Qt::CaseSensitive`
- `CategorizationMode categorizationMode = NoCategories`
- `SortOrder categorizationSortOrder = DefaultOrder`
- `Qt::SortOrder currentCategorizationSortOrder = Qt::AscendingOrder`  
*Only Ascending or Descending, never DefaultOrder.*
- `Qt::SortOrder currentSortOrder = Qt::AscendingOrder`
- `Qt::CaseSensitivity sortCaseSensitivity = Qt::CaseSensitive`
- `SortOrder sortOrder = DefaultOrder`
- `SortRole sortRole = SortByFileName`
- `bool strTypeNatural = true`

## 9.872.1 Member Enumeration Documentation

### 9.872.1.1 CategorizationMode

```
enum Digikam::ItemSortSettings::CategorizationMode
```

#### Enumerator

NoCategories	categorization switched off
OneCategory	all items in one global category

### 9.872.1.2 SortOrder

```
enum Digikam::ItemSortSettings::SortOrder
```



## Enumerator

DefaultOrder	sort order depends on the chosen sort role
--------------	--

## 9.872.1.3 SortRole

```
enum Digikam::ItemSortSettings::SortRole
```

## Enumerator

SortByImageSize	pixel number
SortByAspectRatio	width / height * 100000
SortByFaces	count of unconfirmed faces

## 9.872.2 Member Function Documentation

## 9.872.2.1 compare()

```
int Digikam::ItemSortSettings::compare (
    const ItemInfo & left,
    const ItemInfo & right ) const
```

Return -1 if left is less than right, 1 if left is greater than right, and 0 if left equals right comparing the current sort role's value. Adheres to set sort role and sort order.

## 9.872.2.2 compareCategories()

```
int Digikam::ItemSortSettings::compareCategories (
    const ItemInfo & left,
    const ItemInfo & right,
    const FaceTagsIface & leftFace,
    const FaceTagsIface & rightFace ) const
```

Return -1 if left is less than right, 0 if both fall in the same category, and 1 if left is greater than right. Adheres to set categorization mode and current category sort order. Face passed in to allow Categorization by Faces. Pass in an empty Face if not needed.

## 9.872.2.3 lessThan() [1/2]

```
bool Digikam::ItemSortSettings::lessThan (
    const ItemInfo & left,
    const ItemInfo & right ) const
```

Adheres to current sort role and sort order.

### 9.872.2.4 lessThan() [2/2]

```
bool Digikam::ItemSortSettings::lessThan (
    const QVariant & left,
    const QVariant & right ) const
```

Adheres to current sort role and sort order. Use for extraValue, if necessary.

### 9.872.2.5 lessThanByOrder()

```
template<typename T >
static bool Digikam::ItemSortSettings::lessThanByOrder (
    const T & a,
    const T & b,
    Qt::SortOrder sortOrder ) [inline], [static]
```

Returns  $a < b$  if sortOrder is Ascending, or  $b < a$  if order is descending.

### 9.872.2.6 watchFlags()

```
DatabaseFields::Set Digikam::ItemSortSettings::watchFlags ( ) const
```

Returns database fields a change in which would affect the current sorting.

## 9.873 Digikam::ItemTagPair Class Reference

### Public Member Functions

- [ItemTagPair](#) ()
  - This class provides a wrapper over the Database methods to access the properties of tag / image association.*
- **ItemTagPair** (const [ItemInfo](#) &info, int tagId)
- **ItemTagPair** (const [ItemTagPair](#) &other)
- **ItemTagPair** (qulonglong imageId, int tagId)
  - Access the properties of the given image - tag pair.*
- void [addProperty](#) (const QString &key, const QString &value)
  - Adds the given property.*
- QStringList **allValues** (const QStringList &keys) const
  - Returns value() concatenated for all given keys.*
- void **assignTag** ()
  - Assigns the tag to the image.*
- void **clearProperties** ()
  - Removes all properties.*
- bool **hasAnyProperty** (const QStringList &keys) const
  - Returns true if any of the properties is set.*
- bool **hasProperty** (const QString &key) const
  - Returns true if the property is set.*
- bool **hasValue** (const QString &key, const QString &value) const
  - Returns true of the given property and value is set.*
- qulonglong **imageId** () const

- bool **isAssigned** () const  
*Returns if the tag is assigned to the image.*
- bool **isNull** () const
- [ItemTagPair](#) & **operator=** (const [ItemTagPair](#) &other)
- QMap< QString, QString > **properties** () const  
*Returns a map of all key->value pairs.*
- QStringList **propertyKeys** () const  
*Returns all set property keys.*
- void **removeProperties** (const QString &key)  
*Remove all occurrences of the property.*
- void **removeProperty** (const QString &key, const QString &value)  
*Remove all occurrences of the property.*
- void **setProperty** (const QString &key, const QString &value)  
*Set the given property. Replaces all previous occurrences of this property.*
- int **tagId** () const
- void **unAssignTag** ()  
*Removes the tag from the image.*
- QString **value** (const QString &key) const  
*Returns the value of the given property, or a null string if not set.*
- QStringList **values** (const QString &key) const  
*Returns a list of values with the given property.*

### Static Public Member Functions

- static QList< [ItemTagPair](#) > **availablePairs** (const [ItemInfo](#) &info)
- static QList< [ItemTagPair](#) > **availablePairs** (qulonglong imageId)  
*Return all pairs for the given image for which entries exist.*

## 9.873.1 Constructor & Destructor Documentation

### 9.873.1.1 ItemTagPair()

```
Digikam::ItemTagPair::ItemTagPair ( )
```

It is meant to be a short-lived object, it does not listen to external database changes. Creates a null pair.

## 9.873.2 Member Function Documentation

### 9.873.2.1 addProperty()

```
void Digikam::ItemTagPair::addProperty (
    const QString & key,
    const QString & value )
```

Does not change any previous occurrences of this property, allowing multiple properties with the same key. (duplicates of same key *and* value are not added, though)



## Public Slots

- void **assignRating** (const QList< QModelIndex > &index, int rating)
- void **slotDockLocationChanged** (Qt::DockWidgetArea area)

## Public Slots inherited from [Digikam::ItemCategorizedView](#)

- void **hintAt** (const [ItemInfo](#) &info)
 

*Does something to gain attention for info, but not changing current selection.*
- void **openAlbum** (const QList< [Album](#) \* > &album)
- void **setCurrentInfo** (const [ItemInfo](#) &info)
 

*Set as current item the item identified by the imageinfo.*
- void **setCurrentUrl** (const [QUrl](#) &url)
 

*Set as current item the item identified by its file url.*
- void **setCurrentUrlWhenAvailable** (const [QUrl](#) &url)
 

*Set as current item when it becomes available, the item identified by its file url.*
- void **setCurrentWhenAvailable** (qulonglong imageId)
 

*Scroll the view to the given item when it becomes available.*
- void **setSelectedItemInfos** (const QList< [ItemInfo](#) > &infos)
 

*Set selected items.*
- void **setSelectedUrls** (const QList< [QUrl](#) > &urlList)
 

*Set selected items identified by their file urls.*
- void **setThumbnailSize** (int size)

## Public Slots inherited from [Digikam::ItemViewCategorized](#)

- void **copy** () override
- void **cut** () override
- void **hideIndexNotification** ()
- void **paste** () override
- void **showIndexNotification** (const QModelIndex &index, const [QString](#) &message)

## Public Slots inherited from [Digikam::DCategorizedView](#)

- void **reset** () override

## Public Member Functions

- [ItemThumbnailBar](#) ([QWidget](#) \*const parent=nullptr)
- QModelIndex **firstIndex** () const
- void **installOverlays** ()
- QModelIndex **lastIndex** () const
- QModelIndex **nextIndex** (const QModelIndex &index) const
- QModelIndex **previousIndex** (const QModelIndex &index) const
- void **setFlow** ([QListView::Flow](#) newFlow)
- void **setModelsFiltered** ([ItemModel](#) \*model, [ImageSortFilterModel](#) \*filterModel)
 

*This installs a duplicate filter model, if the [ItemModel](#) may contain duplicates.*
- void **setScrollBarPolicy** (Qt::ScrollBarPolicy policy)
 

*Sets the policy always for the one scroll bar which is relevant, depending on orientation.*

## Public Member Functions inherited from [Digikam::ItemCategorizedView](#)

- **ItemCategorizedView** (QWidget \*const parent=nullptr)
- void **addOverlay** ([ItemDelegateOverlay](#) \*overlay, [ItemDelegate](#) \*delegate=nullptr)
 

*Add and remove an overlay. It will as well be removed automatically when destroyed. Unless you pass a different delegate, the current delegate will be used.*
- void **addSelectionOverlay** ([ItemDelegate](#) \*delegate=nullptr)
- [Album](#) \* **albumAt** (const QPoint &pos) const
 

*If the model is categorized by an album, returns the album of the category that contains the position.*
- [ItemInfoList](#) **allItemInfos** () const
- QList< [QUrl](#) > **allUrls** () const
- [Album](#) \* **currentAlbum** () const
- [ItemInfo](#) **currentInfo** () const
- [QUrl](#) **currentUrl** () const
- [ItemDelegate](#) \* **delegate** () const
- [QItemSelectionModel](#) \* **getSelectionModel** () const
- [ItemAlbumFilterModel](#) \* **imageAlbumFilterModel** () const
- [ItemAlbumModel](#) \* **imageAlbumModel** () const
 

*Returns 0 if the [ItemModel](#) is not an [ItemAlbumModel](#).*
- [ItemFilterModel](#) \* **imageFilterModel** () const
 

*Returns any [ItemFilterMode](#) in chain. May not be [sourceModel\(\)](#)*
- [ItemModel](#) \* **imageModel** () const
- [ImageSortFilterModel](#) \* **imageSortFilterModel** () const
- [ItemThumbnailModel](#) \* **imageThumbnailModel** () const
 

*Returns 0 if the [ItemModel](#) is not an [ItemThumbnailModel](#).*
- [QModelIndex](#) **indexForInfo** (const [ItemInfo](#) &info) const
- [ItemInfo](#) **nextInfo** (const [ItemInfo](#) &info)
- [ItemInfo](#) **nextInOrder** (const [ItemInfo](#) &startingPoint, int nth)
 

*Returns the n-th info after the given one.*
- [ItemInfo](#) **previousInfo** (const [ItemInfo](#) &info)
- void **removeOverlay** ([ItemDelegateOverlay](#) \*overlay)
- [ItemInfoList](#) **selectedItemInfos** () const
- [ItemInfoList](#) **selectedItemInfosCurrentFirst** () const
- void **setModels** ([ItemModel](#) \*model, [ImageSortFilterModel](#) \*filterModel)
- virtual void **setThumbnailSize** (const [ThumbnailSize](#) &size)
- [ThumbnailSize](#) **thumbnailSize** () const
- void **toIndex** (const [QUrl](#) &url)
 

*Selects the index as current and scrolls to it.*

## Public Member Functions inherited from [Digikam::ItemViewCategorized](#)

- **ItemViewCategorized** (QWidget \*const parent=nullptr)
- void **awayFromSelection** ()
- [DItemDelegate](#) \* **delegate** () const
- void **invertSelection** ()
- bool **isToolTipEnabled** () const
- int **numberOfSelectedIndexes** () const
- void **scrollTo** (const [QModelIndex](#) &index, [ScrollHint](#) hint=EnsureVisible) override
- void **scrollToRelaxed** (const [QModelIndex](#) &index, [ScrollHint](#) hint=EnsureVisible)
 

*Like [scrollTo](#), but only scrolls if the index is not visible, regardless of hint.*
- void **setInitialSelectedItem** (bool enabled)
 

*Ensure a initial selected item.*

- void **setScrollCurrentToCenter** (bool enabled)  
*Scroll automatically the current index to center of the view.*
- void **setScrollStepGranularity** (int factor)  
*Determine a step size for scrolling: The larger this number, the smaller and more precise is the scrolling.*
- void **setSelectedIndexes** (const QList< QModelIndex > &indexes)
- void **setSpacing** (int spacing)  
*Sets the spacing.*
- void **setToolTipEnabled** (bool enabled)
- void **setUsePointingHandCursor** (bool useCursor)  
*Set if the PointingHand Cursor should be shown over the activation area.*
- void **toFirstIndex** ()  
*Selects the index as current and scrolls to it.*
- void **toIndex** (const QModelIndex &index)
- void **toLastIndex** ()
- void **toNextIndex** ()
- void **toPreviousIndex** ()

## Public Member Functions inherited from Digikam::DCategorizedView

- **DCategorizedView** (QWidget \*const parent=nullptr)
- virtual QModelIndexList **categorizedIndexesIn** (const QRect &rect) const  
*This method will return all indexes whose visual rect intersects rect.*
- virtual QModelIndex **categoryAt** (const QPoint &point) const  
*This method will return the first index of the category in the region of which point is found.*
- **DCategoryDrawer \* categoryDrawer** () const
- virtual QItemSelectionRange **categoryRange** (const QModelIndex &index) const  
*This method returns the range of indexes contained in the category in which index is sorted.*
- virtual QRect **categoryVisualRect** (const QModelIndex &index) const  
*This method will return the visual rect of the header of the category in which index is sorted.*
- QModelIndex **indexAt** (const QPoint &point) const override
- void **setCategoryDrawer** (DCategoryDrawer \*categoryDrawer)
- void **setDrawDraggedItems** (bool drawDraggedItems)  
*Switch on drawing of dragged items.*
- void **setGridSize** (const QSize &size)
- void **setModel** (QAbstractItemModel \*model) override
- QRect **visualRect** (const QModelIndex &index) const override

## Public Member Functions inherited from Digikam::DragDropViewImplementation

- virtual void **copy** ()
- virtual void **cut** ()
- virtual void **paste** ()

## Public Member Functions inherited from Digikam::GroupingViewImplementation

- **ItemInfoList getHiddenGroupedInfos** (const ItemInfoList &infos) const
- bool **needGroupResolving** (OperationType type, const ItemInfoList &infos) const
- **ItemInfoList resolveGrouping** (const ItemInfoList &infos) const

### Protected Member Functions

- bool **event** (QEvent \*) override
- bool **hasHiddenGroupedImages** (const [ItemInfo](#) &info) const override  
*must be implemented by parent view*
- void **slotSetupChanged** () override

### Protected Member Functions inherited from [Digikam::ItemCategorizedView](#)

- virtual void **activated** (const [ItemInfo](#) &info, Qt::KeyboardModifiers modifiers)  
*Reimplement these in a subclass.*
- void **currentChanged** (const QModelIndex &index, const QModelIndex &previous) override
- [AbstractItemDragDropHandler](#) \* **dragDropHandler** () const override  
*You need to implement these three methods Returns the drag drop handler.*
- QSortFilterProxyModel \* **filterModel** () const override
- [ItemInfo](#) **imageInfo** (const QModelIndex &index) const
- [ItemInfoList](#) **imageInfos** (const QList< QModelIndex > &indexes) const
- void **indexActivated** (const QModelIndex &index, Qt::KeyboardModifiers modifiers) override
- void **installDefaultModels** ()  
*install default [ItemAlbumModel](#) and filter model, ready for use*
- QModelIndex **nextIndexHint** (const QModelIndex &indexToAnchor, const QItemSelectionRange &removed) const override  
*Assuming the given indexes would be removed (hypothetically!), return the index to be selected instead, starting from anchor.*
- void **selectionChanged** (const QItemSelection &, const QItemSelection &) override
- void **setItemDelegate** ([ItemDelegate](#) \*delegate)
- void **showContextMenuOnIndex** (QContextMenuEvent \*event, const QModelIndex &index) override  
*Reimplement these in a subclass.*
- virtual void **showContextMenuOnInfo** (QContextMenuEvent \*event, const [ItemInfo](#) &info)
- void **updateGeometries** () override

### Protected Member Functions inherited from [Digikam::ItemViewCategorized](#)

- void **contextMenuEvent** (QContextMenuEvent \*event) override  
*reimplemented from parent class*
- bool **decodelsCutSelection** (const QMimeData \*mimeData)
- void **encodelsCutSelection** (QMimeData \*mime, bool isCutSelection)
- QModelIndex **indexForCategoryAt** (const QPoint &pos) const  
*Returns an index that is representative for the category at position pos.*
- void **keyPressEvent** (QKeyEvent \*event) override
- void **leaveEvent** (QEvent \*event) override
- QModelIndex **mapIndexForDragDrop** (const QModelIndex &index) const override  
*Note: pure virtual [dragDropHandler\(\)](#) still open from [DragDropViewImplementation](#).*
- void **mouseMoveEvent** (QMouseEvent \*event) override
- void **mousePressEvent** (QMouseEvent \*event) override
- void **mouseReleaseEvent** (QMouseEvent \*event) override
- QModelIndex **moveCursor** (CursorAction cursorAction, Qt::KeyboardModifiers modifiers) override
- QPixmap **pixmapForDrag** (const QList< QModelIndex > &indexes) const override  
*Creates a pixmap for dragging the given indexes.*
- void **reset** () override
- void **resizeEvent** (QResizeEvent \*e) override



- void **rowsAboutToBeRemoved** (const QModelIndex &parent, int start, int end) override
- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- void **rowsRemoved** (const QModelIndex &parent, int start, int end) override
- void **selectionChanged** (const QItemSelection &, const QItemSelection &) override
- void **setItemDelegate** (DItemDelegate \*delegate)
- void **setToolTip** (ItemViewToolTip \*tip)
- virtual void **showContextMenu** (QContextMenuEvent \*event)
- virtual bool **showToolTip** (const QModelIndex &index, QStyleOptionViewItem &option, QHelpEvent \*e=nullptr)
 

*Provides default behavior, can reimplement in a subclass.*
- void **updateDelegateSizes** ()
- void **userInteraction** ()
- bool **viewportEvent** (QEvent \*event) override
- void **wheelEvent** (QWheelEvent \*event) override

### Protected Member Functions inherited from Digikam::DCategorizedView

- void **dragLeaveEvent** (QDragLeaveEvent \*event) override
- void **dragMoveEvent** (QDragMoveEvent \*event) override
- void **dropEvent** (QDropEvent \*event) override
- void **leaveEvent** (QEvent \*event) override
- void **mouseMoveEvent** (QMouseEvent \*event) override
- void **mousePressEvent** (QMouseEvent \*event) override
- void **mouseReleaseEvent** (QMouseEvent \*event) override
- QModelIndex **moveCursor** (CursorAction cursorAction, Qt::KeyboardModifiers modifiers) override
- void **paintEvent** (QPaintEvent \*event) override
- void **resizeEvent** (QResizeEvent \*event) override
- void **setSelection** (const QRect &rect, QItemSelectionModel::SelectionFlags flags) override
- void **startDrag** (Qt::DropActions supportedActions) override

### Protected Member Functions inherited from Digikam::DragDropViewImplementation

- virtual QAbstractItemView \* **asView** ()=0
 

*This one is implemented by DECLARE\_VIEW\_DRAG\_DROP\_METHODS.*
- bool **decodelsCutSelection** (const QMimeData \*mimeData)
- void **dragEnterEvent** (QDragEnterEvent \*event)
 

*Implements the relevant QAbstractItemView methods for drag and drop.*
- void **dragMoveEvent** (QDragMoveEvent \*e)
- void **dropEvent** (QDropEvent \*e)
- void **encodelsCutSelection** (QMimeData \*mime, bool isCutSelection)
- void **startDrag** (Qt::DropActions supportedActions)

### Additional Inherited Members

### Signals inherited from Digikam::ItemCategorizedView

- void **currentChanged** (const ItemInfo &info)
- void **deselected** (const QList< ItemInfo > &nowDeselectedInfos)
 

*Emitted when items are deselected. There may be other selected infos left. This signal is not emitted when the model is reset; then only selectionCleared is emitted.*
- void **imageActivated** (const ItemInfo &info)
 

*Emitted when the given image is activated. Info is never null.*
- void **modelChanged** ()
 

*Emitted when a new model is set.*
- void **selected** (const QList< ItemInfo > &newSelectedInfos)
 

*Emitted when new items are selected. The parameter includes only the newly selected infos, there may be other already selected infos.*

## Signals inherited from [Digikam::ItemViewCategorized](#)

- void [clicked](#) (const QMouseEvent \*e, const QModelIndex &index)  
*For overlays: Like the respective parent class signals, but with additional info.*
- void [entered](#) (const QMouseEvent \*e, const QModelIndex &index)
- void [keyPressed](#) (QKeyEvent \*e)  
*Remember you may want to check if the event is accepted or ignored.*
- void [selectionChanged](#) ()  
*Emitted when any selection change occurs.*
- void [selectionCleared](#) ()  
*Emitted when the selection is completely cleared.*
- void [viewportClicked](#) (const QMouseEvent \*e)  
*While [clicked\(\)](#) is emitted with a valid index, this corresponds to clicking on empty space.*
- void [zoomInStep](#) ()
- void [zoomOutStep](#) ()

## Protected Slots inherited from [Digikam::ItemCategorizedView](#)

- void [slotCurrentUrlTimer](#) ()
- void [slotItemInfosAdded](#) ()

## Protected Slots inherited from [Digikam::ItemViewCategorized](#)

- void [layoutAboutToBeChanged](#) ()
- void [layoutWasChanged](#) ()
- void [slotActivated](#) (const QModelIndex &index)
- void [slotClicked](#) (const QModelIndex &index)
- void [slotEntered](#) (const QModelIndex &index)
- virtual void [slotThemeChanged](#) ()

## Protected Slots inherited from [Digikam::DCategorizedView](#)

- void [currentChanged](#) (const QModelIndex &current, const QModelIndex &previous) override
- void [rowsInserted](#) (const QModelIndex &parent, int start, int end) override
- virtual void [rowsInsertedArtificial](#) (const QModelIndex &parent, int start, int end)
- virtual void [slotLayoutChanged](#) ()
- void [updateGeometries](#) () override

## 9.874.1 Member Function Documentation

### 9.874.1.1 [hasHiddenGroupedImages\(\)](#)

```
bool Digikam::ItemThumbnailBar::hasHiddenGroupedImages (
    const ItemInfo & ) const [override], [protected], [virtual]
```

Reimplemented from [Digikam::GroupingViewImplementation](#).

### 9.874.1.2 setModelsFiltered()

```
void Digikam::ItemThumbnailBar::setModelsFiltered (
    ItemModel * model,
    ImageSortFilterModel * filterModel )
```

Otherwise, just use setModels().

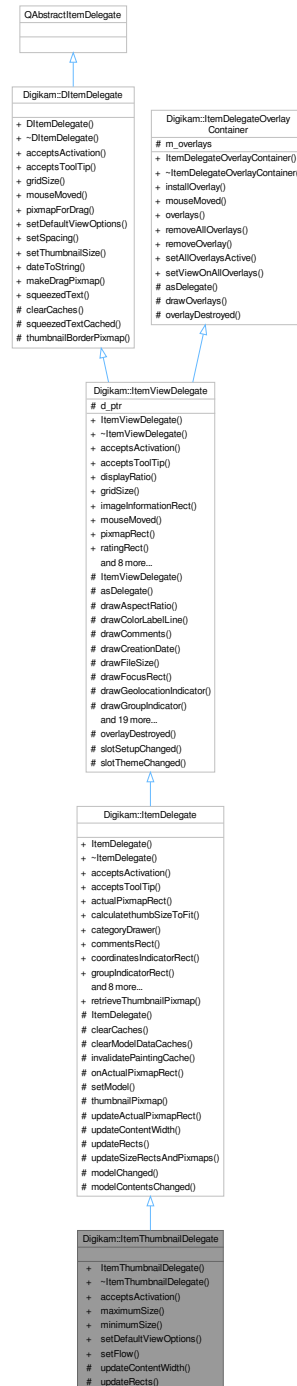
### 9.874.1.3 slotSetupChanged()

```
void Digikam::ItemThumbnailBar::slotSetupChanged ( ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ItemViewCategorized](#).

## 9.875 Digikam::ItemThumbnailDelegate Class Reference

Inheritance diagram for Digikam::ItemThumbnailDelegate:



### Public Member Functions

- **ItemThumbnailDelegate** (**ItemCategorizedView** \*const parent)
- bool **acceptsActivation** (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*activationRect) const override

- int **maximumSize** () const  
*Returns the minimum or maximum viewport size in the limiting dimension, width or height, depending on current flow.*
- int **minimumSize** () const
- void **setDefaultViewOptions** (const QStyleOptionViewItem &option) override  
*Style option with standard values to use for cached rendering.*
- void **setFlow** (QListView::Flow flow)

## Public Member Functions inherited from Digikam::ItemDelegate

- **ItemDelegate** (QWidget \*const parent)
- bool **acceptsToolTip** (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*tooltipRect=nullptr) const override  
*These methods take four parameters: The position on viewport, the rect on viewport, the index, and optionally a parameter into which, if the return value is true, a rectangle can be written for which the return value will be true as well.*
- QRect **actualPixmapRect** (const QModelIndex &index) const
- int **calculatethumbSizeToFit** (int ws)
- **ItemCategoryDrawer** \* **categoryDrawer** () const
- QRect **commentsRect** () const
- QRect **coordinatesIndicatorRect** () const
- QRect **groupIndicatorRect** () const
- QRect **imageInformationRect** () const override  
*Returns the area where the image information is drawn, or null if empty / not supported.*
- void **paint** (QPainter \*painter, const QStyleOptionViewItem &option, const QModelIndex &index) const override
- QPixmap **pixmapForDrag** (const QStyleOptionViewItem &option, const QList< QModelIndex > &indexes) const override
- QRect **pixmapRect** () const override  
*Returns the area where the pixmap is drawn, or null if not supported.*
- void **setSpacing** (int spacing) override
- void **setView** (ItemCategorizedView \*view)
- QRect **tagsRect** () const

## Public Member Functions inherited from Digikam::ItemViewDelegate

- **ItemViewDelegate** (QWidget \*const parent)
- bool **acceptsActivation** (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*activationRect=nullptr) const override
- bool **acceptsToolTip** (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*tooltipRect=nullptr) const override  
*These methods take four parameters: The position on viewport, the rect on viewport, the index, and optionally a parameter into which, if the return value is true, a rectangle can be written for which the return value will be true as well.*
- double **displayRatio** () const
- QSize **gridSize** () const override  
*Returns the gridsize to be set by the view.*
- void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index) override
- virtual QRect **ratingRect** () const  
*Returns the rectangle where the rating is drawn, or a null rectangle if not supported.*
- QRect **rect** () const
- void **setDefaultViewOptions** (const QStyleOptionViewItem &option) override  
*Style option with standard values to use for cached rendering.*

- void [setRatingEdited](#) (const QModelIndex &index)  
*Can be used to temporarily disable drawing of the rating.*
- void [setSpacing](#) (int spacing) override
- void [setThumbnailSize](#) (const [ThumbnailSize](#) &thumbSize) override  
*You must set these options from the view.*
- QSize **sizeHint** (const QStyleOptionViewItem &option, const QModelIndex &index) const override
- int **spacing** () const
- [ThumbnailSize](#) **thumbnailSize** () const

## Public Member Functions inherited from [Digikam::DItemDelegate](#)

- **DItemDelegate** (QObject \*const parent=nullptr)

## Public Member Functions inherited from [Digikam::ItemDelegateOverlayContainer](#)

- [ItemDelegateOverlayContainer](#) ()=default  
*This is a sample implementation for delegate management methods, to be inherited by a delegate.*
- void **installOverlay** ([ItemDelegateOverlay](#) \*overlay)
- void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)
- QList< [ItemDelegateOverlay](#) \* > **overlays** () const
- void **removeAllOverlays** ()
- void **removeOverlay** ([ItemDelegateOverlay](#) \*overlay)
- void **setAllOverlaysActive** (bool active)
- void **setViewOnAllOverlays** (QAbstractItemView \*view)

## Protected Member Functions

- void [updateContentWidth](#) () override  
*Reimplement this to set contentWidth.*
- void [updateRects](#) () override  
*In a subclass, you need to implement this method to set up the rects for drawing.*

## Protected Member Functions inherited from [Digikam::ItemDelegate](#)

- **ItemDelegate** (ItemDelegate::ItemDelegatePrivate &dd, QWidget \*const parent)
- void [clearCaches](#) () override
- virtual void **clearModelDataCaches** ()  
*Reimplement to clear caches based on model indexes (hash on row number etc.) Change signals are listened to this is called whenever such properties become invalid.*
- void [invalidatePaintingCache](#) () override
- bool **onActualPixmapRect** (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*actualRect) const
- void **setModel** (QAbstractItemModel \*model)
- virtual QPixmap **thumbnailPixmap** (const QModelIndex &index) const
- void **updateActualPixmapRect** (const QModelIndex &index, const QRect &rect)
- void [updateSizeRectsAndPixmaps](#) () override

## Protected Member Functions inherited from Digikam::ItemViewDelegate

- **ItemViewDelegate** (ItemViewDelegatePrivate &dd, QWidget \*const parent)
- QAbstractItemDelegate \* **asDelegate** () override
 

*Returns the delegate, typically, the derived class.*
- void **drawAspectRatio** (QPainter \*p, const QRect &dimsRect, const QSize &dims) const
- void **drawColorLabelLine** (QPainter \*p, const QRect &pixRect, int colorId) const
- void **drawComments** (QPainter \*p, const QRect &commentsRect, const QString &comments) const
- void **drawCreationDate** (QPainter \*p, const QRect &dateRect, const QDateTime &date) const
- void **drawFileSize** (QPainter \*p, const QRect &r, qlonglong bytes) const
- void **drawFocusRect** (QPainter \*p, const QStyleOptionViewItem &option, bool isSelected) const
- void **drawGeolocationIndicator** (QPainter \*p, const QRect &r) const
- void **drawGroupIndicator** (QPainter \*p, const QRect &r, int numberOfGroupedImages, bool open) const
- void **drawImageFormat** (QPainter \*p, const QRect &r, const QString &f, bool drawTop) const
- void **drawImageSize** (QPainter \*p, const QRect &dimsRect, const QSize &dims) const
- void **drawModificationDate** (QPainter \*p, const QRect &dateRect, const QDateTime &date) const
- void **drawMouseOverRect** (QPainter \*p, const QStyleOptionViewItem &option) const
- void **drawName** (QPainter \*p, const QRect &nameRect, const QString &name) const
- void **drawPanelSidelcon** (QPainter \*p, bool left, bool right) const
- void **drawPickLabelIcon** (QPainter \*p, const QRect &r, int pickLabel) const
- void **drawRating** (QPainter \*p, const QModelIndex &index, const QRect &ratingRect, int rating, bool isSelected) const
- void **drawSpecialInfo** (QPainter \*p, const QRect &r, const QString &text) const
- void **drawTags** (QPainter \*p, const QRect &r, const QString &tagsString, bool isSelected) const
- QRect **drawThumbnail** (QPainter \*p, const QRect &thumbRect, const QPixmap &background, const QPixmap &thumbnail, bool isGrouped) const
 

*Use the tool methods for painting in subclasses.*
- void **drawTitle** (QPainter \*p, const QRect &titleRect, const QString &title) const
- void **prepareBackground** ()
- void **prepareFonts** ()
- void **prepareMetrics** (int maxWidth)
- void **prepareRatingPixmap** (bool composeOverBackground=true)
- QPixmap **ratingPixmap** (int rating, bool selected) const
 

*Returns the relevant pixmap from the cached rating pixmaps.*

## Protected Member Functions inherited from Digikam::DItemDelegate

- QString **squeezedTextCached** (QPainter \*const p, int width, const QString &text) const
- QPixmap **thumbnailBorderPixmap** (const QSize &pixSize, bool isGrouped=false) const

## Protected Member Functions inherited from Digikam::ItemDelegateOverlayContainer

- virtual void **drawOverlays** (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index) const
- virtual void **overlayDestroyed** (QObject \*o)
 

*Declare as slot in the derived class calling this method.*

## Additional Inherited Members

## Signals inherited from Digikam::ItemViewDelegate

- void **hideNotification** ()
- void **requestNotification** (const QModelIndex &index, const QString &message)

## Signals inherited from [Digikam::DItemDelegate](#)

- void **gridSizeChanged** (const QSize &newSize)
- void **visualChange** ()

## Static Public Member Functions inherited from [Digikam::ItemDelegate](#)

- static QPixmap **retrieveThumbnailPixmap** (const QModelIndex &index, int thumbnailSize)  
*Retrieve the thumbnail pixmap in given size for the [ItemModel::ThumbnailRole](#) for the given index from the given index, which must adhere to [ItemThumbnailModel](#) semantics.*

## Static Public Member Functions inherited from [Digikam::DItemDelegate](#)

- static QString **dateToString** (const QDateTime &datetime)
- static QPixmap **makeDragPixmap** (const QStyleOptionViewItem &option, const QList< QModelIndex > &indexes, double displayRatio, const QPixmap &suggestedPixmap=QPixmap())
- static QString **squeezedText** (const QFontMetrics &fm, int width, const QString &text)

## Protected Slots inherited from [Digikam::ItemDelegate](#)

- void **modelChanged** ()
- void **modelContentsChanged** ()

## Protected Slots inherited from [Digikam::ItemViewDelegate](#)

- void **overlayDestroyed** (QObject \*o) override
- void **slotSetupChanged** ()
- void **slotThemeChanged** ()

## Protected Attributes inherited from [Digikam::ItemViewDelegate](#)

- ItemViewDelegatePrivate \*const **d\_ptr** = nullptr

## Protected Attributes inherited from [Digikam::ItemDelegateOverlayContainer](#)

- QList< [ItemDelegateOverlay](#) \* > **m\_overlays**

## 9.875.1 Member Function Documentation

### 9.875.1.1 acceptsActivation()

```
bool Digikam::ItemThumbnailDelegate::acceptsActivation (
    const QPoint & pos,
    const QRect & visualRect,
    const QModelIndex & index,
    QRect * activationRect ) const [override], [virtual]
```

Reimplemented from [Digikam::ItemDelegate](#).



### 9.875.1.2 setDefaultViewOptions()

```
void Digikam::ItemThumbnailDelegate::setDefaultViewOptions (
    const QStyleOptionViewItem & option ) [override], [virtual]
```

option.rect shall be the viewport rectangle. Call on resize, font change.

Reimplemented from [Digikam::ItemDelegate](#).

### 9.875.1.3 updateContentWidth()

```
void Digikam::ItemThumbnailDelegate::updateContentWidth ( ) [override], [protected], [virtual]
```

This is the maximum width of all content rectangles, typically excluding margins on both sides.

Reimplemented from [Digikam::ItemDelegate](#).

### 9.875.1.4 updateRects()

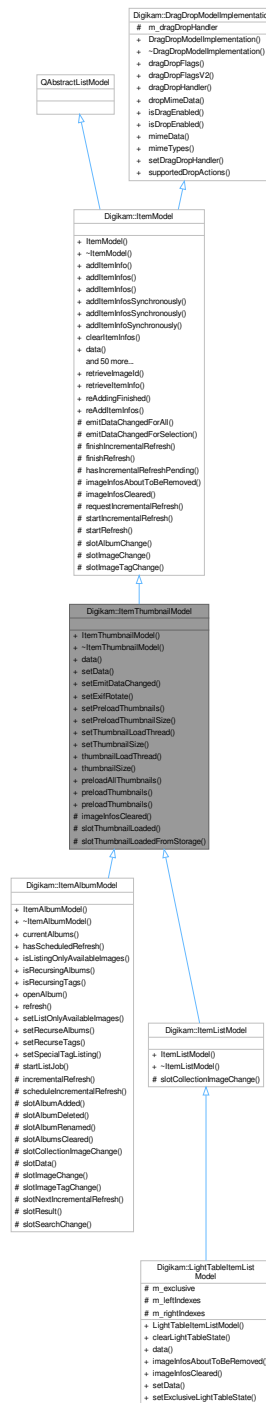
```
void Digikam::ItemThumbnailDelegate::updateRects ( ) [override], [protected], [virtual]
```

The paint() method operates depending on these rects.

Implements [Digikam::ItemDelegate](#).

## 9.876 Digikam::ItemThumbnailModel Class Reference

Inheritance diagram for Digikam::ItemThumbnailModel:



### Public Slots

- void **preloadAllThumbnails** ()
- void **preloadThumbnails** (const QList< [ItemInfo](#) > &)  
*Preload thumbnail for the given infos resp.*
- void **preloadThumbnails** (const QList< [QModelIndex](#) > &)

## Public Slots inherited from Digikam::ItemModel

- void **reAddingFinished** ()
- void **reAddItemInfos** (const QList< [ItemInfo](#) > &infos, const QList< QVariant > &extraValues)

## Signals

- void **thumbnailAvailable** (const QModelIndex &index, int requestedSize)
- void **thumbnailFailed** (const QModelIndex &index, int requestedSize)

## Signals inherited from Digikam::ItemModel

- void **allRefreshingFinished** ()  
*Signals that the model has finished currently with all scheduled refreshing, full or incremental, and all preprocessing.*
- void **imageChange** (const [ImageChangeset](#) &, const QItemSelection &)  
*If an [ImageChangeset](#) affected indexes of this model with changes as set in [watchFlags\(\)](#), this signal contains the changeset and the affected indexes.*
- void **imageInfosAboutToBeAdded** (const QList< [ItemInfo](#) > &infos)  
*Informs that ItemInfos will be added to the model.*
- void **imageInfosAboutToBeRemoved** (const QList< [ItemInfo](#) > &infos)  
*Informs that ItemInfos will be removed from the model.*
- void **imageInfosAdded** (const QList< [ItemInfo](#) > &infos)  
*Informs that ItemInfos have been added to the model.*
- void **imageInfosRemoved** (const QList< [ItemInfo](#) > &infos)  
*Informs that ItemInfos have been removed from the model.*
- void **imageTagChange** (const [ImageTagChangeset](#) &, const QItemSelection &)  
*If an [ImageTagChangeset](#) affected indexes of this model, this signal contains the changeset and the affected indexes.*
- void **preprocess** (const QList< [ItemInfo](#) > &infos, const QList< QVariant > &)  
*Connect to this signal only if you are the current preprocessor.*
- void **processAdded** (const QList< [ItemInfo](#) > &infos, const QList< QVariant > &)
- void **readyForIncrementalRefresh** ()  
*Signals that the model is right now ready to start an incremental refresh.*

## Public Member Functions

- [ItemThumbnailModel](#) (QWidget \*const parent)  
*An [ItemModel](#) that supports thumbnail loading.*
- QVariant **data** (const QModelIndex &index, int role=Qt::DisplayRole) const override  
*Handles the ThumbnailRole.*
- bool **setData** (const QModelIndex &index, const QVariant &value, int role=Qt::DisplayRole) override  
*You can override the current thumbnail size by giving an integer value for ThumbnailRole.*
- void **setEmitDataChanged** (bool emitSignal)  
*Enable emitting [dataChanged\(\)](#) when a thumbnail becomes available.*
- void **setExifRotate** (bool rotate)
- void **setPreloadThumbnails** (bool preload)  
*Enable preloading of thumbnails: If preloading is enabled, for every entry in the model a thumbnail generation is started.*
- void **setPreloadThumbnailSize** (const [ThumbnailSize](#) &thumbSize)  
*If you want to fix a size for preloading, do it here.*
- void **setThumbnailLoadThread** ([ThumbnailLoadThread](#) \*const thread)  
*Enable thumbnail loading and set the thread that shall be used.*
- void **setThumbnailSize** (const [ThumbnailSize](#) &thumbSize)  
*Set the thumbnail size to use.*
- [ThumbnailLoadThread](#) \* **thumbnailLoadThread** () const
- [ThumbnailSize](#) **thumbnailSize** () const

## Public Member Functions inherited from [Digikam::ItemModel](#)

- **ItemModel** (QObject \*const parent=nullptr)
- void **addItemInfo** (const [ItemInfo](#) &info)
  - Main entry point for subclasses adding image infos to the model.*
- void **addItemInfos** (const QList< [ItemInfo](#) > &infos)
- void **addItemInfos** (const QList< [ItemInfo](#) > &infos, const QList< QVariant > &extraValues)
- void **addItemInfosSynchronously** (const QList< [ItemInfo](#) > &infos)
- void **addItemInfosSynchronously** (const QList< [ItemInfo](#) > &infos, const QList< QVariant > &extraValues)
- void **addItemInfoSynchronously** (const [ItemInfo](#) &info)
  - addItemInfo() is asynchronous if a preprocessor is set.*
- void **clearItemInfos** ()
  - Clears image infos and resets model.*
- QVariant **data** (const QModelIndex &index, int role=Qt::DisplayRole) const override
- void **ensureHasGroupedImages** (const [ItemInfo](#) &groupLeader)
  - Ensure that all images grouped on the given leader are contained in the model.*
- void **ensureHasItemInfo** (const [ItemInfo](#) &info)
  - Add the given entries.*
- void **ensureHasItemInfos** (const QList< [ItemInfo](#) > &infos)
- void **ensureHasItemInfos** (const QList< [ItemInfo](#) > &infos, const QList< QVariant > &extraValues)
- Qt::ItemFlags **flags** (const QModelIndex &index) const override
- bool **hasImage** (const [ItemInfo](#) &info) const
- bool **hasImage** (const [ItemInfo](#) &info, const QVariant &extraValue) const
- bool **hasImage** (qulonglong id) const
- bool **hasImage** (qulonglong id, const QVariant &extraValue) const
- QVariant **headerData** (int section, Qt::Orientation orientation, int role=Qt::DisplayRole) const override
- qulonglong **imageId** (const QModelIndex &index) const
- qulonglong **imageId** (int row) const
- QList< qulonglong > **imageIds** () const
- QList< qulonglong > **imageIds** (const QList< QModelIndex > &indexes) const
- [ItemInfo](#) **imageInfo** (const QModelIndex &index) const
  - Returns the ItemInfo object, reference or image id from the underlying data pointed to by the index.*
- [ItemInfo](#) **imageInfo** (const QString &filePath) const
- [ItemInfo](#) **imageInfo** (int row) const
  - Returns the ItemInfo object, reference or image id from the underlying data of the given row (parent is the invalid QModelIndex, column is 0).*
- [ItemInfo](#) & **imageInfoRef** (const QModelIndex &index) const
- [ItemInfo](#) & **imageInfoRef** (int row) const
- QList< [ItemInfo](#) > **imageInfos** () const
- QList< [ItemInfo](#) > **imageInfos** (const QList< QModelIndex > &indexes) const
- QList< [ItemInfo](#) > **imageInfos** (const QString &filePath) const
- QModelIndex **index** (int row, int column=0, const QModelIndex &parent=QModelIndex()) const override
- QList< QModelIndex > **indexesForImageId** (qulonglong id) const
- QList< QModelIndex > **indexesForItemInfo** (const [ItemInfo](#) &info) const
- QList< QModelIndex > **indexesForPath** (const QString &filePath) const
- QModelIndex **indexForImageId** (qulonglong id) const
- QModelIndex **indexForImageId** (qulonglong id, const QVariant &extraValue) const
- QModelIndex **indexForItemInfo** (const [ItemInfo](#) &info) const
  - Return the index for the given ItemInfo or id, if contained in this model.*
- QModelIndex **indexForItemInfo** (const [ItemInfo](#) &info, const QVariant &extraValue) const
- QModelIndex **indexForPath** (const QString &filePath) const
  - Returns the index or ItemInfo object from the underlying data for the given file path.*
- bool **isEmpty** () const

- bool **isRefreshing** () const  
*Returns true if this model is currently refreshing.*
- int **itemCount** () const
- bool **keepsFilePathCache** () const
- int **numberOfIndexesForImageId** (qulonglong id) const
- int **numberOfIndexesForItemInfo** (const [ItemInfo](#) &info) const
- void **removeIndex** (const QModelIndex &indexes)  
*Directly remove the given indexes or infos from the model.*
- void **removeIndexes** (const QList< QModelIndex > &indexes)
- void **removeItemInfo** (const [ItemInfo](#) &info)
- void **removeItemInfos** (const QList< [ItemInfo](#) > &infos)
- void **removeItemInfos** (const QList< [ItemInfo](#) > &infos, const QList< QVariant > &extraValues)
- int **rowCount** (const QModelIndex &parent=QModelIndex()) const override
- void **setItemInfos** (const QList< [ItemInfo](#) > &infos)  
*Clears and adds the infos.*
- void **setKeepsFilePathCache** (bool keepCache)  
*If a cache is kept, lookup by file path is fast, without a cache it is O(n).*
- DECLARE\_MODEL\_DRAG\_DROP\_METHODS void **setPreprocessor** (QObject \*const processor)  
*Install an object as a preprocessor for ItemInfos added to this model.*
- void **setSendRemovalSignals** (bool send)  
*Enable sending of imageInfosAboutToBeRemoved and imageInfosRemoved signals.*
- void **setWatchFlags** (const [DatabaseFields::Set](#) &set)  
*Set a set of database fields to watch.*
- QList< [ItemInfo](#) > **uniqueItemInfos** () const
- void **unsetPreprocessor** (QObject \*const processor)

## Public Member Functions inherited from [Digikam::DragDropModelImplementation](#)

- [DragDropModelImplementation](#) ()=default  
*A class providing a sample implementation for a QAbstractItemModel redirecting drag-and-drop support to a handler.*
- virtual Qt::ItemFlags **dragDropFlags** (const QModelIndex &index) const  
*Call from your flags() method, adding the relevant drag drop flags.*
- Qt::ItemFlags **dragDropFlagsV2** (const QModelIndex &index) const  
*This is an alternative approach to dragDropFlags().*
- [AbstractItemDragDropHandler](#) \* **dragDropHandler** () const
- bool **dropMimeData** (const QMimeData \*, Qt::DropAction, int, int, const QModelIndex &)
- virtual bool **isDragEnabled** (const QModelIndex &index) const
- virtual bool **isDropEnabled** (const QModelIndex &index) const
- QMimeData \* **mimeData** (const QModelIndexList &indexes) const
- QStringList **mimeTypes** () const
- void **setDragDropHandler** ([AbstractItemDragDropHandler](#) \*handler)  
*Set a drag drop handler.*
- Qt::DropActions **supportedDropActions** () const  
*Implements the relevant QAbstractItemModel methods for drag and drop.*

## Protected Slots

- void **slotThumbnailLoaded** (const [LoadingDescription](#) &loadingDescription, const QPixmap &thumb)
- void **slotThumbnailLoadedFromStorage** (const [LoadingDescription](#) &loadingDescription, const QPixmap &thumb)

## Protected Slots inherited from [Digikam::ItemModel](#)

- virtual void **slotAlbumChange** (const [AlbumChangeset](#) &changeset)
- virtual void **slotImageChange** (const [ImageChangeset](#) &changeset)
- virtual void **slotImageTagChange** (const [ImageTagChangeset](#) &changeset)

## Protected Member Functions

- void [imageInfosCleared](#) () override  
*Called when the internal storage is cleared.*

## Protected Member Functions inherited from [Digikam::ItemModel](#)

- void **emitDataChangedForAll** ()
- void **emitDataChangedForSelection** (const [QItemSelection](#) &selection)
- void **finishIncrementalRefresh** ()
- void **finishRefresh** ()
- bool **hasIncrementalRefreshPending** () const
- virtual void **imageInfosAboutToBeRemoved** (int, int)  
*Called before rowsAboutToBeRemoved.*
- void [requestIncrementalRefresh](#) ()  
*As soon as the model is ready to start an incremental refresh, the signal [readyForIncrementalRefresh\(\)](#) will be emitted.*
- void [startIncrementalRefresh](#) ()  
*Starts an incremental refresh operation.*
- void [startRefresh](#) ()  
*Subclasses that add ItemInfos in batches shall call [startRefresh\(\)](#) when they start sending batches and [finishRefresh\(\)](#) when they have finished.*

## Additional Inherited Members

## Public Types inherited from [Digikam::ItemModel](#)

- enum [ItemModelRoles](#) {  
[ItemModelPointerRole](#) = [Qt::UserRole](#) , [ItemModelInternalId](#) = [Qt::UserRole](#) + 1 , [ThumbnailRole](#) = [Qt::UserRole](#) + 2 , [CreationDateRole](#) = [Qt::UserRole](#) + 3 ,  
[ExtraDataRole](#) = [Qt::UserRole](#) + 5 , [ExtraDataDuplicateCount](#) = [Qt::UserRole](#) + 6 , [LTLeftPanelRole](#) = [Qt::UserRole](#) + 50 , [LTRightPanelRole](#) = [Qt::UserRole](#) + 51 ,  
[SubclassRoles](#) = [Qt::UserRole](#) + 100 , [FilterModelRoles](#) = [Qt::UserRole](#) + 500 }

## Static Public Member Functions inherited from [Digikam::ItemModel](#)

- static qulonglong **retrieveImageId** (const [QModelIndex](#) &index)
- static [ItemInfo](#) **retrieveItemInfo** (const [QModelIndex](#) &index)  
*Retrieves the imageInfo object from the data() method of the given index.*

## Protected Attributes inherited from [Digikam::DragDropModelImplementation](#)

- [AbstractItemDragDropHandler](#) \* **m\_dragDropHandler** = nullptr

## 9.876.1 Constructor & Destructor Documentation

### 9.876.1.1 ItemThumbnailModel()

```
Digikam::ItemThumbnailModel::ItemThumbnailModel (
    QWidget *const parent ) [explicit]
```

You need to set a [ThumbnailLoadThread](#) to enable thumbnail loading. Adjust the thumbnail size to your needs. Note that `setKeepsFilePathCache` is enabled per default.

## 9.876.2 Member Function Documentation

### 9.876.2.1 data()

```
QVariant Digikam::ItemThumbnailModel::data (
    const QModelIndex & index,
    int role = Qt::DisplayRole ) const [override]
```

If the pixmap is available, returns it in the QVariant. If it still needs to be loaded, returns a null QVariant and emits `thumbnailAvailable()` as soon as it is available.

### 9.876.2.2 imageInfosCleared()

```
void Digikam::ItemThumbnailModel::imageInfosCleared ( ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ItemModel](#).

### 9.876.2.3 preloadThumbnails

```
void Digikam::ItemThumbnailModel::preloadThumbnails (
    const QList< ItemInfo > & infos ) [slot]
```

indexes. Note: Use `setPreloadThumbnails` to automatically preload all entries in the model. Note: This only ensures thumbnail generation. It is not guaranteed that pixmaps are stored in the cache. For thumbnails that are expect to be drawn immediately, include them in `prepareThumbnails()`. Note: Stops preloading of previously added thumbnails.

### 9.876.2.4 setData()

```
bool Digikam::ItemThumbnailModel::setData (
    const QModelIndex & index,
    const QVariant & value,
    int role = Qt::DisplayRole ) [override]
```

Set a null QVariant to use the thumbnail size set by `setThumbnailSize()` again. The index given here is ignored for this purpose.

### 9.876.2.5 setEmitDataChanged()

```
void Digikam::ItemThumbnailModel::setEmitDataChanged (
    bool emitSignal )
```

The thumbnailAvailable() signal will be emitted in any case. Default is true.

### 9.876.2.6 setPreloadThumbnails()

```
void Digikam::ItemThumbnailModel::setPreloadThumbnails (
    bool preload )
```

Default: false.

### 9.876.2.7 setThumbnailLoadThread()

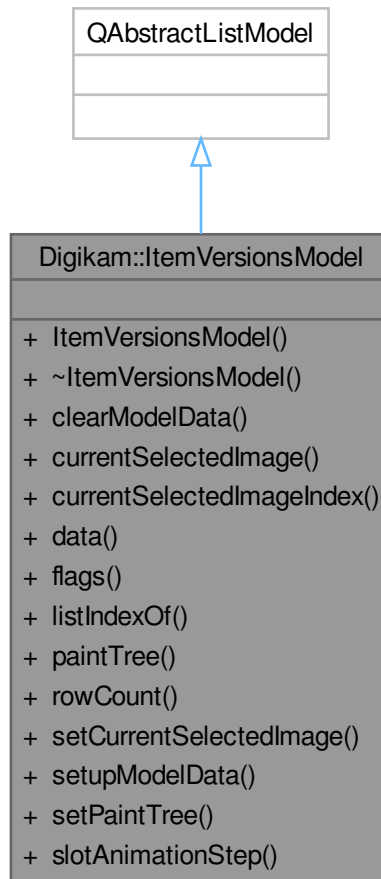
```
void Digikam::ItemThumbnailModel::setThumbnailLoadThread (
    ThumbnailLoadThread *const thread )
```

The thumbnail size of this thread will be adjusted.



## 9.877 Digikam::ItemVersionsModel Class Reference

Inheritance diagram for Digikam::ItemVersionsModel:



### Public Slots

- void **setPaintTree** (bool paint)
- void **slotAnimationStep** ()

### Public Member Functions

- **ItemVersionsModel** (QObject \*const parent=nullptr)
- void **clearModelData** ()
- QString **currentSelectedImage** () const
- QModelIndex **currentSelectedImageIndex** () const
- QVariant **data** (const QModelIndex &index, int role=Qt::DisplayRole) const override
- Qt::ItemFlags **flags** (const QModelIndex &index) const override
- int **listIndexOf** (const QString &item) const
- bool **paintTree** () const
- int **rowCount** (const QModelIndex &parent=QModelIndex()) const override
- void **setCurrentSelectedImage** (const QString &path)
- void **setupModelData** (QList< QPair< QString, int > > &data)



- void **hideIndexNotification** ()
- void **paste** () override
- void **showIndexNotification** (const QModelIndex &index, const QString &message)

## Public Slots inherited from [Digikam::DCategorizedView](#)

- void **reset** () override

## Signals

- void **clicked** (const QMouseEvent \*e, const QModelIndex &index)  
*For overlays: Like the respective parent class signals, but with additional info.*
- void **entered** (const QMouseEvent \*e, const QModelIndex &index)
- void **keyPressed** (QKeyEvent \*e)  
*Remember you may want to check if the event is accepted or ignored.*
- void **selectionChanged** ()  
*Emitted when any selection change occurs.*
- void **selectionCleared** ()  
*Emitted when the selection is completely cleared.*
- void **viewportClicked** (const QMouseEvent \*e)  
*While [clicked\(\)](#) is emitted with a valid index, this corresponds to clicking on empty space.*
- void **zoomInStep** ()
- void **zoomOutStep** ()

## Public Member Functions

- **ItemViewCategorized** (QWidget \*const parent=nullptr)
- void **awayFromSelection** ()
- [DItemDelegate](#) \* **delegate** () const
- virtual QSortFilterProxyModel \* **filterModel** () const =0
- void **invertSelection** ()
- bool **isToolTipEnabled** () const
- int **numberOfSelectedIndexes** () const
- void **scrollTo** (const QModelIndex &index, ScrollHint hint=EnsureVisible) override
- void **scrollToRelaxed** (const QModelIndex &index, ScrollHint hint=EnsureVisible)  
*Like [scrollTo](#), but only scrolls if the index is not visible, regardless of hint.*
- void **setInitialSelectedItem** (bool enabled)  
*Ensure a initial selected item.*
- void **setScrollCurrentToCenter** (bool enabled)  
*Scroll automatically the current index to center of the view.*
- void **setScrollStepGranularity** (int factor)  
*Determine a step size for scrolling: The larger this number, the smaller and more precise is the scrolling.*
- void **setSelectedIndexes** (const QList< QModelIndex > &indexes)
- void **setSpacing** (int spacing)  
*Sets the spacing.*
- void **setToolTipEnabled** (bool enabled)
- void **setUsePointingHandCursor** (bool useCursor)  
*Set if the PointingHand Cursor should be shown over the activation area.*
- void **toFirstIndex** ()  
*Selects the index as current and scrolls to it.*
- void **toIndex** (const QModelIndex &index)
- void **toLastIndex** ()
- void **toNextIndex** ()
- void **toPreviousIndex** ()

## Public Member Functions inherited from [Digikam::DCategorizedView](#)

- **DCategorizedView** (QWidget \*const parent=nullptr)
- virtual QModelIndexList [categorizedIndexesIn](#) (const QRect &rect) const  
*This method will return all indexes whose visual rect intersects rect.*
- virtual QModelIndex [categoryAt](#) (const QPoint &point) const  
*This method will return the first index of the category in the region of which point is found.*
- **DCategoryDrawer** \* [categoryDrawer](#) () const
- virtual QItemSelectionRange [categoryRange](#) (const QModelIndex &index) const  
*This method returns the range of indexes contained in the category in which index is sorted.*
- virtual QRect [categoryVisualRect](#) (const QModelIndex &index) const  
*This method will return the visual rect of the header of the category in which index is sorted.*
- QModelIndex [indexAt](#) (const QPoint &point) const override
- void [setCategoryDrawer](#) ([DCategoryDrawer](#) \*categoryDrawer)
- void [setDrawDraggedItems](#) (bool drawDraggedItems)  
*Switch on drawing of dragged items.*
- void [setGridSize](#) (const QSize &size)
- void [setModel](#) (QAbstractItemModel \*model) override
- QRect [visualRect](#) (const QModelIndex &index) const override

## Public Member Functions inherited from [Digikam::DragDropViewImplementation](#)

- virtual void [copy](#) ()
- virtual void [cut](#) ()
- virtual void [paste](#) ()

## Protected Slots

- void [layoutAboutToBeChanged](#) ()
- void [layoutWasChanged](#) ()
- void [slotActivated](#) (const QModelIndex &index)
- void [slotClicked](#) (const QModelIndex &index)
- void [slotEntered](#) (const QModelIndex &index)
- virtual void [slotSetupChanged](#) ()
- virtual void [slotThemeChanged](#) ()

## Protected Slots inherited from [Digikam::DCategorizedView](#)

- void [currentChanged](#) (const QModelIndex &current, const QModelIndex &previous) override
- void [rowsInserted](#) (const QModelIndex &parent, int start, int end) override
- virtual void [rowsInsertedArtificial](#) (const QModelIndex &parent, int start, int end)
- virtual void [slotLayoutChanged](#) ()
- void [updateGeometries](#) () override

## Protected Member Functions

- void **contextMenuEvent** (QContextMenuEvent \*event) override  
*reimplemented from parent class*
- bool **decodelsCutSelection** (const QMimeData \*mimeData)
- void **encodelsCutSelection** (QMimeData \*mime, bool isCutSelection)
- virtual void **indexActivated** (const QModelIndex &index, Qt::KeyboardModifiers modifiers)
- QModelIndex **indexForCategoryAt** (const QPoint &pos) const  
*Returns an index that is representative for the category at position pos.*
- void **keyPressEvent** (QKeyEvent \*event) override
- void **leaveEvent** (QEvent \*event) override
- QModelIndex **mapIndexForDragDrop** (const QModelIndex &index) const override  
*Note: pure virtual [dragDropHandler\(\)](#) still open from [DragDropViewImplementation](#).*
- void **mouseMoveEvent** (QMouseEvent \*event) override
- void **mousePressEvent** (QMouseEvent \*event) override
- void **mouseReleaseEvent** (QMouseEvent \*event) override
- QModelIndex **moveCursor** (CursorAction cursorAction, Qt::KeyboardModifiers modifiers) override
- virtual QModelIndex **nextIndexHint** (const QModelIndex &indexToAnchor, const QItemSelectionRange &removed) const  
*Assuming the given indexes would be removed (hypothetically!), return the index to be selected instead, starting from anchor.*
- QPixmap **pixmapForDrag** (const QList< QModelIndex > &indexes) const override  
*Creates a pixmap for dragging the given indexes.*
- void **reset** () override
- void **resizeEvent** (QResizeEvent \*e) override
- void **rowsAboutToBeRemoved** (const QModelIndex &parent, int start, int end) override
- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- void **rowsRemoved** (const QModelIndex &parent, int start, int end) override
- void **selectionChanged** (const QItemSelection &, const QItemSelection &) override
- void **setItemDelegate** (DItemDelegate \*delegate)
- void **setToolTip** (ItemViewToolTip \*tip)
- virtual void **showContextMenu** (QContextMenuEvent \*event)
- virtual void **showContextMenuOnIndex** (QContextMenuEvent \*event, const QModelIndex &index)  
*Reimplement these in a subclass.*
- virtual bool **showToolTip** (const QModelIndex &index, QStyleOptionViewItem &option, QHelpEvent \*e=nullptr)  
*Provides default behavior, can reimplement in a subclass.*
- void **updateDelegateSizes** ()
- void **userInteraction** ()
- bool **viewportEvent** (QEvent \*event) override
- void **wheelEvent** (QWheelEvent \*event) override

## Protected Member Functions inherited from [Digikam::DCategorizedView](#)

- void **dragLeaveEvent** (QDragLeaveEvent \*event) override
- void **dragMoveEvent** (QDragMoveEvent \*event) override
- void **dropEvent** (QDropEvent \*event) override
- void **leaveEvent** (QEvent \*event) override
- void **mouseMoveEvent** (QMouseEvent \*event) override
- void **mousePressEvent** (QMouseEvent \*event) override
- void **mouseReleaseEvent** (QMouseEvent \*event) override
- QModelIndex **moveCursor** (CursorAction cursorAction, Qt::KeyboardModifiers modifiers) override
- void **paintEvent** (QPaintEvent \*event) override
- void **resizeEvent** (QResizeEvent \*event) override
- void **setSelection** (const QRect &rect, QItemSelectionModel::SelectionFlags flags) override
- void **startDrag** (Qt::DropActions supportedActions) override

## Protected Member Functions inherited from [Digikam::DragDropViewImplementation](#)

- virtual `QAbstractItemView * asView ()=0`  
*This one is implemented by `DECLARE_VIEW_DRAG_DROP_METHODS`.*
- bool `decodelsCutSelection` (const `QMimeData *mimeData`)
- virtual `AbstractItemDragDropHandler * dragDropHandler ()` const =0  
*You need to implement these three methods Returns the drag drop handler.*
- void `dragEnterEvent` (`QDragEnterEvent *event`)  
*Implements the relevant QAbstractItemView methods for drag and drop.*
- void `dragMoveEvent` (`QDragMoveEvent *e`)
- void `dropEvent` (`QDropEvent *e`)
- void `encodelsCutSelection` (`QMimeData *mime`, bool `isCutSelection`)
- void `startDrag` (`Qt::DropActions supportedActions`)

### 9.878.1 Member Function Documentation

#### 9.878.1.1 clicked

```
void Digikam::ItemViewCategorized::clicked (
    const QMouseEvent * e,
    const QModelIndex & index ) [signal]
```

Do not change the mouse events.

#### 9.878.1.2 filterModel()

```
virtual QSortFilterProxyModel * Digikam::ItemViewCategorized::filterModel ( ) const [pure virtual]
```

Implemented in [ShowFoto::ShowfotoCategorizedView](#), and [Digikam::ImportCategorizedView](#).

#### 9.878.1.3 keyPressed

```
void Digikam::ItemViewCategorized::keyPressed (
    QKeyEvent * e ) [signal]
```

This signal is emitted after being handled by this widget. You can accept it if ignored.

#### 9.878.1.4 mapIndexForDragDrop()

```
QModelIndex Digikam::ItemViewCategorized::mapIndexForDragDrop (
    const QModelIndex & index ) const [override], [protected], [virtual]
```

`cut()`, `copy()`, `paste()`, [dragEnterEvent\(\)](#), `dragMoveEvent()`, `dropEvent()`, `startDrag()` are implemented by [DragDropViewImplementation](#)

Implements [Digikam::DragDropViewImplementation](#).

### 9.878.1.5 nextIndexHint()

```
QModelIndex Digikam::ItemViewCategorized::nextIndexHint (
    const QModelIndex & indexToAnchor,
    const QItemSelectionRange & removed ) const [protected], [virtual]
```

The default implementation returns the next remaining sibling.

Reimplemented in [Digikam::ItemCategorizedView](#), [ShowFoto::ShowfotoCategorizedView](#), and [Digikam::ImportCategorizedView](#).

### 9.878.1.6 pixmapForDrag()

```
QPixmap Digikam::ItemViewCategorized::pixmapForDrag (
    const QList< QModelIndex > & indexes ) const [override], [protected], [virtual]
```

Implements [Digikam::DragDropViewImplementation](#).

### 9.878.1.7 rowsRemoved()

```
void Digikam::ItemViewCategorized::rowsRemoved (
    const QModelIndex & parent,
    int start,
    int end ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::DCategorizedView](#).

### 9.878.1.8 selectionChanged

```
void Digikam::ItemViewCategorized::selectionChanged ( ) [signal]
```

Any of the signals below will be emitted before.

### 9.878.1.9 setScrollStepGranularity()

```
void Digikam::ItemViewCategorized::setScrollStepGranularity (
    int factor )
```

Default is 10.

### 9.878.1.10 setSpacing()

```
void Digikam::ItemViewCategorized::setSpacing (
    int spacing )
```

Does not use [setSpacing\(\)/spacing\(\)](#) from [QListView](#)

### 9.878.1.11 showContextMenuOnIndex()

```
void Digikam::ItemViewCategorized::showContextMenuOnIndex (
    QContextMenuEvent * event,
    const QModelIndex & index ) [protected], [virtual]
```

Reimplemented in [Digikam::ItemCategorizedView](#), [ShowFoto::ShowfotoCategorizedView](#), and [Digikam::ImportCategorizedView](#).

### 9.878.1.12 showToolTip()

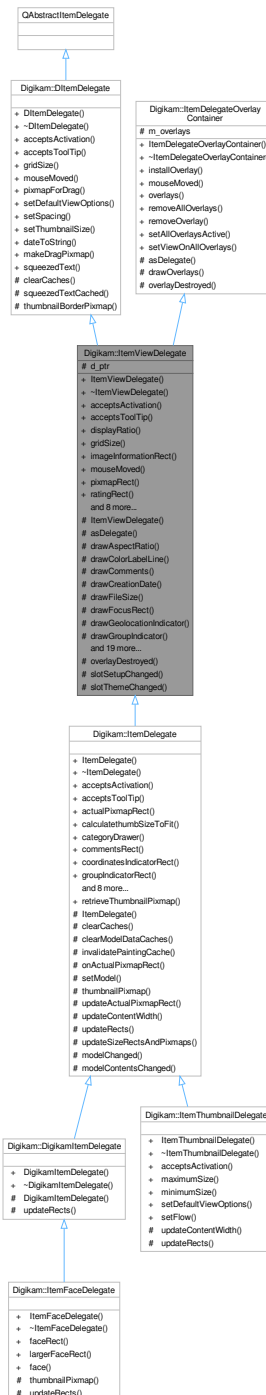
```
bool Digikam::ItemViewCategorized::showToolTip (
    const QModelIndex & index,
    QStyleOptionViewItem & option,
    QHelpEvent * e = nullptr ) [protected], [virtual]
```

Returns true if a tooltip was shown. The help event is optional.



## 9.879 Digikam::ItemViewDelegate Class Reference

Inheritance diagram for Digikam::ItemViewDelegate:



### Signals

- void **hideNotification** ()
- void **requestNotification** (const QModelIndex &index, const QString &message)

## Signals inherited from [Digikam::DItemDelegate](#)

- void **gridSizeChanged** (const QSize &newSize)
- void **visualChange** ()

## Public Member Functions

- **ItemViewDelegate** (QWidget \*const parent)
- bool **acceptsActivation** (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*activationRect=nullptr) const override
- bool **acceptsToolTip** (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*tooltipRect=nullptr) const override
 

*These methods take four parameters: The position on viewport, the rect on viewport, the index, and optionally a parameter into which, if the return value is true, a rectangle can be written for which the return value will be true as well.*
- double **displayRatio** () const
- QSize **gridSize** () const override
 

*Returns the gridsize to be set by the view.*
- virtual QRect **imageInformationRect** () const
 

*Returns the area where the image information is drawn, or null if empty / not supported.*
- void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index) override
- virtual QRect  **pixmapRect** () const
 

*Returns the area where the pixmap is drawn, or null if not supported.*
- virtual QRect **ratingRect** () const
 

*Returns the rectangle where the rating is drawn, or a null rectangle if not supported.*
- QRect **rect** () const
- void **setDefaultViewOptions** (const QStyleOptionViewItem &option) override
 

*Style option with standard values to use for cached rendering.*
- void **setRatingEdited** (const QModelIndex &index)
 

*Can be used to temporarily disable drawing of the rating.*
- void **setSpacing** (int spacing) override
- void **setThumbnailSize** (const ThumbnailSize &thumbSize) override
 

*You must set these options from the view.*
- QSize **sizeHint** (const QStyleOptionViewItem &option, const QModelIndex &index) const override
- int **spacing** () const
- [ThumbnailSize](#) **thumbnailSize** () const

## Public Member Functions inherited from [Digikam::DItemDelegate](#)

- **DItemDelegate** (QObject \*const parent=nullptr)
- virtual QPixmap **pixmapForDrag** (const QStyleOptionViewItem &option, const QList< QModelIndex > &indexes) const =0

## Public Member Functions inherited from [Digikam::ItemDelegateOverlayContainer](#)

- **ItemDelegateOverlayContainer** ()=default
 

*This is a sample implementation for delegate management methods, to be inherited by a delegate.*
- void **installOverlay** ([ItemDelegateOverlay](#) \*overlay)
- void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)
- QList< [ItemDelegateOverlay](#) \* > **overlays** () const
- void **removeAllOverlays** ()
- void **removeOverlay** ([ItemDelegateOverlay](#) \*overlay)
- void **setAllOverlaysActive** (bool active)
- void **setViewOnAllOverlays** (QAbstractItemView \*view)

## Protected Slots

- void **overlayDestroyed** (QObject \*o) override
- void **slotSetupChanged** ()
- void **slotThemeChanged** ()

## Protected Member Functions

- **ItemViewDelegate** (ItemViewDelegatePrivate &dd, QWidget \*const parent)
- QAbstractItemDelegate \* **asDelegate** () override  
*Returns the delegate, typically, the derived class.*
- void **drawAspectRatio** (QPainter \*p, const QRect &dimsRect, const QSize &dims) const
- void **drawColorLabelLine** (QPainter \*p, const QRect &pixRect, int colorId) const
- void **drawComments** (QPainter \*p, const QRect &commentsRect, const QString &comments) const
- void **drawCreationDate** (QPainter \*p, const QRect &dateRect, const QDateTime &date) const
- void **drawFileSize** (QPainter \*p, const QRect &r, qlonglong bytes) const
- void **drawFocusRect** (QPainter \*p, const QStyleOptionViewItem &option, bool isSelected) const
- void **drawGeolocationIndicator** (QPainter \*p, const QRect &r) const
- void **drawGroupIndicator** (QPainter \*p, const QRect &r, int numberOfGroupedImages, bool open) const
- void **drawImageFormat** (QPainter \*p, const QRect &r, const QString &f, bool drawTop) const
- void **drawImageSize** (QPainter \*p, const QRect &dimsRect, const QSize &dims) const
- void **drawModificationDate** (QPainter \*p, const QRect &dateRect, const QDateTime &date) const
- void **drawMouseOverRect** (QPainter \*p, const QStyleOptionViewItem &option) const
- void **drawName** (QPainter \*p, const QRect &nameRect, const QString &name) const
- void **drawPanelSidelcon** (QPainter \*p, bool left, bool right) const
- void **drawPickLabelIcon** (QPainter \*p, const QRect &r, int pickLabel) const
- void **drawRating** (QPainter \*p, const QModelIndex &index, const QRect &ratingRect, int rating, bool isSelected) const
- void **drawSpecialInfo** (QPainter \*p, const QRect &r, const QString &text) const
- void **drawTags** (QPainter \*p, const QRect &r, const QString &tagsString, bool isSelected) const
- QRect **drawThumbnail** (QPainter \*p, const QRect &thumbRect, const QPixmap &background, const QPixmap &thumbnail, bool isGrouped) const  
*Use the tool methods for painting in subclasses.*
- void **drawTitle** (QPainter \*p, const QRect &titleRect, const QString &title) const
- virtual void **invalidatePaintingCache** ()
- void **prepareBackground** ()
- void **prepareFonts** ()
- void **prepareMetrics** (int maxWidth)
- void **prepareRatingPixmap** (bool composeOverBackground=true)
- QPixmap **ratingPixmap** (int rating, bool selected) const  
*Returns the relevant pixmap from the cached rating pixmaps.*
- virtual void **updateSizeRectsAndPixmap** ()=0

## Protected Member Functions inherited from Digikam::DItemDelegate

- virtual void **clearCaches** ()
- QString **squeezedTextCached** (QPainter \*const p, int width, const QString &text) const
- QPixmap **thumbnailBorderPixmap** (const QSize &pixSize, bool isGrouped=false) const

## Protected Member Functions inherited from [Digikam::ItemDelegateOverlayContainer](#)

- virtual void **drawOverlays** (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index) const
- virtual void **overlayDestroyed** (QObject \*o)

*Declare as slot in the derived class calling this method.*

## Protected Attributes

- ItemViewDelegatePrivate \*const **d\_ptr** = nullptr

## Protected Attributes inherited from [Digikam::ItemDelegateOverlayContainer](#)

- QList< [ItemDelegateOverlay](#) \* > **m\_overlays**

## Additional Inherited Members

## Static Public Member Functions inherited from [Digikam::DItemDelegate](#)

- static QString **dateToString** (const QDateTime &datetime)
- static QPixmap **makeDragPixmap** (const QStyleOptionViewItem &option, const QList< QModelIndex > &indexes, double displayRatio, const QPixmap &suggestedPixmap=QPixmap())
- static QString **squeezedText** (const QFontMetrics &fm, int width, const QString &text)

## 9.879.1 Member Function Documentation

### 9.879.1.1 acceptsActivation()

```
bool Digikam::ItemViewDelegate::acceptsActivation (
    const QPoint & pos,
    const QRect & visualRect,
    const QModelIndex & index,
    QRect * activationRect = nullptr ) const [override], [virtual]
```

Implements [Digikam::DItemDelegate](#).

### 9.879.1.2 acceptsToolTip()

```
bool Digikam::ItemViewDelegate::acceptsToolTip (
    const QPoint & pos,
    const QRect & visualRect,
    const QModelIndex & index,
    QRect * tooltipRect = nullptr ) const [override], [virtual]
```

Implements [Digikam::DItemDelegate](#).

### 9.879.1.3 asDelegate()

```
QAbstractItemDelegate * Digikam::ItemViewDelegate::asDelegate ( ) [override], [protected], [virtual]
```

Implements [Digikam::ItemDelegateOverlayContainer](#).

### 9.879.1.4 gridSize()

```
QSize Digikam::ItemViewDelegate::gridSize ( ) const [override], [virtual]
```

It's sizeHint plus spacing.

Implements [Digikam::DItemDelegate](#).

### 9.879.1.5 imageInformationRect()

```
QRect Digikam::ItemViewDelegate::imageInformationRect ( ) const [virtual]
```

The image information is textual or graphical information, but not the pixmap. The [ratingRect\(\)](#) will e.g. typically be contained in this area.

Reimplemented in [Digikam::ItemDelegate](#).

### 9.879.1.6 mouseMoved()

```
void Digikam::ItemViewDelegate::mouseMoved (
    QMouseEvent * e,
    const QRect & visualRect,
    const QModelIndex & index ) [override], [virtual]
```

#### Note

to be called by [ItemViewCategorized](#) only

Implements [Digikam::DItemDelegate](#).

### 9.879.1.7 pixmapRect()

```
QRect Digikam::ItemViewDelegate::pixmapRect ( ) const [virtual]
```

Reimplemented in [Digikam::ItemDelegate](#).

### 9.879.1.8 setDefaultViewOptions()

```
void Digikam::ItemViewDelegate::setDefaultViewOptions (
    const QStyleOptionViewItem & option ) [override], [virtual]
```

option.rect shall be the viewport rectangle. Call on resize, font change.

Implements [Digikam::DItemDelegate](#).

### 9.879.1.9 setRatingEdited()

```
void Digikam::ItemViewDelegate::setRatingEdited (
    const QModelIndex & index )
```

Call with QModelIndex() afterwards.

### 9.879.1.10 setSpacing()

```
void Digikam::ItemViewDelegate::setSpacing (
    int spacing ) [override], [virtual]
```

Implements [Digikam::DItemDelegate](#).

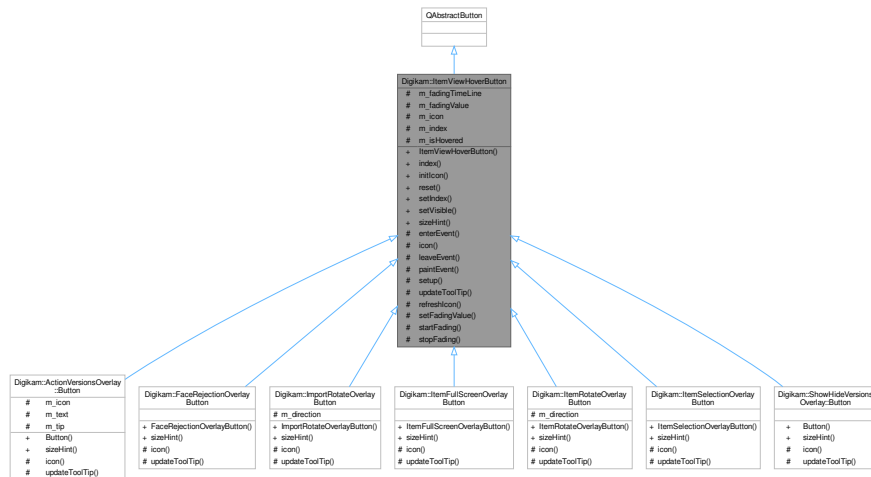
### 9.879.1.11 setThumbnailSize()

```
void Digikam::ItemViewDelegate::setThumbnailSize (
    const ThumbnailSize & thumbSize ) [override], [virtual]
```

Implements [Digikam::DItemDelegate](#).

## 9.880 Digikam::ItemViewHoverButton Class Reference

Inheritance diagram for Digikam::ItemViewHoverButton:



### Public Member Functions

- **ItemViewHoverButton** (QAbstractItemView \*const parentView)
- QModelIndex **index** () const
- void **initIcon** ()
- void **reset** ()
- void **setIndex** (const QModelIndex &index)
- void **setVisible** (bool visible) override
- QSize **sizeHint** () const override=0

*Reimplement to match the size of your icon.*

## Protected Slots

- void **refreshIcon** ()
- void **setFadingValue** (int value)
- void **startFading** ()
- void **stopFading** ()

## Protected Member Functions

- void **enterEvent** (QEnterEvent \*event)
- virtual QIcon **icon** ()=0  
*Return your icon here.*
- void **leaveEvent** (QEvent \*event)
- void **paintEvent** (QPaintEvent \*event)
- void **setup** ()  
*to call in children class constructors to init signal/slot connections.*
- virtual void **updateToolTip** ()  
*Optionally update tooltip here.*

## Protected Attributes

- QTimerLine \* **m\_fadingTimeLine** = nullptr
- int **m\_fadingValue** = 0
- QIcon **m\_icon**
- QPersistentModelIndex **m\_index**
- bool **m\_isHovered** = false

## 9.880.1 Member Function Documentation

### 9.880.1.1 icon()

```
virtual QIcon Digikam::ItemViewHoverButton::icon ( ) [protected], [pure virtual]
```

Will be queried again on toggle.

Implemented in [Digikam::FaceRejectionOverlayButton](#), [Digikam::ItemFullScreenOverlayButton](#), [Digikam::ItemRotateOverlayButton](#), [Digikam::ItemSelectionOverlayButton](#), and [Digikam::ImportRotateOverlayButton](#).

### 9.880.1.2 sizeHint()

```
QSize Digikam::ItemViewHoverButton::sizeHint ( ) const [override], [pure virtual]
```

Implemented in [Digikam::FaceRejectionOverlayButton](#), [Digikam::ItemFullScreenOverlayButton](#), [Digikam::ItemRotateOverlayButton](#), [Digikam::ItemSelectionOverlayButton](#), and [Digikam::ImportRotateOverlayButton](#).

### 9.880.1.3 updateToolTip()

```
void Digikam::ItemViewHoverButton::updateToolTip ( ) [protected], [virtual]
```

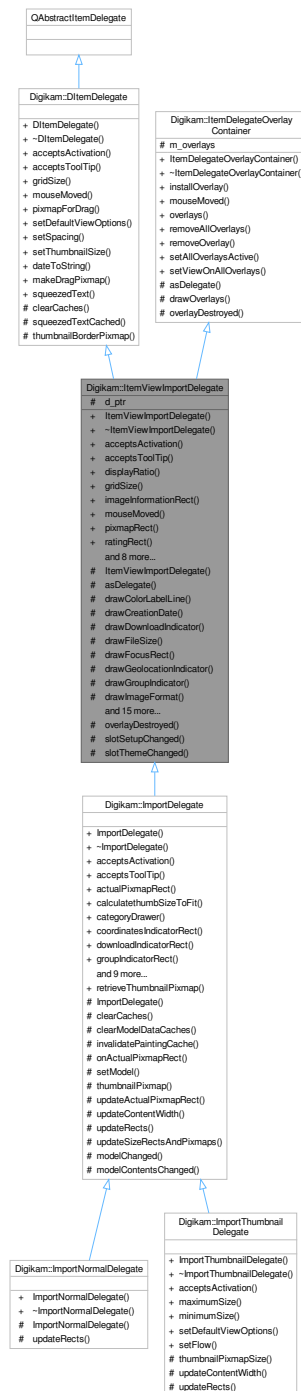
Will be called again on state change.

Reimplemented in [Digikam::FaceRejectionOverlayButton](#), [Digikam::ItemFullScreenOverlayButton](#), [Digikam::ItemRotateOverlayButton](#), [Digikam::ItemSelectionOverlayButton](#), and [Digikam::ImportRotateOverlayButton](#).



## 9.881 Digikam::ItemViewImportDelegate Class Reference

Inheritance diagram for Digikam::ItemViewImportDelegate:



### Signals

- void **hideNotification** ()
- void **requestNotification** (const QModelIndex &index, const QString &message)

## Signals inherited from [Digikam::DItemDelegate](#)

- void **gridSizeChanged** (const QSize &newSize)
- void **visualChange** ()

## Public Member Functions

- **ItemViewImportDelegate** (QWidget \*const parent)
- bool **acceptsActivation** (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*activationRect=nullptr) const override
- bool **acceptsToolTip** (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*tooltipRect=nullptr) const override
 

*These methods take four parameters: The position on viewport, the rect on viewport, the index, and optionally a parameter into which, if the return value is true, a rectangle can be written for which the return value will be true as well.*
- double **displayRatio** () const
- QSize **gridSize** () const override
 

*Returns the gridsize to be set by the view.*
- virtual QRect **imageInformationRect** () const
 

*Returns the area where the image information is drawn, or null if empty / not supported.*
- void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index) override
- virtual QRect  **pixmapRect** () const
 

*Returns the area where the pixmap is drawn, or null if not supported.*
- virtual QRect **ratingRect** () const
 

*Returns the rectangle where the rating is drawn, or a null rectangle if not supported.*
- QRect **rect** () const
- void **setDefaultViewOptions** (const QStyleOptionViewItem &option) override
 

*Style option with standard values to use for cached rendering.*
- void **setRatingEdited** (const QModelIndex &index)
 

*Can be used to temporarily disable drawing of the rating.*
- void **setSpacing** (int spacing) override
- void **setThumbnailSize** (const ThumbnailSize &thumbSize) override
 

*reimplemented from [DItemDelegate](#)*
- QSize **sizeHint** (const QStyleOptionViewItem &option, const QModelIndex &index) const override
- int **spacing** () const
- [ThumbnailSize](#) **thumbnailSize** () const

## Public Member Functions inherited from [Digikam::DItemDelegate](#)

- **DItemDelegate** (QObject \*const parent=nullptr)
- virtual QPixmap **pixmapForDrag** (const QStyleOptionViewItem &option, const QList< QModelIndex > &indexes) const =0

## Public Member Functions inherited from [Digikam::ItemDelegateOverlayContainer](#)

- **ItemDelegateOverlayContainer** ()=default
 

*This is a sample implementation for delegate management methods, to be inherited by a delegate.*
- void **installOverlay** ([ItemDelegateOverlay](#) \*overlay)
- void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)
- QList< [ItemDelegateOverlay](#) \* > **overlays** () const
- void **removeAllOverlays** ()
- void **removeOverlay** ([ItemDelegateOverlay](#) \*overlay)
- void **setAllOverlaysActive** (bool active)
- void **setViewOnAllOverlays** (QAbstractItemView \*view)

### Protected Slots

- void **overlayDestroyed** (QObject \*o) override
- void **slotSetupChanged** ()
- void **slotThemeChanged** ()

### Protected Member Functions

- **ItemViewImportDelegate** (ItemViewImportDelegatePrivate &dd, QWidget \*const parent)
- QAbstractItemDelegate \* **asDelegate** () override
 

*Returns the delegate, typically, the derived class.*
- void **drawColorLabelLine** (QPainter \*p, const QRect &pixRect, int colorId) const
- void **drawCreationDate** (QPainter \*p, const QRect &dateRect, const QDateTime &date) const
- void **drawDownloadIndicator** (QPainter \*p, const QRect &r, int itemType) const
- void **drawFileSize** (QPainter \*p, const QRect &r, qlonglong bytes) const
- void **drawFocusRect** (QPainter \*p, const QStyleOptionViewItem &option, bool isSelected) const
- void **drawGeolocationIndicator** (QPainter \*p, const QRect &r) const
- void **drawGroupIndicator** (QPainter \*p, const QRect &r, int numberOfGroupedImages, bool open) const
- void **drawImageFormat** (QPainter \*p, const QRect &dimsRect, const QString &mime) const
- void **drawImageSize** (QPainter \*p, const QRect &dimsRect, const QSize &dims) const
- void **drawLockIndicator** (QPainter \*p, const QRect &r, int lockStatus) const
- void **drawMouseOverRect** (QPainter \*p, const QStyleOptionViewItem &option) const
- void **drawName** (QPainter \*p, const QRect &nameRect, const QString &name) const
- void **drawPickLabelIcon** (QPainter \*p, const QRect &r, int pickLabel) const
- void **drawRating** (QPainter \*p, const QModelIndex &index, const QRect &ratingRect, int rating, bool isSelected) const
- void **drawTags** (QPainter \*p, const QRect &r, const QString &tagsString, bool isSelected) const
- QRect **drawThumbnail** (QPainter \*p, const QRect &thumbRect, const QPixmap &background, const QPixmap &thumbnail) const
 

*Use the tool methods for painting in subclasses.*
- virtual void **invalidatePaintingCache** ()
 

*reimplement these in subclasses*
- void **prepareBackground** ()
- void **prepareFonts** ()
- void **prepareMetrics** (int maxWidth)
- void **prepareRatingPixmap** (bool composeOverBackground=true)
- QPixmap **ratingPixmap** (int rating, bool selected) const
 

*Returns the relevant pixmap from the cached rating pixmaps.*
- virtual void **updateSizeRectsAndPixmap** ()=0

### Protected Member Functions inherited from [Digikam::DItemDelegate](#)

- virtual void **clearCaches** ()
- QString **squeezedTextCached** (QPainter \*const p, int width, const QString &text) const
- QPixmap **thumbnailBorderPixmap** (const QSize &pixSize, bool isGrouped=false) const

### Protected Member Functions inherited from [Digikam::ItemDelegateOverlayContainer](#)

- virtual void **drawOverlays** (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index) const
- virtual void **overlayDestroyed** (QObject \*o)
 

*Declare as slot in the derived class calling this method.*

## Protected Attributes

- `ItemViewImportDelegatePrivate *const d_ptr = nullptr`

## Protected Attributes inherited from [Digikam::ItemDelegateOverlayContainer](#)

- `QList< ItemDelegateOverlay * > m_overlays`

## Additional Inherited Members

## Static Public Member Functions inherited from [Digikam::DItemDelegate](#)

- static `QString dateToString` (const `QDateTime` &datetime)
- static `QPixmap makeDragPixmap` (const `QStyleOptionViewItem` &option, const `QList< QModelIndex >` &indexes, double displayRatio, const `QPixmap` &suggestedPixmap=`QPixmap()`)
- static `QString squeezedText` (const `QFontMetrics` &fm, int width, const `QString` &text)

## 9.881.1 Member Function Documentation

### 9.881.1.1 `acceptsActivation()`

```
bool Digikam::ItemViewImportDelegate::acceptsActivation (
    const QPoint & pos,
    const QRect & visualRect,
    const QModelIndex & index,
    QRect * activationRect = nullptr ) const [override], [virtual]
```

Implements [Digikam::DItemDelegate](#).

### 9.881.1.2 `acceptsToolTip()`

```
bool Digikam::ItemViewImportDelegate::acceptsToolTip (
    const QPoint & pos,
    const QRect & visualRect,
    const QModelIndex & index,
    QRect * tooltipRect = nullptr ) const [override], [virtual]
```

Implements [Digikam::DItemDelegate](#).

### 9.881.1.3 `asDelegate()`

```
QAbstractItemDelegate * Digikam::ItemViewImportDelegate::asDelegate ( ) [override], [protected],
[virtual]
```

Implements [Digikam::ItemDelegateOverlayContainer](#).

#### 9.881.1.4 gridSize()

```
QSize Digikam::ItemViewImportDelegate::gridSize ( ) const [override], [virtual]
```

It's sizeHint plus spacing.

Implements [Digikam::DItemDelegate](#).

#### 9.881.1.5 imageInformationRect()

```
QRect Digikam::ItemViewImportDelegate::imageInformationRect ( ) const [virtual]
```

The image information is textual or graphical information, but not the pixmap. The [ratingRect\(\)](#) will e.g. typically be contained in this area.

Reimplemented in [Digikam::ImportDelegate](#).

#### 9.881.1.6 invalidatePaintingCache()

```
void Digikam::ItemViewImportDelegate::invalidatePaintingCache ( ) [protected], [virtual]
```

Reimplemented in [Digikam::ImportDelegate](#).

#### 9.881.1.7 mouseMoved()

```
void Digikam::ItemViewImportDelegate::mouseMoved (
    QMouseEvent * e,
    const QRect & visualRect,
    const QModelIndex & index ) [override], [virtual]
```

##### Note

to be called by [ItemViewCategorized](#) only

Implements [Digikam::DItemDelegate](#).

#### 9.881.1.8 pixmapRect()

```
QRect Digikam::ItemViewImportDelegate::pixmapRect ( ) const [virtual]
```

Reimplemented in [Digikam::ImportDelegate](#).

#### 9.881.1.9 prepareRatingPixmaps()

```
void Digikam::ItemViewImportDelegate::prepareRatingPixmaps (
    bool composeOverBackground = true ) [protected]
```

Please call this method after [prepareBackground\(\)](#) and when `d->ratingPixmap` is set

#### 9.881.1.10 setDefaultViewOptions()

```
void Digikam::ItemViewImportDelegate::setDefaultViewOptions (
    const QStyleOptionViewItem & option ) [override], [virtual]
```

option.rect shall be the viewport rectangle. Call on resize, font change.

Implements [Digikam::DItemDelegate](#).

#### 9.881.1.11 setRatingEdited()

```
void Digikam::ItemViewImportDelegate::setRatingEdited (
    const QModelIndex & index )
```

Call with QModelIndex() afterwards.

#### 9.881.1.12 setSpacing()

```
void Digikam::ItemViewImportDelegate::setSpacing (
    int spacing ) [override], [virtual]
```

Implements [Digikam::DItemDelegate](#).

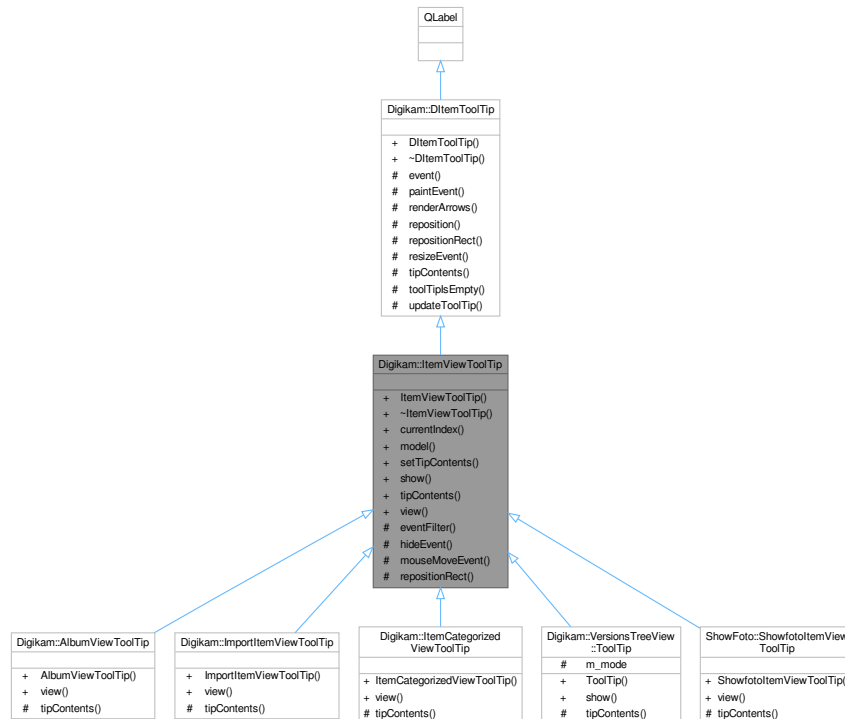
#### 9.881.1.13 setThumbnailSize()

```
void Digikam::ItemViewImportDelegate::setThumbnailSize (
    const ThumbnailSize & thumbSize ) [override], [virtual]
```

Implements [Digikam::DItemDelegate](#).

## 9.882 Digikam::ItemViewToolTip Class Reference

Inheritance diagram for Digikam::ItemViewToolTip:



### Public Member Functions

- **ItemViewToolTip** (QAbstractItemView \*const view)
- QModelIndex **currentIndex** () const
- QAbstractItemModel \* **model** () const
- void **setTipContents** (const QString &tipContents)
- void **show** (const QStyleOptionViewItem &option, const QModelIndex &index)
  - Show the tooltip for the given item.*
- QString **tipContents** () override
  - Default implementation is based on setTipContents().*
- QAbstractItemView \* **view** () const

### Public Member Functions inherited from Digikam::DItemToolTip

- **DItemToolTip** (QWidget \*const parent=nullptr)

### Protected Member Functions

- bool **eventFilter** (QObject \*o, QEvent \*e) override
- void **hideEvent** (QHideEvent \*) override
- void **mouseMoveEvent** (QMouseEvent \*e) override
- QRect **repositionRect** () override

## Protected Member Functions inherited from [Digikam::DItemToolTip](#)

- bool **event** (QEvent \*) override
- void **paintEvent** (QPaintEvent \*) override
- void **renderArrows** ()
- void **reposition** ()
- void **resizeEvent** (QResizeEvent \*) override
- bool **toolTipsEmpty** () const
- void **updateToolTip** ()

### 9.882.1 Member Function Documentation

#### 9.882.1.1 repositionRect()

```
QRect Digikam::ItemViewToolTip::repositionRect ( ) [override], [protected], [virtual]
```

Implements [Digikam::DItemToolTip](#).

#### 9.882.1.2 show()

```
void Digikam::ItemViewToolTip::show (
    const QStyleOptionViewItem & option,
    const QModelIndex & index )
```

The rect of the given option is taken as area for which the tooltip is shown.

#### 9.882.1.3 tipContents()

```
QString Digikam::ItemViewToolTip::tipContents ( ) [override], [virtual]
```

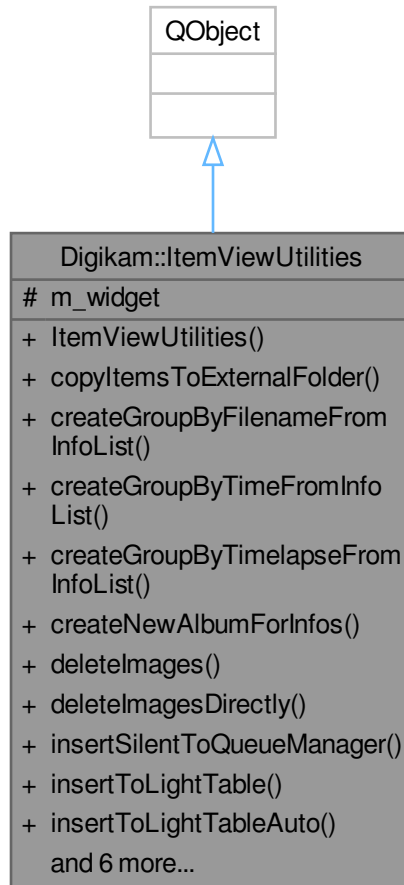
Reimplement if you dynamically provide the contents.

Implements [Digikam::DItemToolTip](#).



## 9.883 Digikam::ItemViewUtilities Class Reference

Inheritance diagram for Digikam::ItemViewUtilities:



### Public Types

- enum `DeleteMode` { `DeletePermanently` = 1 , `DeleteUseTrash` = 2 }

### Public Slots

- void `copyItemsToExternalFolder` (const QList< [ItemInfo](#) > &infos)
- void `createGroupByFilenameFromInfoList` (const [ItemInfoList](#) &itemInfoList)
- void `createGroupByTimeFromInfoList` (const [ItemInfoList](#) &itemInfoList)
- void `createGroupByTimelapseFromInfoList` (const [ItemInfoList](#) &itemInfoList)
- void `createNewAlbumForInfos` (const QList< [ItemInfo](#) > &infos, [Album](#) \*currentAlbum)
- bool `deleteImages` (const QList< [ItemInfo](#) > &infos, const DeleteMode deleteMode)
- void `deleteImagesDirectly` (const QList< [ItemInfo](#) > &infos, const DeleteMode deleteMode)
- void `insertSilentToQueueManager` (const QList< [ItemInfo](#) > &list, const [ItemInfo](#) &currentInfo, int queueid)

- void **insertToLightTable** (const QList< [ItemInfo](#) > &list, const [ItemInfo](#) &current, bool addTo)
- void **insertToLightTableAuto** (const QList< [ItemInfo](#) > &all, const QList< [ItemInfo](#) > &selected, const [ItemInfo](#) &current)
- void **insertToQueueManager** (const QList< [ItemInfo](#) > &list, const [ItemInfo](#) &currentInfo, bool newQueue)
- void **notifyFileContentChanged** (const QList< QUrl > &urls)
- void **openInfos** (const [ItemInfo](#) &info, const QList< [ItemInfo](#) > &allInfosToOpen, [Album](#) \*currentAlbum)
- void **openInfosWithDefaultApplication** (const QList< [ItemInfo](#) > &allInfosToOpen)
- void **rename** (const QUrl &imageUrl, const QString &newName, bool overwrite=false)
- void **setAsAlbumThumbnail** ([Album](#) \*album, const [ItemInfo](#) &itemInfo)

### Signals

- void **editorCurrentUrlChanged** (const QUrl &url)
- void **signalImagesDeleted** (const QList< qlonglong > &imageIds)

### Public Member Functions

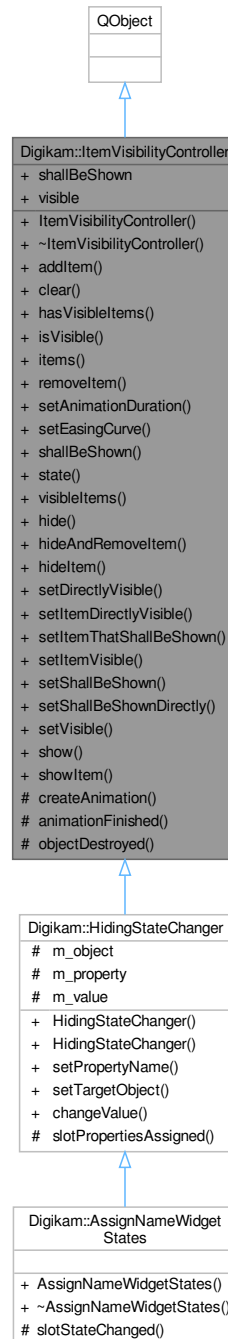
- **ItemViewUtilities** (QWidget \*const parentWidget)

### Protected Attributes

- QWidget \* **m\_widget** = nullptr

## 9.884 Digikam::ItemVisibilityController Class Reference

Inheritance diagram for Digikam::ItemVisibilityController:



### Public Types

- enum `IncludeFadingOutMode` { `IncludeFadingOut` , `ExcludeFadingOut` }
- enum `State` { `Hidden` , `FadingIn` , `Visible` , `FadingOut` }

*This class handles complex visibility situations for items.*

## Public Slots

- void **hide** ()
- void **hideAndRemoveItem** (QObject \*item)
  - Hide the item, and then remove it.*
- void **hideItem** (QObject \*item)
- void **setDirectlyVisible** (bool visible)
- void **setItemDirectlyVisible** (QObject \*item, bool visible)
- void **setItemThatShallBeShown** (QObject \*item)
  - Sets a single item to be shown.*
- void **setItemVisible** (QObject \*item, bool visible)
- void **setShallBeShown** (bool shallBeShown)
  - Adjusts the first condition - the items are shown if shallBeShown is true and isVisible is true.*
- void **setShallBeShownDirectly** (bool shallBeShown)
- void **setVisible** (bool visible)
- void **show** ()
  - Adjusts the main condition.*
- void **showItem** (QObject \*item)
  - Shows or hides a single item.*

## Signals

- void **hiddenAndRemoved** (QObject \*item)
  - Emitted when hideAndRemoveItem has finished.*
- void **propertiesAssigned** (bool visible)
  - Emitted when the (main) transition has finished.*
- void **propertiesAssigned** (QObject \*item, bool visible)
  - Emitted when a transition for a single item finished (see setItemVisible())*

## Public Member Functions

- **ItemVisibilityController** (QObject \*const parent=nullptr)
- void **addItem** (QObject \*const object)
  - Add and remove objects.*
- void **clear** ()
  - Remove all animations.*
- bool **hasVisibleItems** (**IncludeFadingOutMode** mode=**IncludeFadingOut**) const
  - This returns the "result" of isVisible and shallBeShown: Something is indeed visible on the scene.*
- bool **isVisible** () const
- QList< QObject \* > **items** () const
  - Returns all items under control.*
- void **removeItem** (QObject \*const object)
- void **setAnimationDuration** (int msec)
- void **setEasingCurve** (const QEasingCurve &easing)
  - Allows to change the default parameters of all animations.*
- bool **shallBeShown** () const
- **State state** () const
- QList< QObject \* > **visibleItems** (**IncludeFadingOutMode** mode=**IncludeFadingOut**) const
  - Returns all currently visible items.*

## Protected Slots

- void **animationFinished** ()
- void **objectDestroyed** (QObject \*)

## Protected Member Functions

- virtual QPropertyAnimation \* **createAnimation** (QObject \*item)  
*Creates the animation for showing and hiding the given item.*

## Properties

- bool **shallBeShown**
- bool **visible**

## 9.884.1 Member Enumeration Documentation

### 9.884.1.1 IncludeFadingOutMode

```
enum Digikam::ItemVisibilityController::IncludeFadingOutMode
```

#### Enumerator

IncludeFadingOut	In addition to items visible or fading in, return those fading out.
ExcludeFadingOut	Do not return those items currently fading out (soon to be hidden)

### 9.884.1.2 State

```
enum Digikam::ItemVisibilityController::State
```

There is a 3-tiered approach: 1) shallBeShown determines if the items shall at any time be shown. If it is false, items will never be shown. Default is true, so you can ignore this setting. 2) visible determines if the items shall be shown now. Only takes effect if shallBeShown is true. Default is false: Initially, controlled items are hidden. 3) Opacity and individual item visibility: When showing, items are first set to individually visible, then their opacity is increased from 0 to 1. When hiding, opacity is first decreased from 1 to 0, then they are set individually to hidden. Different types of items can be handled:

- a group of items with an "opacity" and "visible" property
- a single item with an "opacity" and "visible" property
- a proxy object with these properties (see above)

## 9.884.2 Member Function Documentation

### 9.884.2.1 addItem()

```
void Digikam::ItemVisibilityController::addItem (
    QObject *const object )
```

The given objects shall provide an "opacity" and a "visible" property. You can, for convenience, use a [ItemVisibilityControllerPropertyObject](#) as a value container, if your items do not provide these properties directly. No ownership is taken, so the objects should live as long as this object is used.

#### 9.884.2.2 createAnimation()

```
QPropertyAnimation * Digikam::ItemVisibilityController::createAnimation (
    QObject * item ) [protected], [virtual]
```

The item is given for information only, you do not need to use it. The default implementation creates an animation for "opacity" from 0.0 to 1.0, using default easing curve and duration, which can and will be changed by [setEasingCurve](#) and [setAnimationDuration](#).

#### 9.884.2.3 hasVisibleItems()

```
bool Digikam::ItemVisibilityController::hasVisibleItems (
    IncludeFadingOutMode mode = IncludeFadingOut ) const
```

Also returns false if no items are available.

#### 9.884.2.4 hideAndRemoveItem

```
void Digikam::ItemVisibilityController::hideAndRemoveItem (
    QObject * item ) [slot]
```

When finished, [hiddenAndRemoved\(\)](#) is emitted.

#### 9.884.2.5 setItemThatShallBeShown

```
void Digikam::ItemVisibilityController::setItemThatShallBeShown (
    QObject * item ) [slot]
```

Calling [setVisible\(\)](#) will effectively effect only this single item, as if calling [setItemVisible\(\)](#). Reset by calling with 0 or [setShallBeShown\(\)](#).

#### 9.884.2.6 show

```
void Digikam::ItemVisibilityController::show ( ) [slot]
```

All items are affected. If any items were shown or hidden separately, they will be resynchronized. "Directly" means no animation is employed.

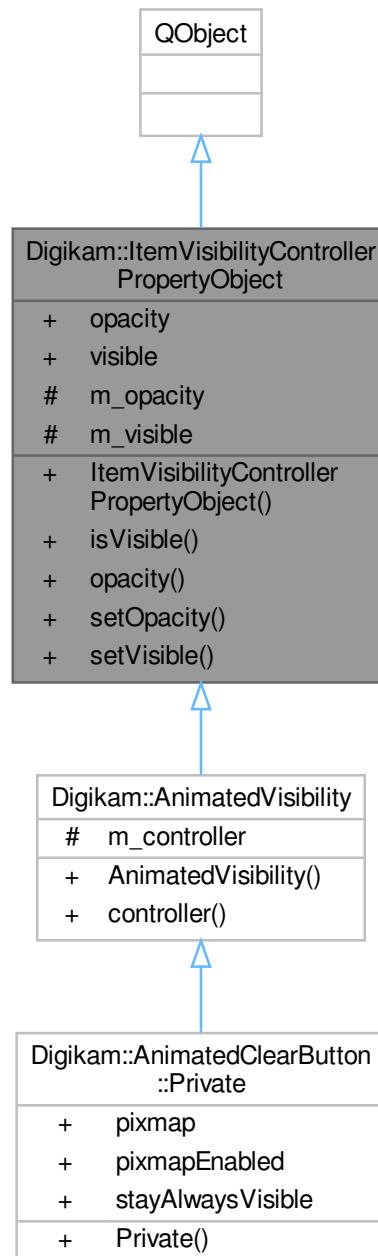
#### 9.884.2.7 showItem

```
void Digikam::ItemVisibilityController::showItem (
    QObject * item ) [slot]
```

The item's status is changed individually. The next call to the "global" method will take precedence again. "Directly" means no animation is employed.

## 9.885 Digikam::ItemVisibilityControllerPropertyObject Class Reference

Inheritance diagram for Digikam::ItemVisibilityControllerPropertyObject:



### Signals

- void **opacityChanged** ()
- void **visibleChanged** ()

## Public Member Functions

- [ItemVisibilityControllerPropertyObject](#) (QObject \*const parent=nullptr)  
*You can use this object as a container providing the properties set by [ItemVisibilityController](#).*
- bool **isVisible** () const
- qreal **opacity** () const
- void **setOpacity** (qreal opacity)
- void **setVisible** (bool visible)

## Protected Attributes

- qreal **m\_opacity** = 0.0
- bool **m\_visible** = false

## Properties

- qreal **opacity**
- bool **visible**

## 9.885.1 Constructor & Destructor Documentation

### 9.885.1.1 ItemVisibilityControllerPropertyObject()

```
Digikam::ItemVisibilityControllerPropertyObject::ItemVisibilityControllerPropertyObject (
    QObject *const parent = nullptr ) [explicit]
```

Connect to the signals accordingly, e.g. to trigger a repaint.

## 9.886 Digikam::JPEGUtils::digikam\_source\_mgr Struct Reference

### Public Attributes

- JOCTET **eoI** [2]
- struct jpeg\_source\_mgr **pub**

## 9.887 Digikam::JPEGUtils::JpegRotator Class Reference

### Public Member Functions

- [JpegRotator](#) (const QString &file)  
*Create a [JpegRotator](#) reading from the given file.*
- [~JpegRotator](#) ()  
*Destructor.*
- bool [autoExifTransform](#) ()  
*Rotate the JPEG file's content according to the current orientation, resetting the current orientation to normal.*
- bool [exifTransform](#) (const [MetaEngineRotation](#) &matrix)  
*Rotate the given image by the given [Matrix](#).*
- bool [exifTransform](#) ([TransformAction](#) action)  
*Rotate the given image by the given [TransformAction](#).*
- void [setCurrentOrientation](#) ([MetaEngine::ImageOrientation](#) orientation)  
*Per default, the orientation is read from the metadata of the file.*
- void [setDestinationFile](#) (const QString &dest)  
*Set the destination file.*
- void [setDocumentName](#) (const QString &documentName)  
*Set the Exif document name of the destination file.*



## Protected Member Functions

- bool **performJpegTransform** ([TransformAction](#) action, const QString &src, const QString &dest)
- void **updateMetadata** (const QString &fileName, const [MetaEngineRotation](#) &matrix)

## Protected Attributes

- QString **m\_destFile**
- QString **m\_documentName**
- QString **m\_file**
- [DMetadata](#) \* **m\_metadata** = nullptr
- [MetaEngine::ImageOrientation](#) **m\_orientation** = MetaEngine::ORIENTATION\_UNSPECIFIED
- QSize **m\_originalSize**

## 9.887.1 Constructor & Destructor Documentation

### 9.887.1.1 JpegRotator()

```
Digikam::JPEGUtils::JpegRotator::JpegRotator (
    const QString & file ) [explicit]
```

Per default, it will replace the file, read the current orientation from the metadata, and use the src file name as documentName.

## 9.887.2 Member Function Documentation

### 9.887.2.1 autoExifTransform()

```
bool Digikam::JPEGUtils::JpegRotator::autoExifTransform ( )
```

The final result of loading the image does not change.

### 9.887.2.2 exifTransform() [1/2]

```
bool Digikam::JPEGUtils::JpegRotator::exifTransform (
    const MetaEngineRotation & matrix )
```

The matrix describes the final transformation, it is not adjusted by current rotation.

### 9.887.2.3 exifTransform() [2/2]

```
bool Digikam::JPEGUtils::JpegRotator::exifTransform (
    TransformAction action )
```

The current orientation will be taken into account

#### 9.887.2.4 setCurrentOrientation()

```
void Digikam::JPEGUtils::JpegRotator::setCurrentOrientation (
    MetaEngine::ImageOrientation orientation )
```

You can override this value

#### 9.887.2.5 setDestinationFile()

```
void Digikam::JPEGUtils::JpegRotator::setDestinationFile (
    const QString & dest )
```

By default, the source file will be overwritten by atomic operation if the operation had succeeded.

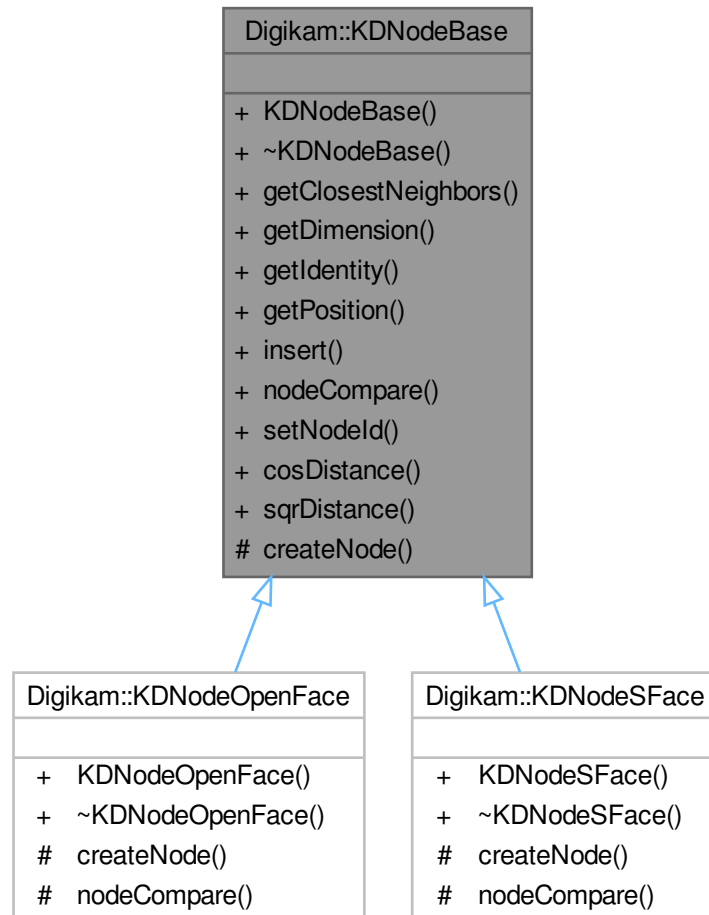
#### 9.887.2.6 setDocumentName()

```
void Digikam::JPEGUtils::JpegRotator::setDocumentName (
    const QString & documentName )
```

Default value is the source's file name

## 9.888 Digikam::KNodeBase Class Reference

Inheritance diagram for Digikam::KNodeBase:



### Classes

- struct [NodeCompareResult](#)

### Public Member Functions

- **KNodeBase** (const cv::Mat &nodePos, const int identity, int splitAxis, int dimension)
- double **getClosestNeighbors** (QMap< double, QVector< int > > &neighborList, const cv::Mat &position, float sqRange, float cosThreshold, int maxNbNeighbors) const  
*Return a list of closest neighbors, limited by maxNbNeighbors and sqRange.*
- int **getDimension** ()
- int **getIdentity** ()  
*Return identity of the node.*

- `cv::Mat` **getPosition** () const  
*Return position vector of a node.*
- `KDNodeBase` \* **insert** (const `cv::Mat` &nodePos, const int identity)  
*Insert a new node to the sub-tree.*
- virtual `NodeCompareResult` **nodeCompare** (const `cv::Mat` &queryPosition, const `cv::Mat` &currentPosition, float sqRange, float cosThreshold, int nbDimension) const =0
- void **setNodeid** (int id)  
*Set database entry ID of the node.*

### Static Public Member Functions

- static float **cosDistance** (const float \*const pos1, const float \*const pos2, int dimension)
- static float **sqrDistance** (const float \*const pos1, const float \*const pos2, int dimension)

### Protected Member Functions

- virtual `KDNodeBase` \* **createNode** (const `cv::Mat` &nodePos, const int identity, int splitAxis, int dimension)=0  
*Pure virtual functions to be overridden in child classes.*

## 9.888.1 Member Function Documentation

### 9.888.1.1 createNode()

```
virtual KDNodeBase * Digikam::KDNodeBase::createNode (
    const cv::Mat & nodePos,
    const int identity,
    int splitAxis,
    int dimension ) [protected], [pure virtual]
```

Implemented in [Digikam::KDNodeOpenFace](#), and [Digikam::KDNodeSFace](#).

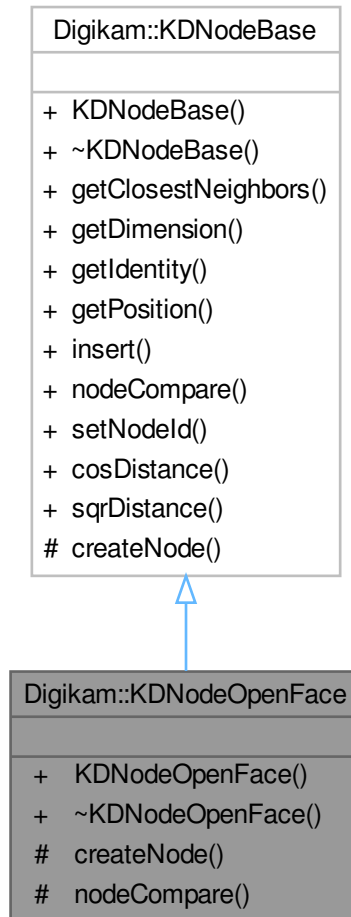
## 9.889 Digikam::KDNodeBase::NodeCompareResult Struct Reference

### Public Attributes

- double **distance1**
- double **distance2**
- bool **result**

## 9.890 Digikam::KNodeOpenFace Class Reference

Inheritance diagram for Digikam::KNodeOpenFace:



### Public Member Functions

- **KNodeOpenFace** (const cv::Mat &nodePos, const int identity, int splitAxis, int dimension)

### Public Member Functions inherited from [Digikam::KNodeBase](#)

- **KNodeBase** (const cv::Mat &nodePos, const int identity, int splitAxis, int dimension)
- double **getClosestNeighbors** (QMap< double, QVector< int > > &neighborList, const cv::Mat &position, float sqRange, float cosThreshold, int maxNbNeighbors) const
  - Return a list of closest neighbors, limited by maxNbNeighbors and sqRange.*
- int **getDimension** ()
- int **getIdentity** ()
  - Return identity of the node.*

- `cv::Mat` **getPosition** () const  
*Return position vector of a node.*
- `KDNodeBase` \* **insert** (const `cv::Mat` &nodePos, const int identity)  
*Insert a new node to the sub-tree.*
- void **setNodeId** (int id)  
*Set database entry ID of the node.*

### Protected Member Functions

- `KDNodeBase` \* **createNode** (const `cv::Mat` &nodePos, const int identity, int splitAxis, int dimension) override  
*Pure virtual functions to be overridden in child classes.*
- `KDNodeBase::NodeCompareResult` **nodeCompare** (const `cv::Mat` &queryPosition, const `cv::Mat` &currentPosition, float sqRange, float cosThreshold, int nbDimension) const override

### Additional Inherited Members

### Static Public Member Functions inherited from `Digikam::KDNodeBase`

- static float **cosDistance** (const float \*const pos1, const float \*const pos2, int dimension)
- static float **sqrDistance** (const float \*const pos1, const float \*const pos2, int dimension)

## 9.890.1 Member Function Documentation

### 9.890.1.1 createNode()

```
KDNodeBase * Digikam::KDNodeOpenFace::createNode (
    const cv::Mat & nodePos,
    const int identity,
    int splitAxis,
    int dimension ) [override], [protected], [virtual]
```

Implements `Digikam::KDNodeBase`.

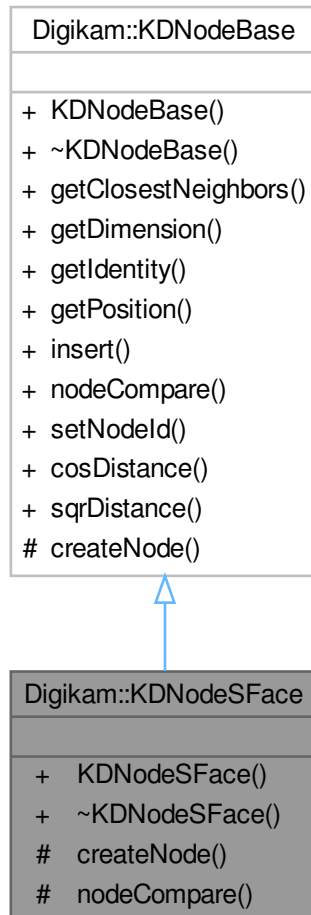
### 9.890.1.2 nodeCompare()

```
KDNodeBase::NodeCompareResult Digikam::KDNodeOpenFace::nodeCompare (
    const cv::Mat & queryPosition,
    const cv::Mat & currentPosition,
    float sqRange,
    float cosThreshold,
    int nbDimension ) const [override], [protected], [virtual]
```

Implements `Digikam::KDNodeBase`.

## 9.891 Digikam::KNodeSFace Class Reference

Inheritance diagram for Digikam::KNodeSFace:



### Public Member Functions

- **KNodeSFace** (const cv::Mat &nodePos, const int identity, int splitAxis, int dimension)

### Public Member Functions inherited from [Digikam::KNodeBase](#)

- **KNodeBase** (const cv::Mat &nodePos, const int identity, int splitAxis, int dimension)
- double **getClosestNeighbors** (QMap< double, QVector< int > > &neighborList, const cv::Mat &position, float sqRange, float cosThreshold, int maxNbNeighbors) const  
*Return a list of closest neighbors, limited by maxNbNeighbors and sqRange.*
- int **getDimension** ()
- int **getIdentity** ()  
*Return identity of the node.*

- cv::Mat **getPosition** () const  
*Return position vector of a node.*
- [KDNNodeBase](#) \* **insert** (const cv::Mat &nodePos, const int identity)  
*Insert a new node to the sub-tree.*
- void **setNodeId** (int id)  
*Set database entry ID of the node.*

### Protected Member Functions

- [KDNNodeBase](#) \* **createNode** (const cv::Mat &nodePos, const int identity, int splitAxis, int dimension) override  
*Pure virtual functions to be overridden in child classes.*
- [KDNNodeBase::NodeCompareResult](#) **nodeCompare** (const cv::Mat &queryPosition, const cv::Mat &currentPosition, float sqRange, float cosThreshold, int nbDimension) const override

### Additional Inherited Members

### Static Public Member Functions inherited from [Digikam::KDNNodeBase](#)

- static float **cosDistance** (const float \*const pos1, const float \*const pos2, int dimension)
- static float **sqrDistance** (const float \*const pos1, const float \*const pos2, int dimension)

## 9.891.1 Member Function Documentation

### 9.891.1.1 createNode()

```
KDNNodeBase * Digikam::KDNNodeSFace::createNode (
    const cv::Mat & nodePos,
    const int identity,
    int splitAxis,
    int dimension ) [override], [protected], [virtual]
```

Implements [Digikam::KDNNodeBase](#).

### 9.891.1.2 nodeCompare()

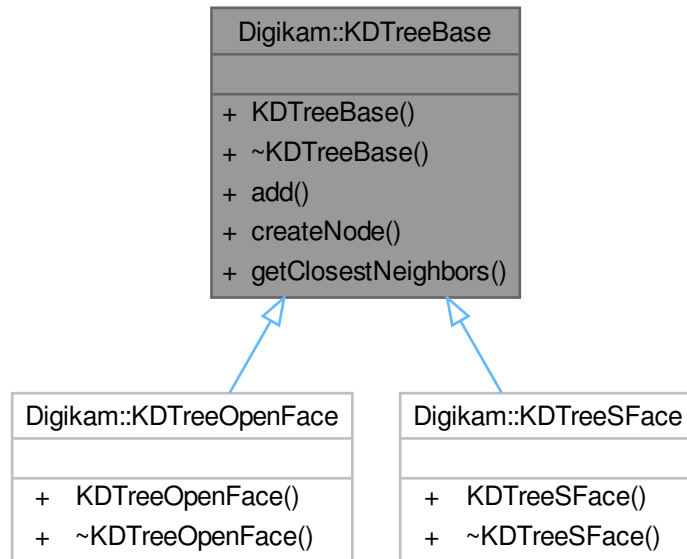
```
KDNNodeBase::NodeCompareResult Digikam::KDNNodeSFace::nodeCompare (
    const cv::Mat & queryPosition,
    const cv::Mat & currentPosition,
    float sqRange,
    float cosThreshold,
    int nbDimension ) const [override], [protected], [virtual]
```

Implements [Digikam::KDNNodeBase](#).



## 9.892 Digikam::KdTreeBase Class Reference

Inheritance diagram for Digikam::KdTreeBase:



### Public Member Functions

- [KdTreeBase](#) (int dim, int kdTreeThreshold=KDTREE\_MAP\_THRESHOLD)  
*Constructor of the class implementing the KD-Tree for vector space partitioning.*
- virtual [KdNodeBase](#) \* [add](#) (const cv::Mat &position, const int identity)  
*add new node to KD-Tree*
- virtual [KdNodeBase](#) \* [createNode](#) (const cv::Mat &nodePos, const int identity, int splitAxis, int dimension)=0  
*create a new node*
- virtual QMap< double, QVector< int > > [getClosestNeighbors](#) (const cv::Mat &position, float sqRange, int maxNbNeighbors) const

### 9.892.1 Constructor & Destructor Documentation

#### 9.892.1.1 KdTreeBase()

```

Digikam::KdTreeBase::KdTreeBase (
    int dim,
    int kdTreeThreshold = KDTREE_MAP_THRESHOLD ) [explicit]
  
```

#### Parameters

<i>dim</i>	The dimension of the tree.
<i>kdTreeThreshold</i>	The KD-Tree threshold. Above this value, we start using the KD-Tree instead of the vector. If the vector grows to default KDTREE_MAP_THRESHOLD items, start using the KDTree.

**Note**

Due to sparse data density in the tree, we initially use a vector of nodes to compare the target to the samples once we have achieved a suitable data density we delete the vector (but not the nodes) and begin using the tree.

Using this to compare brute force vs kdtree performance due to sparse data in k-dimensions (128 dimensions for face features).

**9.892.2 Member Function Documentation****9.892.2.1 add()**

```
KDNodeBase * Digikam::KDTreeBase::add (
    const cv::Mat & position,
    const int identity ) [virtual]
```

**Parameters**

<i>position</i>	K-dimension vector
<i>identity</i>	identity of this face vector

**Returns**

the KD-Tree node base instance

**9.892.2.2 createNode()**

```
virtual KDNodeBase * Digikam::KDTreeBase::createNode (
    const cv::Mat & nodePos,
    const int identity,
    int splitAxis,
    int dimension ) [pure virtual]
```

**Parameters**

<i>nodePos</i>	extracted face vectors
<i>identity</i>	identity of this face vector
<i>splitAxis</i>	current axis/dimension of the vector
<i>dimension</i>	number of dimensions (usually 128)

**Returns**

the KD-Tree node base instance

**9.892.2.3 getClosestNeighbors()**

```
QMap< double, QVector< int > > Digikam::KDTreeBase::getClosestNeighbors (
    const cv::Mat & position,
```

```
float sqRange,
int maxNbNeighbors ) const [virtual]
```

**Returns**

Map of N-nearest neighbors, sorted by distance

**9.893 Digikam::KdTreeOpenFace Class Reference**

Inheritance diagram for Digikam::KdTreeOpenFace:

**Public Member Functions**

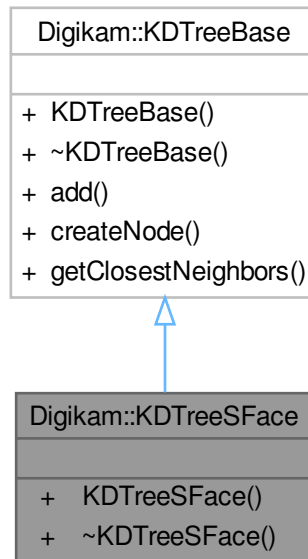
- **KdTreeOpenFace** (int dim, int threshold=KDTREE\_MAP\_THRESHOLD)

**Public Member Functions inherited from [Digikam::KdTreeBase](#)**

- [KdTreeBase](#) (int dim, int kdtreeThreshold=KDTREE\_MAP\_THRESHOLD)  
*Constructor of the class implementing the KD-Tree for vector space partitioning.*
- virtual [KdNodeBase](#) \* **add** (const cv::Mat &position, const int identity)  
*add new node to KD-Tree*
- virtual QMap< double, QVector< int > > **getClosestNeighbors** (const cv::Mat &position, float sqRange, int maxNbNeighbors) const

## 9.894 Digikam::KDTreeSFace Class Reference

Inheritance diagram for Digikam::KDTreeSFace:



### Public Member Functions

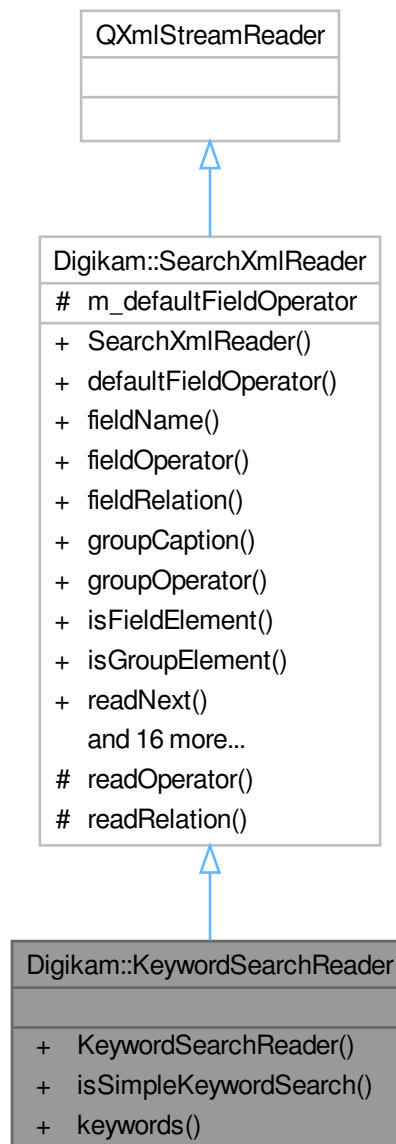
- **KDTreeSFace** (int dim, int threshold=KDTREE\_MAP\_THRESHOLD)

### Public Member Functions inherited from [Digikam::KDTreeBase](#)

- [KDTreeBase](#) (int dim, int kdtreeThreshold=KDTREE\_MAP\_THRESHOLD)  
*Constructor of the class implementing the KD-Tree for vector space partitioning.*
- virtual [KDNodeBase](#) \* **add** (const cv::Mat &position, const int identity)  
*add new node to KD-Tree*
- virtual QMap< double, QVector< int > > **getClosestNeighbors** (const cv::Mat &position, float sqRange, int maxNbNeighbors) const

## 9.895 Digikam::KeywordSearchReader Class Reference

Inheritance diagram for Digikam::KeywordSearchReader:



### Public Member Functions

- **KeywordSearchReader** (const QString &xml)
- bool **isSimpleKeywordSearch** ()  
*Checks if the XML is a simple keyword search, compatible with [keywords\(\)](#).*
- QStringList **keywords** ()  
*Returns the keywords from this search, merged in a list.*

## Public Member Functions inherited from [Digikam::SearchXmlReader](#)

- **SearchXmlReader** (const QString &xml)
- SearchXml::Operator [defaultFieldOperator](#) () const  
*Returns the default field operator.*
- QString **fieldName** () const
- SearchXml::Operator [fieldOperator](#) () const  
*Returns the field attributes.*
- SearchXml::Relation **fieldRelation** () const
- QString [groupCaption](#) () const  
*Returns the (optional) group caption.*
- SearchXml::Operator [groupOperator](#) () const  
*Returns the group operator.*
- bool **isFieldElement** () const  
*Returns if the current element is a field element (start or end element).*
- bool **isGroupElement** () const  
*Returns if the current element is a group element (start or end element).*
- SearchXml::Element [readNext](#) ()  
*Continue parsing the document.*
- void **readToEndOfElement** ()  
*General helper method: Reads XML until the end element of the current start element in reached.*
- void **readToFirstField** ()  
*General helper method: Reads XML until the first field of the next or first found group is reached.*
- bool [readToStartOfElement](#) (const QString &name)  
*General helper method: Reads XML a start element with the given name is found.*
- QString [value](#) ()  
*Returns the field values.*
- QDateTime **valueToDateTime** ()
- QList< QDateTime > **valueToDateTimeList** ()
- double **valueToDouble** ()
- QList< double > **valueToDoubleList** ()
- QList< double > **valueToDoubleOrDoubleList** ()
- int **valueToInt** ()
- QList< int > **valueToIntList** ()
- QList< int > **valueToIntOrIntList** ()
- qlonglong **valueToLongLong** ()
- QList< qlonglong > **valueToLongLongList** ()
- QStringList **valueToStringList** ()
- QList< QString > **valueToStringOrStringList** ()

## Additional Inherited Members

## Protected Member Functions inherited from [Digikam::SearchXmlReader](#)

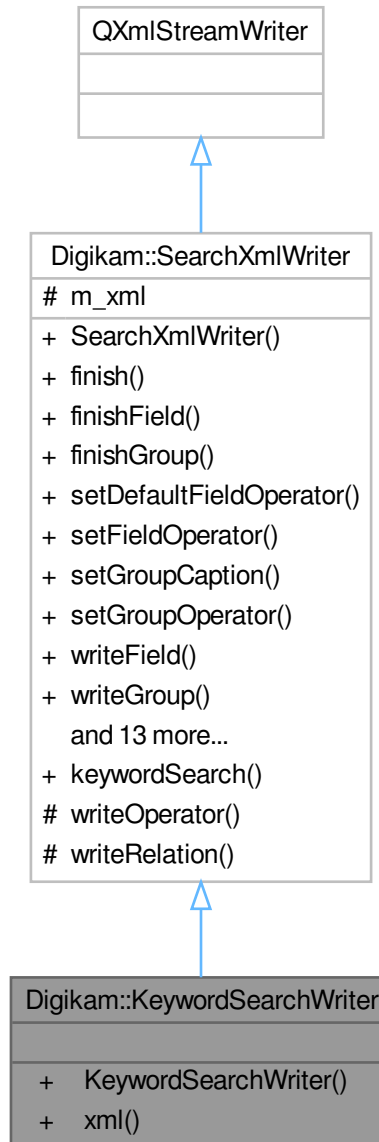
- SearchXml::Operator **readOperator** (const QString &, SearchXml::Operator) const
- SearchXml::Relation **readRelation** (const QString &, SearchXml::Relation) const

## Protected Attributes inherited from [Digikam::SearchXmlReader](#)

- SearchXml::Operator **m\_defaultFieldOperator**

## 9.896 Digikam::KeywordSearchWriter Class Reference

Inheritance diagram for Digikam::KeywordSearchWriter:



### Public Member Functions

- `QString xml (const QStringList &keywordList)`

### Public Member Functions inherited from [Digikam::SearchXmlWriter](#)

- `SearchXmlWriter ()`

Note that [SearchXmlWriter](#) and [SearchXmlGroupWriter](#) rely on you calling the methods following the restrictions set by the documentation; Otherwise you will not produce the desired output.

- void [finish](#) ()  
*Finish the XML.*
- void [finishField](#) ()  
*Finish writing the current field.*
- void [finishGroup](#) ()  
*Finish the current group.*
- void [setDefaultFieldOperator](#) (SearchXml::Operator op)  
*Sets the default operator for fields in this group "(field1 AND field2 AND ... fieldn)".*
- void [setFieldOperator](#) (SearchXml::Operator op)  
*Adds an optional operator overriding the default field operator of the group.*
- void [setGroupCaption](#) (const QString &caption)  
*Sets an optional caption.*
- void [setGroupOperator](#) (SearchXml::Operator op)  
*Sets the operator applied to the group as a whole "OR (field1 ... fieldn)".*
- void [writeField](#) (const QString &name, SearchXml::Relation relation)  
*Adds a new field with the given name (entity) and relation, "Rating less than ...".*
- void [writeGroup](#) ()  
*Adds a group.*
- void [writeValue](#) (const QDateTime &dateTime)
- void [writeValue](#) (const QList< double > &valueList, int precision=8)
- void [writeValue](#) (const QList< float > &valueList, int precision=6)
- void [writeValue](#) (const QList< int > &valueList)
- void [writeValue](#) (const QList< QDateTime > &valueList)
- void [writeValue](#) (const QList< qlonglong > &valueList)
- void [writeValue](#) (const QString &value)  
*Adds the value, "4" in the case of "Rating less than 4".*
- void [writeValue](#) (const QStringList &valueList)
- void [writeValue](#) (double value, int precision=8)
- void [writeValue](#) (float value, int precision=6)
- void [writeValue](#) (int value)
- void [writeValue](#) (qlonglong value)
- QString [xml](#) () const  
*Get the created XML.*

### Additional Inherited Members

### Static Public Member Functions inherited from [Digikam::SearchXmlWriter](#)

- static QString [keywordSearch](#) (const QString &keyword)  
*Returns ready-made XML for a query of type "keyword" with the specified text as keyword.*

### Protected Member Functions inherited from [Digikam::SearchXmlWriter](#)

- void [writeOperator](#) (const QString &, SearchXml::Operator)
- void [writeRelation](#) (const QString &, SearchXml::Relation)

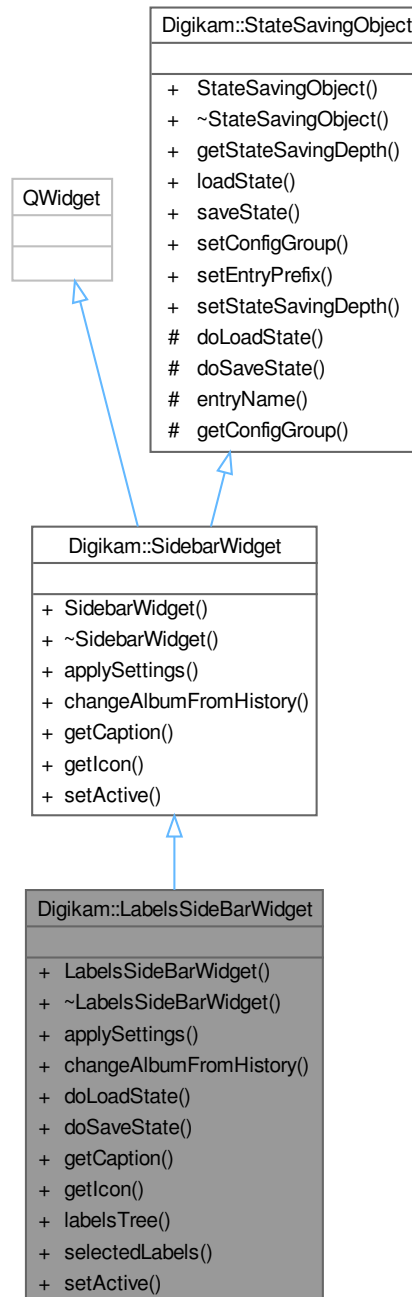
### Protected Attributes inherited from [Digikam::SearchXmlWriter](#)

- QString [m\\_xml](#)



## 9.897 Digikam::LabelsSideBarWidget Class Reference

Inheritance diagram for Digikam::LabelsSideBarWidget:



### Public Member Functions

- **LabelsSideBarWidget** (`QWidget *const parent`)
- void `applySettings()` override

- This method is invoked when the application settings should be (re-) applied to this widget.*

  - void [changeAlbumFromHistory](#) (const QList< Album \* > &album) override
    - This is called on this widget when the history requires to move back to the specified album.*
  - void [doLoadState](#) () override
    - Implement this hook method for state loading.*
  - void [doSaveState](#) () override
    - Implement this hook method for state saving.*
  - const QString [getCaption](#) () override
    - Must be implemented to return the title of this sidebar's tab.*
  - const QIcon [getIcon](#) () override
    - Must be implemented and return the icon that shall be visible for this sidebar widget.*
  - [LabelsTreeView](#) \* [labelsTree](#) ()
  - QHash< LabelsTreeView::Labels, QList< int > > [selectedLabels](#) ()
  - void [setActive](#) (bool active) override
    - This method is called if the visible sidebar widget is changed.*

## Public Member Functions inherited from [Digikam::SidebarWidget](#)

- [SidebarWidget](#) (QWidget \*const parent)
  - Constructor.*
- [~SidebarWidget](#) () override=default
  - Destructor.*

## Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (QObject \*const host)
  - Constructor.*
- virtual [~StateSavingObject](#) ()
  - Destructor.*
- [StateSavingDepth](#) [getStateSavingDepth](#) () const
  - Returns the depth used for state saving or loading.*
- void [loadState](#) ()
  - Invokes loading the class' state.*
- void [saveState](#) ()
  - Invokes saving the class' state.*
- virtual void [setConfigGroup](#) (const KConfigGroup &group)
  - Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void [setEntryPrefix](#) (const QString &prefix)
  - Define a prefix that will be used for every entry in the config group.*
- void [setStateSavingDepth](#) (const [StateSavingDepth](#) depth)
  - Sets the depth used for state saving or loading.*

## Additional Inherited Members

## Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }
  - This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

## Signals inherited from [Digikam::SidebarWidget](#)

- void **requestActiveTab** ([SidebarWidget](#) \*)  
*This signal can be emitted if this sidebar widget wants to be the one that is active.*
- void **signalNotificationError** (const QString &message, int type)  
*To dispatch error message to temporized pop-up notification widget hosted with icon-view.*

## Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- QString **entryName** (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup **getConfigGroup** () const  
*Returns the config group that must be used for state saving and loading.*

### 9.897.1 Member Function Documentation

#### 9.897.1.1 **applySettings()**

```
void Digikam::LabelsSideBarWidget::applySettings ( ) [override], [virtual]
```

Implements [Digikam::SidebarWidget](#).

#### 9.897.1.2 **changeAlbumFromHistory()**

```
void Digikam::LabelsSideBarWidget::changeAlbumFromHistory (
    const QList< Album * > & album ) [override], [virtual]
```

Implements [Digikam::SidebarWidget](#).

#### 9.897.1.3 **doLoadState()**

```
void Digikam::LabelsSideBarWidget::doLoadState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

#### 9.897.1.4 **doSaveState()**

```
void Digikam::LabelsSideBarWidget::doSaveState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

### 9.897.1.5 `getCaption()`

```
const QString Digikam::LabelsSideBarWidget::getCaption ( ) [override], [virtual]
```

#### Returns

localized title string

Implements [Digikam::SidebarWidget](#).

### 9.897.1.6 `getIcon()`

```
const QIcon Digikam::LabelsSideBarWidget::getIcon ( ) [override], [virtual]
```

#### Returns

pixmap icon

Implements [Digikam::SidebarWidget](#).

### 9.897.1.7 `setActive()`

```
void Digikam::LabelsSideBarWidget::setActive (
    bool active ) [override], [virtual]
```

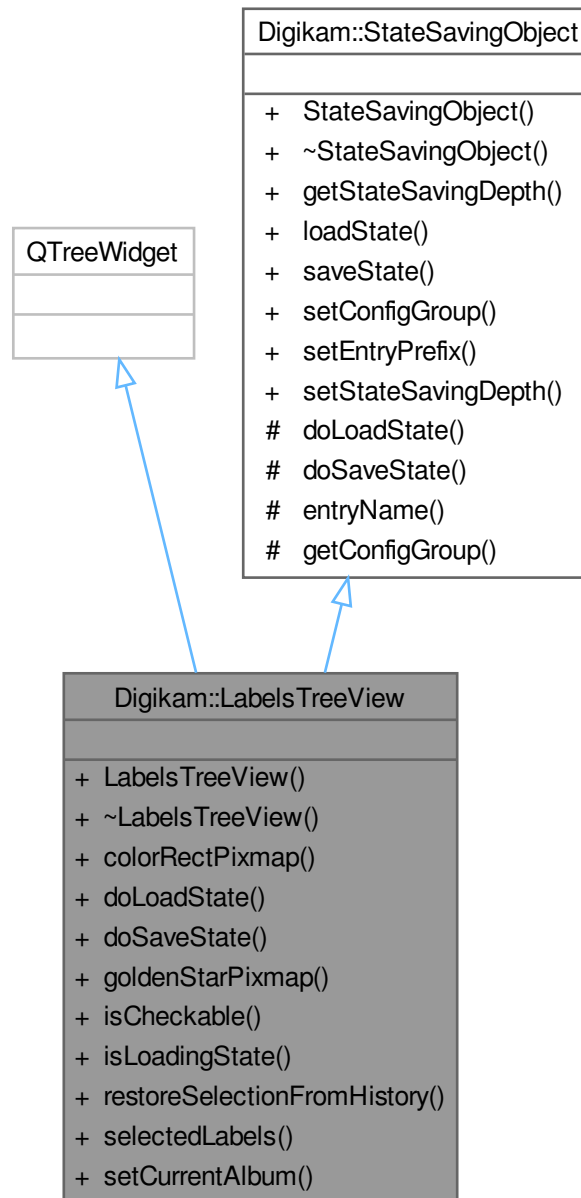
#### Parameters

<i>active</i>	if true, this widget is the new active widget, if false another widget is active
---------------	--

Implements [Digikam::SidebarWidget](#).

## 9.898 Digikam::LabelsTreeView Class Reference

Inheritance diagram for Digikam::LabelsTreeView:



### Public Types

- enum **Labels** { **Ratings** = 0 , **Picks** , **Colors** }

### Public Types inherited from Digikam::StateSavingObject

- enum **StateSavingDepth** { **INSTANCE** , **DIRECT\_CHILDREN** , **RECURSIVE** }

*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

## Signals

- void **signalSetCurrentAlbum** ()

## Public Member Functions

- **LabelsTreeView** (QWidget \*const parent=nullptr, bool setCheckable=false)
- QPixmap **colorRectPixmap** (const QColor &color) const  
*Creates a 30\*30 rectangular pixmap with specific color.*
- void **doLoadState** () override  
*Loading and saving state function inherited from [StateSavingObject](#).*
- void **doSaveState** () override  
*Implement this hook method for state saving.*
- QPixmap **goldenStarPixmap** (bool fillin=true) const
- bool **isCheckable** () const
- bool **isLoadingState** () const
- void **restoreSelectionFromHistory** (QHash< Labels, QList< int > > neededLabels)  
*Restores the selection state from the [AlbumHistory](#) class.*
- QHash< Labels, QList< int > > **selectedLabels** ()  
*Provide the current selection from the tree-view hierarchy.*
- void **setCurrentAlbum** ()  
*Emits a signal to the search handler to set the Current album from currently selected labels.*

## Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (QObject \*const host)  
*Constructor.*
- virtual ~**StateSavingObject** ()  
*Destructor.*
- [StateSavingDepth](#) **getStateSavingDepth** () const  
*Returns the depth used for state saving or loading.*
- void **loadState** ()  
*Invokes loading the class' state.*
- void **saveState** ()  
*Invokes saving the class' state.*
- virtual void **setConfigGroup** (const KConfigGroup &group)  
*Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void **setEntryPrefix** (const QString &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void **setStateSavingDepth** (const [StateSavingDepth](#) depth)  
*Sets the depth used for state saving or loading.*

## Additional Inherited Members

## Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- QString **entryName** (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup **getConfigGroup** () const  
*Returns the config group that must be used for state saving and loading.*

## 9.898.1 Member Function Documentation

### 9.898.1.1 colorRectPixmap()

```
QPixmap Digikam::LabelsTreeView::colorRectPixmap (
    const QColor & color ) const
```

#### Parameters

<i>color</i>	wanted to be set
--------------	------------------

#### Returns

pixmap has a rectangle filled with the color

### 9.898.1.2 doLoadState()

```
void Digikam::LabelsTreeView::doLoadState ( ) [override], [virtual]
```

Implements [Digikam::StateSavingObject](#).

### 9.898.1.3 doSaveState()

```
void Digikam::LabelsTreeView::doSaveState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

### 9.898.1.4 goldenStarPixmap()

```
QPixmap Digikam::LabelsTreeView::goldenStarPixmap (
    bool fillin = true ) const
```

#### Returns

a QPixmap of a 30\*30 pixels golden star used for rating and widget icon

### 9.898.1.5 isCheckable()

```
bool Digikam::LabelsTreeView::isCheckable ( ) const
```

#### Returns

true if the tree widget is checkable and false if not

### 9.898.1.6 isLoadingState()

```
bool Digikam::LabelsTreeView::isLoadingState ( ) const
```

#### Returns

true if Loading state function is running

### 9.898.1.7 restoreSelectionFromHistory()

```
void Digikam::LabelsTreeView::restoreSelectionFromHistory (
    QHash< Labels, QList< int > > neededLabels )
```

#### Parameters

<i>neededLabels</i>	is a QHash to restore the selection from it, the hash is formatted just like the hash generated from
---------------------	--

#### See also

[selectedLabels\(\)](#)

### 9.898.1.8 selectedLabels()

```
QHash< LabelsTreeView::Labels, QList< int > > Digikam::LabelsTreeView::selectedLabels ( )
```

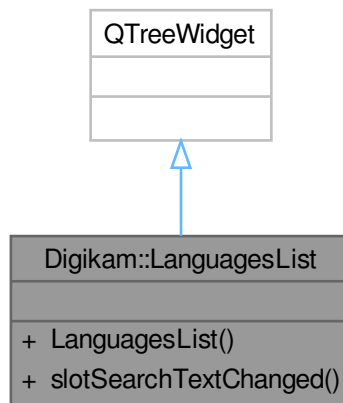


**Returns**

a QMap with three keys: "Ratings", "Picks", and "Colors", every key dedicated to an int list which holds the rows selected

## 9.899 Digikam::LanguagesList Class Reference

Inheritance diagram for Digikam::LanguagesList:

**Public Slots**

- void **slotSearchTextChanged** (const [SearchTextSettings](#) &settings)

**Signals**

- void **signalSearchResult** (int)

**Public Member Functions**

- **LanguagesList** (QWidget \*const parent)

## 9.900 Digikam::LcmsLock Class Reference

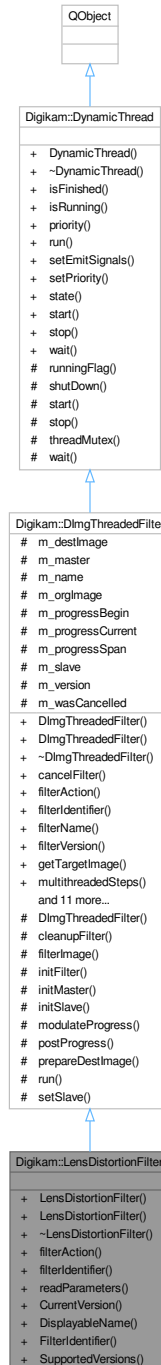
**Public Member Functions**

- **LcmsLock** ()

Obtain an [LcmsLock](#) if you access not clearly thread-safe LittleCMS methods.

## 9.901 Digikam::LensDistortionFilter Class Reference

Inheritance diagram for Digikam::LensDistortionFilter:



### Public Member Functions

- **LensDistortionFilter** (`DImg *const orgImage`, `QObject *const parent=nullptr`, `double main=0.0`, `double edge=0.0`, `double rescale=0.0`, `double brighten=0.0`, `int center_x=0`, `int center_y=0`)

- **LensDistortionFilter** (QObject \*const parent=nullptr)
- **FilterAction filterAction** () override  
*Returns the action description corresponding to currently set options.*
- **QString filterIdentifier** () const override  
*Return the identifier for this filter in the image history.*
- void **readParameters** (const **FilterAction** &action) override

## Public Member Functions inherited from Digikam::DImgThreadedFilter

- **DImgThreadedFilter** (DImg \*const orgImage, QObject \*const parent, const QString &name=QString())  
*Constructs a filter with all arguments (ready to use).*
- **DImgThreadedFilter** (QObject \*const parent=nullptr, const QString &name=QString())  
*Constructs a filter without argument.*
- virtual void **cancelFilter** ()  
*Cancel the threaded computation.*
- const QString & **filterName** ()
- int **filterVersion** () const
- **DImg getTargetImage** ()
- QList< int > **multithreadedSteps** (int stop, int start=0) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool **parametersSuccessfullyRead** () const  
*Optional: error handling for readParameters.*
- virtual QString **readParametersError** (const **FilterAction** &actionThatFailed) const
- void **setFilterName** (const QString &name)
- void **setFilterVersion** (int version)  
*Replaying a filter action: Set the filter version.*
- void **setOriginalImage** (const **DImg** &orgImage)
- void **setupAndStartDirectly** (const **DImg** &orgImage, **DImgThreadedFilter** \*const master, int progress←Begin=0, int progressEnd=100)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void **setupFilter** (const **DImg** &orgImage)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void **startFilter** ()  
*Start the threaded computation.*
- virtual void **startFilterDirectly** ()  
*Start computation of this filter, directly in this thread.*
- virtual QList< int > **supportedVersions** () const

## Public Member Functions inherited from Digikam::DynamicThread

- **DynamicThread** (QObject \*const parent=nullptr)  
*This class extends QRunnable, so you have to reimplement virtual void run().*
- **~DynamicThread** () override  
*The destructor calls stop() and wait(), but if you, in your destructor, delete any data that is accessed by your run() method, you must call stop() and wait() before yourself.*
- bool **isFinished** () const
- bool **isRunning** () const
- QThread::Priority **priority** () const
- void **setEmitSignals** (bool emitThem)
- void **setPriority** (QThread::Priority priority)  
*Sets the priority for this dynamic thread.*
- State **state** () const

### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from Digikam::DImgThreadedFilter

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from Digikam::DynamicThread

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from Digikam::DImgThreadedFilter

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.901.1 Member Function Documentation

### 9.901.1.1 filterAction()

`FilterAction` Digikam::LensDistortionFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.901.1.2 filterIdentifier()

`QString` Digikam::LensDistortionFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.901.1.3 readParameters()

```
void Digikam::LensDistortionFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

## 9.902 Digikam::LensDistortionPixelAccess Class Reference

[LensDistortionPixelAccess](#) class: solving the eternal problem: random, cubic-interpolated, sub-pixel coordinate access to an image.

### Public Member Functions

- `LensDistortionPixelAccess` (`DImg *srcImage`)
- void `pixelAccessGetCubic` (double srcX, double srcY, double brighten, uchar \*dst)

### Protected Member Functions

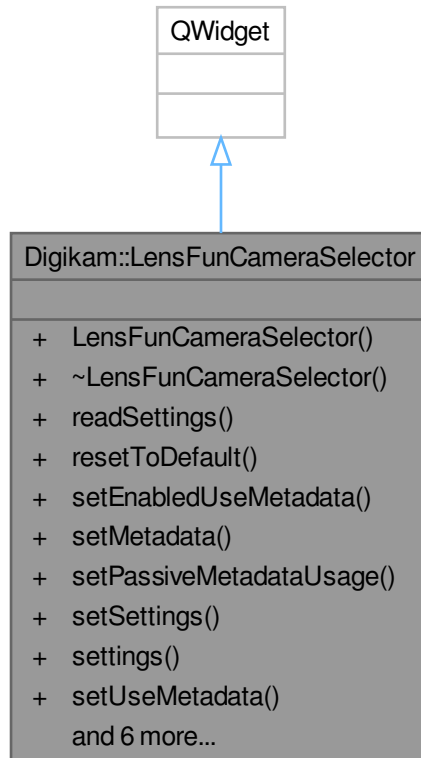
- void `cubicInterpolate` (uchar \*src, int rowStride, uchar \*dst, bool sixteenBit, double dx, double dy, double brighten)
- uchar \* `pixelAccessAddress` (int i, int j)
- void `pixelAccessDoEdge` (int i, int j)
- void `pixelAccessReposition` (int xInt, int yInt)
- void `pixelAccessSelectRegion` (int n)

### 9.902.1 Detailed Description

Assuming that accesses are at least slightly coherent, [LensDistortionPixelAccess](#) keeps `LensDistortionPixelAccessRegions` buffers, each containing a `LensDistortionPixelAccessWidth` x `LensDistortionPixelAccessHeight` region of pixels. `Buffer[0]` is always checked first, so move the last accessed region into that position. When a request arrives which is outside all the regions, get a new region. The new region is placed so that the requested pixel is positioned at [`LensDistortionPixelAccessXOffset`, `LensDistortionPixelAccessYOffset`] in the region.

## 9.903 Digikam::LensFunCameraSelector Class Reference

Inheritance diagram for Digikam::LensFunCameraSelector:



### Public Types

- typedef `QMap< QString, QString >` **Device**

### Signals

- void **signalLensSettingsChanged** ()

### Public Member Functions

- **LensFunCameraSelector** (`QWidget *const parent=nullptr`)
- void **readSettings** (`const KConfigGroup &group`)
- void **resetToDefault** ()
- void **setEnabledUseMetadata** (`bool b`)
- void **setMetadata** (`const MetaEngineData &`)
- void **setPassiveMetadataUsage** (`bool b`)

*Special mode used with BQM which processes multiple items at the same time.*

- void **setSettings** (const [LensFunContainer](#) &settings)
- [LensFunContainer](#) **settings** ()
- void **setUseMetadata** (bool b)
- bool **supportsCCA** () const
- bool **supportsDistortion** () const
- bool **supportsGeometry** () const
- bool **supportsVig** () const
- bool **useMetadata** () const
- void **writeSettings** (KConfigGroup &group)

## 9.904 Digikam::LensFunContainer Class Reference

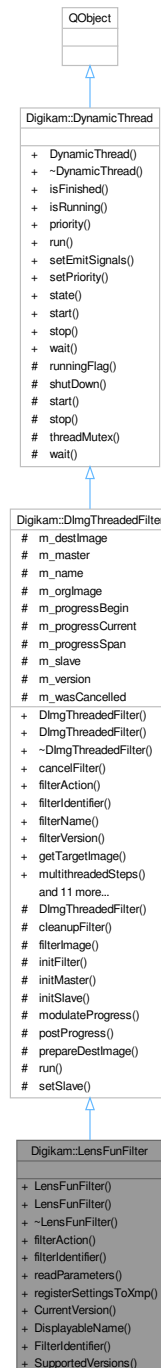
### Public Attributes

- double **aperture** = -1.0
- QString **cameraMake**
- QString **cameraModel**
- double **cropFactor** = -1.0
- bool **filterCCA** = true  
*Chromatic Aberration Corrections.*
- bool **filterDST** = true  
*Distortion Corrections.*
- bool **filterGEO** = true  
*Geometry Corrections.*
- bool **filterVIG** = true  
*Vignetting Corrections.*
- double **focalLength** = -1.0
- QString **lensModel**
- double **subjectDistance** = -1.0



## 9.905 Digikam::LensFunFilter Class Reference

Inheritance diagram for Digikam::LensFunFilter:



### Public Member Functions

- **LensFunFilter** (`DImg *const origImage`, `QObject *const parent`, `const LensFunContainer &settings`)
- **LensFunFilter** (`QObject *const parent=nullptr`)

- [FilterAction filterAction](#) () override  
*Returns the action description corresponding to currently set options.*
- [QString filterIdentifier](#) () const override  
*Return the identifier for this filter in the image history.*
- void [readParameters](#) (const [FilterAction](#) &action) override
- bool [registerSettingsToXmp](#) ([MetaEngineData](#) &data) const

## Public Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImg](#) \*const orgImage, [QObject](#) \*const parent, const [QString](#) &name=[QString](#)())  
*Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter](#) ([QObject](#) \*const parent=nullptr, const [QString](#) &name=[QString](#)())  
*Constructs a filter without argument.*
- virtual void [cancelFilter](#) ()  
*Cancel the threaded computation.*
- const [QString](#) & [filterName](#) ()
- int [filterVersion](#) () const
- [DImg](#) [getTargetImage](#) ()
- [QList](#)< int > [multithreadedSteps](#) (int stop, int start=0) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool [parametersSuccessfullyRead](#) () const  
*Optional: error handling for readParameters.*
- virtual [QString](#) [readParametersError](#) (const [FilterAction](#) &actionThatFailed) const
- void [setFilterName](#) (const [QString](#) &name)
- void [setFilterVersion](#) (int version)  
*Replaying a filter action: Set the filter version.*
- void [setOriginalImage](#) (const [DImg](#) &orgImage)
- void [setupAndStartDirectly](#) (const [DImg](#) &orgImage, [DImgThreadedFilter](#) \*const master, int progress←Begin=0, int progressEnd=100)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void [setupFilter](#) (const [DImg](#) &orgImage)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void [startFilter](#) ()  
*Start the threaded computation.*
- virtual void [startFilterDirectly](#) ()  
*Start computation of this filter, directly in this thread.*
- virtual [QList](#)< int > [supportedVersions](#) () const

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread](#) ([QObject](#) \*const parent=nullptr)  
*This class extends [QRunnable](#), so you have to reimplement virtual void [run\(\)](#).*
- [~DynamicThread](#) () override  
*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool [isFinished](#) () const
- bool [isRunning](#) () const
- [QThread::Priority](#) [priority](#) () const
- void [setEmitSignals](#) (bool emitThem)
- void [setPriority](#) ([QThread::Priority](#) priority)  
*Sets the priority for this dynamic thread.*
- State [state](#) () const

### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from [Digikam::DImgThreadedFilter](#)

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.905.1 Member Function Documentation

### 9.905.1.1 filterAction()

`FilterAction` Digikam::LensFunFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.905.1.2 filterIdentifier()

`QString` Digikam::LensFunFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.905.1.3 readParameters()

```
void Digikam::LensFunFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

## 9.906 Digikam::LensFunface Class Reference

### Public Types

- typedef const IfCamera \* **DevicePtr**
- typedef QList< LensPtr > **LensList**
- typedef const IfLens \* **LensPtr**
- enum **MetadataMatch** { **MetadataUnavailable** = -2 , **MetadataNoMatch** = -1 , **MetadataPartialMatch** = 0 , **MetadataExactMatch** = 1 }

### Public Member Functions

- DevicePtr **findCamera** (const QString &make, const QString &model) const
- MetadataMatch **findFromMetadata** (const [DMetadata](#) \*const meta)
- LensPtr **findLens** (const QString &model) const
- QString **lensDescription** ( ) const
 

*Return Lens string description found in metadata.*
- const IfCamera \*const \* **lensFunCameras** ( ) const
- IfDatabase \* **lensFunDataBase** ( ) const
- QString **makeDescription** ( ) const
 

*Return Camera maker string description found in metadata.*
- QString **modelDescription** ( ) const
 

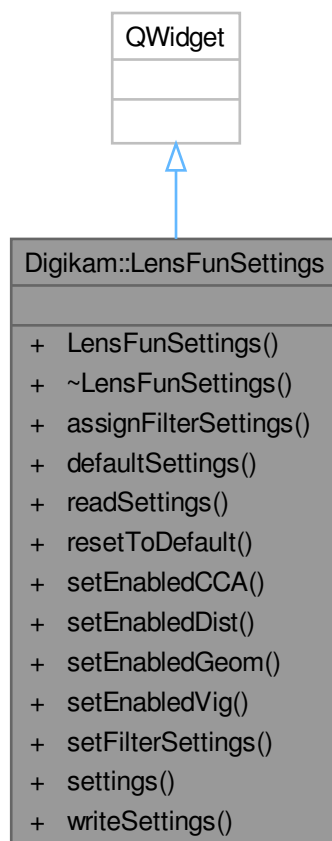
*Return Camera model string description found in metadata.*
- void **setFilterSettings** (const [LensFunContainer](#) &other)
- void **setSettings** (const [LensFunContainer](#) &other)
- [LensFunContainer](#) **settings** ( ) const
- void **setUsedCamera** (DevicePtr cam)
- void **setUsedLens** (LensPtr lens)
- bool **supportsCCA** ( ) const
- bool **supportsDistortion** ( ) const
- bool **supportsGeometry** ( ) const
- bool **supportsVig** ( ) const
- DevicePtr **usedCamera** ( ) const
- LensPtr **usedLens** ( ) const

### Static Public Member Functions

- static QString **lensFunVersion** ()

## 9.907 Digikam::LensFunSettings Class Reference

Inheritance diagram for Digikam::LensFunSettings:



### Signals

- void **signalSettingsChanged** ()

### Public Member Functions

- **LensFunSettings** (QWidget \*const parent=nullptr)
- void **assignFilterSettings** ([LensFunContainer](#) &prm)
- [LensFunContainer](#) **defaultSettings** () const
- void **readSettings** (const KConfigGroup &group)

- void **resetToDefault** ()
- void **setEnabledCCA** (bool b)
- void **setEnabledDist** (bool b)
- void **setEnabledGeom** (bool b)
- void **setEnabledVig** (bool b)
- void **setFilterSettings** (const [LensFunContainer](#) &settings)
- [LensFunContainer](#) **settings** () const
- void **writeSettings** (KConfigGroup &group)

## 9.908 Digikam::LevelsContainer Class Reference

### Public Attributes

- double **gamma** [5] = { 1.0 }
- int **hInput** [5] = { 65535 }
- int **hOutput** [5] = { 65535 }
- int **lInput** [5] = { 0 }
- int **lOutput** [5] = { 0 }

## 9.909 Digikam::LevelsFilter Class Reference

Inheritance diagram for Digikam::LevelsFilter:



### Public Member Functions

- **LevelsFilter** (const [LevelsContainer](#) &settings, [DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, [DImg](#) &destImage, int progressBegin=0, int progressEnd=100)



- **LevelsFilter** (*DImg* \*const orgImage, *QObject* \*const parent=nullptr, const [LevelsContainer](#) &settings=[LevelsContainer](#)())
- **LevelsFilter** (*QObject* \*const parent=nullptr)
- **FilterAction** [filterAction](#) () override
 

*Returns the action description corresponding to currently set options.*
- *QString* [filterIdentifier](#) () const override
 

*Return the identifier for this filter in the image history.*
- void [readParameters](#) (const [FilterAction](#) &action) override

## Public Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) (*DImg* \*const orgImage, *QObject* \*const parent, const *QString* &name=*QString*())
 

*Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter](#) (*QObject* \*const parent=nullptr, const *QString* &name=*QString*())
 

*Constructs a filter without argument.*
- virtual void [cancelFilter](#) ()
 

*Cancel the threaded computation.*
- const *QString* & **filterName** ()
- int **filterVersion** () const
- [DImg](#) **getTargetImage** ()
- *QList*< int > [multithreadedSteps](#) (int stop, int start=0) const
 

*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool [parametersSuccessfullyRead](#) () const
 

*Optional: error handling for readParameters.*
- virtual *QString* **readParametersError** (const [FilterAction](#) &actionThatFailed) const
- void **setFilterName** (const *QString* &name)
- void [setFilterVersion](#) (int version)
 

*Replaying a filter action: Set the filter version.*
- void **setOriginalImage** (const [DImg](#) &orgImage)
- void **setupAndStartDirectly** (const [DImg](#) &orgImage, [DImgThreadedFilter](#) \*const master, int progress←Begin=0, int progressEnd=100)
 

*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void [setupFilter](#) (const [DImg](#) &orgImage)
 

*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void **startFilter** ()
 

*Start the threaded computation.*
- virtual void **startFilterDirectly** ()
 

*Start computation of this filter, directly in this thread.*
- virtual *QList*< int > **supportedVersions** () const

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread](#) (*QObject* \*const parent=nullptr)
 

*This class extends *QRunnable*, so you have to reimplement virtual void [run\(\)](#).*
- ~**DynamicThread** () override
 

*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool **isFinished** () const
- bool **isRunning** () const
- *QThread*::Priority **priority** () const
- void **setEmitSignals** (bool emitThem)
- void [setPriority](#) (*QThread*::Priority priority)
 

*Sets the priority for this dynamic thread.*
- State **state** () const

### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from Digikam::DImgThreadedFilter

- **DImgThreadedFilter** (**DImgThreadedFilter** \*const master, const **DImg** &orgImage, const **DImg** &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void **cleanupFilter** ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void **initFilter** ()  
*Start filter operation before threaded method.*
- void **initMaster** ()
- void **initSlave** (**DImgThreadedFilter** \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int **modulateProgress** (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void **postProgress** (int progress)  
*Emit progress info.*
- virtual void **prepareDestImage** ()
- void **run** () override  
*List of threaded operations by filter.*
- void **setSlave** (**DImgThreadedFilter** \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from Digikam::DynamicThread

- bool **runningFlag** () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void **shutDown** ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void **start** (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void **stop** (const QMutexLocker< QMutex > &locker)
- QMutex \* **threadMutex** () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void **wait** (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from Digikam::DImgThreadedFilter

- **DImg m\_destImage**  
*Output image data.*
- **DImgThreadedFilter \* m\_master** = nullptr  
*The master of this slave filter.*
- **QString m\_name**  
*Filter name.*
- **DImg m\_orgImage**  
*Copy of original Image data.*
- **int m\_progressBegin** = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- **int m\_progressCurrent** = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- **int m\_progressSpan** = 0
- **DImgThreadedFilter \* m\_slave** = nullptr  
*The current slave.*
- **int m\_version** = 1
- **bool m\_wasCancelled** = false

## 9.909.1 Member Function Documentation

### 9.909.1.1 filterAction()

`FilterAction` Digikam::LevelsFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.909.1.2 filterIdentifier()

`QString` Digikam::LevelsFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

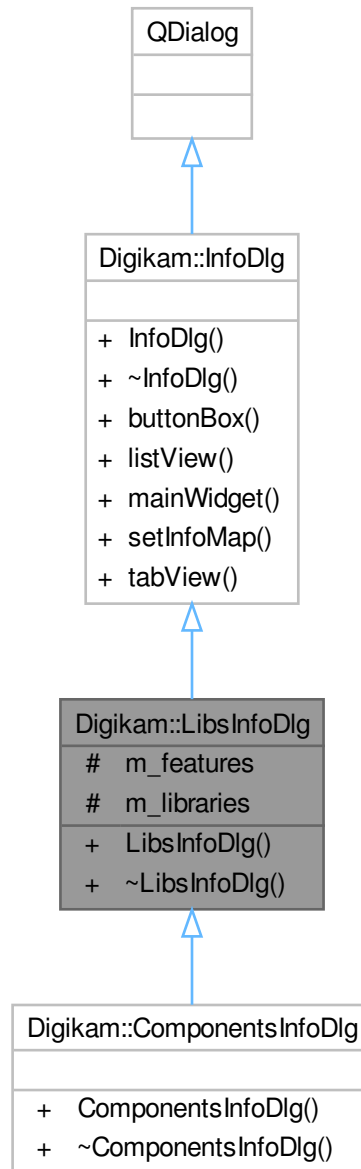
### 9.909.1.3 readParameters()

```
void Digikam::LevelsFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

## 9.910 Digikam::LibsInfoDlg Class Reference

Inheritance diagram for Digikam::LibsInfoDlg:



### Public Member Functions

- [LibsInfoDlg](#) (QWidget \*const parent)

### Public Member Functions inherited from [Digikam::InfoDlg](#)

- `InfoDlg` (QWidget \*const parent)

- QDialogButtonBox \* **buttonBox** () const
- QTreeWidget \* **listView** () const
- QWidget \* **mainWidget** () const
- virtual void **setInfoMap** (const QMap< QString, QString > &list)
- QTabWidget \* **tabView** () const

### Protected Attributes

- QTreeWidgetItem \* **m\_features** = nullptr
- QTreeWidgetItem \* **m\_libraries** = nullptr

## 9.910.1 Constructor & Destructor Documentation

### 9.910.1.1 LibsInfoDlg()

```
Digikam::LibsInfoDlg::LibsInfoDlg (  
    QWidget *const parent ) [explicit]
```

NOTE: MANIFEST.txt is a text file generated with the bundles and listing all git revisions of rolling release components. One section title start with '+'. All component revisions are listed below line by line with the name and the revision separated by ':'. More than one section can be listed in manifest.

## 9.911 Digikam::LightTablePreview Class Reference

Inheritance diagram for Digikam::LightTablePreview:



### Signals

- void **signalDroppedItems** (const [ItemInfoList](#) &)

### Signals inherited from [Digikam::ItemPreviewView](#)

- void **signalAddToExistingQueue** (int)
- void **signalDeleteItem** ()
- void **signalEscapePreview** ()
- void **signalGotoAlbumAndItem** (const [ItemInfo](#) &)
- void **signalGotoDateAndItem** (const [ItemInfo](#) &)
- void **signalGotoTagAndItem** (int)
- void **signalNextItem** ()
- void **signalPopupTagsView** ()
- void **signalPreviewLoaded** (bool success)
- void **signalPrevItem** ()
- void **signalSlideShowCurrent** ()

### Signals inherited from [Digikam::GraphicsDImgView](#)

- void **activated** ()
- void **contentsMoved** (bool panningFinished)
- void **contentsMoving** (int, int)
- void **leftButtonClicked** ()
- void **leftButtonDoubleClicked** ()
- void **resized** ()
- void **rightButtonClicked** ()
- void **toNextImage** ()
- void **toPreviousImage** ()
- void **viewportRectChanged** (const [QRectF](#) &viewportRect)

### Public Member Functions

- **LightTablePreview** ([QWidget](#) \*const parent=nullptr)
- void **setDragAndDropEnabled** (bool b)
- void **showDragAndDropMessage** ()

### Public Member Functions inherited from [Digikam::ItemPreviewView](#)

- **ItemPreviewView** ([QWidget](#) \*const parent, Mode mode=IconViewPreview, [Album](#) \*const currAlbum=nullptr)
- [ItemInfo](#) **getItemInfo** () const
- void **reload** ()
- void **setImagePath** (const [QString](#) &path=[QString](#)())
- void **setItemInfo** (const [ItemInfo](#) &info=[ItemInfo](#)(), const [ItemInfo](#) &previous=[ItemInfo](#)(), const [ItemInfo](#) &next=[ItemInfo](#)())
- void **setPreviousNextPaths** (const [QString](#) &previous, const [QString](#) &next)



## Public Member Functions inherited from [Digikam::GraphicsDImgView](#)

- **GraphicsDImgView** (QWidget \*const parent=nullptr)
- int **contentsX** () const
- int **contentsY** () const
- void **drawText** (QPainter \*p, const QRectF &rect, const QString &text)
- void **fitToWindow** ()
- [GraphicsDImgItem](#) \* **item** () const  
*Return the instance of item set by [setItem\(\)](#).*
- [SinglePhotoPreviewLayout](#) \* **layout** () const
- [DImgPreviewItem](#) \* **previewItem** () const  
*Return a cast of item instance of item set by [setItem\(\)](#) as [DImgPreviewItem](#) Note: if you store a [GraphicsDImgItem](#) object using [setItem\(\)](#), this method will return 0.*
- void **scrollPointOnPoint** (const QPointF &scenePos, const QPoint &viewportPos)  
*Scrolls the view such that scenePos (in scene coordinates) is displayed on the viewport at viewportPos (in viewport coordinates).*
- void **setContentsPos** (int x, int y)
- void **setItem** ([GraphicsDImgItem](#) \*const item)  
*Store internal instance of item as [GraphicsDImgItem](#).*
- void **toggleFullScreen** (bool set)
- QRect **visibleArea** () const

## Additional Inherited Members

## Public Types inherited from [Digikam::ItemPreviewView](#)

- enum **Mode** { [IconViewPreview](#) , [LightTablePreview](#) }

## Protected Slots inherited from [Digikam::GraphicsDImgView](#)

- void **slotContentsMoved** ()
- void **slotCornerButtonPressed** ()
- void **slotPanIconHidden** ()
- virtual void **slotPanIconSelectionMoved** (const QRect &, bool)

## Protected Member Functions inherited from [Digikam::ItemPreviewView](#)

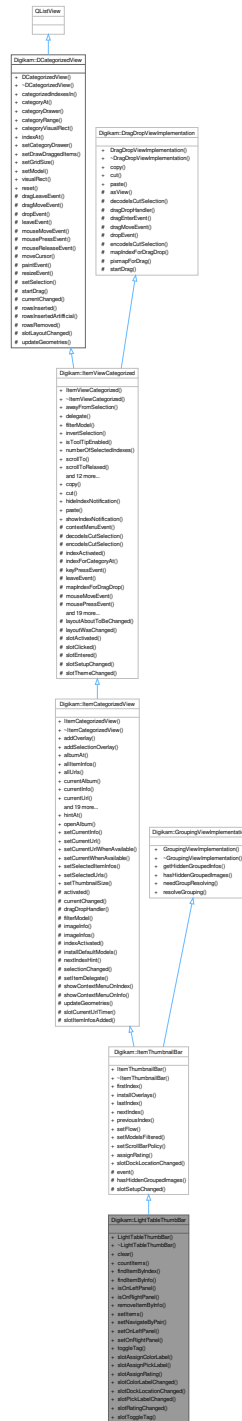
- bool **acceptsMouseClicked** (QMouseEvent \*e) override
- void **dragEnterEvent** (QDragEnterEvent \*e) override
- void **dragMoveEvent** (QDragMoveEvent \*e) override
- void **dropEvent** (QDropEvent \*e) override
- void **enterEvent** (QEnterEvent \*) override
- void **leaveEvent** (QEvent \*e) override
- void **mousePressEvent** (QMouseEvent \*e) override
- void **showEvent** (QShowEvent \*e) override

## Protected Member Functions inherited from [Digikam::GraphicsDImgView](#)

- void **continuePanning** (const QPoint &pos)
- void **drawForeground** (QPainter \*painter, const QRectF &rect) override
- void **finishPanning** ()
- void **installPanIcon** ()
- void **mouseDoubleClickEvent** (QMouseEvent \*) override
- void **mouseMoveEvent** (QMouseEvent \*) override
- void **mousePressEvent** (QMouseEvent \*) override
- void **mouseReleaseEvent** (QMouseEvent \*) override
- void **resizeEvent** (QResizeEvent \*) override
- void **scrollContentsBy** (int dx, int dy) override
- void **setScaleFitToWindow** (bool value)
- void **setShowText** (bool value)
- void **startPanning** (const QPoint &pos)
- void **wheelEvent** (QWheelEvent \*) override

## 9.912 Digikam::LightTableThumbBar Class Reference

Inheritance diagram for Digikam::LightTableThumbBar:



### Public Slots

- void **slotAssignColorLabel** (int)
- void **slotAssignPickLabel** (int)

- void **slotAssignRating** (int)
- void **slotColorLabelChanged** (const QUrl &, int)
- void **slotDockLocationChanged** (Qt::DockWidgetArea area)
- void **slotPickLabelChanged** (const QUrl &, int)
- void **slotRatingChanged** (const QUrl &, int)
- void **slotToggleTag** (const QUrl &, int)

### Public Slots inherited from [Digikam::ItemThumbnailBar](#)

- void **assignRating** (const QList< QModelIndex > &index, int rating)
- void **slotDockLocationChanged** (Qt::DockWidgetArea area)

### Public Slots inherited from [Digikam::ItemCategorizedView](#)

- void **hintAt** (const [ItemInfo](#) &info)
 

*Does something to gain attention for info, but not changing current selection.*
- void **openAlbum** (const QList< [Album](#) \* > &album)
- void **setCurrentInfo** (const [ItemInfo](#) &info)
 

*Set as current item the item identified by the imageinfo.*
- void **setCurrentUrl** (const QUrl &url)
 

*Set as current item the item identified by its file url.*
- void **setCurrentUrlWhenAvailable** (const QUrl &url)
 

*Set as current item when it becomes available, the item identified by its file url.*
- void **setCurrentWhenAvailable** (qulonglong imageId)
 

*Scroll the view to the given item when it becomes available.*
- void **setSelectedItemInfos** (const QList< [ItemInfo](#) > &infos)
 

*Set selected items.*
- void **setSelectedUrls** (const QList< QUrl > &urlList)
 

*Set selected items identified by their file urls.*
- void **setThumbnailSize** (int size)

### Public Slots inherited from [Digikam::ItemViewCategorized](#)

- void **copy** () override
- void **cut** () override
- void **hideIndexNotification** ()
- void **paste** () override
- void **showIndexNotification** (const QModelIndex &index, const QString &message)

### Public Slots inherited from [Digikam::DCategorizedView](#)

- void **reset** () override

### Signals

- void **signalClearAll** ()
- void **signalContentChanged** ()
- void **signalDroppedItems** (const QList< [ItemInfo](#) > &)
- void **signalEditItem** (const [ItemInfo](#) &)
- void **signalRemoveItem** (const [ItemInfo](#) &)
- void **signalSetItemOnLeftPanel** (const [ItemInfo](#) &)
- void **signalSetItemOnRightPanel** (const [ItemInfo](#) &)

## Signals inherited from Digikam::ItemCategorizedView

- void **currentChanged** (const [ItemInfo](#) &info)
- void **deselected** (const QList< [ItemInfo](#) > &nowDeselectedInfos)
 

*Emitted when items are deselected. There may be other selected infos left. This signal is not emitted when the model is reset; then only selectionCleared is emitted.*
- void **imageActivated** (const [ItemInfo](#) &info)
 

*Emitted when the given image is activated. Info is never null.*
- void **modelChanged** ()
 

*Emitted when a new model is set.*
- void **selected** (const QList< [ItemInfo](#) > &newSelectedInfos)
 

*Emitted when new items are selected. The parameter includes only the newly selected infos, there may be other already selected infos.*

## Signals inherited from Digikam::ItemViewCategorized

- void **clicked** (const QMouseEvent \*e, const QModelIndex &index)
 

*For overlays: Like the respective parent class signals, but with additional info.*
- void **entered** (const QMouseEvent \*e, const QModelIndex &index)
- void **keyPressed** (QKeyEvent \*e)
 

*Remember you may want to check if the event is accepted or ignored.*
- void **selectionChanged** ()
 

*Emitted when any selection change occurs.*
- void **selectionCleared** ()
 

*Emitted when the selection is completely cleared.*
- void **viewportClicked** (const QMouseEvent \*e)
 

*While [clicked\(\)](#) is emitted with a valid index, this corresponds to clicking on empty space.*
- void **zoomInStep** ()
- void **zoomOutStep** ()

## Public Member Functions

- **LightTableThumbBar** (QWidget \*const parent)
- void **clear** ()
- int **countItems** () const
- [ItemInfo](#) **findItemByIndex** (const QModelIndex &index) const
- QModelIndex **findItemByInfo** (const [ItemInfo](#) &info) const
- bool **isOnLeftPanel** (const [ItemInfo](#) &info) const
- bool **isOnRightPanel** (const [ItemInfo](#) &info) const
- void **removeItemByInfo** (const [ItemInfo](#) &info)
- void **setItems** (const [ItemInfoList](#) &list)
- void **setNavigateByPair** (bool b)
- void **setOnLeftPanel** (const [ItemInfo](#) &info)
- void **setOnRightPanel** (const [ItemInfo](#) &info)
- void **toggleTag** (int tagID)

## Public Member Functions inherited from [Digikam::ItemThumbnailBar](#)

- **ItemThumbnailBar** (QWidget \*const parent=nullptr)
- QModelIndex **firstIndex** () const
- void **installOverlays** ()
- QModelIndex **lastIndex** () const
- QModelIndex **nextIndex** (const QModelIndex &index) const
- QModelIndex **previousIndex** (const QModelIndex &index) const
- void **setFlow** (QListView::Flow newFlow)
- void **setModelsFiltered** (ItemModel \*model, ImageSortFilterModel \*filterModel)
 

*This installs a duplicate filter model, if the [ItemModel](#) may contain duplicates.*
- void **setScrollBarPolicy** (Qt::ScrollBarPolicy policy)
 

*Sets the policy always for the one scroll bar which is relevant, depending on orientation.*

## Public Member Functions inherited from [Digikam::ItemCategorizedView](#)

- **ItemCategorizedView** (QWidget \*const parent=nullptr)
- void **addOverlay** (ItemDelegateOverlay \*overlay, ItemDelegate \*delegate=nullptr)
 

*Add and remove an overlay. It will as well be removed automatically when destroyed. Unless you pass a different delegate, the current delegate will be used.*
- void **addSelectionOverlay** (ItemDelegate \*delegate=nullptr)
- Album \* **albumAt** (const QPoint &pos) const
 

*If the model is categorized by an album, returns the album of the category that contains the position.*
- ItemInfoList **allItemInfos** () const
- QList< QUrl > **allUrls** () const
- Album \* **currentAlbum** () const
- ItemInfo **currentInfo** () const
- QUrl **currentUrl** () const
- ItemDelegate \* **delegate** () const
- QItemSelectionModel \* **getSelectionModel** () const
- ItemAlbumFilterModel \* **imageAlbumFilterModel** () const
- ItemAlbumModel \* **imageAlbumModel** () const
 

*Returns 0 if the [ItemModel](#) is not an [ItemAlbumModel](#).*
- ItemFilterModel \* **imageFilterModel** () const
 

*Returns any [ItemFilterMode](#) in chain. May not be [sourceModel\(\)](#)*
- ItemModel \* **imageModel** () const
- ImageSortFilterModel \* **imageSortFilterModel** () const
- ItemThumbnailModel \* **imageThumbnailModel** () const
 

*Returns 0 if the [ItemModel](#) is not an [ItemThumbnailModel](#).*
- QModelIndex **indexForInfo** (const ItemInfo &info) const
- ItemInfo **nextInfo** (const ItemInfo &info)
- ItemInfo **nextInOrder** (const ItemInfo &startingPoint, int nth)
 

*Returns the n-th info after the given one.*
- ItemInfo **previousInfo** (const ItemInfo &info)
- void **removeOverlay** (ItemDelegateOverlay \*overlay)
- ItemInfoList **selectedItemInfos** () const
- ItemInfoList **selectedItemInfosCurrentFirst** () const
- void **setModels** (ItemModel \*model, ImageSortFilterModel \*filterModel)
- virtual void **setThumbnailSize** (const ThumbnailSize &size)
- ThumbnailSize **thumbnailSize** () const
- void **toIndex** (const QUrl &url)
 

*Selects the index as current and scrolls to it.*

## Public Member Functions inherited from Digikam::ItemViewCategorized

- **ItemViewCategorized** (QWidget \*const parent=nullptr)
- void **awayFromSelection** ()
- **DItemDelegate** \* **delegate** () const
- void **invertSelection** ()
- bool **isToolTipEnabled** () const
- int **numberOfSelectedIndexes** () const
- void **scrollTo** (const QModelIndex &index, ScrollHint hint=EnsureVisible) override
- void **scrollToRelaxed** (const QModelIndex &index, ScrollHint hint=EnsureVisible)
  - Like scrollTo, but only scrolls if the index is not visible, regardless of hint.*
- void **setInitialSelectedItem** (bool enabled)
  - Ensure a initial selected item.*
- void **setScrollCurrentToCenter** (bool enabled)
  - Scroll automatically the current index to center of the view.*
- void **setScrollStepGranularity** (int factor)
  - Determine a step size for scrolling: The larger this number, the smaller and more precise is the scrolling.*
- void **setSelectedIndexes** (const QList< QModelIndex > &indexes)
- void **setSpacing** (int spacing)
  - Sets the spacing.*
- void **setToolTipEnabled** (bool enabled)
- void **setUsePointingHandCursor** (bool useCursor)
  - Set if the PointingHand Cursor should be shown over the activation area.*
- void **toFirstIndex** ()
  - Selects the index as current and scrolls to it.*
- void **toIndex** (const QModelIndex &index)
- void **toLastIndex** ()
- void **toNextIndex** ()
- void **toPreviousIndex** ()

## Public Member Functions inherited from Digikam::DCategorizedView

- **DCategorizedView** (QWidget \*const parent=nullptr)
- virtual QModelIndexList **categorizedIndexesIn** (const QRect &rect) const
  - This method will return all indexes whose visual rect intersects rect.*
- virtual QModelIndex **categoryAt** (const QPoint &point) const
  - This method will return the first index of the category in the region of which point is found.*
- **DCategoryDrawer** \* **categoryDrawer** () const
- virtual QItemSelectionRange **categoryRange** (const QModelIndex &index) const
  - This method returns the range of indexes contained in the category in which index is sorted.*
- virtual QRect **categoryVisualRect** (const QModelIndex &index) const
  - This method will return the visual rect of the header of the category in which index is sorted.*
- QModelIndex **indexAt** (const QPoint &point) const override
- void **setCategoryDrawer** (DCategoryDrawer \*categoryDrawer)
- void **setDrawDraggedItems** (bool drawDraggedItems)
  - Switch on drawing of dragged items.*
- void **setGridSize** (const QSize &size)
- void **setModel** (QAbstractItemModel \*model) override
- QRect **visualRect** (const QModelIndex &index) const override

## Public Member Functions inherited from [Digikam::DragDropViewImplementation](#)

- virtual void **copy** ()
- virtual void **cut** ()
- virtual void **paste** ()

## Public Member Functions inherited from [Digikam::GroupingViewImplementation](#)

- [ItemInfoList](#) **getHiddenGroupedInfos** (const [ItemInfoList](#) &infos) const
- bool **needGroupResolving** ([OperationType](#) type, const [ItemInfoList](#) &infos) const
- [ItemInfoList](#) **resolveGrouping** (const [ItemInfoList](#) &infos) const

## Additional Inherited Members

## Protected Slots inherited from [Digikam::ItemCategorizedView](#)

- void **slotCurrentUrlTimer** ()
- void **slotItemInfosAdded** ()

## Protected Slots inherited from [Digikam::ItemViewCategorized](#)

- void **layoutAboutToBeChanged** ()
- void **layoutWasChanged** ()
- void **slotActivated** (const QModelIndex &index)
- void **slotClicked** (const QModelIndex &index)
- void **slotEntered** (const QModelIndex &index)
- virtual void **slotThemeChanged** ()

## Protected Slots inherited from [Digikam::DCategorizedView](#)

- void **currentChanged** (const QModelIndex &current, const QModelIndex &previous) override
- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- virtual void **rowsInsertedArtificial** (const QModelIndex &parent, int start, int end)
- virtual void **slotLayoutChanged** ()
- void **updateGeometries** () override

## Protected Member Functions inherited from [Digikam::ItemThumbnailBar](#)

- bool **event** (QEvent \*) override
- bool **hasHiddenGroupedImages** (const [ItemInfo](#) &info) const override  
*must be implemented by parent view*
- void **slotSetupChanged** () override



## Protected Member Functions inherited from Digikam::ItemCategorizedView

- virtual void **activated** (const [ItemInfo](#) &info, Qt::KeyboardModifiers modifiers)
  - Reimplement these in a subclass.*
- void **currentChanged** (const QModelIndex &index, const QModelIndex &previous) override
- [AbstractItemDragDropHandler](#) \* **dragDropHandler** () const override
  - You need to implement these three methods Returns the drag drop handler.*
- QSortFilterProxyModel \* **filterModel** () const override
- [ItemInfo](#) **imageInfo** (const QModelIndex &index) const
- [ItemInfoList](#) **imageInfos** (const QList< QModelIndex > &indexes) const
- void **indexActivated** (const QModelIndex &index, Qt::KeyboardModifiers modifiers) override
- void **installDefaultModels** ()
  - install default [ItemAlbumModel](#) and filter model, ready for use*
- QModelIndex **nextIndexHint** (const QModelIndex &indexToAnchor, const QItemSelectionRange &removed) const override
  - Assuming the given indexes would be removed (hypothetically!), return the index to be selected instead, starting from anchor.*
- void **selectionChanged** (const QItemSelection &, const QItemSelection &) override
- void **setItemDelegate** ([ItemDelegate](#) \*delegate)
- void **showContextMenuOnIndex** (QContextMenuEvent \*event, const QModelIndex &index) override
  - Reimplement these in a subclass.*
- void **updateGeometries** () override

## Protected Member Functions inherited from Digikam::ItemViewCategorized

- void **contextMenuEvent** (QContextMenuEvent \*event) override
  - reimplemented from parent class*
- bool **decodelsCutSelection** (const QMimeData \*mimeData)
- void **encodelsCutSelection** (QMimeData \*mime, bool isCutSelection)
- QModelIndex **indexForCategoryAt** (const QPoint &pos) const
  - Returns an index that is representative for the category at position pos.*
- void **keyPressEvent** (QKeyEvent \*event) override
- void **leaveEvent** (QEvent \*event) override
- QModelIndex **mapIndexForDragDrop** (const QModelIndex &index) const override
  - Note: pure virtual [dragDropHandler\(\)](#) still open from [DragDropViewImplementation](#).*
- void **mouseMoveEvent** (QMouseEvent \*event) override
- void **mousePressEvent** (QMouseEvent \*event) override
- void **mouseReleaseEvent** (QMouseEvent \*event) override
- QModelIndex **moveCursor** (CursorAction cursorAction, Qt::KeyboardModifiers modifiers) override
- QPixmap **pixmapForDrag** (const QList< QModelIndex > &indexes) const override
  - Creates a pixmap for dragging the given indexes.*
- void **reset** () override
- void **resizeEvent** (QResizeEvent \*e) override
- void **rowsAboutToBeRemoved** (const QModelIndex &parent, int start, int end) override
- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- void **rowsRemoved** (const QModelIndex &parent, int start, int end) override
- void **selectionChanged** (const QItemSelection &, const QItemSelection &) override
- void **setItemDelegate** ([DItemDelegate](#) \*delegate)
- void **setToolTip** ([ItemViewToolTip](#) \*tip)
- virtual void **showContextMenu** (QContextMenuEvent \*event)
- virtual bool **showToolTip** (const QModelIndex &index, QStyleOptionViewItem &option, QHelpEvent \*e=nullptr)
  - Provides default behavior, can reimplement in a subclass.*
- void **updateDelegateSizes** ()
- void **userInteraction** ()
- bool **viewportEvent** (QEvent \*event) override
- void **wheelEvent** (QWheelEvent \*event) override

### Protected Member Functions inherited from [Digikam::DCategorizedView](#)

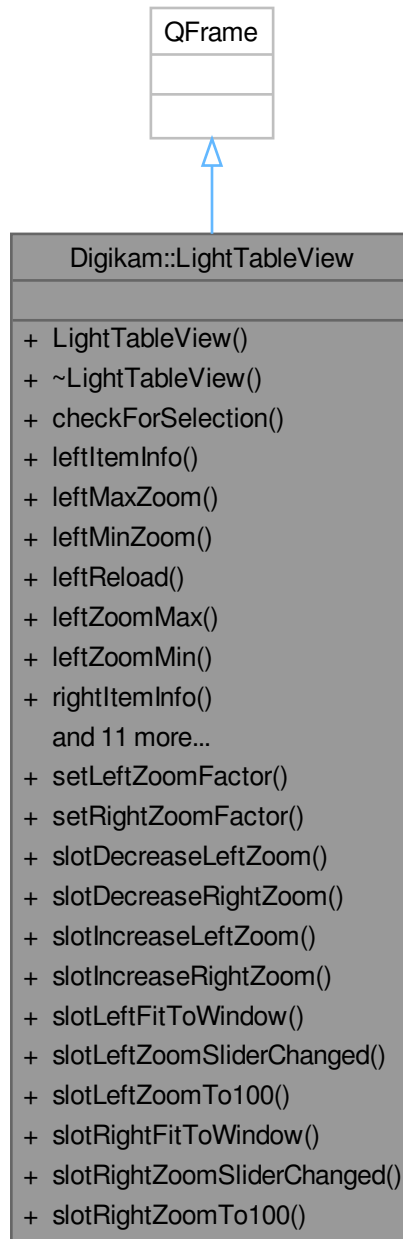
- void **dragLeaveEvent** (QDragLeaveEvent \*event) override
- void **dragMoveEvent** (QDragMoveEvent \*event) override
- void **dropEvent** (QDropEvent \*event) override
- void **leaveEvent** (QEvent \*event) override
- void **mouseMoveEvent** (QMouseEvent \*event) override
- void **mousePressEvent** (QMouseEvent \*event) override
- void **mouseReleaseEvent** (QMouseEvent \*event) override
- QModelIndex **moveCursor** (CursorAction cursorAction, Qt::KeyboardModifiers modifiers) override
- void **paintEvent** (QPaintEvent \*event) override
- void **resizeEvent** (QResizeEvent \*event) override
- void **setSelection** (const QRect &rect, QItemSelectionModel::SelectionFlags flags) override
- void **startDrag** (Qt::DropActions supportedActions) override

### Protected Member Functions inherited from [Digikam::DragDropViewImplementation](#)

- virtual QAbstractItemView \* **asView** ()=0  
*This one is implemented by DECLARE\_VIEW\_DRAG\_DROP\_METHODS.*
- bool **decodelsCutSelection** (const QMimeData \*mimeData)
- void **dragEnterEvent** (QDragEnterEvent \*event)  
*Implements the relevant QAbstractItemView methods for drag and drop.*
- void **dragMoveEvent** (QDragMoveEvent \*e)
- void **dropEvent** (QDropEvent \*e)
- void **encodelsCutSelection** (QMimeData \*mime, bool isCutSelection)
- void **startDrag** (Qt::DropActions supportedActions)

## 9.913 Digikam::LightTableView Class Reference

Inheritance diagram for Digikam::LightTableView:



### Public Slots

- void **setLeftZoomFactor** (double z)
- void **setRightZoomFactor** (double z)
- void **slotDecreaseLeftZoom** ()

- void **slotDecreaseRightZoom** ()
- void **slotIncreaseLeftZoom** ()
- void **slotIncreaseRightZoom** ()
- void **slotLeftFitToWindow** ()
- void **slotLeftZoomSliderChanged** (int)
- void **slotLeftZoomTo100** ()
- void **slotRightFitToWindow** ()
- void **slotRightZoomSliderChanged** (int)
- void **slotRightZoomTo100** ()

## Signals

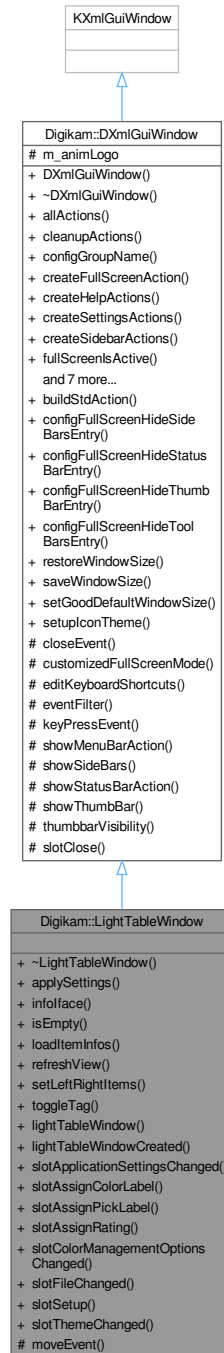
- void **signalDeleteItem** (const [ItemInfo](#) &)
- void **signalEditItem** (const [ItemInfo](#) &)
- void **signalLeftDroppedItems** (const [ItemInfoList](#) &)
- void **signalLeftPanelLeftButtonClicked** ()
- void **signalLeftPopupTagsView** ()
- void **signalLeftPreviewLoaded** (bool)
- void **signalLeftPreviewSelected** (bool)
- void **signalLeftSlideShowCurrent** ()
- void **signalLeftZoomFactorChanged** (double)
- void **signalRightDroppedItems** (const [ItemInfoList](#) &)
- void **signalRightPanelLeftButtonClicked** ()
- void **signalRightPopupTagsView** ()
- void **signalRightPreviewLoaded** (bool)
- void **signalRightPreviewSelected** (bool)
- void **signalRightSlideShowCurrent** ()
- void **signalRightZoomFactorChanged** (double)
- void **signalToggleOnSyncPreview** (bool)

## Public Member Functions

- **LightTableView** (QWidget \*const parent=nullptr)
- void **checkForSelection** (const [ItemInfo](#) &info)
- [ItemInfo](#) **leftItemInfo** () const
- bool **leftMaxZoom** () const
- bool **leftMinZoom** () const
- void **leftReload** ()
- double **leftZoomMax** () const
- double **leftZoomMin** () const
- [ItemInfo](#) **rightItemInfo** () const
- bool **rightMaxZoom** () const
- bool **rightMinZoom** () const
- void **rightReload** ()
- double **rightZoomMax** () const
- double **rightZoomMin** () const
- void **setLeftItemInfo** (const [ItemInfo](#) &info=[ItemInfo](#)())
- void **setNavigateByPair** (bool b)
- void **setPreviewSettings** (const [PreviewSettings](#) &settings)
- void **setRightItemInfo** (const [ItemInfo](#) &info=[ItemInfo](#)())
- void **setSyncPreview** (bool sync)
- void **toggleFullScreen** (bool set)

## 9.914 Digikam::LightTableWindow Class Reference

Inheritance diagram for Digikam::LightTableWindow:



### Public Slots

- void [slotApplicationSettingsChanged](#) ()
- void **slotAssignColorLabel** (int colorId)

- void **slotAssignPickLabel** (int pickId)
- void **slotAssignRating** (int rating)
- void **slotColorManagementOptionsChanged** ()
- void **slotFileChanged** (const QString &filePath)
- void **slotSetup** () override
- void **slotThemeChanged** ()

### Signals

- void **signalWindowHasMoved** ()

### Public Member Functions

- void **applySettings** ()
- [DInfoInterface](#) \* **infoface** ([DPluginAction](#) \*const ac) override  
*Return the interface instance to access to items information.*
- bool **isEmpty** () const
- void **loadItemInfos** (const [ItemInfoList](#) &list, const [ItemInfo](#) &imageInfoCurrent, bool addTo)  
*We get here either.*
- void **refreshView** ()
- void **setLeftRightItems** (const [ItemInfoList](#) &list, bool addTo)  
*Set the images for the left and right panel.*
- void **toggleTag** (int tagID)

### Public Member Functions inherited from [Digikam::DXmlGuiWindow](#)

- [DXmlGuiWindow](#) (QWidget \*const parent=nullptr, Qt::WindowFlags f=Qt::WindowFlags())
- QList< QAction \* > **allActions** () const  
*Return all actions from internal collection.*
- void **cleanupActions** ()  
*Cleanup unwanted actions from action collection.*
- QString **configGroupName** () const
- void **createFullscreenAction** (const QString &name)  
*Create Full-screen action to action collection instance from managed window set through setManagedWindow().*
- void **createHelpActions** (const QString &handbookSection, bool coreOptions=true)  
*Create common actions from Help menu for all digiKam main windows.*
- void **createSettingsActions** ()  
*Create common actions to setup all digiKam main windows.*
- void **createSidebarActions** ()  
*Create common actions to handle side-bar through keyboard shortcuts.*
- bool **fullScreensActive** () const  
*Return true if managed window is currently in Full Screen Mode.*
- void **readFullscreenSettings** (const KConfigGroup &group)  
*Read full-screen settings from KDE config file.*
- virtual void **registerExtraPluginsActions** (QString &)
- void **registerPluginsActions** ()  
*Register all generic plugins action to this instance.*
- void **setConfigGroupName** (const QString &name)  
*Manage config group name used by window instance to get/set settings from config file.*
- void **setFullscreenOptions** (int options)  
*Set full-screen options to managed window.*
- void **unminimizeAndActivateWindow** ()

**Static Public Member Functions**

- static [LightTableWindow](#) \* **lightTableWindow** ()
- static bool **lightTableWindowCreated** ()

**Static Public Member Functions inherited from [Digikam::DXmlGuiWindow](#)**

- static QAction \* **buildStdAction** (StdActionType type, const QObject \*const recvr, const char \*const slot, QObject \*const parent)
  - static QString **configFullScreenHideSideBarsEntry** ()
  - static QString **configFullScreenHideStatusBarEntry** ()
  - static QString **configFullScreenHideThumbBarEntry** ()
  - static QString **configFullScreenHideToolBarsEntry** ()
- Shared with [FullScreenSettings](#).*
- static void **restoreWindowSize** (QWindow \*const win, const KConfigGroup &group)
  - static void **saveWindowSize** (QWindow \*const win, KConfigGroup &group)
  - static void **setGoodDefaultWindowSize** (QWindow \*const win)
  - static void **setupIconTheme** ()

*If we have some local breeze icon resource, prefer it.*

**Protected Member Functions**

- void **moveEvent** (QMoveEvent \*e) override

**Protected Member Functions inherited from [Digikam::DXmlGuiWindow](#)**

- void **closeEvent** (QCloseEvent \*e) override
  - void **editKeyboardShortcuts** (KActionCollection \*const extraac=nullptr, const QString &actitle=QString())
- Call this method from your main window to show keyboard shortcut config dialog with an extra action collection to configure.*
- bool **eventFilter** (QObject \*obj, QEvent \*ev) override
  - void **keyPressEvent** (QKeyEvent \*e) override
  - QAction \* **showMenuBarAction** () const
  - QAction \* **showStatusBarAction** () const
  - virtual void **showThumbBar** (bool visible)
- Re-implement this method if you want to manage thumbbar visibility in full-screen mode.*
- virtual bool **thumbbarVisibility** () const

*Re-implement this method if managed window has a thumbbar.*

**Additional Inherited Members****Protected Slots inherited from [Digikam::DXmlGuiWindow](#)**

- bool **slotClose** ()

**Protected Attributes inherited from [Digikam::DXmlGuiWindow](#)**

- [DLogoAction](#) \* **m\_animLogo** = nullptr

## 9.914.1 Member Function Documentation

### 9.914.1.1 infoIface()

```
DInfoInterface * Digikam::LightTableWindow::infoIface (
    DPluginAction *const ac ) [override], [virtual]
```

Implements [Digikam::DXmlGuiWindow](#).

### 9.914.1.2 loadItemInfos()

```
void Digikam::LightTableWindow::loadItemInfos (
    const ItemInfoList & list,
    const ItemInfo & givenItemInfoCurrent,
    bool addTo )
```

- via CTRL+L (from the albumview) a) digikamapp.cpp: CTRL+key\_L leads to slotImageLightTable() b) digikamview.cpp: void ItemIconView::slotImageLightTable() calls d->iconView->insertToLightTable(list, info); c) albumiconview.cpp: AlbumIconView::insertToLightTable calls lview->loadItemInfos(list, current);
- via drag&drop, i.e. calls issued by the ...Dropped... routines

### 9.914.1.3 slotApplicationSettingsChanged

```
void Digikam::LightTableWindow::slotApplicationSettingsChanged ( ) [slot]
```



## 9.915 Digikam::ListItem Class Reference

Inheritance diagram for Digikam::ListItem:



### Public Member Functions

- `ListItem` (`QList< QVariant > &data`, `ListItem *const parent=nullptr`)
- `QList< ListItem * > allChildren ()` const
- void `appendChild (ListItem *const child)`
- void `appendList (const QList< ListItem * > &items)`
- `ListItem * child (int row)` const
- int `childCount ()` const
- int `columnCount ()` const
- `ListItem * containsItem (ListItem *const item)` const  
*containsItem - search child items if contains a ListItem with the same data as item*
- `QVariant data (int column)` const
- void `deleteChild (int row)`
- void `deleteChild (ListItem *const item)`
- bool `equal (ListItem *const item)` const
- `QList< int > getTagIds ()` const
- `ListItem * parent ()` const
- void `removeAll ()`
- void `removeTagId (int tagId)`
- int `row ()` const
- void `setData (const QList< QVariant > &data)`

## 9.915.1 Member Function Documentation

### 9.915.1.1 containsItem()

```
ListItem * Digikam::ListItem::containsItem (  
    ListItem *const item ) const
```

#### Parameters

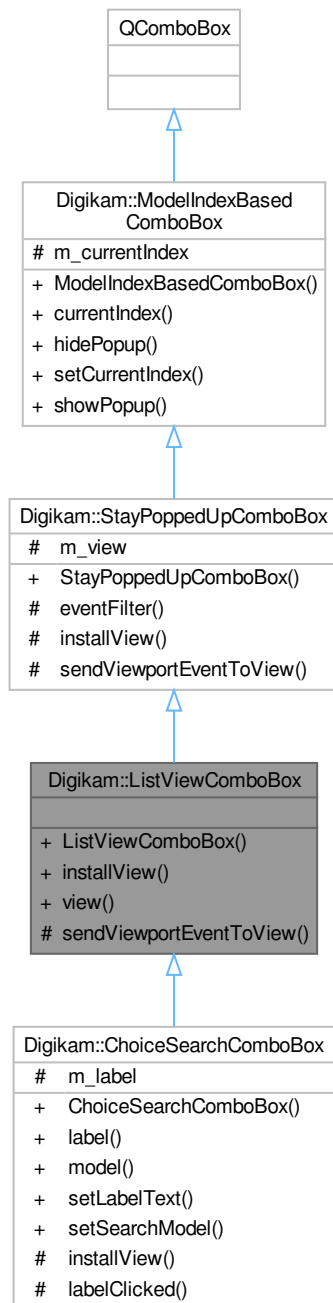
<i>item</i>	- <a href="#">ListItem</a> pointer for which we should search if there is a similar item
-------------	--

#### Returns

- NULL if no similar item was found and a valid [ListItem](#) if a [ListItem](#) with the same data was found

## 9.916 Digikam::ListViewComboBox Class Reference

Inheritance diagram for Digikam::ListViewComboBox:



### Public Member Functions

- [ListViewComboBox](#) (QWidget \*parent=nullptr)

*This class provides an implementation of a [StayPoppedUpComboBox](#) with a [QListView](#).*

- virtual void `installView` (QAbstractItemView \*view=nullptr)  
*Replace the standard combo box list view with a QTreeView.*
- QListView \* `view` () const  
*Returns the QTreeView of this class.*

### Public Member Functions inherited from [Digikam::StayPoppedUpComboBox](#)

- [StayPoppedUpComboBox](#) (QWidget \*const parent=nullptr)  
*This class provides an abstract QComboBox with a custom view (which is created by implementing subclasses) instead of the usual QListView.*

### Public Member Functions inherited from [Digikam::ModelIndexBasedComboBox](#)

- [ModelIndexBasedComboBox](#) (QWidget \*const parent=nullptr)  
*QComboBox has a current index based on a single integer.*
- QModelIndex `currentIndex` () const
- void `hidePopup` () override
- void `setCurrentIndex` (const QModelIndex &index)
- void `showPopup` () override

### Protected Member Functions

- void `sendViewportEventToView` (QEvent \*e) override  
*Implement in subclass: Send the given event to the viewportEvent() method of m\_view.*

### Protected Member Functions inherited from [Digikam::StayPoppedUpComboBox](#)

- bool `eventFilter` (QObject \*watched, QEvent \*event) override
- void `installView` (QAbstractItemView \*view)  
*Replace the standard combo box list view with the given view.*

### Additional Inherited Members

### Protected Attributes inherited from [Digikam::StayPoppedUpComboBox](#)

- QAbstractItemView \* `m_view` = nullptr

### Protected Attributes inherited from [Digikam::ModelIndexBasedComboBox](#)

- QPersistentModelIndex `m_currentIndex`

## 9.916.1 Constructor & Destructor Documentation

### 9.916.1.1 ListViewComboBox()

```
Digikam::ListViewComboBox::ListViewComboBox (
    QWidget * parent = nullptr ) [explicit]
```

This is the standard view of a QComboBox, but in conjunction with [StayPoppedUpComboBox](#) some extra steps are needed. You need three steps: Construct the object, call `setModel()` with an appropriate `QAbstractItemModel`, then call `installView()`.

## 9.916.2 Member Function Documentation

### 9.916.2.1 installView()

```
void Digikam::ListViewComboBox::installView (
    QAbstractItemView * view = nullptr ) [virtual]
```

Call this after installing an appropriate model.

Reimplemented in [Digikam::ChoiceSearchComboBox](#).

### 9.916.2.2 sendViewportEventToView()

```
void Digikam::ListViewComboBox::sendViewportEventToView (
    QEvent * e ) [override], [protected], [virtual]
```

This method is protected for a usual `QAbstractItemView`. You can override, pass a view, and call parent implementation. The existing view will be used. You must then also reimplement `sendViewportEventToView`.

Implements [Digikam::StayPoppedUpComboBox](#).

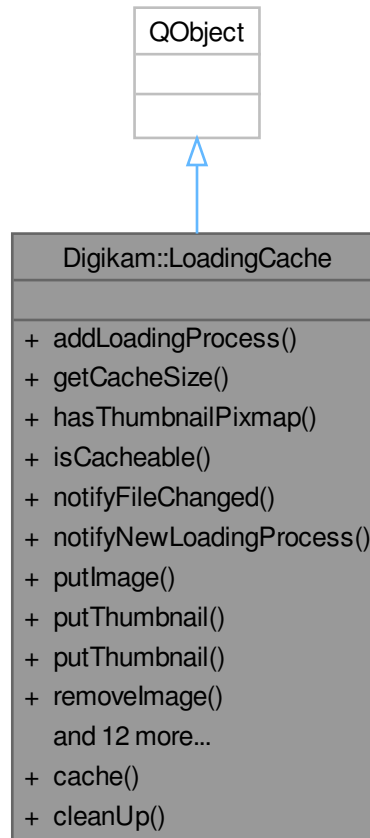
### 9.916.2.3 view()

```
QListView * Digikam::ListViewComboBox::view ( ) const
```

Valid after `installView()` has been called.

## 9.917 Digikam::LoadingCache Class Reference

Inheritance diagram for Digikam::LoadingCache:



### Classes

- class [CacheLock](#)

### Signals

- void [fileChanged](#) (const QString &filePath)  
*This signal is emitted when the cache is notified that a file was changed.*

### Public Member Functions

- void [addLoadingProcess](#) ([LoadingProcess](#) \*const process)  
*Add a loading process to the list.*
- quint64 [getCacheSize](#) () const

*Get the cache size in bytes.*

- bool **hasThumbnailPixmap** (const QString &cacheKey) const
- bool **isCacheable** (const DImg &img) const
 

*Returns whether the given DImg fits in the cache.*
- void **notifyFileChanged** (const QString &filePath, bool notify=true)
 

*Remove all entries from cache that were loaded from filePath.*
- void **notifyNewLoadingProcess** (LoadingProcess \*const process, const LoadingDescription &description)
 

*Notify all currently registered loading processes.*
- bool **putImage** (const QString &cacheKey, const DImg &img, const QString &filePath) const
 

*Put image into for given string into the cache.*
- void **putThumbnail** (const QString &cacheKey, const QImage &thumb, const QString &filePath)
 

*Puts a thumbnail into the thumbnail cache.*
- void **putThumbnail** (const QString &cacheKey, const QPixmap &thumb, const QString &filePath)
- void **removeImage** (const QString &cacheKey)
 

*Remove entries for the given cacheKey from the cache.*
- void **removeImages** ()
 

*Remove all entries from the cache.*
- void **removeLoadingProcess** (LoadingProcess \*const process)
 

*Remove loading process for given cache key.*
- void **removeThumbnail** (const QString &cacheKey)
 

*Remove the thumbnail for the given file path from the thumbnail cache.*
- void **removeThumbnails** ()
 

*Remove all thumbnails.*
- const QPixmap \* **retrieveBufferedTPixmap** (const QString &cacheKey) const
- DImg \* **retrieveImage** (const QString &cacheKey) const
 

*Retrieves an image for the given string from the cache, or 0 if no image is found.*
- LoadingProcess \* **retrieveLoadingProcess** (const QString &cacheKey) const
 

*Find the loading process for given cacheKey, or 0 if not found.*
- const QImage \* **retrieveThumbnail** (const QString &cacheKey) const
 

*The LoadingCache support both the caching of QImage and QPixmap objects.*
- const QPixmap \* **retrieveThumbnailPixmap** (const QString &cacheKey) const
- void **setCacheSize** (int megabytes)
 

*Sets the cache size in megabytes.*
- void **setFileWatch** (LoadingCacheFileWatch \*const watch)
 

*Sets a LoadingCacheFileWatch to watch the files contained in this cache.*
- void **setThumbnailCacheSize** (int numberOfQImages, int numberOfQPixmaps)
 

*Sets the size of the thumbnail cache.*

### Static Public Member Functions

- static LoadingCache \* **cache** ()
- static void **cleanUp** ()

### Friends

- class **CacheLock**
- class **LoadingCacheFileWatch**

## 9.917.1 Member Function Documentation

### 9.917.1.1 addLoadingProcess()

```
void Digikam::LoadingCache::addLoadingProcess (
    LoadingProcess *const process )
```

Only one loading process for the same cache key is registered at a time.

### 9.917.1.2 fileChanged

```
void Digikam::LoadingCache::fileChanged (
    const QString & filePath ) [signal]
```

There is no information in this signal if the file was ever contained in the cache. The signal may be emitted under [CacheLock](#). Strongly consider a queued connection.

### 9.917.1.3 notifyFileChanged()

```
void Digikam::LoadingCache::notifyFileChanged (
    const QString & filePath,
    bool notify = true )
```

Emits relevant signals if notify = true.

### 9.917.1.4 putImage()

```
bool Digikam::LoadingCache::putImage (
    const QString & cacheKey,
    const DImg & img,
    const QString & filePath ) const
```

Returns true if image has been put in the cache, false otherwise. Ownership of the [DImg](#) instance is passed to the cache. When it cannot be put in the cache it is deleted. The third parameter specifies a file path that will be watched. If this file changes, the object will be removed from the cache.

### 9.917.1.5 retrieveThumbnail()

```
const QImage * Digikam::LoadingCache::retrieveThumbnail (
    const QString & cacheKey ) const
```

QPixmap can only be accessed from the main thread, so the tasks cannot access this cache. Retrieves a thumbnail for the given filePath from the thumbnail cache, or a 0 if the thumbnail is not found.

### 9.917.1.6 setCacheSize()

```
void Digikam::LoadingCache::setCacheSize (
    int megabytes )
```

The thumbnail cache is not affected and setThumbnailCacheSize takes the maximum number.



### 9.917.1.7 setFileWatch()

```
void Digikam::LoadingCache::setFileWatch (
    LoadingCacheFileWatch *const watch )
```

Ownership of this object is transferred to the cache.

### 9.917.1.8 setThumbnailCacheSize()

```
void Digikam::LoadingCache::setThumbnailCacheSize (
    int numberOfQImages,
    int numberOfQPixmaps )
```

#### Parameters

<i>numberOfQImages</i>	The maximum number of thumbnails of max possible size in QImage format that will be cached. If the size of the images is smaller, a larger number will be cached.
<i>numberOfQPixmaps</i>	The maximum number of thumbnails of max possible size in QPixmap format that will be cached. If the size of the images is smaller, a larger number will be cached. Note: The main cache is unaffected by this method, and setCacheSize takes megabytes as parameter. Note: A good caching strategy will be to set one of the numbers to 0 Default values: (0, 100)

## 9.918 Digikam::LoadingCache::CacheLock Class Reference

### Public Member Functions

- **CacheLock** ([LoadingCache](#) \*const cache)
- void **timedWait** ()
- void **wakeAll** ()

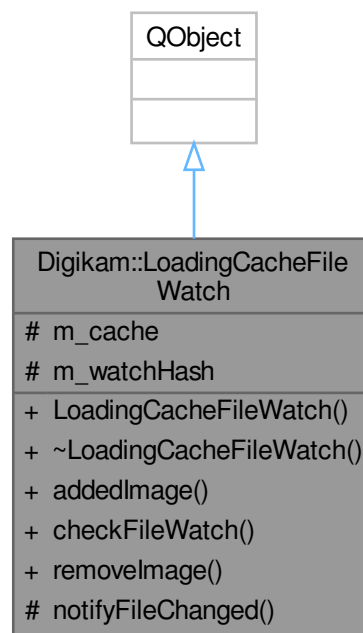
### 9.918.1 Detailed Description

## Warning

All methods of [LoadingCache](#) shall only be called when a [CacheLock](#) is held

## 9.919 Digikam::LoadingCacheFileWatch Class Reference

Inheritance diagram for Digikam::LoadingCacheFileWatch:



### Public Member Functions

- void **addedImage** (const QString &filePath)
- void **checkFileWatch** (const QString &filePath)
- void **removeImage** (const QString &filePath)

### Protected Member Functions

- void [notifyFileChanged](#) (const QString &filePath)  
*Convenience method.*

### Protected Attributes

- class [LoadingCache](#) \* **m\_cache** = nullptr
- QHash< QString, QPair< qint64, QDateTime > > **m\_watchHash**

## Friends

- class **LoadingCache**

## 9.919.1 Member Function Documentation

### 9.919.1.1 notifyFileChanged()

```
void Digikam::LoadingCacheFileWatch::notifyFileChanged (
    const QString & filePath ) [protected]
```

Call this to tell the cache to remove stored images for filePath from the cache. Calling this method is fast, you do not need to check if the file is contained in the cache. Do not hold the CacheLock when calling this method.

## 9.920 Digikam::LoadingCacheInterface Class Reference

### Static Public Member Functions

- static void **cleanCache** ()  
*remove all images from the cache (e.g.*
- static void **cleanThumbnailCache** ()  
*Remove all thumbnails from the thumbnail cache.*
- static void **cleanUp** ()  
*clean up cache at shutdown*
- static void **connectToSignalFileChanged** (QObject \*const object, const char \*slot)  
*Connect the given object/slot to the signal void fileChanged(const QString& filePath); which is emitted when the cache gains knowledge about a possible change of this file on disk.*
- static void **fileChanged** (const QString &filePath, bool notify=true)  
*Remove an image from the cache because it may have changed on disk.*
- static void **initialize** ()
- static void **putImage** (const QString &filePath, const DImg &img)  
*add a copy of the image to cache*
- static void **setCacheOptions** (int cacheSize)  
*Set cache size in Megabytes.*

## 9.920.1 Member Function Documentation

### 9.920.1.1 cleanCache()

```
void Digikam::LoadingCacheInterface::cleanCache ( ) [static]
```

when loading settings changed) Does not affect thumbnails.

### 9.920.1.2 cleanThumbnailCache()

```
void Digikam::LoadingCacheInterface::cleanThumbnailCache ( ) [static]
```

Does not affect main image cache.

### 9.920.1.3 setCacheOptions()

```
void Digikam::LoadingCacheInterface::setCacheOptions (
    int cacheSize ) [static]
```

Set to 0 to disable caching.

## 9.921 Digikam::LoadingDescription Class Reference

### Classes

- class [PostProcessingParameters](#)
- class [PreviewParameters](#)

### Public Types

- enum [ColorManagementSettings](#) { [NoColorConversion](#) , [ApplyTransform](#) , [ConvertForEditor](#) , [ConvertToSRGB](#) , [ConvertForDisplay](#) , [ConvertForOutput](#) }
- enum [RawDecodingHint](#) { [RawDecodingDefaultSettings](#) , [RawDecodingGlobalSettings](#) , [RawDecodingCustomSettings](#) , [RawDecodingTimeOptimized](#) }

### Public Member Functions

- **LoadingDescription** ()  
*An invalid [LoadingDescription](#).*
- **LoadingDescription** (const QString &filePath, [ColorManagementSettings](#)=NoColorConversion)  
*Use this for full loading of non-raw files.*
- **LoadingDescription** (const QString &filePath, const [DRawDecoding](#) &settings, [RawDecodingHint](#) rawDecodingHint=[RawDecodingCustomSettings](#), [ColorManagementSettings](#)=NoColorConversion)  
*Use this for full loading of raw files.*
- **LoadingDescription** (const QString &filePath, const [PreviewSettings](#) &settings, int size, [ColorManagementSettings](#)=NoColorConversion, [PreviewParameters::PreviewType](#)=[PreviewParameters::PreviewImage](#))  
*For preview and thumbnail jobs: Stores preview max size and Exif rotation.*
- QString **cacheKey** () const  
*Return the cache key for this description.*
- bool **equalsIgnoreReducedVersion** (const [LoadingDescription](#) &other) const  
*Returns whether the other loading task equals this one ignoring parameters used to specify a reduced version.*
- bool **equalsOrBetterThan** (const [LoadingDescription](#) &other) const  
*Returns whether this loading task equals the other one or is superior to it, if the other one is a reduced version.*
- bool **isPreviewImage** () const  
*Returns if this description will load a preview.*
- bool **isReducedVersion** () const  
*Returns whether this description describes a loading operation which loads the image in a reduced version (quality, size etc.)*
- bool **isThumbnail** () const  
*Returns if this description will load a thumbnail.*
- QStringList **lookupCacheKeys** () const  
*Return all possible cache keys, starting with the best choice, for which a result may be found in the cache for this description.*

- bool [needCheckRawDecoding](#) () const  
*For some RAW images, the same cache key is not enough to say it is the correct result.*
- bool **operator!=** (const [LoadingDescription](#) &other) const
- bool **operator==** (const [LoadingDescription](#) &other) const  
*Returns whether the other loading task equals this one.*
- [ThumbnailIdentifier](#) **thumbnailIdentifier** () const  
*If this referenced a thumbnail, recreate the identifier.*

### Static Public Member Functions

- static QStringList **possibleCacheKeys** (const QString &filePath)  
*Returns all possible cacheKeys for the given file path (all cache keys under which the given file could be stored in the cache).*
- static QStringList **possibleThumbnailCacheKeys** (const QString &filePath)

### Public Attributes

- QString **filePath**
- [PostProcessingParameters](#) **postProcessingParameters**
- [PreviewParameters](#) **previewParameters**
- [RawDecodingHint](#) **rawDecodingHint** = [RawDecodingDefaultSettings](#)
- [DRawDecoding](#) **rawDecodingSettings**

## 9.921.1 Member Enumeration Documentation

### 9.921.1.1 ColorManagementSettings

enum [Digikam::LoadingDescription::ColorManagementSettings](#)

#### Enumerator

ApplyTransform	IccData is an <a href="#">IccTransform</a> .
ConvertForDisplay	IccData can be the output profile.
ConvertForOutput	IccData is the output profile.

### 9.921.1.2 RawDecodingHint

enum [Digikam::LoadingDescription::RawDecodingHint](#)

#### Enumerator

RawDecodingDefaultSettings	The raw decoding options passed are taken from default, hardcoded settings.
RawDecodingGlobalSettings	The raw decoding options passed are taken from global settings.
RawDecodingCustomSettings	The raw decoding options may be customly edited by the user.
RawDecodingTimeOptimized	The raw decoding options are hardcoded settings optimized for loading time The halfSizeColorImage and 16bit settings can be adjusted separately.

## 9.921.2 Constructor & Destructor Documentation

### 9.921.2.1 LoadingDescription()

```
Digikam::LoadingDescription::LoadingDescription (
    const QString & filePath,
    const PreviewSettings & settings,
    int size,
    ColorManagementSettings cm = NoColorConversion,
    PreviewParameters::PreviewType type = PreviewParameters::PreviewImage )
```

Raw files / preview jobs: If size is not 0, the embedded preview will be loaded if available. If size is 0, [DImg](#) based loading will be used with default raw decoding settings. You can also adjust raw decoding settings and hint in this case.

## 9.921.3 Member Function Documentation

### 9.921.3.1 lookupCacheKeys()

```
QStringList Digikam::LoadingDescription::lookupCacheKeys ( ) const
```

Included in the list are better quality versions, if this description is reduced.

### 9.921.3.2 needCheckRawDecoding()

```
bool Digikam::LoadingDescription::needCheckRawDecoding ( ) const
```

You must check the raw decoding settings in this case.

## 9.922 Digikam::LoadingDescription::PostProcessingParameters Class Reference

### Public Member Functions

- bool **hasProfile** () const
- bool **hasTransform** () const
- bool **needsProcessing** () const
- bool **operator==** (const [PostProcessingParameters](#) &other) const
- [IccProfile](#) **profile** () const
- void **setProfile** (const [IccProfile](#) &profile)
- void **setTransform** (const [IccTransform](#) &transform)
- [IccTransform](#) **transform** () const

### Public Attributes

- [ColorManagementSettings](#) **colorManagement** = NoColorConversion
- QVariant **iccData**

## 9.923 Digikam::LoadingDescription::PreviewParameters Class Reference

### Public Types

- enum **PreviewFlag** { **NoFlags** = 0 , **OnlyPregenerate** = 1 << 0 , **OnlyFromStorage** = 1 << 1 }
- typedef QFlags< PreviewFlag > **PreviewFlags**
- enum **PreviewType** { **NoPreview** , **PreviewImage** , **Thumbnail** , **DetailThumbnail** }

### Public Member Functions

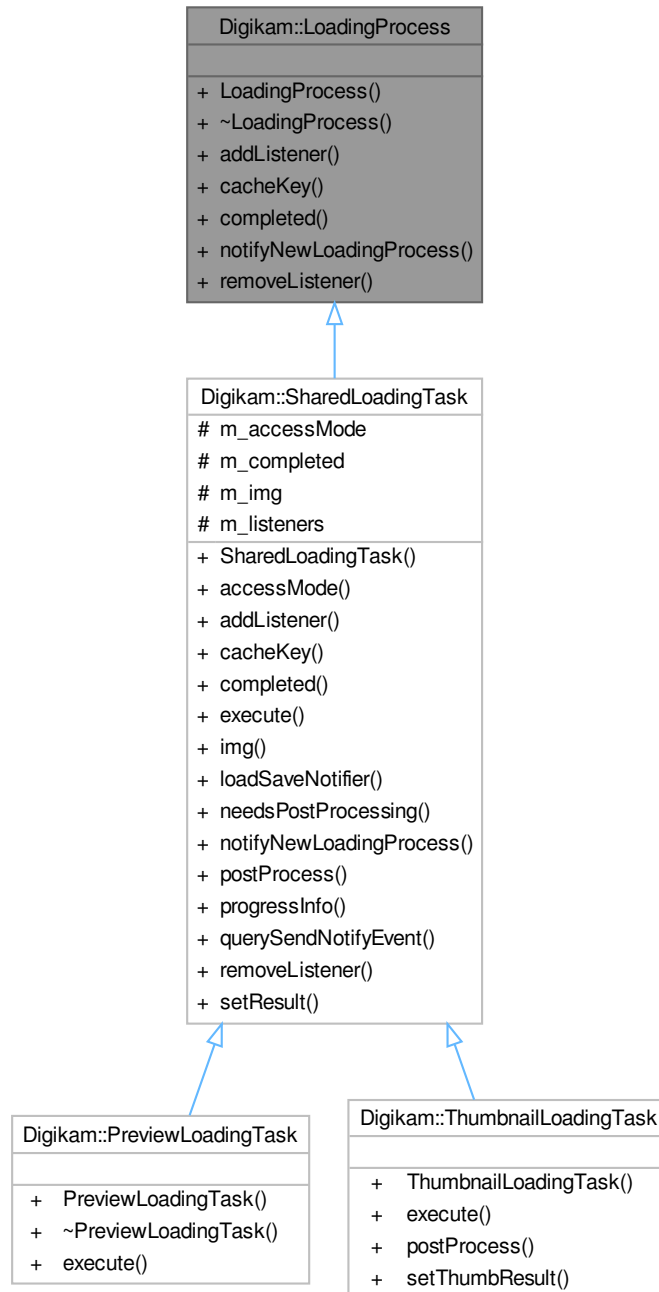
- bool **onlyFromStorage** () const
- bool **onlyPregenerate** () const
- bool **operator==** (const [PreviewParameters](#) &other) const

### Public Attributes

- QVariant **extraParameter**
- PreviewFlags **flags** = NoFlags
- [PreviewSettings](#) **previewSettings**
- int **size** = 0
- QVariant **storageReference**
- PreviewType **type** = NoPreview

## 9.924 Digikam::LoadingProcess Class Reference

Inheritance diagram for Digikam::LoadingProcess:



### Public Member Functions

- virtual void **addListener** ([LoadingProcessListener](#) \*const listener)=0
- virtual QString **cacheKey** () const =0



- virtual bool **completed** () const =0
- virtual void **notifyNewLoadingProcess** ([LoadingProcess](#) \*const process, const [LoadingDescription](#) &description)=0
- virtual void **removeListener** ([LoadingProcessListener](#) \*const listener)=0

## 9.925 Digikam::LoadingProcessListener Class Reference

Inheritance diagram for Digikam::LoadingProcessListener:

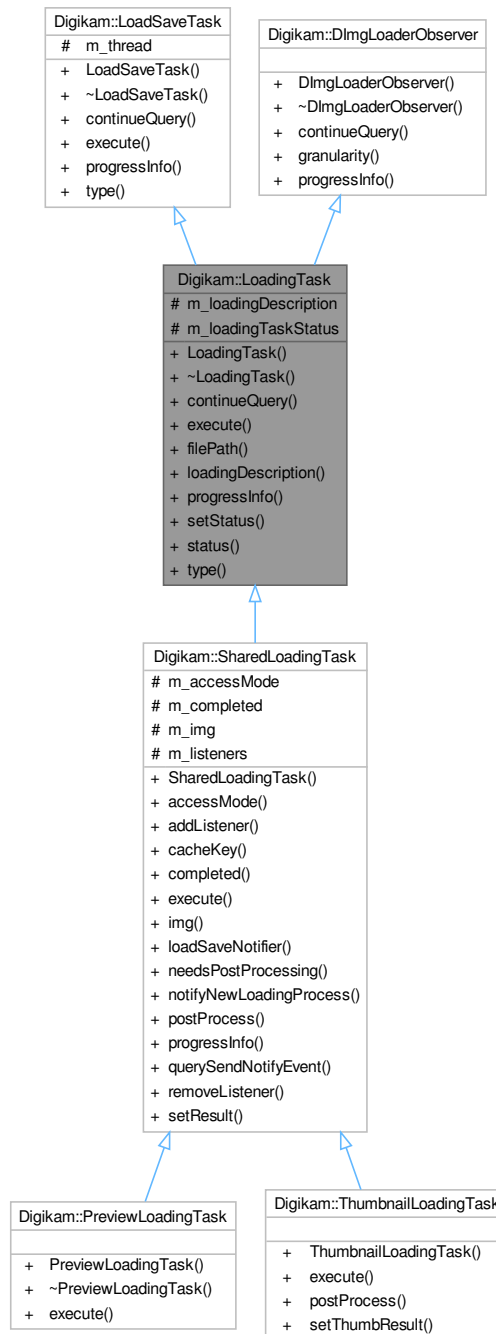


## Public Member Functions

- virtual `LoadSaveThread::AccessMode accessMode ()` const =0
- virtual `LoadSaveNotifier * loadSaveNotifier ()` const =0
- virtual `bool querySendNotifyEvent ()` const =0
- virtual `void setResult (const LoadingDescription &loadingDescription, const DImg &img)=0`

## 9.926 Digikam::LoadingTask Class Reference

Inheritance diagram for Digikam::LoadingTask:



## Public Types

- enum **LoadingTaskStatus** { **LoadingTaskStatusLoading** , **LoadingTaskStatusPreloading** , **LoadingTaskStatusStopping** }

## Public Types inherited from [Digikam::LoadSaveTask](#)

- enum **TaskType** { **TaskTypeLoading** , **TaskTypeSaving** }

## Public Member Functions

- **LoadingTask** ([LoadSaveThread](#) \*const thread, const [LoadingDescription](#) &description, LoadingTaskStatus loadingTaskStatus=LoadingTaskStatusLoading)
- bool [continueQuery](#) () override
- void [execute](#) () override
- QString [filePath](#) () const
- const [LoadingDescription](#) & [loadingDescription](#) () const
- void [progressInfo](#) (float progress) override
- void [setStatus](#) (LoadingTaskStatus status)
- LoadingTaskStatus [status](#) () const
- TaskType [type](#) () override

## Public Member Functions inherited from [Digikam::LoadSaveTask](#)

- **LoadSaveTask** ([LoadSaveThread](#) \*const thread)

## Public Member Functions inherited from [Digikam::DImgLoaderObserver](#)

- virtual float [granularity](#) ()  
*Return a relative value which determines the granularity, the frequency with which the [DImgLoaderObserver](#) is checked and progress is posted.*

## Protected Attributes

- [LoadingDescription](#) **m\_loadingDescription**
- volatile LoadingTaskStatus **m\_loadingTaskStatus** = LoadingTaskStatusLoading

## Protected Attributes inherited from [Digikam::LoadSaveTask](#)

- [LoadSaveThread](#) \* **m\_thread** = nullptr

## 9.926.1 Member Function Documentation

### 9.926.1.1 continueQuery()

```
bool Digikam::LoadingTask::continueQuery ( ) [override], [virtual]
```

Implements [Digikam::LoadSaveTask](#).

### 9.926.1.2 execute()

```
void Digikam::LoadingTask::execute ( ) [override], [virtual]
```

Implements [Digikam::LoadSaveTask](#).

### 9.926.1.3 progressInfo()

```
void Digikam::LoadingTask::progressInfo (
    float progress ) [override], [virtual]
```

Implements [Digikam::LoadSaveTask](#).

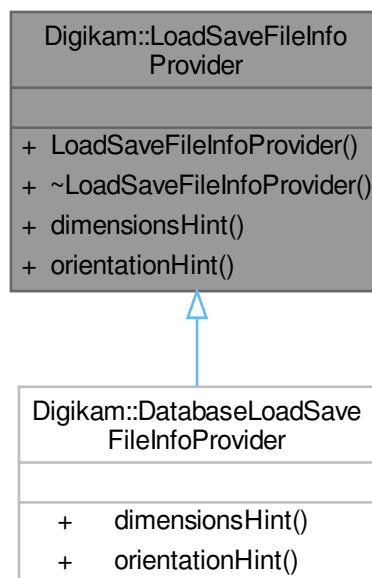
### 9.926.1.4 type()

```
LoadingTask::TaskType Digikam::LoadingTask::type ( ) [override], [virtual]
```

Implements [Digikam::LoadSaveTask](#).

## 9.927 Digikam::LoadSaveFileInfoProvider Class Reference

Inheritance diagram for Digikam::LoadSaveFileInfoProvider:



## Public Member Functions

- virtual QSize [dimensionsHint](#) (const QString &path)=0  
*Gives a hint at the size of the image.*
- virtual int [orientationHint](#) (const QString &path)=0  
*Gives a hint at the orientation of the image.*

## 9.927.1 Member Function Documentation

### 9.927.1.1 dimensionsHint()

```
virtual QSize Digikam::LoadSaveFileInfoProvider::dimensionsHint (  
    const QString & path ) [pure virtual]
```

This can be used to supersede the Exif information in the file.

Implemented in [Digikam::DatabaseLoadSaveFileInfoProvider](#).

### 9.927.1.2 orientationHint()

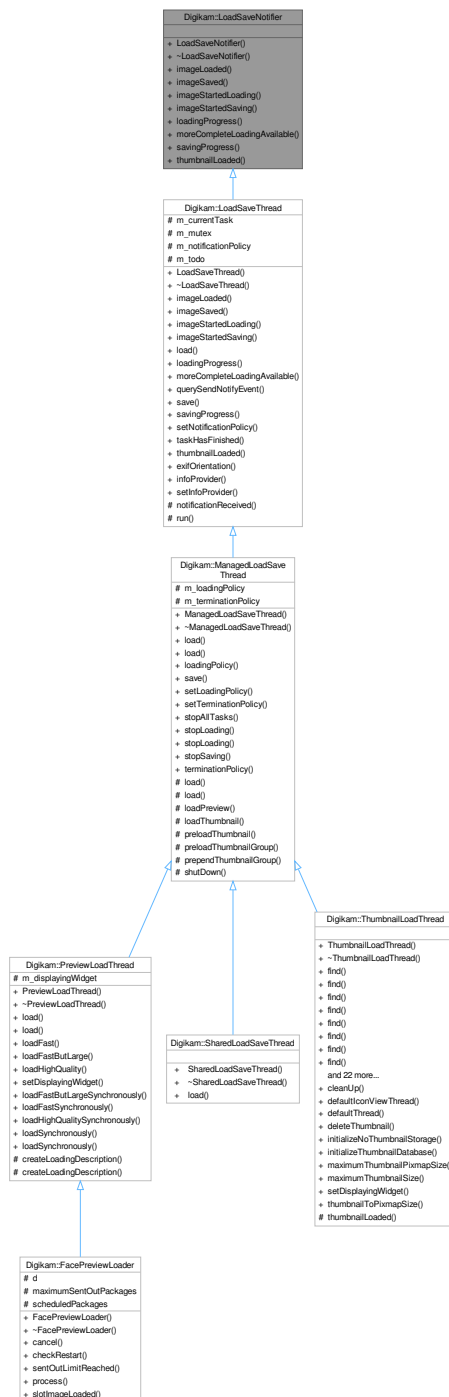
```
virtual int Digikam::LoadSaveFileInfoProvider::orientationHint (  
    const QString & path ) [pure virtual]
```

This can be used to supersede the Exif information in the file. Will not be used if DMetadata::ORIENTATION\_↔ UNSPECIFIED (default value)

Implemented in [Digikam::DatabaseLoadSaveFileInfoProvider](#).

## 9.928 Digikam::LoadSaveNotifier Class Reference

Inheritance diagram for Digikam::LoadSaveNotifier:



### Public Member Functions

- virtual void **imageLoaded** (const [LoadingDescription](#) &loadingDescription, const [DImg](#) &img)=0
- virtual void **imageSaved** (const QString &filePath, bool success)=0

- virtual void **imageStartedLoading** (const [LoadingDescription](#) &loadingDescription)=0
- virtual void **imageStartedSaving** (const QString &filePath)=0
- virtual void **loadingProgress** (const [LoadingDescription](#) &loadingDescription, float progress)=0
- virtual void **moreCompleteLoadingAvailable** (const [LoadingDescription](#) &oldLoadingDescription, const [LoadingDescription](#) &newLoadingDescription)=0
- virtual void **savingProgress** (const QString &filePath, float progress)=0
- virtual void **thumbnailLoaded** (const [LoadingDescription](#) &loadingDescription, const QImage &img)=0

## 9.928.1 Member Function Documentation

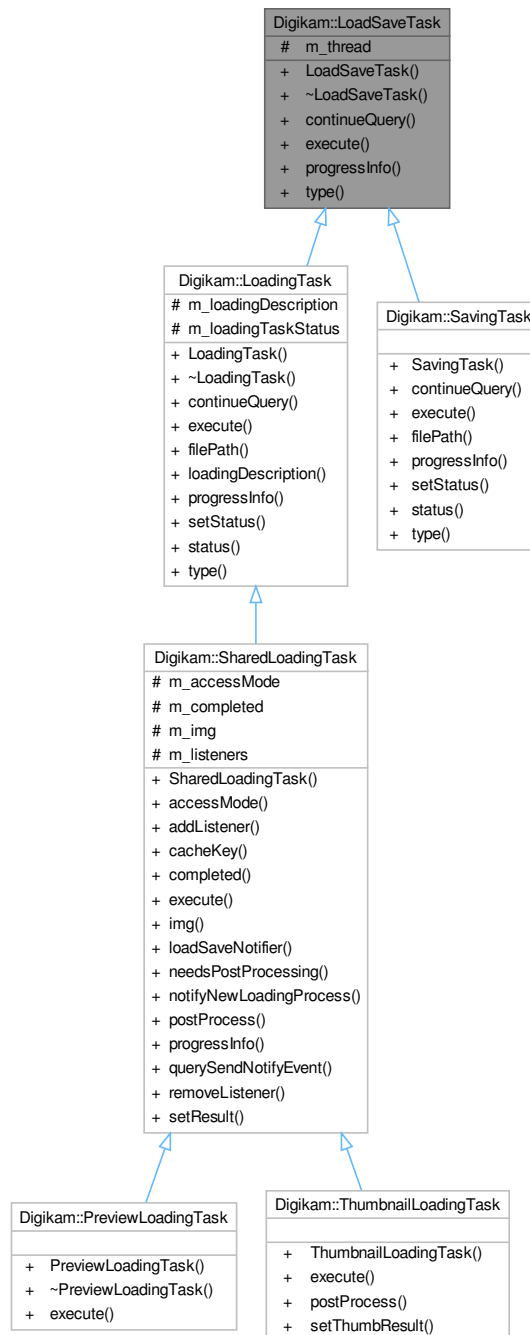
### 9.928.1.1 thumbnailLoaded()

```
virtual void Digikam::LoadSaveNotifier::thumbnailLoaded (  
    const LoadingDescription & loadingDescription,  
    const QImage & img ) [pure virtual]
```

Implemented in [Digikam::ThumbnailLoadThread](#).

## 9.929 Digikam::LoadSaveTask Class Reference

Inheritance diagram for Digikam::LoadSaveTask:



### Public Types

- enum `TaskType` { `TaskTypeLoading` , `TaskTypeSaving` }



### Public Member Functions

- **LoadSaveTask** ([LoadSaveThread](#) \*const thread)
- virtual bool **continueQuery** ()=0
- virtual void **execute** ()=0
- virtual void **progressInfo** (float progress)=0
- virtual TaskType **type** ()=0

### Protected Attributes

- [LoadSaveThread](#) \* **m\_thread** = nullptr

# 9.930 Digikam::LoadSaveThread Class Reference

Inheritance diagram for Digikam::LoadSaveThread:



**Public Types**

- enum [AccessMode](#) { [AccessModeRead](#) , [AccessModeReadWrite](#) }  
 used by [SharedLoadSaveThread](#) only
- enum [NotificationPolicy](#) { [NotificationPolicyDirect](#) , [NotificationPolicyTimeLimited](#) }

## Public Types inherited from Digikam::DynamicThread

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

## Signals

- void **signalImageLoaded** (const [LoadingDescription](#) &loadingDescription, const [DImg](#) &img)  
*This signal is emitted when the loading process has finished.*
- void **signalImageSaved** (const [QString](#) &filePath, bool success)
- void **signalImageStartedLoading** (const [LoadingDescription](#) &loadingDescription)  
*All signals are delivered to the thread from where the [LoadSaveThread](#) object has been created.*
- void **signalImageStartedSaving** (const [QString](#) &filePath)
- void **signalLoadingProgress** (const [LoadingDescription](#) &loadingDescription, float progress)  
*This signal is emitted whenever new progress info is available and the notification policy allows emitting the signal.*
- void **signalMoreCompleteLoadingAvailable** (const [LoadingDescription](#) &oldLoadingDescription, const [LoadingDescription](#) &newLoadingDescription)  
*This signal is emitted if.*
- void **signalSavingProgress** (const [QString](#) &filePath, float progress)
- void **signalThumbnailLoaded** (const [LoadingDescription](#) &loadingDescription, const [QImage](#) &img)

## Signals inherited from Digikam::DynamicThread

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Public Member Functions

- **LoadSaveThread** (QObject \*const parent=nullptr)
- **~LoadSaveThread** () override  
*Destructor: The thread will execute all pending tasks and wait for this upon destruction.*
- void **imageLoaded** (const [LoadingDescription](#) &loadingDescription, const [DImg](#) &img) override
- void **imageSaved** (const [QString](#) &filePath, bool success) override
- void **imageStartedLoading** (const [LoadingDescription](#) &loadingDescription) override
- void **imageStartedSaving** (const [QString](#) &filePath) override
- void **load** (const [LoadingDescription](#) &description)  
*Append a task to load the given file to the task list.*
- void **loadingProgress** (const [LoadingDescription](#) &loadingDescription, float progress) override
- void **moreCompleteLoadingAvailable** (const [LoadingDescription](#) &oldLoadingDescription, const [LoadingDescription](#) &newLoadingDescription) override
- virtual bool **querySendNotifyEvent** () const
- void **save** (const [DImg](#) &image, const [QString](#) &filePath, const [QString](#) &format)  
*Append a task to save the image to the task list.*
- void **savingProgress** (const [QString](#) &filePath, float progress) override
- void **setNotificationPolicy** ([NotificationPolicy](#) notificationPolicy)
- virtual void **taskHasFinished** ()
- void **thumbnailLoaded** (const [LoadingDescription](#) &loadingDescription, const [QImage](#) &img) override

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread](#) (QObject \*const parent=nullptr)
 

*This class extends QRunnable, so you have to reimplement virtual void [run\(\)](#).*
- [~DynamicThread](#) () override
 

*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool [isFinished](#) () const
- bool [isRunning](#) () const
- QThread::Priority [priority](#) () const
- void [setEmitSignals](#) (bool emitThem)
- void [setPriority](#) (QThread::Priority priority)
 

*Sets the priority for this dynamic thread.*
- State [state](#) () const

## Static Public Member Functions

- static int [exifOrientation](#) (const QString &filePath, const [DMetadata](#) &metadata, bool isRaw, bool fromRaw↔ EmbeddedPreview)
 

*Retrieves the Exif orientation, either from the info provider if available, or from the metadata.*
- static [LoadSaveFileInfoProvider](#) \* [infoProvider](#) ()
- static void [setInfoProvider](#) ([LoadSaveFileInfoProvider](#) \*const infoProvider)

## Protected Member Functions

- void [notificationReceived](#) ()
- void [run](#) () override
 

*Implement this pure virtual function in your subclass.*

## Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool [runningFlag](#) () const volatile
 

*In you [run\(\)](#) method, you shall regularly check for [runningFlag\(\)](#) and cleanup and return if false.*
- void [shutDown](#) ()
 

*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call [stop\(\)](#) and [wait\(\)](#), knowing that nothing will call [start\(\)](#) anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)
 

*Doing the same as [start\(\)](#), [stop\(\)](#) and [wait](#) above, provide it with a locked QMutexLocker on [mutex\(\)](#).*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const
 

*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes

- [LoadSaveTask](#) \* [m\\_currentTask](#) = nullptr
- QMutex [m\\_mutex](#)
- [NotificationPolicy](#) [m\\_notificationPolicy](#) = [NotificationPolicyTimeLimited](#)
- QList< [LoadSaveTask](#) \* > [m\\_todo](#)

## Additional Inherited Members

## Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()
  - Stop computation, sets the running flag to false.*
- void **wait** ()
  - Waits until the thread finishes.*

## 9.930.1 Member Enumeration Documentation

### 9.930.1.1 AccessMode

```
enum Digikam::LoadSaveThread::AccessMode
```

#### Enumerator

AccessModeRead	image will only be used for reading
AccessModeReadWrite	image data will possibly be changed

### 9.930.1.2 NotificationPolicy

```
enum Digikam::LoadSaveThread::NotificationPolicy
```

#### Enumerator

NotificationPolicyDirect	Always send notification, unless the last event is still in the event queue.
NotificationPolicyTimeLimited	Always wait for a certain amount of time after the last event sent. In particular, the first event will be sent only after waiting for this time span. (Or no event will be sent, when the loading has finished before) This is the default.

## 9.930.2 Member Function Documentation

### 9.930.2.1 imageLoaded()

```
void Digikam::LoadSaveThread::imageLoaded (
    const LoadingDescription & loadingDescription,
    const DImg & img ) [override], [virtual]
```

Implements [Digikam::LoadSaveNotifier](#).

### 9.930.2.2 imageSaved()

```
void Digikam::LoadSaveThread::imageSaved (
    const QString & filePath,
    bool success ) [override], [virtual]
```

Implements [Digikam::LoadSaveNotifier](#).

### 9.930.2.3 imageStartedLoading()

```
void Digikam::LoadSaveThread::imageStartedLoading (
    const LoadingDescription & loadingDescription ) [override], [virtual]
```

Implements [Digikam::LoadSaveNotifier](#).

### 9.930.2.4 imageStartedSaving()

```
void Digikam::LoadSaveThread::imageStartedSaving (
    const QString & filePath ) [override], [virtual]
```

Implements [Digikam::LoadSaveNotifier](#).

### 9.930.2.5 loadingProgress()

```
void Digikam::LoadSaveThread::loadingProgress (
    const LoadingDescription & loadingDescription,
    float progress ) [override], [virtual]
```

Implements [Digikam::LoadSaveNotifier](#).

### 9.930.2.6 moreCompleteLoadingAvailable()

```
void Digikam::LoadSaveThread::moreCompleteLoadingAvailable (
    const LoadingDescription & oldLoadingDescription,
    const LoadingDescription & newLoadingDescription ) [override], [virtual]
```

Implements [Digikam::LoadSaveNotifier](#).

### 9.930.2.7 run()

```
void Digikam::LoadSaveThread::run ( ) [override], [protected], [virtual]
```

Implements [Digikam::DynamicThread](#).

### 9.930.2.8 savingProgress()

```
void Digikam::LoadSaveThread::savingProgress (
    const QString & filePath,
    float progress ) [override], [virtual]
```

Implements [Digikam::LoadSaveNotifier](#).

### 9.930.2.9 signalImageLoaded

```
void Digikam::LoadSaveThread::signalImageLoaded (
    const LoadingDescription & loadingDescription,
    const QImage & img ) [signal]
```

If the process failed, img is null.

### 9.930.2.10 signalImageStartedLoading

```
void Digikam::LoadSaveThread::signalImageStartedLoading (
    const LoadingDescription & loadingDescription ) [signal]
```

This thread must use its event loop to get the signals. You must connect to these signals with Qt::AutoConnection (default) or Qt::QueuedConnection. This signal is emitted when the loading process begins.

### 9.930.2.11 signalLoadingProgress

```
void Digikam::LoadSaveThread::signalLoadingProgress (
    const LoadingDescription & loadingDescription,
    float progress ) [signal]
```

No progress info will be sent for preloaded images ([ManagedLoadSaveThread](#)).

### 9.930.2.12 signalMoreCompleteLoadingAvailable

```
void Digikam::LoadSaveThread::signalMoreCompleteLoadingAvailable (
    const LoadingDescription & oldLoadingDescription,
    const LoadingDescription & newLoadingDescription ) [signal]
```

- you are doing shared loading ([SharedLoadSaveThread](#))
- you started a loading operation with a [LoadingDescription](#) for a reduced version of the image
- another thread started a loading operation for a more complete version You may want to cancel the current operation and start with the given loadingDescription

### 9.930.2.13 thumbnailLoaded()

```
void Digikam::LoadSaveThread::thumbnailLoaded (
    const LoadingDescription & loadingDescription,
    const QImage & img ) [override], [virtual]
```

Implements [Digikam::LoadSaveNotifier](#).

Reimplemented in [Digikam::ThumbnailLoadThread](#).

## 9.931 Digikam::LocalContrastContainer Class Reference

### Public Member Functions

- double **getBlur** (int nstage) const
- double **getPower** (int nstage) const

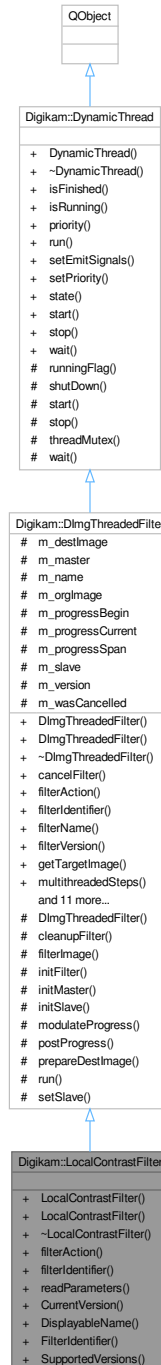
### Public Attributes

- int **functionId** = 0
- int **highSaturation** = 100
- int **lowSaturation** = 100
- struct {
  - double **blur** = 80.0
  - bool **enabled** = false
  - double **power** = 30.0
  - } **stage** [TONEMAPPING\_MAX\_STAGES]
- bool **stretchContrast** = true



## 9.932 Digikam::LocalContrastFilter Class Reference

Inheritance diagram for Digikam::LocalContrastFilter:



### Public Member Functions

- **LocalContrastFilter** (`DImg *const image`, `QObject *const parent=nullptr`, `const LocalContrastContainer &par=LocalContrastContainer()`)

- **LocalContrastFilter** (QObject \*const parent=nullptr)
- **FilterAction filterAction** () override  
*Returns the action description corresponding to currently set options.*
- **QString filterIdentifier** () const override  
*Return the identifier for this filter in the image history.*
- void **readParameters** (const **FilterAction** &action) override

## Public Member Functions inherited from **Digikam::DImgThreadedFilter**

- **DImgThreadedFilter** (DImg \*const orgImage, QObject \*const parent, const QString &name=QString())  
*Constructs a filter with all arguments (ready to use).*
- **DImgThreadedFilter** (QObject \*const parent=nullptr, const QString &name=QString())  
*Constructs a filter without argument.*
- virtual void **cancelFilter** ()  
*Cancel the threaded computation.*
- const QString & **filterName** ()
- int **filterVersion** () const
- **DImg getTargetImage** ()
- QList< int > **multithreadedSteps** (int stop, int start=0) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool **parametersSuccessfullyRead** () const  
*Optional: error handling for readParameters.*
- virtual QString **readParametersError** (const **FilterAction** &actionThatFailed) const
- void **setFilterName** (const QString &name)
- void **setFilterVersion** (int version)  
*Replaying a filter action: Set the filter version.*
- void **setOriginalImage** (const **DImg** &orgImage)
- void **setupAndStartDirectly** (const **DImg** &orgImage, **DImgThreadedFilter** \*const master, int progress←Begin=0, int progressEnd=100)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void **setupFilter** (const **DImg** &orgImage)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void **startFilter** ()  
*Start the threaded computation.*
- virtual void **startFilterDirectly** ()  
*Start computation of this filter, directly in this thread.*
- virtual QList< int > **supportedVersions** () const

## Public Member Functions inherited from **Digikam::DynamicThread**

- **DynamicThread** (QObject \*const parent=nullptr)  
*This class extends QRunnable, so you have to reimplement virtual void [run\(\)](#).*
- **~DynamicThread** () override  
*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool **isFinished** () const
- bool **isRunning** () const
- QThread::Priority **priority** () const
- void **setEmitSignals** (bool emitThem)
- void **setPriority** (QThread::Priority priority)  
*Sets the priority for this dynamic thread.*
- State **state** () const

### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from [Digikam::DImgThreadedFilter](#)

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.932.1 Member Function Documentation

### 9.932.1.1 filterAction()

`FilterAction` Digikam::LocalContrastFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.932.1.2 filterIdentifier()

`QString` Digikam::LocalContrastFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

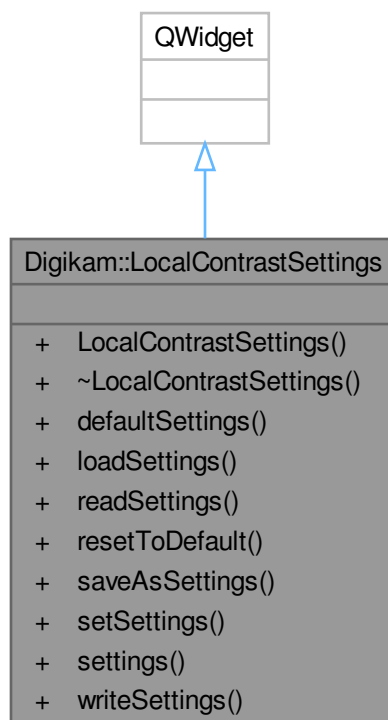
### 9.932.1.3 readParameters()

```
void Digikam::LocalContrastFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

## 9.933 Digikam::LocalContrastSettings Class Reference

Inheritance diagram for Digikam::LocalContrastSettings:



## Signals

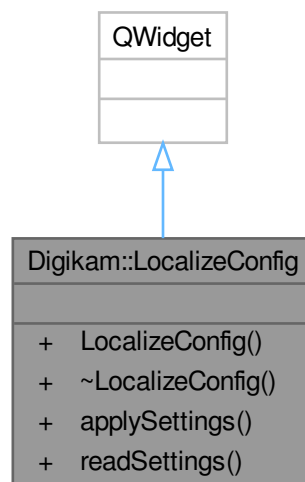
- void **signalSettingsChanged** ()

## Public Member Functions

- **LocalContrastSettings** (QWidget \*const parent)
- [LocalContrastContainer](#) **defaultSettings** () const
- void **loadSettings** ()
- void **readSettings** (KConfigGroup &group)
- void **resetToDefault** ()
- void **saveAsSettings** ()
- void **setSettings** (const [LocalContrastContainer](#) &settings)
- [LocalContrastContainer](#) **settings** () const
- void **writeSettings** (KConfigGroup &group)

## 9.934 Digikam::LocalizeConfig Class Reference

Inheritance diagram for Digikam::LocalizeConfig:



## Public Member Functions

- **LocalizeConfig** (QWidget \*const parent=nullptr)
- void **applySettings** ()
- void **readSettings** ()

## 9.935 Digikam::LocalizeContainer Class Reference

The class [LocalizeContainer](#) encapsulates all spell-check and localize related settings.

### Public Member Functions

- void **readFromConfig** (const KConfigGroup &group)
- void **writeToConfig** (KConfigGroup &group) const

### Public Attributes

- QStringList **alternativeLang**  
*List of langges to use with Alternative Languages Text editor.*
- QString **defaultLanguage**
- bool **enableSpellCheck** = false  
*Enable spell-checking feature.*
- QStringList **ignoredWords**  
*Default language code to use with x-default (empty for auto-detection).*
- [DOnlineTranslator::Engine](#) **translatorEngine** = [DOnlineTranslator::Google](#)  
*Online translator to use.*
- QStringList **translatorLang**  
*List of langues to use with Online translator.*

### 9.935.1 Member Data Documentation

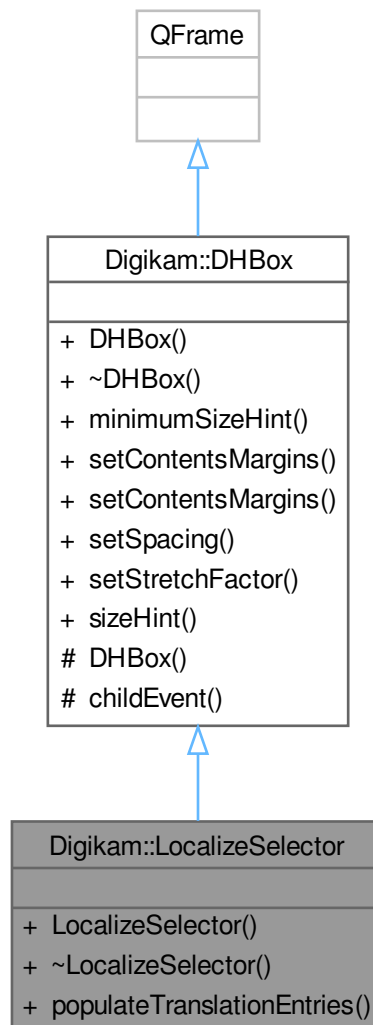
#### 9.935.1.1 ignoredWords

```
QStringList Digikam::LocalizeContainer::ignoredWords
```

Words to ignore with spell-checking.

## 9.936 Digikam::LocalizeSelector Class Reference

Inheritance diagram for Digikam::LocalizeSelector:



### Signals

- void **signalTranslate** (const QString &lang)

### Public Member Functions

- **LocalizeSelector** (QWidget \*const parent)
- void **populateTranslationEntries** ()



## Public Member Functions inherited from Digikam::DHBox

- **DHBox** (QWidget \*const parent=nullptr)
- QSize **minimumSizeHint** () const override
- void **setContentsMargins** (const QMargins &margins)
- void **setContentsMargins** (int left, int top, int right, int bottom)
- void **setSpacing** (int space)
- void **setStretchFactor** (QWidget \*const widget, int stretch)
- QSize **sizeHint** () const override

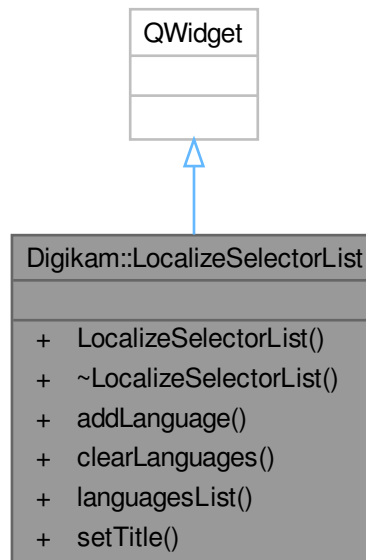
## Additional Inherited Members

## Protected Member Functions inherited from Digikam::DHBox

- **DHBox** (bool vertical, QWidget \*const parent)
- void **childEvent** (QChildEvent \*e) override

## 9.937 Digikam::LocalizeSelectorList Class Reference

Inheritance diagram for Digikam::LocalizeSelectorList:



## Signals

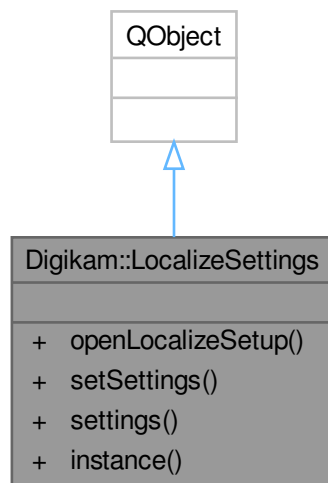
- void **signalSettingsChanged** ()

### Public Member Functions

- **LocalizeSelectorList** (QWidget \*const parent)
- void **addLanguage** (const QString &code)
- void **clearLanguages** ()
- QStringList **languagesList** () const
- void **setTitle** (const QString &title)

## 9.938 Digikam::LocalizeSettings Class Reference

Inheritance diagram for Digikam::LocalizeSettings:



### Public Types

- enum **ConfigPart** { **LocalizeConfig** , **SpellCheckConfig** , **AllConfig** }

### Signals

- void **signalOpenLocalizeSetup** ()
- void **signalSettingsChanged** ()

### Public Member Functions

- void **openLocalizeSetup** ()
- void **setSettings** (const [LocalizeContainer](#) &settings, ConfigPart config)  
*Sets the current Metadata settings and writes them to config.*
- [LocalizeContainer](#) **settings** () const  
*Returns the current Metadata settings.*

### Static Public Member Functions

- static [LocalizeSettings](#) \* [instance](#) ()  
*Global container for spell-check and localize settings.*

### Friends

- class [LocalizeSettingsCreator](#)

## 9.938.1 Member Function Documentation

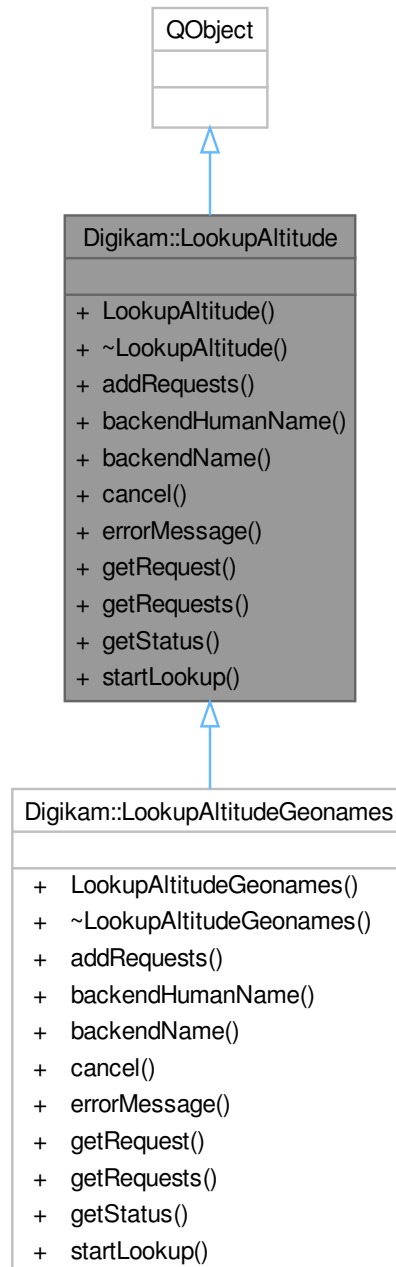
### 9.938.1.1 [instance\(\)](#)

`LocalizeSettings * Digikam::LocalizeSettings::instance ( ) [static]`

All accessor methods are thread-safe.

## 9.939 Digikam::LookupAltitude Class Reference

Inheritance diagram for Digikam::LookupAltitude:



### Classes

- class [Request](#)

### Public Types

- typedef QFlags< StatusEnum > **StatusAltitude**
- enum **StatusEnum** { **StatusInProgress** = 0 , **StatusSuccess** = 1 , **StatusCanceled** = 2 , **StatusError** = 3 }

### Signals

- void **signalDone** ()
- void **signalRequestsReady** (const QList< int > &readyRequests)

### Public Member Functions

- **LookupAltitude** (QObject \*const parent)
- virtual void **addRequests** (const Request::List &requests)=0
- virtual QString **backendHumanName** () const =0
- virtual QString **backendName** () const =0
- virtual void **cancel** ()=0
- virtual QString **errorMessage** () const =0
- virtual [Request](#) **getRequest** (const int index) const =0
- virtual Request::List **getRequests** () const =0
- virtual StatusAltitude **getStatus** () const =0
- virtual void **startLookup** ()=0

## 9.940 Digikam::LookupAltitude::Request Class Reference

### Public Types

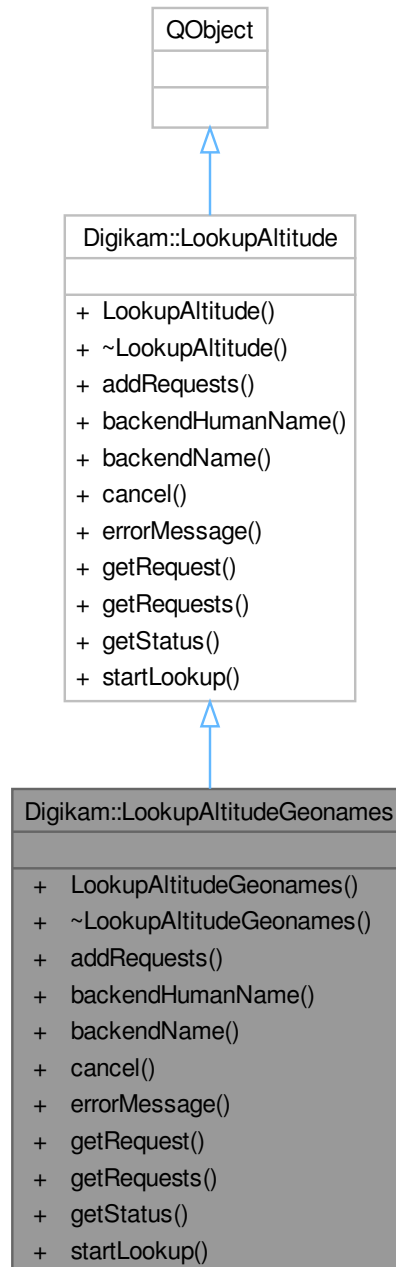
- typedef QList< [Request](#) > **List**

### Public Attributes

- [GeoCoordinates](#) **coordinates**
- QVariant **data**
- bool **success** = false

## 9.941 Digikam::LookupAltitudeGeonames Class Reference

Inheritance diagram for Digikam::LookupAltitudeGeonames:



### Public Member Functions

- **LookupAltitudeGeonames** (QObject \*const parent)
- void **addRequests** (const Request::List &requests) override

- QString [backendHumanName](#) () const override
- QString [backendName](#) () const override
- void [cancel](#) () override
- QString [errorMessage](#) () const override
- [Request](#) [getRequest](#) (const int index) const override
- Request::List [getRequests](#) () const override
- StatusAltitude [getStatus](#) () const override
- void [startLookup](#) () override

## Public Member Functions inherited from [Digikam::LookupAltitude](#)

- **LookupAltitude** (QObject \*const parent)
- virtual void **addRequests** (const Request::List &requests)=0

## Additional Inherited Members

## Public Types inherited from [Digikam::LookupAltitude](#)

- typedef QFlags< StatusEnum > **StatusAltitude**
- enum **StatusEnum** { **StatusInProgress** = 0 , **StatusSuccess** = 1 , **StatusCanceled** = 2 , **StatusError** = 3 }

## Signals inherited from [Digikam::LookupAltitude](#)

- void **signalDone** ()
- void **signalRequestsReady** (const QList< int > &readyRequests)

## 9.941.1 Member Function Documentation

### 9.941.1.1 backendHumanName()

```
QString Digikam::LookupAltitudeGeonames::backendHumanName ( ) const [override], [virtual]
```

Implements [Digikam::LookupAltitude](#).

### 9.941.1.2 backendName()

```
QString Digikam::LookupAltitudeGeonames::backendName ( ) const [override], [virtual]
```

Implements [Digikam::LookupAltitude](#).

### 9.941.1.3 cancel()

```
void Digikam::LookupAltitudeGeonames::cancel ( ) [override], [virtual]
```

Implements [Digikam::LookupAltitude](#).

#### 9.941.1.4 errorMessage()

```
QString Digikam::LookupAltitudeGeonames::errorMessage ( ) const [override], [virtual]
```

Implements [Digikam::LookupAltitude](#).

#### 9.941.1.5 getRequest()

```
LookupAltitude::Request Digikam::LookupAltitudeGeonames::getRequest (
    const int index ) const [override], [virtual]
```

Implements [Digikam::LookupAltitude](#).

#### 9.941.1.6 getRequests()

```
LookupAltitude::Request::List Digikam::LookupAltitudeGeonames::getRequests ( ) const [override],
[virtual]
```

Implements [Digikam::LookupAltitude](#).

#### 9.941.1.7 getStatus()

```
LookupAltitude::StatusAltitude Digikam::LookupAltitudeGeonames::getStatus ( ) const [override],
[virtual]
```

Implements [Digikam::LookupAltitude](#).

#### 9.941.1.8 startLookup()

```
void Digikam::LookupAltitudeGeonames::startLookup ( ) [override], [virtual]
```

Implements [Digikam::LookupAltitude](#).

## 9.942 Digikam::LookupFactory Class Reference

### Static Public Member Functions

- static [LookupAltitude](#) \* **getAltitudeLookup** (const QString &backendName, QObject \*const parent)



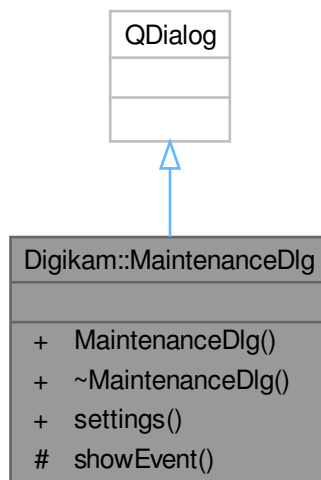
## 9.943 Digikam::MaintenanceData Class Reference

### Public Member Functions

- [Identity](#) `getIdentity ()` const
- `qulonglong` `getImageld ()` const
- `QString` `getImagePath ()` const
- [ItemInfo](#) `getItemInfo ()` const
- `qulonglong` `getSimilarityImageld ()` const
- `int` `getThumbnailId ()` const
- `void` `setIdentities (const QList< Identity > &identities)`
- `void` `setImagelds (const QList< qulonglong > &ids)`
- `void` `setImagePaths (const QList< QString > &paths)`
- `void` `setItemInfos (const QList< ItemInfo > &infos)`
- `void` `setSimilarityImagelds (const QList< qulonglong > &ids)`
- `void` `setThumbnailIds (const QList< int > &ids)`

## 9.944 Digikam::MaintenanceDlg Class Reference

Inheritance diagram for Digikam::MaintenanceDlg:



### Public Member Functions

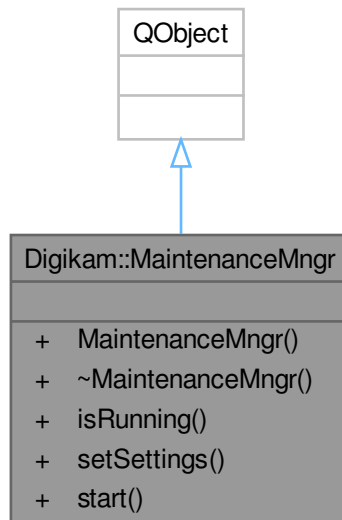
- `MaintenanceDlg` (`QWidget *const parent=nullptr`)
- [MaintenanceSettings](#) `settings ()` const

### Protected Member Functions

- `void` `showEvent (QShowEvent *)` override

## 9.945 Digikam::MaintenanceMngr Class Reference

Inheritance diagram for Digikam::MaintenanceMngr:



### Signals

- void **signalComplete** ()

### Public Member Functions

- **MaintenanceMngr** (QObject \*const parent)
- bool **isRunning** () const
- void **setSettings** (const [MaintenanceSettings](#) &settings)
- void **start** ()

## 9.946 Digikam::MaintenanceSettings Class Reference

### Public Attributes

- AlbumList **albums**
- int **autotaggingScanMode** = [AutoTagsScanSettings::AllItems](#)  
*autotagging scan mode*
- bool **autotagsAssignment** = false  
*Autotags assignment.*
- QStringList **autotagsLanguages**  
*Autotags languages.*

- bool **cleanFacesDb** = false
- bool **cleanSimilarityDb** = false
- bool **cleanThumbDb** = false
- bool **databaseCleanup** = false  
*Perform database cleanup.*
- bool **duplicates** = false  
*Scan for new items.*
- Haarface::DuplicatesSearchRestrictions **duplicatesRestriction** = Haarface::DuplicatesSearchRestrictions↔  
::None  
*The type of restrictions to apply on duplicates search results.*
- bool **faceManagement** = false  
*Scan for faces.*
- [FaceScanSettings](#) **faceSettings**  
*Face detection settings.*
- bool **fingerPrints** = false  
*Generate finger-prints.*
- int **maxSimilarity** = 100  
*Maximal similarity between items to compare, in percents.*
- bool **metadataSync** = false  
*Sync metadata and DB.*
- int **minSimilarity** = 90  
*Minimal similarity between items to compare, in percents.*
- int **modelSelectionMode** = [AutoTagsScanSettings::YOLOV5NANO](#)  
*model selection mode*
- bool **newItems** = false  
*Find new items on whole collection.*
- [ImageQualityContainer](#) **quality**  
*Image Quality Sorting Settings.*
- int **qualityScanMode** = [ImageQualitySorter::AllItems](#)  
*Mode to assign Pick Labels to items.*
- int **qualitySettingsSelected** = [ImageQualityConfSelector::GlobalSettings](#)  
*Type of quality settings selected.*
- bool **qualitySort** = false  
*Perform Image Quality Sorting.*
- bool **scanFingerPrints** = false  
*Rebuild all fingerprints or only scan missing items.*
- bool **scanThumbs** = false  
*Rebuild all thumbnails or only scan missing items.*
- bool **shrinkDatabases** = false
- int **syncDirection** = [MetadataSynchronizer::WriteFromDatabaseToFile](#)  
*Sync direction (image metadata <-> DB).*
- AlbumList **tags**
- bool **thumbnails** = false  
*Generate thumbnails.*
- bool **useMutiCoreCPU** = false  
*Use Multi-core CPU to process items.*
- bool **wholeAlbums** = true
- bool **wholeTags** = true

## 9.946.1 Member Data Documentation

### 9.946.1.1 qualityScanMode

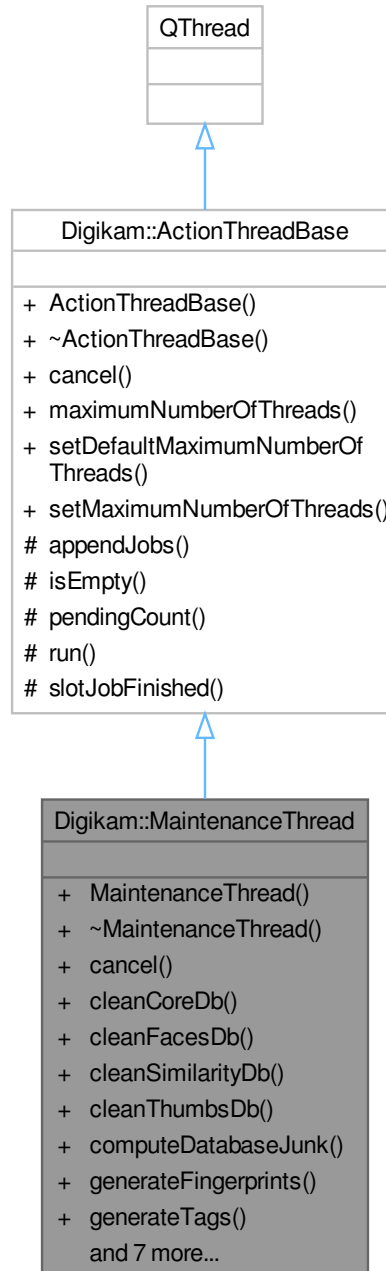
```
int Digikam::MaintenanceSettings::qualityScanMode = ImageQualitySorter::AllItems
```

#### Note

turn all items by default to prevent clearing whole Pick Labels from Collection

## 9.947 Digikam::MaintenanceThread Class Reference

Inheritance diagram for Digikam::MaintenanceThread:



### Signals

- void **signalAddItemsToProcess** (int count)  
*Signal to emit the count of additional items to process.*

- void **signalAdvance** ()
  - Emit when an item was processed and on additional information is necessary.*
- void **signalAdvance** (const [ItemInfo](#) &, const QImage &)
- void **signalAdvance** (const [ItemInfo](#) &, const QImage &, const QStringList &)
- void [signalAdvance](#) (const [ItemInfo](#) &, const QImage &, int)
  - Emit when an item have been processed.*
- void **signalAdvance** (const QImage &)
- void **signalCanceled** ()
  - Signal to emit to sub-tasks to cancel processing.*
- void **signalCompleted** ()
  - Emit when a items list have been fully processed.*
- void **signalData** (const QList< qlonglong > &staleImageIds, const QList< int > &staleThumbIds, const QList< [Identity](#) > &staleIdentities, const QList< qlonglong > &staleSimilarityImageIds)
  - Signal to emit junk data for db cleaner.*
- void **signalFinished** (bool done, bool errorFree)
  - Signal to emit after processing with info if the processing was done and if yes, without errors.*
- void **signalRemovePending** (const [ItemInfo](#) &info)
  - Signal to remove pending item from lazy sync.*
- void **signalStarted** ()
  - Emit when the task has started it's work.*

## Public Member Functions

- **MaintenanceThread** (QObject \*const parent)
- void **cancel** ()
- void **cleanCoreDb** (const QList< qlonglong > &imageIds)
- void **cleanFacesDb** (const QList< [Identity](#) > &staleIdentities)
- void **cleanSimilarityDb** (const QList< qlonglong > &imageIds)
- void **cleanThumbsDb** (const QList< int > &thumbnaillds)
- void **computeDatabaseJunk** (bool thumbsDb=false, bool facesDb=false, bool similarityDb=false)
- void **generateFingerprints** (const QList< qlonglong > &itemIds, bool rebuildAll)
- void **generateTags** (const QStringList &paths, int modelType, const QStringList &langs)
- void **generateThumbs** (const QStringList &paths)
- QString **getThumbFingerprintPath** ()
- void **removeMetadata** (const [ItemInfoList](#) &items, MetadataRemover::RemoveAction action)
- void **setUseMultiCore** (const bool b)
- void **shrinkDatabases** ()
- void **sortByImageQuality** (const QStringList &paths, const [ImageQualityContainer](#) &quality)
- void **syncMetadata** (const [ItemInfoList](#) &items, MetadataSynchronizer::SyncDirection dir, bool tagsOnly)

## Public Member Functions inherited from [Digikam::ActionThreadBase](#)

- **ActionThreadBase** (QObject \*const parent=nullptr)
- void **cancel** (bool isCancel=true)
  - Cancel processing of current jobs under progress.*
- int **maximumNumberOfThreads** () const
  - Return the maximum number of threads used to parallelize collection of job processing.*
- void [setDefaultMaximumNumberOfThreads](#) ()
  - Reset maximum number of threads used to parallelize collection of job processing to max core detected on computer.*
- void **setMaximumNumberOfThreads** (int n)
  - Adjust maximum number of threads used to parallelize collection of job processing.*

## Additional Inherited Members

### Protected Slots inherited from [Digikam::ActionThreadBase](#)

- void `slotJobFinished ()`

### Protected Member Functions inherited from [Digikam::ActionThreadBase](#)

- void `appendJobs` (const [ActionJobCollection](#) &jobs)  
*Append a collection of jobs to process into QThreadPool.*
- bool `isEmpty ()` const  
*Return true if list of pending jobs to process is empty.*
- int `pendingCount ()` const  
*Return the number of pending jobs to process.*
- void `run ()` override  
*Main thread loop used to process jobs in todo list.*

## 9.947.1 Member Function Documentation

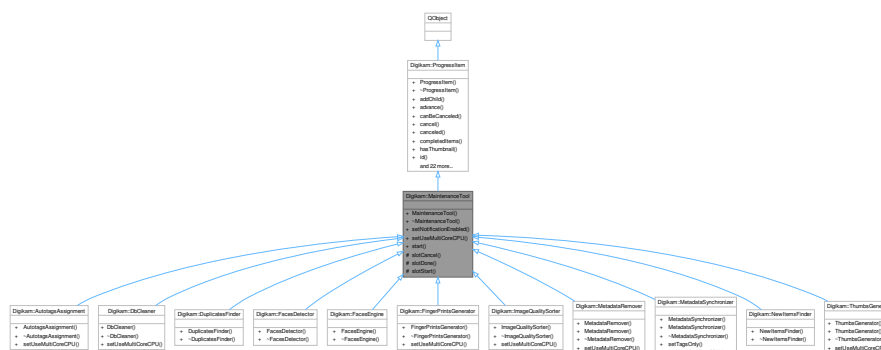
### 9.947.1.1 signalAdvance

```
void Digikam::MaintenanceThread::signalAdvance (
    const ItemInfo & ,
    const QImage & ,
    int ) [signal]
```

[QImage](#) can be used to pass item thumbnail processed.

## 9.948 Digikam::MaintenanceTool Class Reference

Inheritance diagram for [Digikam::MaintenanceTool](#):



## Public Slots

- void `start ()`

## Signals

- void **signalCanceled** ()  
*Emit when process is canceled.*
- void **signalComplete** ()  
*Emit when process is done (not canceled).*

## Signals inherited from [Digikam::ProgressItem](#)

- void [progressItemAdded](#) ([ProgressItem](#) \*item)  
*Emitted when a new [ProgressItem](#) is added.*
- void [progressItemCanceled](#) ([ProgressItem](#) \*item)  
*Emitted when an item was canceled.*
- void **progressItemCanceledById** (const QString &id)
- void [progressItemCompleted](#) ([ProgressItem](#) \*item)  
*Emitted when a progress item was completed.*
- void [progressItemLabel](#) ([ProgressItem](#) \*item, const QString &label)  
*Emitted when the label of an item changed.*
- void [progressItemProgress](#) ([ProgressItem](#) \*item, unsigned int v)  
*Emitted when the progress value of an item changes.*
- void [progressItemStatus](#) ([ProgressItem](#) \*item, const QString &mess)  
*Emitted when the status message of an item changed.*
- void [progressItemThumbnail](#) ([ProgressItem](#) \*item, const QPixmap &thumb)  
*Emitted when the thumbnail data must be set in item.*
- void [progressItemUsesBusyIndicator](#) ([ProgressItem](#) \*item, bool value)  
*Emitted when the busy indicator state of an item changes.*

## Public Member Functions

- **MaintenanceTool** (const QString &id, [ProgressItem](#) \*const parent=nullptr)
- void **setNotificationEnabled** (bool b)  
*If true, show a notification message on desktop notification manager with time elapsed to run process.*
- virtual void [setUseMultiCoreCPU](#) (bool)  
*Re-implement this method if your tool is able to use multi-core CPU to process item in parallel.*

## Public Member Functions inherited from [Digikam::ProgressItem](#)

- **ProgressItem** ([ProgressItem](#) \*const parent, const QString &id, const QString &label, const QString &status, bool [canBeCanceled](#), bool hasThumb)
- void **addChild** ([ProgressItem](#) \*const kiddo)
- bool [advance](#) (unsigned int v)  
*Advance total items processed by n values and update percentage in progressbar.*
- bool [canBeCanceled](#) () const
- void **cancel** ()
- bool **canceled** () const
- unsigned int **completedItems** () const
- bool [hasThumbnail](#) () const
- const QString & [id](#) () const
- bool **incCompletedItems** (unsigned int v=1)
- void **incTotalItems** (unsigned int v=1)



- const QString & **label** () const
- ProgressItem \* **parent** () const
- unsigned int **progress** () const
- void **removeChild** (ProgressItem \*const kiddo)
- void **reset** ()
  - Reset the progress value of this item to 0 and the status string to the empty string.*
- void **setComplete** ()
  - Tell the item it has finished.*
- bool **setCompletedItems** (unsigned int v)
- void **setLabel** (const QString &v)
- void **setProgress** (unsigned int v)
  - Set the progress (percentage of completion) value of this item.*
- void **setShowAtStart** (bool showAtStart)
  - Set the property to pop-up item when it's added in progress manager.*
- void **setStatus** (const QString &v)
  - Set the string to be used for showing this item's current status.*
- void **setThumbnail** (const QIcon &icon)
  - Sets whether this item has a thumbnail.*
- void **setTotalItems** (unsigned int v)
- void **setUsesBusyIndicator** (bool useBusyIndicator)
  - Sets whether this item uses a busy indicator instead of real progress for its progress bar.*
- bool **showAtStart** () const
- const QString & **status** () const
- bool **totalCompleted** () const
- unsigned int **totalItems** () const
- void **updateProgress** ()
  - Recalculate progress according to total/completed items and update.*
- bool **usesBusyIndicator** () const

### Protected Slots

- virtual void **slotCancel** ()
- virtual void **slotDone** ()
- virtual void **slotStart** ()

## 9.948.1 Member Function Documentation

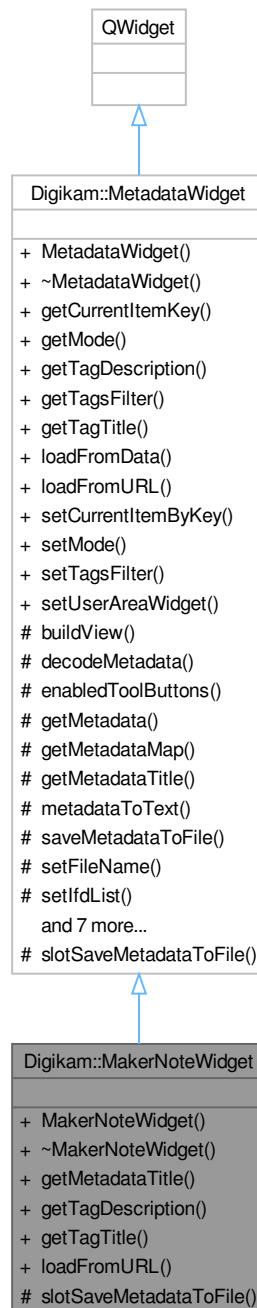
### 9.948.1.1 setUseMultiCoreCPU()

```
virtual void Digikam::MaintenanceTool::setUseMultiCoreCPU (
    bool ) [inline], [virtual]
```

Reimplemented in [Digikam::AutotagsAssignment](#), [Digikam::DbCleaner](#), [Digikam::FingerPrintsGenerator](#), [Digikam::ImageQualitySorter](#), [Digikam::MetadataRemover](#), [Digikam::MetadataSynchronizer](#), and [Digikam::ThumbsGenerator](#).

## 9.949 Digikam::MakerNoteWidget Class Reference

Inheritance diagram for Digikam::MakerNoteWidget:



### Public Member Functions

- **MakerNoteWidget** (QWidget \*const parent, const QString &name=QString())
- QString [getMetadataTitle](#) () const override

- QString [getTagDescription](#) (const QString &key) override
- QString [getTagTitle](#) (const QString &key) override
- bool [loadFromURL](#) (const QUrl &url) override

### Public Member Functions inherited from [Digikam::MetadataWidget](#)

- **MetadataWidget** (QWidget \*const parent, const QString &name=QString())
- QString [getCurrentItemKey](#) () const
- int [getMode](#) () const
- QStringList [getTagsFilter](#) () const
- virtual bool [loadFromData](#) (const QString &fileName, const [DMetadata](#) &data=[DMetadata](#)())
- void [setCurrentItemByKey](#) (const QString &itemKey)
- void [setMode](#) (int mode)
- void [setTagsFilter](#) (const QStringList &list)
- void [setUserAreaWidget](#) (QWidget \*const w)

### Protected Slots

- void [slotSaveMetadataToFile](#) () override

### Protected Slots inherited from [Digikam::MetadataWidget](#)

- virtual void [slotSaveMetadataToFile](#) ()=0

### Additional Inherited Members

### Public Types inherited from [Digikam::MetadataWidget](#)

- enum [TagFilters](#) { NONE = 0 , PHOTO , CUSTOM }

### Signals inherited from [Digikam::MetadataWidget](#)

- void [signalSetupMetadataFilters](#) ()

### Protected Member Functions inherited from [Digikam::MetadataWidget](#)

- void [enabledToolButtons](#) (bool)
- [DMetadata](#) \* [getMetadata](#) () const
- const [DMetadata::MetaDatum](#) & [getMetadataMap](#) ()
- QString [metadataToText](#) () const
- QUrl [saveMetadataToFile](#) (const QString &caption, const QString &fileFilter)
- void [setFileName](#) (const QString &fileName)
- void [setIfdList](#) (const [DMetadata::MetaDatum](#) &ifds, const QStringList &keysFilter, const QStringList &tagsFilter)
- void [setIfdList](#) (const [DMetadata::MetaDatum](#) &ifds, const QStringList &tagsFilter=QStringList())
- bool [setMetadata](#) (const [DMetadata](#) &data=[DMetadata](#)())
- virtual void [setMetadataEmpty](#) ()
- void [setMetadataMap](#) (const [DMetadata::MetaDatum](#) &data=[DMetadata::MetaDatum](#)())
- void [setup](#) ()
  - *Call this method in children class constructors to init signal/slots connections.*
- bool [storeMetadataToFile](#) (const QUrl &url, const QByteArray &metaData)
- [MetadataListView](#) \* [view](#) () const

## 9.949.1 Member Function Documentation

### 9.949.1.1 getMetadataTitle()

```
QString Digikam::MakerNoteWidget::getMetadataTitle ( ) const [override], [virtual]
```

Implements [Digikam::MetadataWidget](#).

### 9.949.1.2 getTagDescription()

```
QString Digikam::MakerNoteWidget::getTagDescription (
    const QString & key ) [override], [virtual]
```

Reimplemented from [Digikam::MetadataWidget](#).

### 9.949.1.3 getTagTitle()

```
QString Digikam::MakerNoteWidget::getTagTitle (
    const QString & key ) [override], [virtual]
```

Reimplemented from [Digikam::MetadataWidget](#).

### 9.949.1.4 loadFromURL()

```
bool Digikam::MakerNoteWidget::loadFromURL (
    const QUrl & url ) [override], [virtual]
```

Implements [Digikam::MetadataWidget](#).



- enum [LoadingPolicy](#) { [LoadingPolicyFirstRemovePrevious](#) , [LoadingPolicyPrepend](#) , [LoadingPolicySimplePrepend](#) , [LoadingPolicyAppend](#) , [LoadingPolicySimpleAppend](#) , [LoadingPolicyPreload](#) }
- enum [LoadingTaskFilter](#) { [LoadingTaskFilterAll](#) , [LoadingTaskFilterPreloading](#) }
- enum [TerminationPolicy](#) { [TerminationPolicyTerminateLoading](#) , [TerminationPolicyTerminatePreloading](#) , [TerminationPolicyWait](#) , [TerminationPolicyTerminateAll](#) }

## Public Types inherited from [Digikam::LoadSaveThread](#)

- enum [AccessMode](#) { [AccessModeRead](#) , [AccessModeReadWrite](#) }  
*used by [SharedLoadSaveThread](#) only*
- enum [NotificationPolicy](#) { [NotificationPolicyDirect](#) , [NotificationPolicyTimeLimited](#) }

## Public Types inherited from [Digikam::DynamicThread](#)

- enum [State](#) { [Inactive](#) , [Scheduled](#) , [Running](#) , [Deactivating](#) }

## Public Member Functions

- **ManagedLoadSaveThread** (QObject \*const parent=nullptr)  
*Termination is controlled by setting the TerminationPolicy Default is TerminationPolicyTerminateLoading.*
- void [load](#) (const [LoadingDescription](#) &description)  
*Append a task to load the given file to the task list.*
- void **load** (const [LoadingDescription](#) &description, [LoadingPolicy](#) policy)
- [LoadingPolicy](#) **loadingPolicy** () const
- void **save** (const [DImg](#) &image, const QString &filePath, const QString &format)  
*Append a task to save the image to the task list.*
- void [setLoadingPolicy](#) ([LoadingPolicy](#) policy)  
*Set the loading policy.*
- void **setTerminationPolicy** ([TerminationPolicy](#) terminationPolicy)
- void **stopAllTasks** ()
- void **stopLoading** (const [LoadingDescription](#) &desc, [LoadingTaskFilter](#) filter=[LoadingTaskFilterAll](#))  
*Same than previous method, but Stop and remove tasks filtered by LoadingDescription.*
- void [stopLoading](#) (const QString &filePath=QString(), [LoadingTaskFilter](#) filter=[LoadingTaskFilterAll](#))  
*Stop and remove tasks filtered by filePath and policy.*
- void [stopSaving](#) (const QString &filePath=QString())  
*Stop and remove saving tasks filtered by filePath.*
- [TerminationPolicy](#) **terminationPolicy** () const

## Public Member Functions inherited from Digikam::LoadSaveThread

- **LoadSaveThread** (QObject \*const parent=nullptr)
- **~LoadSaveThread** () override
 

*Destructor: The thread will execute all pending tasks and wait for this upon destruction.*
- void **imageLoaded** (const [LoadingDescription](#) &loadingDescription, const [DImg](#) &img) override
- void **imageSaved** (const QString &filePath, bool success) override
- void **imageStartedLoading** (const [LoadingDescription](#) &loadingDescription) override
- void **imageStartedSaving** (const QString &filePath) override
- void **load** (const [LoadingDescription](#) &description)
 

*Append a task to load the given file to the task list.*
- void **loadingProgress** (const [LoadingDescription](#) &loadingDescription, float progress) override
- void **moreCompleteLoadingAvailable** (const [LoadingDescription](#) &oldLoadingDescription, const [LoadingDescription](#) &newLoadingDescription) override
- virtual bool **querySendNotifyEvent** () const
- void **save** (const [DImg](#) &image, const QString &filePath, const QString &format)
 

*Append a task to save the image to the task list.*
- void **savingProgress** (const QString &filePath, float progress) override
- void **setNotificationPolicy** ([NotificationPolicy](#) notificationPolicy)
- virtual void **taskHasFinished** ()
- void **thumbnailLoaded** (const [LoadingDescription](#) &loadingDescription, const QImage &img) override

## Public Member Functions inherited from Digikam::DynamicThread

- **DynamicThread** (QObject \*const parent=nullptr)
 

*This class extends QRunnable, so you have to reimplement virtual void run().*
- **~DynamicThread** () override
 

*The destructor calls stop() and wait(), but if you, in your destructor, delete any data that is accessed by your run() method, you must call stop() and wait() before yourself.*
- bool **isFinished** () const
- bool **isRunning** () const
- QThread::Priority **priority** () const
- void **setEmitSignals** (bool emitThem)
- void **setPriority** (QThread::Priority priority)
 

*Sets the priority for this dynamic thread.*
- State **state** () const

## Protected Member Functions

- void **load** (const [LoadingDescription](#) &description, [LoadingMode](#) loadingMode, [AccessMode](#) mode=[AccessModeReadWrite](#))
- void **load** (const [LoadingDescription](#) &description, [LoadingMode](#) loadingMode, [LoadingPolicy](#) policy, [AccessMode](#) mode=[AccessModeReadWrite](#))
- void **loadPreview** (const [LoadingDescription](#) &description, [LoadingPolicy](#) policy)
- void **loadThumbnail** (const [LoadingDescription](#) &description)
- void **preloadThumbnail** (const [LoadingDescription](#) &description)
- void **preloadThumbnailGroup** (const QList< [LoadingDescription](#) > &descriptions)
- void **prependThumbnailGroup** (const QList< [LoadingDescription](#) > &descriptions)
- void **shutDown** ()

## Protected Member Functions inherited from [Digikam::LoadSaveThread](#)

- void **notificationReceived** ()
- void **run** () override

*Implement this pure virtual function in your subclass.*

## Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool **runningFlag** () const volatile

*In your [run\(\)](#) method, you shall regularly check for [runningFlag\(\)](#) and cleanup and return if false.*

- void **shutDown** ()

*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call [stop\(\)](#) and [wait\(\)](#), knowing that nothing will call [start\(\)](#) anymore after this 3) Be sure the thread will never be running at destruction.*

- void **start** (QMutexLocker< QMutex > &locker)

*Doing the same as [start\(\)](#), [stop\(\)](#) and [wait](#) above, provide it with a locked QMutexLocker on [mutex\(\)](#).*

- void **stop** (const QMutexLocker< QMutex > &locker)
- QMutex \* **threadMutex** () const

*This is the non-recursive mutex used to protect state variables and waiting in this class.*

- void **wait** (QMutexLocker< QMutex > &locker)

## Protected Attributes

- [LoadingPolicy](#) **m\_loadingPolicy** = [LoadingPolicyAppend](#)
- [TerminationPolicy](#) **m\_terminationPolicy** = [TerminationPolicyTerminateLoading](#)

## Protected Attributes inherited from [Digikam::LoadSaveThread](#)

- [LoadSaveTask](#) \* **m\_currentTask** = nullptr
- QMutex **m\_mutex**
- [NotificationPolicy](#) **m\_notificationPolicy** = [NotificationPolicyTimeLimited](#)
- QList< [LoadSaveTask](#) \* > **m\_todo**

## Additional Inherited Members

## Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()

*Stop computation, sets the running flag to false.*

- void **wait** ()

*Waits until the thread finishes.*



## Signals inherited from Digikam::LoadSaveThread

- void `signalImageLoaded` (const `LoadingDescription` &loadingDescription, const `DImg` &img)  
*This signal is emitted when the loading process has finished.*
- void `signalImageSaved` (const `QString` &filePath, bool success)
- void `signalImageStartedLoading` (const `LoadingDescription` &loadingDescription)  
*All signals are delivered to the thread from where the `LoadSaveThread` object has been created.*
- void `signalImageStartedSaving` (const `QString` &filePath)
- void `signalLoadingProgress` (const `LoadingDescription` &loadingDescription, float progress)  
*This signal is emitted whenever new progress info is available and the notification policy allows emitting the signal.*
- void `signalMoreCompleteLoadingAvailable` (const `LoadingDescription` &oldLoadingDescription, const `LoadingDescription` &newLoadingDescription)  
*This signal is emitted if.*
- void `signalSavingProgress` (const `QString` &filePath, float progress)
- void `signalThumbnailLoaded` (const `LoadingDescription` &loadingDescription, const `QImage` &img)

## Signals inherited from Digikam::DynamicThread

- void `finished` ()
- void `starting` ()  
*Emitted if `emitSignals` is enabled.*

## Static Public Member Functions inherited from Digikam::LoadSaveThread

- static int `exifOrientation` (const `QString` &filePath, const `DMetadata` &metadata, bool isRaw, bool fromRaw↔  
EmbeddedPreview)  
*Retrieves the Exif orientation, either from the info provider if available, or from the metadata.*
- static `LoadSaveFileInfoProvider` \* `infoProvider` ()
- static void `setInfoProvider` (`LoadSaveFileInfoProvider` \*const infoProvider)

### 9.950.1 Member Enumeration Documentation

#### 9.950.1.1 LoadingMode

enum `Digikam::ManagedLoadSaveThread::LoadingMode`

Enumerator

<code>LoadingModeNormal</code>	no sharing of loading process, no caching of image
<code>LoadingModeShared</code>	loading process is shared, image is cached

#### 9.950.1.2 LoadingPolicy

enum `Digikam::ManagedLoadSaveThread::LoadingPolicy`

## Enumerator

LoadingPolicyFirstRemovePrevious	Load image immediately, remove and stop all previous loading tasks.
LoadingPolicyPrepend	Prepend loading in front of all other tasks, but wait for the current task to finish. No other tasks will be removed, preloading tasks will be stopped and postponed.
LoadingPolicySimplePrepend	Prepend in front of all other tasks (not touching the current task). Do not check for duplicate tasks, do not check for preloading tasks.
LoadingPolicyAppend	Append loading task to the end of the list, but in front of all preloading tasks. No other tasks will be removed, preloading tasks will be stopped and postponed. This is similar to the simple <code>load()</code> operation from <a href="#">LoadSaveThread</a> , except for the special care taken for preloading.
LoadingPolicySimpleAppend	Append to the lists of tasks. Do not check for duplicate tasks, do not check for preloading tasks.
LoadingPolicyPreload	Preload image, i.e. load it with low priority when no other tasks are scheduled. All other tasks will take precedence, and preloading tasks will be stopped and postponed when another task is added. No progress info will be sent for preloaded images

## 9.950.1.3 LoadingTaskFilter

```
enum Digikam::ManagedLoadSaveThread::LoadingTaskFilter
```

## Enumerator

LoadingTaskFilterAll	filter all loading tasks
LoadingTaskFilterPreloading	filter only tasks with preloading policy

## 9.950.1.4 TerminationPolicy

```
enum Digikam::ManagedLoadSaveThread::TerminationPolicy
```

## Enumerator

TerminationPolicyTerminateLoading	Wait for saving tasks, stop and remove loading tasks This is the default.
TerminationPolicyTerminatePreloading	Wait for loading and saving tasks, stop and remove preloading tasks.
TerminationPolicyWait	Wait for all pending tasks.
TerminationPolicyTerminateAll	Stop all pending tasks.

## 9.950.2 Member Function Documentation

## 9.950.2.1 load()

```
void Digikam::ManagedLoadSaveThread::load (
    const LoadingDescription & description )
```

If there is already a task for the given file, it will possibly be rescheduled, but no second task will be added. Only loading tasks will - if required by the policy - be stopped or removed, saving tasks will not be touched.

### 9.950.2.2 setLoadingPolicy()

```
void Digikam::ManagedLoadSaveThread::setLoadingPolicy (
    LoadingPolicy policy )
```

Default is LoadingPolicyAppend. You can override the default value for each operation.

### 9.950.2.3 stopLoading()

```
void Digikam::ManagedLoadSaveThread::stopLoading (
    const QString & filePath = QString(),
    LoadingTaskFilter filter = LoadingTaskFilterAll )
```

If filePath isNull, applies to all file paths.

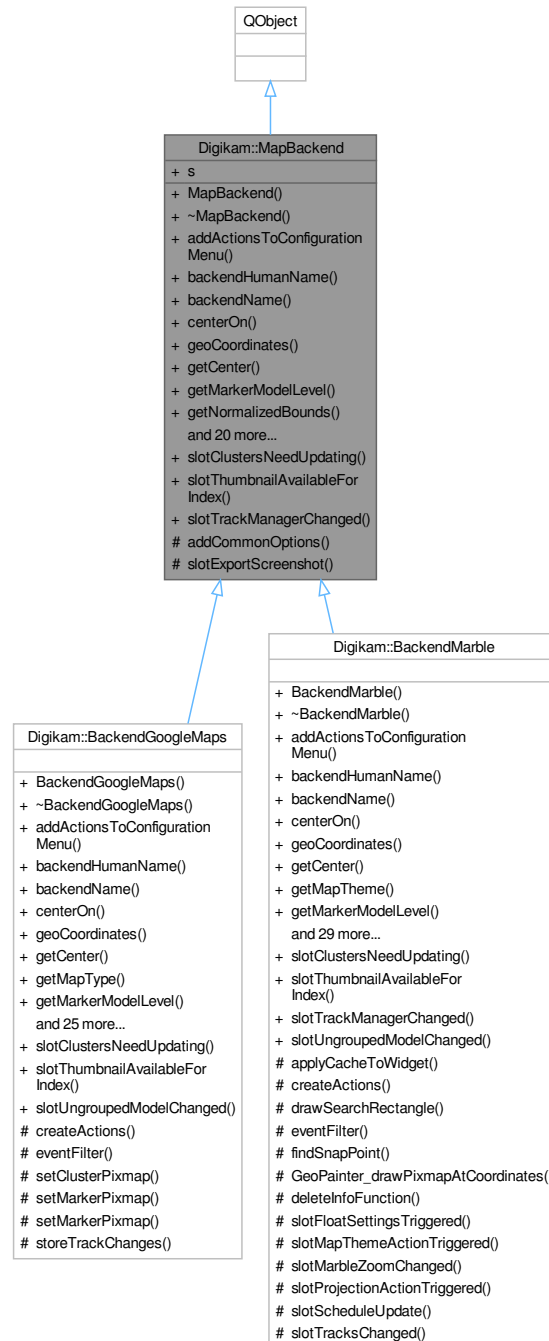
### 9.950.2.4 stopSaving()

```
void Digikam::ManagedLoadSaveThread::stopSaving (
    const QString & filePath = QString() )
```

If filePath isNull, applies to all file paths.

## 9.951 Digikam::MapBackend Class Reference

Inheritance diagram for Digikam::MapBackend:



### Public Slots

- virtual void **slotClustersNeedUpdating** ()=0
- virtual void **slotThumbnailAvailableForIndex** (const QVariant &index, const QPixmap &pixmap)
- virtual void **slotTrackManagerChanged** ()

## Signals

- void **signalBackendReadyChanged** (const QString &backendName)
- void **signalClustersClicked** (const QList &clusterIndices)
- void **signalClustersMoved** (const QList &clusterIndices, const QPair< int, QModelIndex > &snapTarget)
- void **signalMarkersMoved** (const QList &markerIndices)
- void **signalSelectionHasBeenMade** (const Digikam::GeoCoordinates::Pair &coordinates)
- void **signalZoomChanged** (const QString &newZoom)

## Public Member Functions

- **MapBackend** (const QExplicitlySharedDataPointer< [GeofaceSharedData](#) > &sharedData, QObject \*const parent)
- virtual void **addActionToConfigurationMenu** (QMenu \*const configurationMenu)=0
- virtual QString **backendHumanName** () const =0
- virtual QString **backendName** () const =0
- virtual void **centerOn** (const Marble::GeoDataLatLonBox &box, const bool useSaneZoomLevel=true)=0
- virtual bool **geoCoordinates** (const QPoint &point, [GeoCoordinates](#) \*const coordinates) const =0
- virtual [GeoCoordinates](#) **getCenter** () const =0
- virtual int **getMarkerModelLevel** ()=0
- virtual GeoCoordinates::PairList **getNormalizedBounds** ()=0
- virtual QString **getZoom** () const =0
- virtual bool **isReady** () const =0
- virtual QSize **mapSize** () const =0
- virtual QWidget \* **mapWidget** ()=0
- virtual void **mapWidgetDocked** (const bool state)=0
- virtual void **mouseModeChanged** ()=0
- virtual void **readSettingsFromGroup** (const KConfigGroup \*const group)=0
- virtual void **regionSelectionChanged** ()=0
- virtual void **releaseWidget** ([GeofaceInternalWidgetInfo](#) \*const info)=0
- virtual void **reload** ()=0
- virtual void **saveSettingsToGroup** (KConfigGroup \*const group)=0
- virtual bool **screenCoordinates** (const [GeoCoordinates](#) &coordinates, QPoint \*const point)=0
- virtual void **setActive** (const bool state)=0
- virtual void **setCenter** (const [GeoCoordinates](#) &coordinate)=0
- virtual void **setZoom** (const QString &newZoom)=0
- virtual void **updateActionAvailability** ()=0
- virtual void **updateClusters** ()=0
- virtual void **updateMarkers** ()=0
- virtual void **zoomIn** ()=0
- virtual void **zoomOut** ()=0

## Public Attributes

- const QExplicitlySharedDataPointer< [GeofaceSharedData](#) > **s**

## Protected Slots

- void **slotExportScreenshot** ()

## Protected Member Functions

- void **addCommonOptions** (QMenu \*const configurationMenu)

## 9.951.1 Member Function Documentation

### 9.951.1.1 centerOn()

```
virtual void Digikam::MapBackend::centerOn (
    const Marble::GeoDataLatLonBox & box,
    const bool useSaneZoomLevel = true ) [pure virtual]
```

Implemented in [Digikam::BackendMarble](#), and [Digikam::BackendGoogleMaps](#).

### 9.951.1.2 mapWidget()

```
virtual QWidget * Digikam::MapBackend::mapWidget ( ) [pure virtual]
```

Implemented in [Digikam::BackendMarble](#).

### 9.951.1.3 mouseModeChanged()

```
virtual void Digikam::MapBackend::mouseModeChanged ( ) [pure virtual]
```

Implemented in [Digikam::BackendGoogleMaps](#).

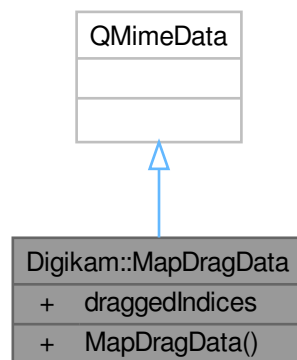
### 9.951.1.4 setActive()

```
virtual void Digikam::MapBackend::setActive (
    const bool state ) [pure virtual]
```

Implemented in [Digikam::BackendGoogleMaps](#).

## 9.952 Digikam::MapDragData Class Reference

Inheritance diagram for Digikam::MapDragData:

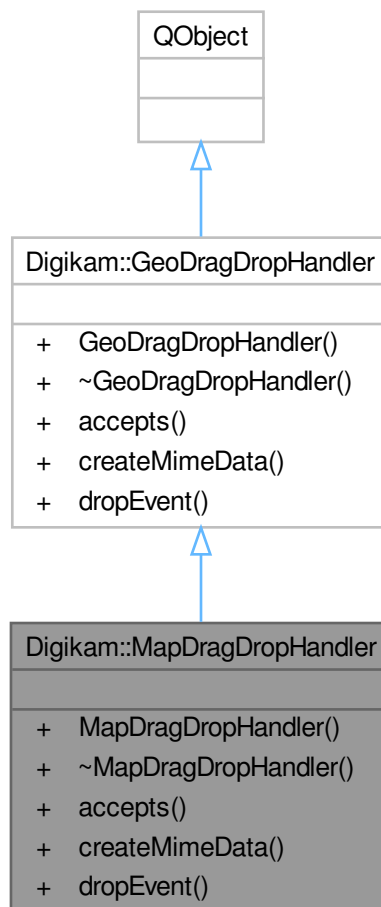


**Public Attributes**

- `QList< QPersistentModelIndex >` **draggedIndices**

**9.953 Digikam::MapDragDropHandler Class Reference**

Inheritance diagram for Digikam::MapDragDropHandler:

**Public Member Functions**

- **MapDragDropHandler** (`QAbstractItemModel *const`, [GPSGeofaceModelHelper](#) \*const parent)
- `Qt::DropAction` [accepts](#) (`const QDropEvent *e`) override
- `QMimeData *` [createMimeData](#) (`const QList< QPersistentModelIndex > &modelIndices`) override
- `bool` [dropEvent](#) (`const QDropEvent *e`, `const GeoCoordinates &dropCoordinates`) override

**Public Member Functions inherited from [Digikam::GeoDragDropHandler](#)**

- **GeoDragDropHandler** (`QObject *const parent=nullptr`)

## 9.953.1 Member Function Documentation

### 9.953.1.1 accepts()

```
Qt::DropAction Digikam::MapDragDropHandler::accepts (
    const QDropEvent * e ) [override], [virtual]
```

Implements [Digikam::GeoDragDropHandler](#).

### 9.953.1.2 createMimeData()

```
QMimeData * Digikam::MapDragDropHandler::createMimeData (
    const QList< QPersistentModelIndex > & modelIndices ) [override], [virtual]
```

Implements [Digikam::GeoDragDropHandler](#).

### 9.953.1.3 dropEvent()

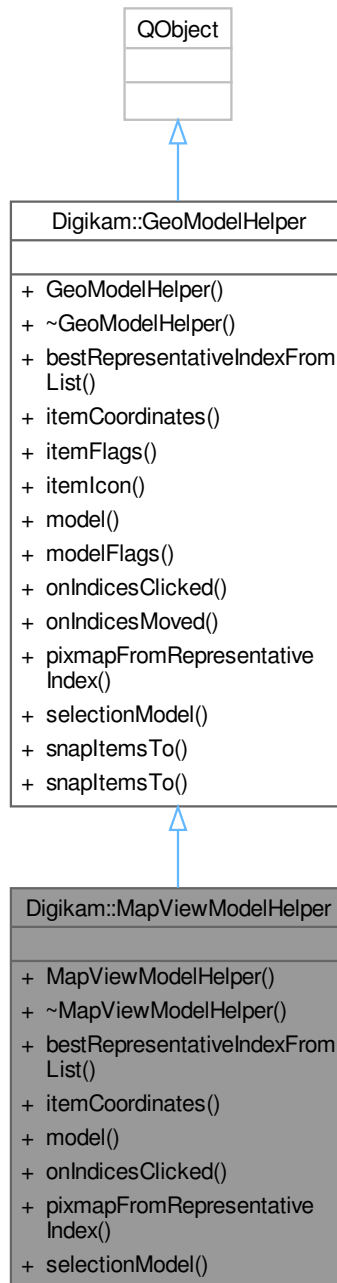
```
bool Digikam::MapDragDropHandler::dropEvent (
    const QDropEvent * e,
    const GeoCoordinates & dropCoordinates ) [override], [virtual]
```

Implements [Digikam::GeoDragDropHandler](#).



## 9.954 Digikam::MapViewModelHelper Class Reference

Inheritance diagram for Digikam::MapViewModelHelper:



### Signals

- void **signalFilteredImages** (const QList< qlonglong > &idList)

## Signals inherited from [Digikam::GeoModelHelper](#)

- void **signalModelChangedDrastically** ()
- void **signalThumbnailAvailableForIndex** (const QPersistentModelIndex &index, const QPixmap &pixmap)
- void **signalVisibilityChanged** ()

## Public Member Functions

- **MapViewModelHelper** (QItemSelectionModel \*const selection, [DCategorizedSortFilterProxyModel](#) \*const filterModel, QObject \*const parent, const MapWidgetView::Application application)
- **~MapViewModelHelper** () override  
*Destructor.*
- QPersistentModelIndex **bestRepresentativeIndexFromList** (const QList< QPersistentModelIndex > &list, const int sortKey) override  
*This function finds the best representative marker from a group of markers.*
- bool **itemCoordinates** (const QModelIndex &index, [GeoCoordinates](#) \*const coordinates) const override  
*Gets the coordinates of a marker found at current model index.*
- QAbstractItemModel \* **model** () const override
- void **onIndicesClicked** (const QList< QPersistentModelIndex > &clickedIndices) override  
*This functions is called when one clicks on a thumbnail.*
- QPixmap **pixmapFromRepresentativeIndex** (const QPersistentModelIndex &index, const QSize &size) override  
*This function retrieves the thumbnail for an index.*
- QItemSelectionModel \* **selectionModel** () const override

## Public Member Functions inherited from [Digikam::GeoModelHelper](#)

- **GeoModelHelper** (QObject \*const parent=nullptr)
- virtual PropertyFlags **itemFlags** (const QModelIndex &index) const
- virtual bool **itemIcon** (const QModelIndex &index, QPoint \*const offset, QSize \*const size, QPixmap \*const pixmap, QUrl \*const url) const  
*these are necessary for ungrouped models*
- virtual PropertyFlags **modelFlags** () const
- virtual void **onIndicesMoved** (const QList< QPersistentModelIndex > &movedIndices, const [GeoCoordinates](#) &targetCoordinates, const QPersistentModelIndex &targetSnapIndex)
- virtual void **snapItemsTo** (const QModelIndex &targetIndex, const QList< QModelIndex > &snappedIndices)
- void **snapItemsTo** (const QModelIndex &targetIndex, const QList< QPersistentModelIndex > &snappedIndices)

## Additional Inherited Members

## Public Types inherited from [Digikam::GeoModelHelper](#)

- enum **PropertyFlag** { **FlagNull** = 0 , **FlagVisible** = 1 , **FlagMovable** = 2 , **FlagSnaps** = 4 }
- typedef QFlags< PropertyFlag > **PropertyFlags**

### 9.954.1 Member Function Documentation

#### 9.954.1.1 bestRepresentativeIndexFromList()

```
QPersistentModelIndex Digikam::MapViewModelHelper::bestRepresentativeIndexFromList (
    const QList< QPersistentModelIndex > & list,
    const int sortKey ) [override], [virtual]
```

This is needed to display a thumbnail for a marker group.

## Parameters

<i>list</i>	A list containing markers.
<i>sortKey</i>	Determines the sorting options and is actually of type <code>GPSItemInfoSorter::SortOptions</code>

## Returns

Returns the index of the marker.

Reimplemented from [Digikam::GeoModelHelper](#).

**9.954.1.2 itemCoordinates()**

```
bool Digikam::MapViewModelHelper::itemCoordinates (
    const QModelIndex & index,
    GeoCoordinates *const coordinates ) const [override], [virtual]
```

## Parameters

<i>index</i>	Current model index.
<i>coordinates</i>	Here will be returned the coordinates of the current marker.

## Returns

True, if the marker has coordinates.

Implements [Digikam::GeoModelHelper](#).

**9.954.1.3 model()**

```
QAbstractItemModel * Digikam::MapViewModelHelper::model ( ) const [override], [virtual]
```

## Returns

Returns digiKam's filter model.

Implements [Digikam::GeoModelHelper](#).

**9.954.1.4 onIndicesClicked()**

```
void Digikam::MapViewModelHelper::onIndicesClicked (
    const QList< QPersistentModelIndex > & clickedIndices ) [override], [virtual]
```

## Parameters

<i>clickedIndices</i>	A list containing the marker indices belonging to the group whose thumbnail has been clicked.
-----------------------	---

Reimplemented from [Digikam::GeoModelHelper](#).

### 9.954.1.5 pixmapFromRepresentativeIndex()

```
QPixmap Digikam::MapViewModelHelper::pixmapFromRepresentativeIndex (
    const QPersistentModelIndex & index,
    const QSize & size ) [override], [virtual]
```

#### Parameters

<i>index</i>	The marker's index.
<i>size</i>	The size of the thumbnail.

#### Returns

If the thumbnail has been loaded in the [ThumbnailLoadThread](#) instance, it is returned. If not, a QPixmap is returned and [ThumbnailLoadThread](#)'s signal named `signalThumbnailLoaded` is emitted when the thumbnail becomes available.

Reimplemented from [Digikam::GeoModelHelper](#).

### 9.954.1.6 selectionModel()

```
QItemSelectionModel * Digikam::MapViewModelHelper::selectionModel ( ) const [override], [virtual]
```

#### Returns

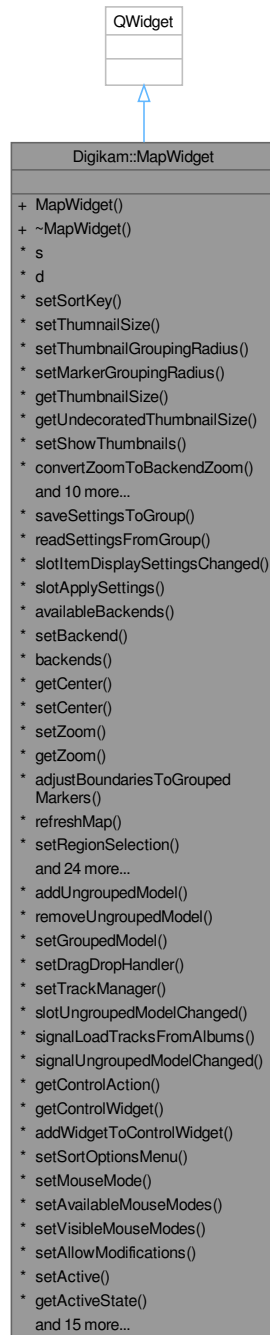
Returns digiKam's selection model.

Implements [Digikam::GeoModelHelper](#).

## 9.955 Digikam::MapWidget Class Reference

The central map view class of geolocation interface.

Inheritance diagram for Digikam::MapWidget:



### Public Member Functions

- **MapWidget** (QWidget \*const parent=nullptr)
- **~MapWidget** () override

### Appearance

- void **setSortKey** (const int sortKey)

- void **setThumbnailSize** (const int newThumbnailSize)
- void **setThumbnailGroupingRadius** (const int newGroupingRadius)
- void **setMarkerGroupingRadius** (const int newGroupingRadius)
- int **getThumbnailSize** () const
- int **getUndecoratedThumbnailSize** () const
- void **setShowThumbnails** (const bool state)
- QString **convertZoomToBackendZoom** (const QString &someZoom, const QString &targetBackend) const
- void **getColorInfos** (const int clusterIndex, QColor \*fillColor, QColor \*strokeColor, Qt::PenStyle \*strokeStyle, QString \*labelText, QColor \*labelColor, const GeoGroupState \*const overrideSelection=nullptr, const int \*const overrideCount=nullptr) const  
*Return color and style information for rendering the cluster.*
- void **getColorInfos** (const GeoGroupState groupState, const int nMarkers, QColor \*fillColor, QColor \*strokeColor, Qt::PenStyle \*strokeStyle, QString \*labelText, QColor \*labelColor) const
- void **slotShowThumbnailsChanged** ()
- void **slotZoomIn** ()
- void **slotZoomOut** ()
- void **slotDecreaseThumbnailSize** ()
- void **slotIncreaseThumbnailSize** ()
- void **stopThumbnailTimer** ()
- void **signalRemoveCurrentFilter** ()
- void **signalStickyModeChanged** ()

#### Settings Management related functions

- void **saveSettingsToGroup** (KConfigGroup \*const group)
- void **readSettingsFromGroup** (const KConfigGroup \*const group)
- void **slotItemDisplaySettingsChanged** ()

#### Map related functions

- QStringList **availableBackends** () const
- bool **setBackend** (const QString &backendName)
- QList< [MapBackend](#) \* > **backends** () const
- [GeoCoordinates](#) **getCenter** () const
- void **setCenter** (const [GeoCoordinates](#) &coordinate)
- void **setZoom** (const QString &newZoom)
- QString **getZoom** ()
- void **adjustBoundariesToGroupedMarkers** (const bool useSaneZoomLevel=true)  
*Adjusts the visible map area such that all grouped markers are visible.*
- void **refreshMap** ()
- void **setRegionSelection** (const [GeoCoordinates::Pair](#) &region)
- [GeoCoordinates::Pair](#) **getRegionSelection** ()
- void **clearRegionSelection** ()
- void **updateMarkers** ()
- void **updateClusters** ()
- void **markClustersAsDirty** ()
- QPixmap **getDecoratedPixmapForCluster** (const int clusterId, const [GeoGroupState](#) \*const selectedState, const int \*const countOverride, QPoint \*const centerPoint)
- QVariant **getClusterRepresentativeMarker** (const int clusterIndex, const int sortKey)
- void **slotBackendReadyChanged** (const QString &backendName)
- void **slotChangeBackend** (QAction \*action)
- void **slotBackendZoomChanged** (const QString &newZoom)
- void **slotClustersMoved** (const QList &clusterIndices, const QPair< int, [QModelIndex](#) > &snapTarget)

- void [slotClustersClicked](#) (const QList &clusterIndices)
- void **slotLazyReclusteringRequestCallback** ()  
*Helper function to buffer reclustering.*
- void **slotRequestLazyReclustering** ()  
*Request reclustering, repeated calls should generate only one actual update of the clusters.*
- void **slotRemoveCurrentRegionSelection** ()
- void [slotNewSelectionFromMap](#) (const Digikam::GeoCoordinates::Pair &sel)
- bool **currentBackendReady** () const
- void [applyCacheToBackend](#) ()
- void **saveBackendToCache** ()
- void **setShowPlaceholderWidget** (const bool state)
- void **setMapWidgetInFrame** (QWidget \*const widgetForFrame)  
*Set widgetForFrame as the widget in the frame, but does not show it.*
- void **removeMapWidgetFromFrame** ()
- void **slotClustersNeedUpdating** ()
- void **signalRegionSelectionChanged** ()

### Data Management

- void [addUngroupedModel](#) (GeoModelHelper \*const modelHelper)
- void [removeUngroupedModel](#) (GeoModelHelper \*const modelHelper)
- void [setGroupedModel](#) (AbstractMarkerTiler \*const markerModel)
- void **setDragDropHandler** (GeoDragDropHandler \*const dragDropHandler)
- void **setTrackManager** (TrackManager \*const trackManager)
- void **slotUngroupedModelChanged** ()
- void **signalLoadTracksFromAlbums** ()
- void **signalUngroupedModelChanged** (const int index)

### UI setup

- QAction \* **getControlAction** (const QString &actionName)
- QWidget \* **getControlWidget** ()  
*Returns the control widget instance.*
- void **addWidgetToControlWidget** (QWidget \*const newWidget)
- void **setSortOptionsMenu** (QMenu \*const sortMenu)
- void **setMouseMode** (const GeoMouseModes mouseMode)
- void **setAvailableMouseModes** (const GeoMouseModes mouseModes)
- void **setVisibleMouseModes** (const GeoMouseModes mouseModes)
- void **setAllowModifications** (const bool state)
- void **setActive** (const bool state)
- bool **getActiveState** ()
- bool **getStickyModeState** () const
- void **setStickyModeState** (const bool state)
- void **setVisibleExtraActions** (const GeoExtraActions actions)
- void **setEnabledExtraActions** (const GeoExtraActions actions)
- void [slotMouseModeChanged](#) (QAction \*triggeredAction)
- void **rebuildConfigurationMenu** ()
- void **createActions** ()
- void **createActionsForBackendSelection** ()
- void **dropEvent** (QDropEvent \*event) override
- void **dragMoveEvent** (QDragMoveEvent \*event) override
- void [dragEnterEvent](#) (QDragEnterEvent \*event) override
- void **dragLeaveEvent** (QDragLeaveEvent \*event) override
- void [slotUpdateActionsEnabled](#) ()
- void **slotStickyModeChanged** ()
- void **signalMouseModeChanged** (const Digikam::GeoMouseModes &currentMouseMode)

## 9.955.1 Detailed Description

The [MapWidget](#) class is the central widget of geolocation interface. It provides a widget which can display maps using either the Marble or Google Maps backend. Using a model, items can be displayed on the map. For models containing only a small number of items, the items can be shown directly, but for models with a larger number of items, the items can also be grouped. Currently, any number of ungrouped models can be shown, but only one grouped model. Item selection models can also be used along with the models, to interact with the selection states of the items on the map. In order to use a model with geolocation interface, however, a model helper has to be implemented, which extracts data from the model that is not provided by the Qt part of a model's API.

Now, a brief introduction on how to get geolocation interface working is provided:

- First, an instance of [MapWidget](#) has to be created.
- Next, [GeoModelHelper](#) has to be subclassed and at least the pure virtual functions have to be implemented.
- To show the model's data ungrouped, the model helper has to be added to [MapWidget](#) instance using `addUngroupedModel`.
- To show the model's data grouped, an instance of [AbstractMarkerTiler](#) has to be created and the model helper has to be set to it using `setMarkerGeoModelHelper`. The [AbstractMarkerTiler](#) has then to be given to [MapWidget](#) using `setGroupedModel`. If the items to be displayed do not reside in a model, a subclass of [AbstractMarkerTiler](#) can be created which returns just the number of items in a particular area, and picks representative items for thumbnails.
- To handle dropping of items from the host applications UI onto the map, `DragDropHandler` has to be subclassed as well and added to the model using `setDragDropHandler`.
- Finally, `setActive()` has to be called to tell the widget that it should start displaying things.

## 9.955.2 Constructor & Destructor Documentation

### 9.955.2.1 `~MapWidget()`

```
Digikam::MapWidget::~MapWidget ( ) [override]
```

## 9.955.3 Member Function Documentation

### 9.955.3.1 `addUngroupedModel()`

```
void Digikam::MapWidget::addUngroupedModel (
    GeoModelHelper *const modelHelper )
```

### 9.955.3.2 `adjustBoundariesToGroupedMarkers()`

```
void Digikam::MapWidget::adjustBoundariesToGroupedMarkers (
    const bool useSaneZoomLevel = true )
```

Note that a call to this function currently has no effect if the widget has been set inactive via `setActive()` or the backend is not yet ready.



## Parameters

<i>useSaneZoomLevel</i>	Stop zooming at a sane level, if markers are too close together.
-------------------------	--

**9.955.3.3 applyCacheToBackend()**

```
void Digikam::MapWidget::applyCacheToBackend ( ) [protected]
```

**9.955.3.4 convertZoomToBackendZoom()**

```
QString Digikam::MapWidget::convertZoomToBackendZoom (
    const QString & someZoom,
    const QString & targetBackend ) const
```

**9.955.3.5 dragEnterEvent()**

```
void Digikam::MapWidget::dragEnterEvent (
    QDragEnterEvent * event ) [override], [protected]
```

**9.955.3.6 getColorInfos() [1/2]**

```
void Digikam::MapWidget::getColorInfos (
    const GeoGroupState groupState,
    const int nMarkers,
    QColor * fillColor,
    QColor * strokeColor,
    Qt::PenStyle * strokeStyle,
    QString * labelText,
    QColor * labelColor ) const
```

**9.955.3.7 getColorInfos() [2/2]**

```
void Digikam::MapWidget::getColorInfos (
    const int clusterIndex,
    QColor * fillColor,
    QColor * strokeColor,
    Qt::PenStyle * strokeStyle,
    QString * labelText,
    QColor * labelColor,
    const GeoGroupState *const overrideSelection = nullptr,
    const int *const overrideCount = nullptr ) const
```

## Parameters

<i>clusterIndex</i>	Index of the cluster
<i>fillColor</i>	Color used to fill the circle
<i>strokeColor</i>	Color used for the stroke around the circle
<i>strokeStyle</i>	Style used to draw the stroke around the circle
<i>labelText</i>	Text for the label
<i>labelColor</i>	Color for the label text
<i>overrideSelection</i>	Get the colors for a different selection state
<i>overrideCount</i>	Get the colors for a different amount of markers

### 9.955.3.8 getDecoratedPixmapForCluster()

```
QPixmap Digikam::MapWidget::getDecoratedPixmapForCluster (
    const int clusterId,
    const GeoGroupState *const selectedStateOverride,
    const int *const countOverride,
    QPoint *const centerPoint )
```

### 9.955.3.9 removeUngroupedModel()

```
void Digikam::MapWidget::removeUngroupedModel (
    GeoModelHelper *const modelHelper )
```

### 9.955.3.10 setBackend()

```
bool Digikam::MapWidget::setBackend (
    const QString & backendName )
```

### 9.955.3.11 setGroupedModel()

```
void Digikam::MapWidget::setGroupedModel (
    AbstractMarkerTiler *const markerModel )
```

### 9.955.3.12 setSortKey()

```
void Digikam::MapWidget::setSortKey (
    const int sortKey )
```

### 9.955.3.13 setThumbnailSize()

```
void Digikam::MapWidget::setThumbnailSize (
    const int newThumbnailSize )
```

### 9.955.3.14 slotClustersClicked

```
void Digikam::MapWidget::slotClustersClicked (
    const QList & clusterIndices ) [protected], [slot]
```

### 9.955.3.15 slotClustersMoved

```
void Digikam::MapWidget::slotClustersMoved (
    const QList & clusterIndices,
    const QPair< int, QModelIndex > & snapTarget ) [protected], [slot]
```

**9.955.3.16 slotItemDisplaySettingsChanged**

```
void Digikam::MapWidget::slotItemDisplaySettingsChanged ( ) [protected], [slot]
```

**9.955.3.17 slotMouseModeChanged**

```
void Digikam::MapWidget::slotMouseModeChanged (
    QAction * triggeredAction ) [protected], [slot]
```

**9.955.3.18 slotNewSelectionFromMap**

```
void Digikam::MapWidget::slotNewSelectionFromMap (
    const Digikam::GeoCoordinates::Pair & sel ) [protected], [slot]
```

**9.955.3.19 slotUpdateActionsEnabled**

```
void Digikam::MapWidget::slotUpdateActionsEnabled ( ) [slot]
```

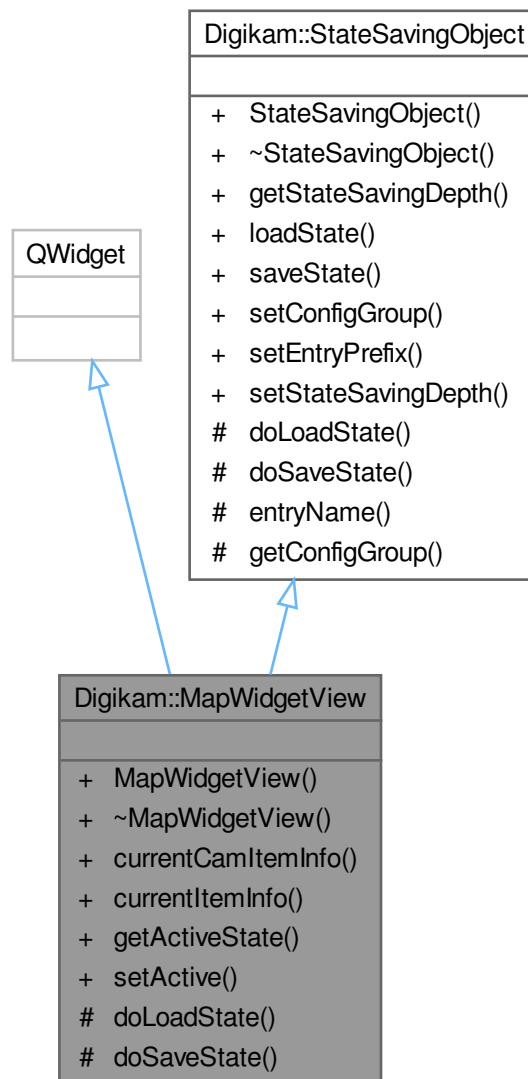
**9.955.3.20 updateClusters()**

```
void Digikam::MapWidget::updateClusters ( )
```

**9.956 Digikam::MapWidgetView Class Reference**

Class containing digiKam's central map view.

Inheritance diagram for Digikam::MapView:



## Public Types

- enum **Application** { **ApplicationDigikam** = 1 , **ApplicationImportUI** = 2 }

## Public Types inherited from Digikam::StateSavingObject

- enum **StateSavingDepth** { **INSTANCE** , **DIRECT\_CHILDREN** , **RECURSIVE** }

*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

## Public Member Functions

- [MapWidgetView](#) (QItemSelectionModel \*const selectionModel, [DCategorizedSortFilterProxyModel](#) \*const imageFilterModel, QWidget \*const parent, const Application application)  
*Constructor.*
- [~MapWidgetView](#) () override  
*Destructor.*
- [CamItemInfo currentCamItemInfo](#) () const  
*Returns the [CamItemInfo](#) for the current image.*
- [ItemInfo currentItemInfo](#) () const  
*Returns the [ItemInfo](#) for the current image.*
- bool [getActiveState](#) () const
- void [setActive](#) (const bool state)  
*Set the map active/inactive.*

## Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (QObject \*const host)  
*Constructor.*
- virtual [~StateSavingObject](#) ()  
*Destructor.*
- [StateSavingDepth getStateSavingDepth](#) () const  
*Returns the depth used for state saving or loading.*
- void [loadState](#) ()  
*Invokes loading the class' state.*
- void [saveState](#) ()  
*Invokes saving the class' state.*
- virtual void [setConfigGroup](#) (const KConfigGroup &group)  
*Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void [setEntryPrefix](#) (const QString &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void [setStateSavingDepth](#) (const [StateSavingDepth](#) depth)  
*Sets the depth used for state saving or loading.*

## Protected Member Functions

- void [doLoadState](#) () override  
*Implement this hook method for state loading.*
- void [doSaveState](#) () override  
*Implement this hook method for state saving.*

## Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- QString [entryName](#) (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup [getConfigGroup](#) () const  
*Returns the config group that must be used for state saving and loading.*

## 9.956.1 Constructor & Destructor Documentation

### 9.956.1.1 MapWidgetView()

```
Digikam::MapWidgetView::MapWidgetView (
    QItemSelectionModel *const selectionModel,
    DategorizedSortFilterProxyModel *const imageFilterModel,
    QWidget *const parent,
    const Application application ) [explicit]
```

#### Parameters

<i>selectionModel</i>	digikam's selection model
<i>imageFilterModel</i>	digikam's filter model
<i>parent</i>	the parent object
<i>application</i>	the type of application host

## 9.956.2 Member Function Documentation

### 9.956.2.1 currentCamItemInfo()

```
CamItemInfo Digikam::MapWidgetView::currentCamItemInfo ( ) const
```

### 9.956.2.2 currentItemInfo()

```
ItemInfo Digikam::MapWidgetView::currentItemInfo ( ) const
```

### 9.956.2.3 doLoadState()

```
void Digikam::MapWidgetView::doLoadState ( ) [override], [protected], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

### 9.956.2.4 doSaveState()

```
void Digikam::MapWidgetView::doSaveState ( ) [override], [protected], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

### 9.956.2.5 getActiveState()

```
bool Digikam::MapWidgetView::getActiveState ( ) const
```

#### Returns

The map's active state

### 9.956.2.6 setActive()

```
void Digikam::MapWidgetView::setActive (
    const bool state )
```

## Parameters

<code>state</code>	If true, the map is active.
--------------------	-----------------------------

## 9.957 Digikam::Mat Struct Reference

[Mat](#):

### Public Attributes

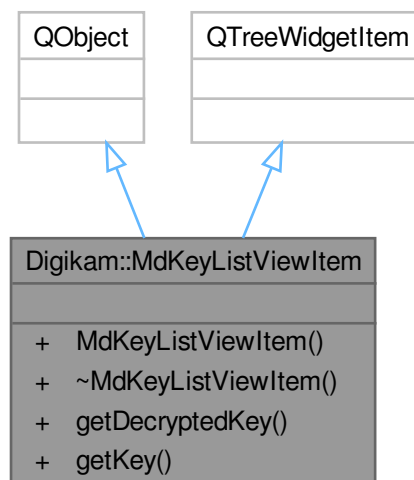
- `int cols`  
*Number of columns in the matrix.*
- `double * data`  
*Content of the matrix.*
- `int rows`  
*Number of rows in the matrix.*

### 9.957.1 Detailed Description

Normal matrix type. Indices range from [0, rows - 1 ] and [0, cols - 1].

## 9.958 Digikam::MdKeyListViewItem Class Reference

Inheritance diagram for Digikam::MdKeyListViewItem:

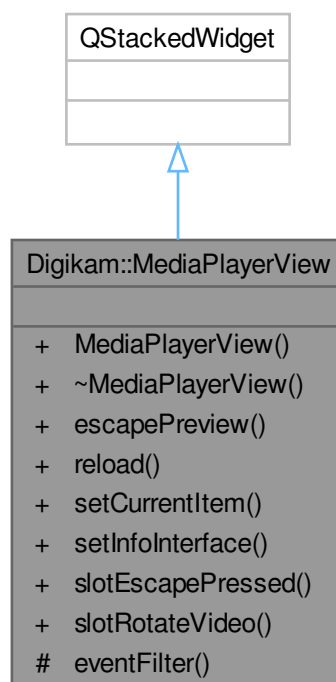


### Public Member Functions

- **MdKeyListViewItem** (QTreeWidget \*const parent, const QString &key)
- QString **getDecryptedKey** () const
- QString **getKey** () const

## 9.959 Digikam::MediaPlayerView Class Reference

Inheritance diagram for Digikam::MediaPlayerView:



### Public Slots

- void **slotEscapePressed** ()
- void **slotRotateVideo** ()

### Signals

- void **signalEscapePreview** ()
- void **signalNextItem** ()
- void **signalPrevItem** ()



**Public Member Functions**

- **MediaPlayerView** (QWidget \*const parent)
- void **escapePreview** ()
- void **reload** ()
- void **setCurrentItem** (const QUrl &url=QUrl(), bool hasPrevious=false, bool hasNext=false)
- void **setInfoInterface** (DInfoInterface \*const iface)

**Protected Member Functions**

- bool **eventFilter** (QObject \*watched, QEvent \*event) override

**9.960 Digikam::MetadataHub Class Reference****Public Types**

- enum **Status** { [MetadataInvalid](#) , [MetadataAvailable](#) }  
*The status enum describes the result of joining several metadata sets.*
- typedef QFlags< WriteComponents > **WriteComponent**
- enum **WriteComponents** {  
  **WRITE\_DATETIME** = 1 , **WRITE\_TITLE** = 2 , **WRITE\_COMMENTS** = 4 , **WRITE\_PICKLABEL** = 8 ,  
  **WRITE\_COLORLABEL** = 16 , **WRITE\_RATING** = 32 , **WRITE\_TEMPLATE** = 64 , **WRITE\_TAGS** = 128 ,  
  **WRITE\_POSITION** = 256 , **WRITE\_ALL** = 511 }
- enum **WriteMode** { [FullWrite](#) , [FullWriteIfChanged](#) , [PartialWrite](#) }

**Public Member Functions**

- **MetadataHub** ()  
*Constructs a [MetadataHub](#).*
- QStringList **cleanupTags** (const QStringList &toClean)  
*cleanupTags remove duplicates and obsolete tags before setting metadata*
- void **load** (const [ItemInfo](#) &info)  
*Add metadata information contained in the [ItemInfo](#) object.*
- void **reset** ()
- bool **willWriteMetadata** (Digikam::MetadataHub::WriteComponent writeMode=WRITE\_ALL, const [MetaEngineSettingsContainer](#) &settings=[MetaEngineSettings::instance\(\)](#) ->settings()) const  
*With the currently applied changes, the given writeMode and settings.*
- bool **write** (const [DImg](#) &image, WriteComponent writeMode=WRITE\_ALL, bool ignoreLazySync=false, const [MetaEngineSettingsContainer](#) &settings=[MetaEngineSettings::instance\(\)](#) ->settings())  
*Constructs a meta engine object from the metadata stored in the given [DImg](#) object, calls the above method, and changes the stored metadata in the [DImg](#) object.*
- bool **write** (const QString &filePath, WriteComponent writeMode=WRITE\_ALL, bool ignoreLazySync=false, const [MetaEngineSettingsContainer](#) &settings=[MetaEngineSettings::instance\(\)](#) ->settings())  
*Constructs a meta engine object for given filePath, calls the above method, writes the changes out to the file, and notifies the [ItemAttributesWatch](#).*
- bool **writeTags** (const [DMetadata](#) &metadata, bool saveTags)  
*Used to deduplicate code from writeTags and usual write, all write to tags operations must be done here.*
- bool **writeTags** (const QString &filePath, WriteComponent writeMode=WRITE\_ALL, const [MetaEngineSettingsContainer](#) &settings=[MetaEngineSettings::instance\(\)](#) ->settings())  
*Will write only Tags to image.*
- void **writeToBaloo** (const QString &filePath, const [MetaEngineSettingsContainer](#) &settings=[MetaEngineSettings::instance\(\)](#) ->settings())  
*write tags, comments and rating to KDE Nepomuk replacement: Baloo*
- bool **writeToMetadata** (const [ItemInfo](#) &info, WriteComponent writeMode=WRITE\_ALL, bool ignoreLazySync=false, const [MetaEngineSettingsContainer](#) &settings=[MetaEngineSettings::instance\(\)](#) ->settings())  
*writeToMetadata write to metadata using image info to retrieve tags and filepath use this method when multiple image infos are loaded in hub*

## Protected Member Functions

- void **applyChangeNotifications** ()
- void **load** (const QDateTime &dateTime, const [CaptionsMap](#) &titles, const [CaptionsMap](#) &comment, int colorLabel, int pickLabel, int rating, const [Template](#) &t)
 

*private common code to load dateTime, comment, color label, pick label, rating*
- void **loadFaceTags** (const [ItemInfo](#) &info)
- void **loadTags** (const QList< int > &loadedTagIds)
 

*private common code to merge tags*
- void **notifyTagDeleted** (int id)
- bool **write** ([DMetadata](#) &metadata, WriteComponent writeMode=WRITE\_ALL, const [MetaEngineSettingsContainer](#) &settings=[MetaEngineSettings::instance\(\)](#) ->settings())
 

*Applies the set of metadata contained in this [MetadataHub](#) to the given meta engine object.*

## 9.960.1 Member Enumeration Documentation

### 9.960.1.1 Status

enum [Digikam::MetadataHub::Status](#)

If only one set has been added, the status is always MetadataAvailable. If no set has been added, the status is always MetadataInvalid

#### Enumerator

MetadataInvalid	not yet filled with any value
MetadataAvailable	only one data set has been added, or a common value is available

### 9.960.1.2 WriteMode

enum [Digikam::MetadataHub::WriteMode](#)

#### Enumerator

FullWrite	Write all available information.
FullWriteIfChanged	Do a full write if and only if. <ul style="list-style-type: none"> <li>• metadata fields changed</li> <li>• the changed fields shall be written according to write settings "Changed" in this context means changed by one of the set... methods, the <a href="#">load()</a> methods are ignored for this attribute. This mode allows to avoid write operations when e.g. the user does not want keywords to be written and only changes keywords.</li> </ul>
PartialWrite	Write only the changed parts. Metadata fields which cannot be changed from <a href="#">MetadataHub</a> (photographer ID etc.) will never be written

## 9.960.2 Member Function Documentation

### 9.960.2.1 cleanupTags()

```
QStringList Digikam::MetadataHub::cleanupTags (
    const QStringList & toClean )
```

#### Parameters

<i>toClean</i>	tag list to be cleared and de-duplicated
----------------	--

#### Returns

clean tag list

### 9.960.2.2 load()

```
void Digikam::MetadataHub::load (
    const ItemInfo & info )
```

This method (or in combination with the other load methods) can be called multiple times on the same [MetadataHub](#) object. In this case, the metadata will be combined.

### 9.960.2.3 willWriteMetadata()

```
bool Digikam::MetadataHub::willWriteMetadata (
    Digikam::MetadataHub::WriteComponent writeMode = WRITE_ALL,
    const MetaEngineSettingsContainer & settings = MetaEngineSettings::instance()->settings()
) const
```

#### Parameters

<i>writeMode</i>	The mode to write metadata
<i>settings</i>	The metadata settings to be set

#### Returns

if write(DMetadata), write(QString) or write(DImg) will actually apply any changes.

### 9.960.2.4 write() [1/3]

```
bool Digikam::MetadataHub::write (
    const DImg & image,
    WriteComponent writeMode = WRITE_ALL,
    bool ignoreLazySync = false,
    const MetaEngineSettingsContainer & settings = MetaEngineSettings::instance()->settings()
)
```

## Parameters

<i>image</i>	The <a href="#">DImg</a> container to retrieve current tags
<i>writeMode</i>	The mode to write metadata
<i>ignoreLazySync</i>	The flag to ignore the lazy sync metadata stage
<i>settings</i>	The metadata settings to be set

## Returns

Returns if the [DImg](#) object has been touched

**9.960.2.5 write()** [2/3]

```
bool Digikam::MetadataHub::write (
    const QString & filePath,
    WriteComponent writeMode = WRITE_ALL,
    bool ignoreLazySync = false,
    const MetaEngineSettingsContainer & settings = MetaEngineSettings::instance()->settings()
)
```

## Parameters

<i>filePath</i>	The file path to retrieve current tags
<i>writeMode</i>	The mode to write metadata
<i>ignoreLazySync</i>	The flag to ignore the lazy sync metadata stage
<i>settings</i>	The metadata settings to be set

## Warning

Do not use this method when multiple image infos are loaded It will result in disjoint tags not being written Use `writeToMetadata(Image info ...)` instead

## Returns

Returns if the file has been touched

**9.960.2.6 write()** [3/3]

```
bool Digikam::MetadataHub::write (
    DMetadata & metadata,
    WriteComponent writeMode = WRITE_ALL,
    const MetaEngineSettingsContainer & settings = MetaEngineSettings::instance()->settings()
) [protected]
```

## Parameters

<i>metadata</i>	The metadata backend instance.
<i>writeMode</i>	The mode to write metadata.

## Parameters

<i>settings</i>	<p>The <a href="#">MetaEngineSettingsContainer</a> determine whether data is actually set or not. The following metadata fields may be set (depending on settings):</p> <ul style="list-style-type: none"> <li>• Comment</li> <li>• Date</li> <li>• Rating</li> <li>• Tags</li> <li>• Photographer ID (data from settings)</li> <li>• Credits (data from settings)</li> </ul>
-----------------	---

## Note

The data fields taken from this [MetadataHub](#) object are only set if their status is `MetadataAvailable`. If the status is `MetadataInvalid` or `MetadataDisjoint`, the respective metadata field is not touched.

## Returns

Returns true if the metadata object has been touched.

**9.960.2.7 writeTags()** [1/2]

```
bool Digikam::MetadataHub::writeTags (
    const DMetadata & metadata,
    bool saveTags )
```

## Parameters

<i>metadata</i>	meta engine object that apply changes
<i>saveTags</i>	save switch

## Returns

if tags were successfully set

**9.960.2.8 writeTags()** [2/2]

```
bool Digikam::MetadataHub::writeTags (
    const QString & filePath,
    WriteComponent writeMode = WRITE_ALL,
    const MetaEngineSettingsContainer & settings = MetaEngineSettings::instance()->settings()
)
```

Used by [TagsManager](#) to write tags to image Other metadata are not updated.

## Parameters

<i>filePath</i>	The file path to update current tags
<i>writeMode</i>	The mode to write metadata
<i>settings</i>	The metadata settings to be set

## Returns

if tags were successfully written.

**9.960.2.9 writeToBaloo()**

```
void Digikam::MetadataHub::writeToBaloo (
    const QString & filePath,
    const MetaEngineSettingsContainer & settings = MetaEngineSettings::instance()->settings()
)
```

## Parameters

<i>filePath</i>	path to file to add comments, tags and rating
<i>settings</i>	metadata settings to be set

**9.960.2.10 writeToMetadata()**

```
bool Digikam::MetadataHub::writeToMetadata (
    const ItemInfo & info,
    WriteComponent writeMode = WRITE_ALL,
    bool ignoreLazySync = false,
    const MetaEngineSettingsContainer & settings = MetaEngineSettings::instance()->settings()
)
```

safe method

## Parameters

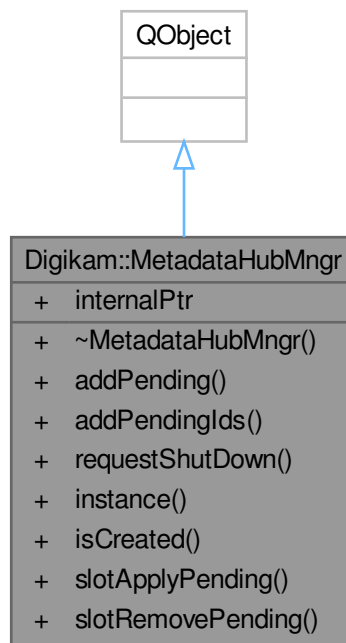
<i>info</i>	The image info to retrieve current tags
<i>writeMode</i>	The mode to write metadata
<i>ignoreLazySync</i>	The flag to ignore the lazy sync metadata stage
<i>settings</i>	The metadata settings to be set

**Returns**

true if everything is successful

## 9.961 Digikam::MetadataHubMngr Class Reference

Inheritance diagram for Digikam::MetadataHubMngr:

**Public Slots**

- void **slotApplyPending** ()
- void **slotRemovePending** (const [ItemInfo](#) &info)

**Signals**

- void **signalPendingMetadata** (int numbers)

**Public Member Functions**

- void **addPending** (const [ItemInfo](#) &info)
- void **addPendingIds** (const QList< qlonglong > &imageIds)
- void **requestShutDown** ()

### Static Public Member Functions

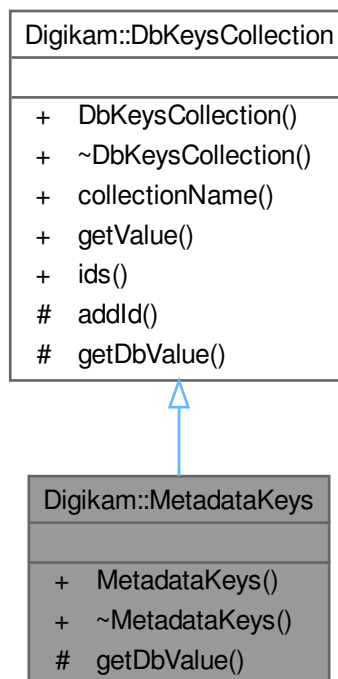
- static [MetadataHubMngr](#) \* **instance** ()
- static bool **isCreated** ()

### Static Public Attributes

- static QPointer< [MetadataHubMngr](#) > **internalPtr** = QPointer<[MetadataHubMngr](#)>()

## 9.962 Digikam::MetadataKeys Class Reference

Inheritance diagram for Digikam::MetadataKeys:



### Protected Member Functions

- QString `getDbValue` (const QString &key, [ParseSettings](#) &settings) override  
*Abstract method for retrieving the value from the database for the given key.*

### Protected Member Functions inherited from [Digikam::DbKeysCollection](#)

- void `addId` (const QString &id, const QString &description)  
*Add an ID to the key collection.*



## Additional Inherited Members

## Public Member Functions inherited from Digikam::DbKeysCollection

- [DbKeysCollection](#) (const QString &n)  
*Default constructor.*
- QString [collectionName](#) () const  
*Get the name of the DbKeysCollection.*
- QString [getValue](#) (const QString &key, [ParseSettings](#) &settings)  
*Get a value from the database.*
- DbKeyIdsMap [ids](#) () const  
*Get all IDs associated with this key collection.*

## 9.962.1 Member Function Documentation

### 9.962.1.1 getDbValue()

```
QString Digikam::MetadataKeys::getDbValue (  
    const QString & key,  
    ParseSettings & settings ) [override], [protected], [virtual]
```

This method has to be implemented by all child classes. It is called by the [getValue\(\)](#) method.

#### Parameters

<i>key</i>	the key representing the value in the database
<i>settings</i>	the ParseSettings object holding all relevant information about the image.

#### Returns

the value of the given database key

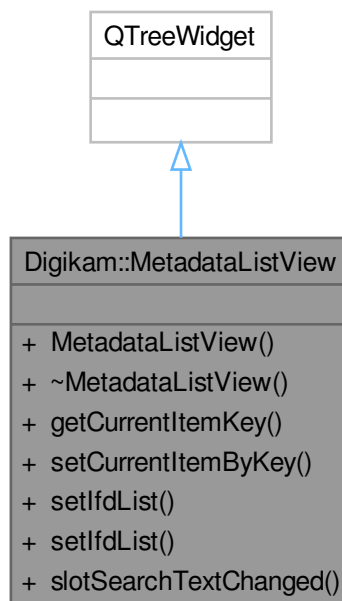
See also

[DbKeysCollection::getValue\(\)](#)

Implements [Digikam::DbKeysCollection](#).

## 9.963 Digikam::MetadataListView Class Reference

Inheritance diagram for Digikam::MetadataListView:



### Public Slots

- void **slotSearchTextChanged** (const [SearchTextSettings](#) &)

### Signals

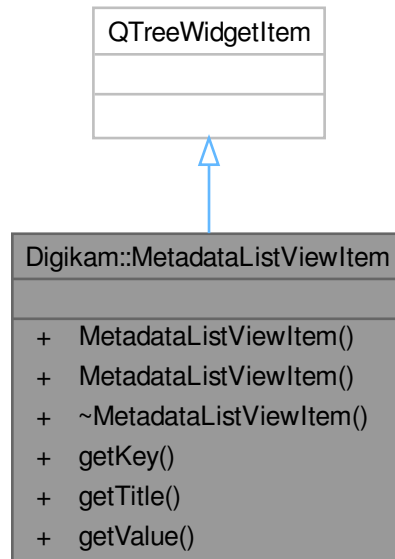
- void **signalTextFilterMatch** (bool)

### Public Member Functions

- **MetadataListView** (QWidget \*const parent)
- QString **getCurrentItemKey** () const
- void **setCurrentItemByKey** (const QString &itemKey)
- void **setIfdList** (const [DMetadata::MetaDataMap](#) &ifds, const QStringList &keysFilter, const QStringList &tagsFilter)
- void **setIfdList** (const [DMetadata::MetaDataMap](#) &ifds, const QStringList &tagsFilter)

## 9.964 Digikam::MetadataListViewItem Class Reference

Inheritance diagram for Digikam::MetadataListViewItem:



### Public Member Functions

- **MetadataListViewItem** (`QTreeWidgetItem *const parent`, `const QString &key`, `const QString &title`)
- **MetadataListViewItem** (`QTreeWidgetItem *const parent`, `const QString &key`, `const QString &title`, `const QString &value`)
- `QString` **getKey** () const
- `QString` **getTitle** () const
- `QString` **getValue** () const

## 9.965 Digikam::MetadataOption Class Reference

Inheritance diagram for Digikam::MetadataOption:



### Protected Member Functions

- `QString parseOperation (ParseSettings &settings, const QRegularExpressionMatch &match)` override  
*TODO: describe me.*

## Protected Member Functions inherited from Digikam::Rule

- bool **addToken** (const QString &id, const QString &description, const QString &actionName=QString())  
*add a token to the parser, every parser should at least assign one token object*
- void **setDescription** (const QString &desc)
- void **setIcon** (const QString &pixmap)
- void **setRegExp** (const QRegularExpression &regExp)
- void **setUseTokenMenu** (bool value)  
*If multiple tokens have been assigned to a rule, a menu will be created.*

## Additional Inherited Members

## Public Types inherited from Digikam::Rule

- enum **IconType** { **Action** = 0 , **Dialog** }

## Signals inherited from Digikam::Rule

- void **signalTokenTriggered** (const QString &)

## Public Member Functions inherited from Digikam::Option

- **Option** (const QString &name, const QString &description)
- **Option** (const QString &name, const QString &description, const QString &icon)

## Public Member Functions inherited from Digikam::Rule

- **Rule** (const QString &name)
- **Rule** (const QString &name, const QString &icon)
- QString **description** () const
- QPixmap **icon** (Rule::IconType type=Rule::Action) const
- bool **isValid** () const  
*Checks the validity of the parse object.*
- ParseResults **parse** (ParseSettings &settings)
- QRegularExpression & **regExp** () const  
*TODO: This is probably not needed anymore.*
- QPushButton \* **registerButton** (QWidget \*parent)  
*Register a button in the parent object.*
- QAction \* **registerMenu** (QMenu \*parent)  
*Register a menu action in the parent object.*
- virtual void **reset** ()  
*Resets the parser to its initial state.*
- TokenList & **tokens** () const
- bool **useTokenMenu** () const  
*Returns true if a token menu is used.*

## Static Public Member Functions inherited from [Digikam::Rule](#)

- static QString [escapeToken](#) (const QString &token)  
*Escape the token characters to make them work in regular expressions.*

## Protected Slots inherited from [Digikam::Rule](#)

- virtual void [slotTokenTriggered](#) (const QString &)

### 9.965.1 Member Function Documentation

#### 9.965.1.1 [parseOperation\(\)](#)

```
QString Digikam::MetadataOption::parseOperation (
    ParseSettings & settings,
    const QRegularExpressionMatch & match ) [override], [protected], [virtual]
```

##### Parameters

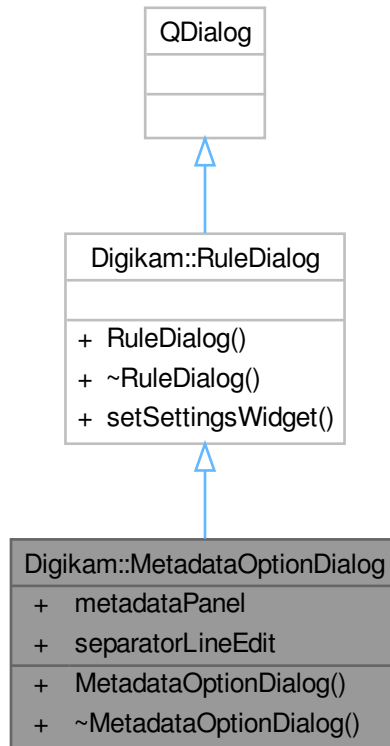
<i>settings</i>	contains settings
<i>match</i>	result of the regular expression match done in <a href="#">Option::parse()</a>

##### Returns

Implements [Digikam::Option](#).

## 9.966 Digikam::MetadataOptionDialog Class Reference

Inheritance diagram for Digikam::MetadataOptionDialog:



### Public Member Functions

- `MetadataOptionDialog` ([Rule](#) \*const parent)

### Public Member Functions inherited from [Digikam::RuleDialog](#)

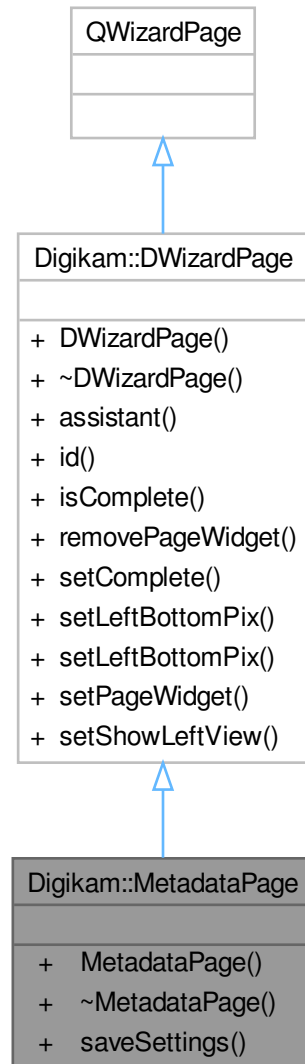
- `RuleDialog` ([Rule](#) \*const parent)
- void `setSettingsWidget` (QWidget \*const settingsWidget)

### Public Attributes

- `MetadataPanel` \* `metadataPanel` = nullptr
- `QLineEdit` \* `separatorLineEdit` = nullptr

## 9.967 Digikam::MetadataPage Class Reference

Inheritance diagram for Digikam::MetadataPage:



### Public Member Functions

- **MetadataPage** (QWizard \*const dlg)
- void **saveSettings** ()

### Public Member Functions inherited from [Digikam::DWizardPage](#)

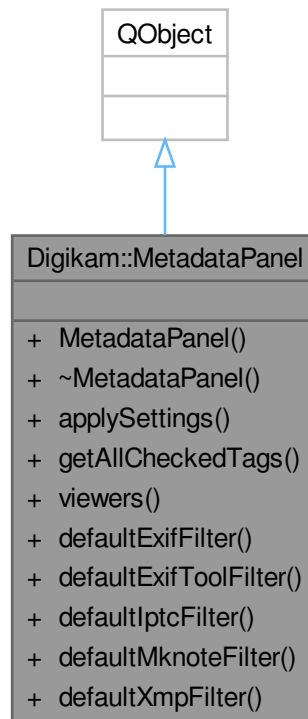
- **DWizardPage** (QWizard \*const dlg, const QString &title)
- QWizard \* **assistant** () const



- int **id** () const
- bool **isComplete** () const override
- void **removePageWidget** (QWidget \*const w)
- void **setComplete** (bool b)
- void **setLeftBottomPix** (const QIcon &icon)
- void **setLeftBottomPix** (const QPixmap &pix)
- void **setPageWidget** (QWidget \*const w)
- void **setShowLeftView** (bool v)

## 9.968 Digikam::MetadataPanel Class Reference

Inheritance diagram for Digikam::MetadataPanel:



### Public Member Functions

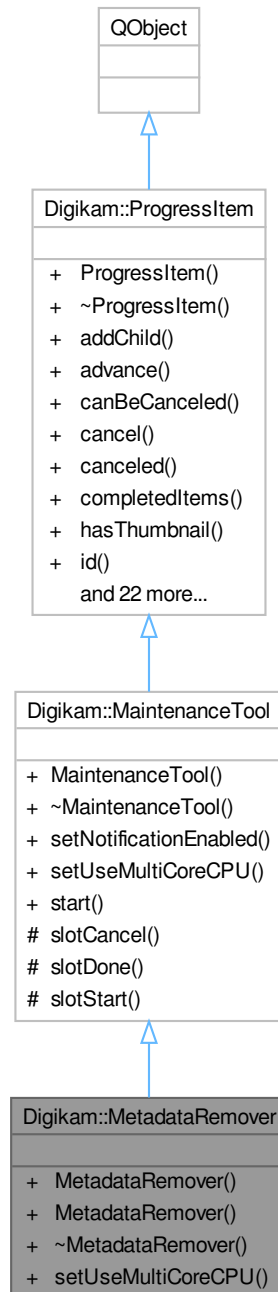
- **MetadataPanel** (QTabWidget \*const tab)
- void **applySettings** ()
- QStringList **getAllCheckedTags** () const
- QList< [MetadataSelectorView](#) \* > **viewers** () const

**Static Public Member Functions**

- static QStringList **defaultExifFilter** ()
- static QStringList **defaultExifToolFilter** ()
- static QStringList **defaultIptcFilter** ()
- static QStringList **defaultMknoteFilter** ()
- static QStringList **defaultXmpFilter** ()

## 9.969 Digikam::MetadataRemover Class Reference

Inheritance diagram for Digikam::MetadataRemover:



### Public Types

- enum `RemoveAction` { `None = 0` , `Faces` , `Tags` }

## Public Member Functions

- [MetadataRemover](#) (const AlbumList &list=AlbumList(), RemoveAction action=None, [ProgressItem](#) \*const parent=nullptr)  
*Constructor which remove all images metadata from an Albums list.*
- [MetadataRemover](#) (const ItemInfoList &list, RemoveAction action=None, [ProgressItem](#) \*const parent=nullptr)  
*Constructor which remove all images metadata from an Images list.*
- void [setUseMultiCoreCPU](#) (bool b) override  
*Re-implement this method if your tool is able to use multi-core CPU to process item in parallel.*

## Public Member Functions inherited from [Digikam::MaintenanceTool](#)

- [MaintenanceTool](#) (const QString &id, [ProgressItem](#) \*const parent=nullptr)
- void [setNotificationEnabled](#) (bool b)  
*If true, show a notification message on desktop notification manager with time elapsed to run process.*

## Public Member Functions inherited from [Digikam::ProgressItem](#)

- [ProgressItem](#) ([ProgressItem](#) \*const parent, const QString &id, const QString &label, const QString &status, bool canBeCanceled, bool hasThumb)
- void [addChild](#) ([ProgressItem](#) \*const kiddo)
- bool [advance](#) (unsigned int v)  
*Advance total items processed by n values and update percentage in progressbar.*
- bool [canBeCanceled](#) () const
- void [cancel](#) ()
- bool [canceled](#) () const
- unsigned int [completedItems](#) () const
- bool [hasThumbnail](#) () const
- const QString & [id](#) () const
- bool [incCompletedItems](#) (unsigned int v=1)
- void [incTotalItems](#) (unsigned int v=1)
- const QString & [label](#) () const
- [ProgressItem](#) \* [parent](#) () const
- unsigned int [progress](#) () const
- void [removeChild](#) ([ProgressItem](#) \*const kiddo)
- void [reset](#) ()  
*Reset the progress value of this item to 0 and the status string to the empty string.*
- void [setComplete](#) ()  
*Tell the item it has finished.*
- bool [setCompletedItems](#) (unsigned int v)
- void [setLabel](#) (const QString &v)
- void [setProgress](#) (unsigned int v)  
*Set the progress (percentage of completion) value of this item.*
- void [setShowAtStart](#) (bool showAtStart)  
*Set the property to pop-up item when it's added in progress manager.*
- void [setStatus](#) (const QString &v)  
*Set the string to be used for showing this item's current status.*
- void [setThumbnail](#) (const QIcon &icon)  
*Sets whether this item has a thumbnail.*
- void [setTotalItems](#) (unsigned int v)
- void [setUsesBusyIndicator](#) (bool useBusyIndicator)

Sets whether this item uses a busy indicator instead of real progress for its progress bar.

- bool `showAtStart` () const
- const QString & `status` () const
- bool `totalCompleted` () const
- unsigned int `totalItems` () const
- void `updateProgress` ()

Recalculate progress according to total/completed items and update.

- bool `usesBusyIndicator` () const

### Additional Inherited Members

### Public Slots inherited from [Digikam::MaintenanceTool](#)

- void `start` ()

### Signals inherited from [Digikam::MaintenanceTool](#)

- void `signalCanceled` ()  
*Emit when process is canceled.*
- void `signalComplete` ()  
*Emit when process is done (not canceled).*

### Signals inherited from [Digikam::ProgressItem](#)

- void `progressItemAdded` ([ProgressItem](#) \*item)  
*Emitted when a new [ProgressItem](#) is added.*
- void `progressItemCanceled` ([ProgressItem](#) \*item)  
*Emitted when an item was canceled.*
- void `progressItemCanceledById` (const QString &id)
- void `progressItemCompleted` ([ProgressItem](#) \*item)  
*Emitted when a progress item was completed.*
- void `progressItemLabel` ([ProgressItem](#) \*item, const QString &label)  
*Emitted when the label of an item changed.*
- void `progressItemProgress` ([ProgressItem](#) \*item, unsigned int v)  
*Emitted when the progress value of an item changes.*
- void `progressItemStatus` ([ProgressItem](#) \*item, const QString &mess)  
*Emitted when the status message of an item changed.*
- void `progressItemThumbnail` ([ProgressItem](#) \*item, const QPixmap &thumb)  
*Emitted when the thumbnail data must be set in item.*
- void `progressItemUsesBusyIndicator` ([ProgressItem](#) \*item, bool value)  
*Emitted when the busy indicator state of an item changes.*

### Protected Slots inherited from [Digikam::MaintenanceTool](#)

- virtual void `slotCancel` ()
- virtual void `slotDone` ()
- virtual void `slotStart` ()

## 9.969.1 Constructor & Destructor Documentation

### 9.969.1.1 MetadataRemover()

```
Digikam::MetadataRemover::MetadataRemover (
    const AlbumList & list = AlbumList(),
    RemoveAction action = None,
    ProgressItem *const parent = nullptr ) [explicit]
```

If list is empty, whole Albums collection is processed.

## 9.969.2 Member Function Documentation

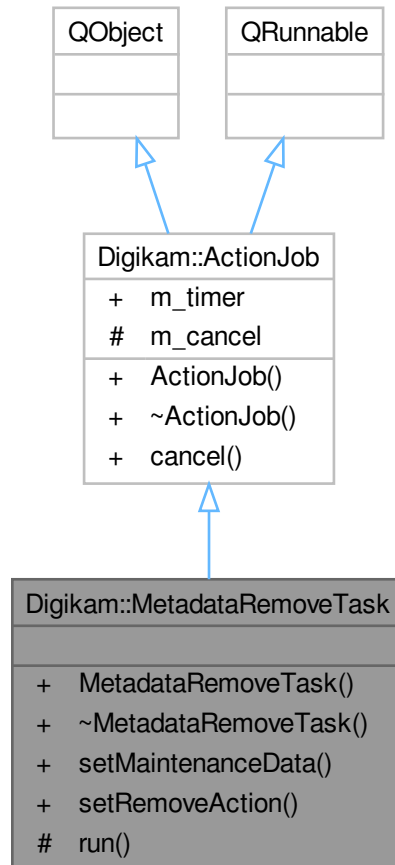
### 9.969.2.1 setUseMultiCoreCPU()

```
void Digikam::MetadataRemover::setUseMultiCoreCPU (
    bool ) [override], [virtual]
```

Reimplemented from [Digikam::MaintenanceTool](#).

## 9.970 Digikam::MetadataRemoveTask Class Reference

Inheritance diagram for Digikam::MetadataRemoveTask:



### Signals

- void **signalFinished** (const [ItemInfo](#) &, const [QImage](#) &)

### Signals inherited from [Digikam::ActionJob](#)

- void **signalDone** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job is done.*
- void **signalProgress** (int)  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job progress.*
- void **signalStarted** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager that job is started.*

### Public Member Functions

- void **setMaintenanceData** ([MaintenanceData](#) \*const data=nullptr)
- void **setRemoveAction** ([MetadataRemover::RemoveAction](#) action)

### Public Member Functions inherited from [Digikam::ActionJob](#)

- **ActionJob** ([QObject](#) \*const parent=nullptr)  
*Constructor which delegate deletion of [QRunnable](#) instance to [ActionThreadBase](#), not [QThreadPool](#).*
- [~ActionJob](#) () override  
*Re-implement destructor in you implementation.*

### Protected Member Functions

- void **run** () override

### Additional Inherited Members

### Public Slots inherited from [Digikam::ActionJob](#)

- void **cancel** ()  
*Call this method to cancel job.*

### Public Attributes inherited from [Digikam::ActionJob](#)

- [QElapsedTimer](#) **m\_timer**  
*Timer to determine the running time of the job.*

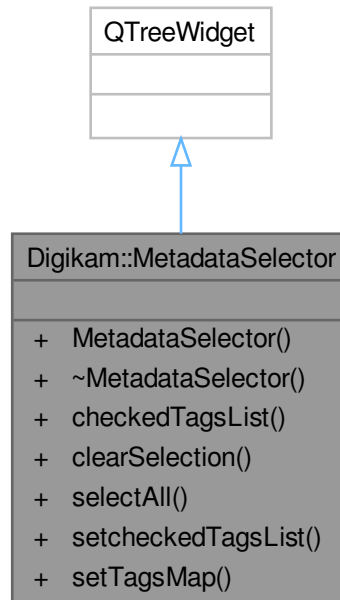
### Protected Attributes inherited from [Digikam::ActionJob](#)

- bool **m\_cancel** = false  
*You can use this boolean in your implementation to know if job must be canceled.*



## 9.971 Digikam::MetadataSelector Class Reference

Inheritance diagram for Digikam::MetadataSelector:

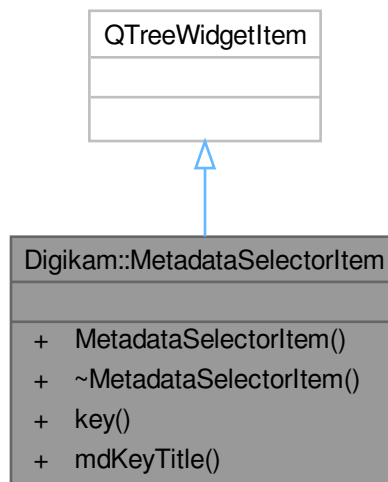


### Public Member Functions

- **MetadataSelector** ([MetadataSelectorView](#) \*const parent)
- `QStringList checkedTagsList ()`
- void **clearSelection** ()
- void **selectAll** () override
- void **setCheckedTagsList** (const QStringList &list)
- void **setTagsMap** (const [DMetadata::TagsMap](#) &map)

## 9.972 Digikam::MetadataSelectorItem Class Reference

Inheritance diagram for Digikam::MetadataSelectorItem:

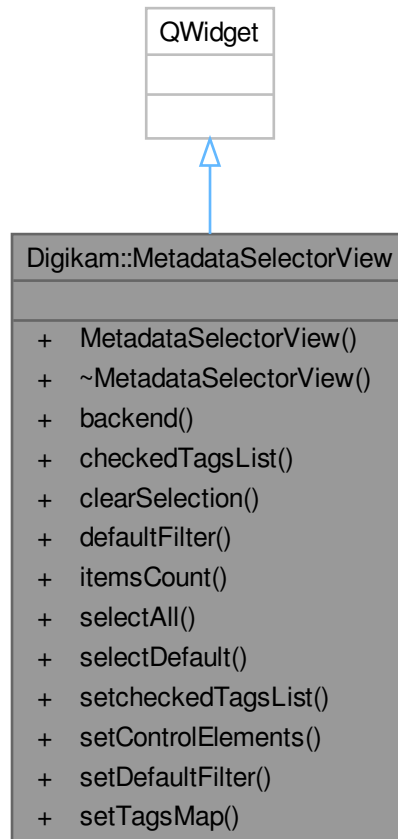


### Public Member Functions

- **MetadataSelectorItem** ([MdKeyListViewItem](#) \*const parent, const QString &key, const QString &title, const QString &desc)
- QString **key** () const
- QString **mdKeyTitle** () const

## 9.973 Digikam::MetadataSelectorView Class Reference

Inheritance diagram for Digikam::MetadataSelectorView:



### Public Types

- enum **Backend** { **Exiv2Backend** = 0 , **ExifToolBackend** }
- enum **ControlElement** { **SelectAllBtn** = 0x01 , **ClearBtn** = 0x02 , **DefaultBtn** = 0x04 , **SearchBar** = 0x08 }
- typedef `QFlags< ControlElement >` **ControlElements**

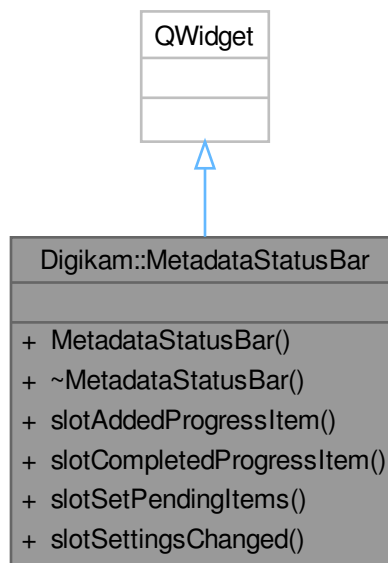
### Public Member Functions

- **MetadataSelectorView** (`QWidget *const parent`, `Backend be`)
- Backend **backend** () const
- `QStringList` **checkedTagsList** () const
- void **clearSelection** ()
- `QStringList` **defaultFilter** () const
- int **itemCount** () const
- void **selectAll** ()
- void **selectDefault** ()

- void **setCheckedTagsList** (const QStringList &list)
- void **setControlElements** (ControlElements controllerMask)
- void **setDefaultFilter** (const QStringList &list)
- void **setTagsMap** (const [DMetadata::TagsMap](#) &map)

## 9.974 Digikam::MetadataStatusBar Class Reference

Inheritance diagram for Digikam::MetadataStatusBar:



### Public Slots

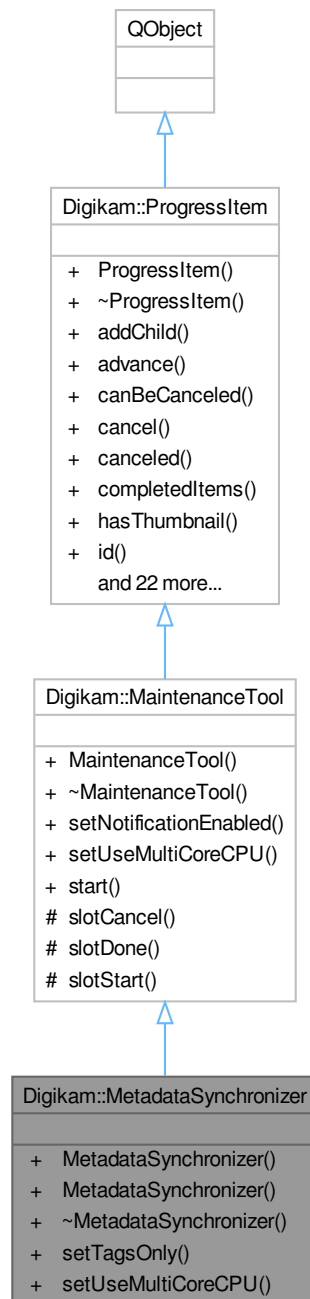
- void **slotAddedProgressItem** ([ProgressItem](#) \*item)
- void **slotCompletedProgressItem** ([ProgressItem](#) \*item)
- void **slotSetPendingItems** (int number)
- void **slotSettingsChanged** ()

### Public Member Functions

- **MetadataStatusBar** ([QWidget](#) \*const parent)

## 9.975 Digikam::MetadataSynchronizer Class Reference

Inheritance diagram for Digikam::MetadataSynchronizer:



### Public Types

- enum **SyncDirection** { **WriteFromDatabaseToFile** = 0 , **ReadFromFileToDatabase** }

## Signals

- void **signalRemovePending** (const [ItemInfo](#) &info)

## Signals inherited from [Digikam::MaintenanceTool](#)

- void **signalCanceled** ()  
*Emit when process is canceled.*
- void **signalComplete** ()  
*Emit when process is done (not canceled).*

## Signals inherited from [Digikam::ProgressItem](#)

- void **progressItemAdded** ([ProgressItem](#) \*item)  
*Emitted when a new [ProgressItem](#) is added.*
- void **progressItemCanceled** ([ProgressItem](#) \*item)  
*Emitted when an item was canceled.*
- void **progressItemCanceledById** (const [QString](#) &id)
- void **progressItemCompleted** ([ProgressItem](#) \*item)  
*Emitted when a progress item was completed.*
- void **progressItemLabel** ([ProgressItem](#) \*item, const [QString](#) &label)  
*Emitted when the label of an item changed.*
- void **progressItemProgress** ([ProgressItem](#) \*item, unsigned int v)  
*Emitted when the progress value of an item changes.*
- void **progressItemStatus** ([ProgressItem](#) \*item, const [QString](#) &mess)  
*Emitted when the status message of an item changed.*
- void **progressItemThumbnail** ([ProgressItem](#) \*item, const [QPixmap](#) &thumb)  
*Emitted when the thumbnail data must be set in item.*
- void **progressItemUsesBusyIndicator** ([ProgressItem](#) \*item, bool value)  
*Emitted when the busy indicator state of an item changes.*

## Public Member Functions

- [MetadataSynchronizer](#) (const [AlbumList](#) &list=[AlbumList](#)(), [SyncDirection](#) direction=[WriteFromDatabaseToFile](#), [ProgressItem](#) \*const parent=nullptr)  
*Constructor which sync all images metadata from an Albums list.*
- [MetadataSynchronizer](#) (const [ItemInfoList](#) &list, [SyncDirection](#)=[WriteFromDatabaseToFile](#), [ProgressItem](#) \*const parent=nullptr)  
*Constructor which sync all images metadata from an Images list.*
- void **setTagsOnly** (bool value)
- void **setUseMultiCoreCPU** (bool b) override  
*Re-implement this method if your tool is able to use multi-core CPU to process item in parallel.*

## Public Member Functions inherited from [Digikam::MaintenanceTool](#)

- **MaintenanceTool** (const [QString](#) &id, [ProgressItem](#) \*const parent=nullptr)
- void **setNotificationEnabled** (bool b)  
*If true, show a notification message on desktop notification manager with time elapsed to run process.*

## Public Member Functions inherited from Digikam::ProgressItem

- **ProgressItem** ([ProgressItem](#) \*const [parent](#), const QString &[id](#), const QString &[label](#), const QString &[status](#), bool [canBeCanceled](#), bool [hasThumb](#))
- void **addChild** ([ProgressItem](#) \*const [kiddo](#))
- bool [advance](#) (unsigned int [v](#))
  - Advance total items processed by n values and update percentage in progressbar.*
- bool [canBeCanceled](#) () const
- void **cancel** ()
- bool **canceled** () const
- unsigned int **completedItems** () const
- bool [hasThumbnail](#) () const
- const QString & [id](#) () const
- bool **incCompletedItems** (unsigned int [v=1](#))
- void **incTotalItems** (unsigned int [v=1](#))
- const QString & [label](#) () const
- [ProgressItem](#) \* [parent](#) () const
- unsigned int [progress](#) () const
- void **removeChild** ([ProgressItem](#) \*const [kiddo](#))
- void **reset** ()
  - Reset the progress value of this item to 0 and the status string to the empty string.*
- void [setComplete](#) ()
  - Tell the item it has finished.*
- bool **setCompletedItems** (unsigned int [v](#))
- void [setLabel](#) (const QString &[v](#))
- void [setProgress](#) (unsigned int [v](#))
  - Set the progress (percentage of completion) value of this item.*
- void [setShowAtStart](#) (bool [showAtStart](#))
  - Set the property to pop-up item when it's added in progress manager.*
- void [setStatus](#) (const QString &[v](#))
  - Set the string to be used for showing this item's current status.*
- void [setThumbnail](#) (const QIcon &[icon](#))
  - Sets whether this item has a thumbnail.*
- void **setTotalItems** (unsigned int [v](#))
- void [setUsesBusyIndicator](#) (bool [useBusyIndicator](#))
  - Sets whether this item uses a busy indicator instead of real progress for its progress bar.*
- bool [showAtStart](#) () const
- const QString & [status](#) () const
- bool **totalCompleted** () const
- unsigned int **totalItems** () const
- void **updateProgress** ()
  - Recalculate progress according to total/completed items and update.*
- bool [usesBusyIndicator](#) () const

## Additional Inherited Members

## Public Slots inherited from Digikam::MaintenanceTool

- void **start** ()

## Protected Slots inherited from [Digikam::MaintenanceTool](#)

- virtual void **slotCancel** ()
- virtual void **slotDone** ()
- virtual void **slotStart** ()

### 9.975.1 Constructor & Destructor Documentation

#### 9.975.1.1 MetadataSynchronizer()

```
Digikam::MetadataSynchronizer::MetadataSynchronizer (  
    const AlbumList & list = AlbumList(),  
    SyncDirection direction = WriteFromDatabaseToFile,  
    ProgressItem *const parent = nullptr ) [explicit]
```

If list is empty, whole Albums collection is processed.

### 9.975.2 Member Function Documentation

#### 9.975.2.1 setUseMultiCoreCPU()

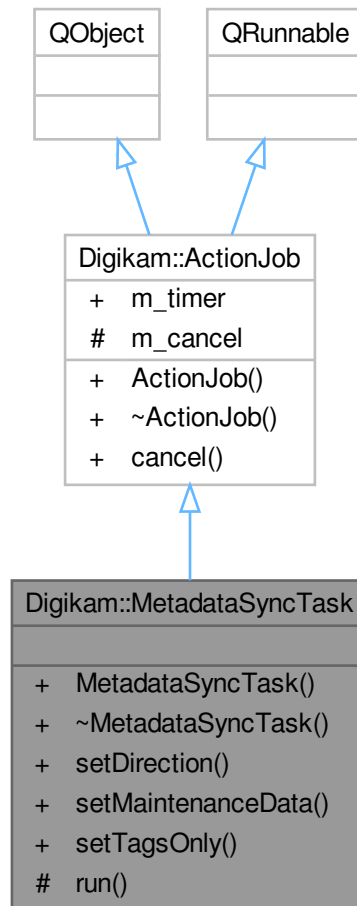
```
void Digikam::MetadataSynchronizer::setUseMultiCoreCPU (  
    bool ) [override], [virtual]
```

Reimplemented from [Digikam::MaintenanceTool](#).



## 9.976 Digikam::MetadataSyncTask Class Reference

Inheritance diagram for Digikam::MetadataSyncTask:



### Signals

- void **signalFinished** (const [ItemInfo](#) &, const `QImage` &)
- void **signalRemovePending** (const [ItemInfo](#) &info)

### Signals inherited from [Digikam::ActionJob](#)

- void **signalDone** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job is done.*
- void **signalProgress** (int)  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job progress.*
- void **signalStarted** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager that job is started.*

### Public Member Functions

- void **setDirection** (MetadataSynchronizer::SyncDirection dir)
- void **setMaintenanceData** ([MaintenanceData](#) \*const data=nullptr)
- void **setTagsOnly** (bool value)

### Public Member Functions inherited from [Digikam::ActionJob](#)

- **ActionJob** (QObject \*const parent=nullptr)  
*Constructor which delegate deletion of QRunnable instance to [ActionThreadBase](#), not QThreadPool.*
- [~ActionJob](#) () override  
*Re-implement destructor in you implementation.*

### Protected Member Functions

- void **run** () override

### Additional Inherited Members

### Public Slots inherited from [Digikam::ActionJob](#)

- void **cancel** ()  
*Call this method to cancel job.*

### Public Attributes inherited from [Digikam::ActionJob](#)

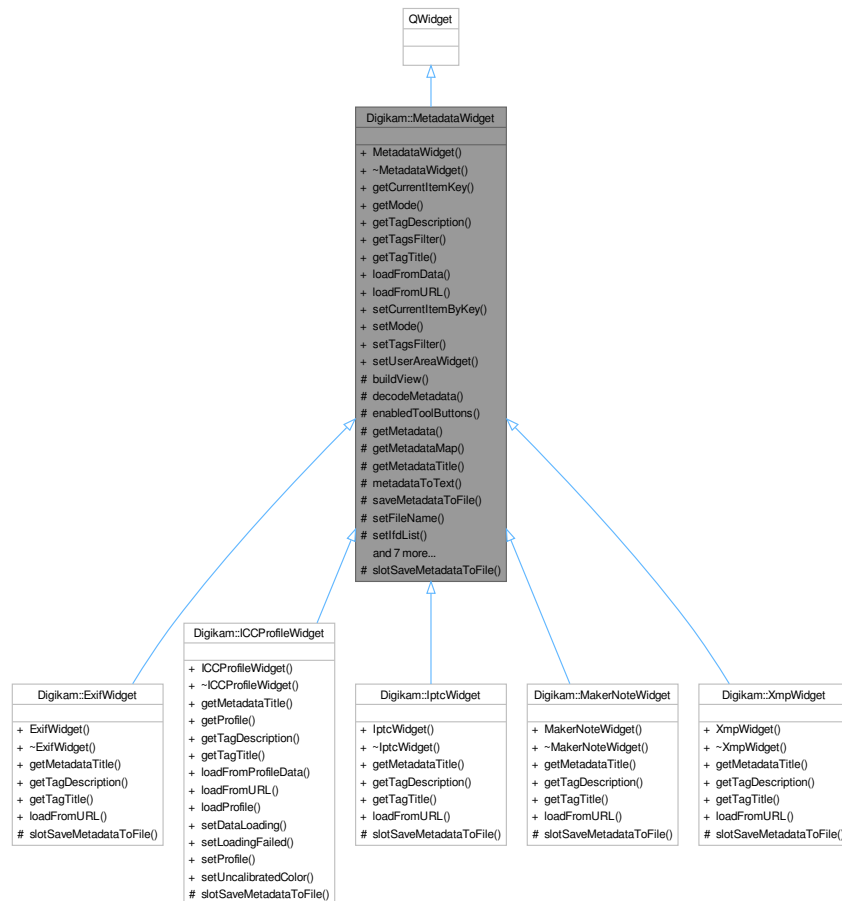
- QElapsedTimer **m\_timer**  
*Timer to determine the running time of the job.*

### Protected Attributes inherited from [Digikam::ActionJob](#)

- bool **m\_cancel** = false  
*You can use this boolean in your implementation to know if job must be canceled.*

## 9.977 Digikam::MetadataWidget Class Reference

Inheritance diagram for Digikam::MetadataWidget:



### Public Types

- enum `TagFilters` { `NONE = 0` , `PHOTO` , `CUSTOM` }

### Signals

- void `signalSetupMetadataFilters ()`

### Public Member Functions

- **MetadataWidget** (`QWidget *const parent, const QString &name=QString()`)
- `QString` `getCurrentItemKey ()` const
- `int` `getMode ()` const
- virtual `QString` `getTagDescription (const QString &key)`
- `QStringList` `getTagsFilter ()` const
- virtual `QString` `getTagTitle (const QString &key)`

- virtual bool **loadFromData** (const QString &fileName, const [DMetadatum](#) &data=[DMetadatum](#)())
- virtual bool **loadFromURL** (const QUrl &url)=0
- void **setCurrentItemByKey** (const QString &itemKey)
- void **setMode** (int mode)
- void **setTagsFilter** (const QStringList &list)
- void **setUserAreaWidget** (QWidget \*const w)

### Protected Slots

- virtual void **slotSaveMetadatumToFile** ()=0

### Protected Member Functions

- virtual void **buildView** ()
  - virtual bool **decodeMetadatum** ()=0
  - void **enabledToolButtons** (bool)
  - [DMetadatum](#) \* **getMetadatum** () const
  - const [DMetadatum::MetaDatumMap](#) & **getMetadatumMap** ()
  - virtual QString **getMetadatumTitle** () const =0
  - QString **metadatumToText** () const
  - QUrl **saveMetadatumToFile** (const QString &caption, const QString &fileFilter)
  - void **setFileName** (const QString &fileName)
  - void **setIfdList** (const [DMetadatum::MetaDatumMap](#) &ifds, const QStringList &keysFilter, const QStringList &tagsFilter)
  - void **setIfdList** (const [DMetadatum::MetaDatumMap](#) &ifds, const QStringList &tagsFilter=[QStringList](#)())
  - bool **setMetadatum** (const [DMetadatum](#) &data=[DMetadatum](#)())
  - virtual void **setMetadatumEmpty** ()
  - void **setMetadatumMap** (const [DMetadatum::MetaDatumMap](#) &data=[DMetadatum::MetaDatumMap](#)())
  - void **setup** ()
- Call this method in children class constructors to init signal/slots connections.*
- bool **storeMetadatumToFile** (const QUrl &url, const QByteArray &metaData)
  - [MetadatumListView](#) \* **view** () const



- enum [Backend](#) {  
[Exiv2Backend](#) = 0 , [LibRawBackend](#) , [LibHeifBackend](#) , [ImageMagickBackend](#) ,  
[FFMpegBackend](#) , [ExifToolBackend](#) , [VideoMergeBackend](#) , [NoBackend](#) }  
*Metadata Backend used to populate information.*
- enum [ImageColorWorkSpace](#) { [WORKSPACE\\_UNSPECIFIED](#) = 0 , [WORKSPACE\\_SRGB](#) = 1 ,  
[WORKSPACE\\_ADOBERGB](#) = 2 , [WORKSPACE\\_UNCALIBRATED](#) = 65535 }  
*The item color workspace values given by Exif metadata.*
- enum [ImageOrientation](#) {  
[ORIENTATION\\_UNSPECIFIED](#) = 0 , [ORIENTATION\\_NORMAL](#) = 1 , [ORIENTATION\\_HFLIP](#) = 2 ,  
[ORIENTATION\\_ROT\\_180](#) = 3 ,  
[ORIENTATION\\_VFLIP](#) = 4 , [ORIENTATION\\_ROT\\_90\\_HFLIP](#) = 5 , [ORIENTATION\\_ROT\\_90](#) = 6 ,  
[ORIENTATION\\_ROT\\_90\\_VFLIP](#) = 7 ,  
[ORIENTATION\\_ROT\\_270](#) = 8 }  
*The item orientation values given by Exif metadata.*
- typedef QMap< QString, QString > **MetaDataMap**  
*A map used to store Tags Key and Tags Value.*
- enum [MetadataWritingMode](#) { [WRITE\\_TO\\_FILE\\_ONLY](#) = 0 , [WRITE\\_TO\\_SIDECAR\\_ONLY](#) = 1 ,  
[WRITE\\_TO\\_SIDECAR\\_AND\\_FILE](#) = 2 , [WRITE\\_TO\\_SIDECAR\\_ONLY\\_FOR\\_READ\\_ONLY\\_FILES](#) = 3  
}
  
*The item metadata writing mode, between item file metadata and XMP sidecar file, depending on the context.*
- typedef QMap< QString, QStringList > [TagsMap](#)  
*A map used to store Tags Key and a list of Tags properties :*
- enum [XmpTagType](#) {  
[NormalTag](#) = 0 , [ArrayBagTag](#) = 1 , [StructureTag](#) = 2 , [ArrayLangTag](#) = 3 ,  
[ArraySeqTag](#) = 4 }  
*Xmp tag types, used by setXmpTag, only first three types are used.*

## Public Member Functions

- **MetaEngine** ()  
*Standard constructor.*
- **MetaEngine** (const [MetaEngineData](#) &data)  
*Constructor to load from parsed data.*
- **MetaEngine** (const QString &filePath)  
*Constructor to Load Metadata from item file.*
- virtual ~**MetaEngine** ()  
*Standard destructor.*

## General methods

- [MetaEngineData](#) **data** () const
- void **setData** (const [MetaEngineData](#) &data)
- bool **loadFromData** (const QByteArray &imgData)  
*Load all metadata (Exif, Iptc, Xmp, and JFIF Comments) from a byte array.*
- bool **loadFromDataAndMerge** (const QByteArray &imgData, const QStringList &exclude=QStringList())  
*Load and merge metadata (Exif, Iptc and Xmp) from a byte array.*
- bool **isEmpty** () const  
*Return 'true' if metadata container in memory as no Comments, Exif, Iptc, and Xmp.*
- QSize **getPixelSize** () const  
*Returns the pixel size of the current item.*
- QString **getMimeType** () const  
*Returns the mime type of this item.*
- void **setReadWithExifTool** (const bool on)  
*Enable or disable reading metadata operations with ExifTool.*

- bool **readWithExifTool** () const  
*Return true if reading metadata operations with ExifTool is enabled.*
- void **setWriteWithExifTool** (const bool on)  
*Enable or disable writing metadata operations with ExifTool.*
- bool **writeWithExifTool** () const  
*Return true if writing metadata operations with ExifTool is enabled.*
- void **setWriteRawFiles** (const bool on)  
*Enable or disable writing metadata operations to RAW files.*
- bool **writeRawFiles** () const  
*Return true if writing metadata operations on RAW files is enabled.*
- void **setWriteDngFiles** (const bool on)  
*Enable or disable writing metadata operations to DNG files.*
- bool **writeDngFiles** () const  
*Return true if writing metadata operations on DNG files is enabled.*
- void **setUseXMPSidecar4Reading** (const bool on)  
*Enable or disable using XMP sidecar for reading metadata.*
- bool **useXMPSidecar4Reading** () const  
*Return true if using XMP sidecar for reading metadata is enabled.*
- void **setUseCompatibleFileName** (const bool on)  
*Enable or disable using compatible file name for sidecar files.*
- bool **useCompatibleFileName** () const  
*Return true if using compatible file name for sidecar files.*
- void **setMetadataWritingMode** (const int mode)  
*Set metadata writing mode.*
- int **metadataWritingMode** () const  
*Return the metadata writing mode.*
- void **setUpdateFileTimeStamp** (bool on)  
*Enable or disable file timestamp updating when metadata are saved.*
- bool **updateFileTimeStamp** () const  
*Return true if file timestamp is updated when metadata are saved.*

### Metadata item information manipulation methods

- bool **setItemProgramId** (const QString &program, const QString &version) const  
*Set Program name and program version in Exif and Iptc Metadata.*
- QSize **getItemDimensions** () const  
*Return the size of item in pixels using Exif tags.*
- bool **setItemDimensions** (const QSize &size) const  
*Set the size of item in pixels in Exif tags.*
- **MetaEngine::ImageOrientation** **getItemOrientation** () const  
*Return the item orientation set in Exif metadata.*
- bool **setItemOrientation** (**ImageOrientation** orientation) const  
*Set the Exif orientation tag of item.*
- **MetaEngine::ImageColorWorkSpace** **getItemColorWorkSpace** () const  
*Return the item color-space set in Exif metadata.*
- bool **setItemColorWorkSpace** (**ImageColorWorkSpace** workspace) const  
*Set the Exif color-space tag of item.*
- QDateTime **getItemDateTime** () const  
*Return the time stamp of item.*
- bool **setItemDateTime** (const QDateTime &dateTime, bool setDateDigitized=false) const  
*Set the Exif and Iptc time stamp.*
- QDateTime **getItemDigitizationDateTime** (bool fallbackToCreationTime=false) const  
*Return the digitization time stamp of the item.*
- bool **getItemPreview** (QImage &preview) const  
*Return a QImage copy of Iptc preview image.*
- bool **setItemPreview** (const QImage &preview) const  
*Set the Iptc preview image.*
- QByteArray **getItemIccProfile** () const  
*Get image ICC profile.*
- bool **setItemIccProfile** (const QByteArray &iccData) const  
*Set image ICC profile.*

## Static Public Member Functions

### Static methods

- static bool `initializeExiv2 ()`  
*Return true if Exiv2 library initialization is done properly.*
- static bool `supportXmp ()`  
*Return true if Exiv2 library is compiled with Xmp metadata support.*
- static bool `supportJpegXL ()`  
*Return true if Exiv2 library is compiled with JpegXL metadata support.*
- static bool `supportBmff ()`  
*Return true if library support Base Media File Format (aka CR3, HEIF, HEIC, and AVIF).*
- static bool `supportMetadataWriting (const QString &typeMime)`  
*Return true if library can write metadata to typeMime file format.*
- static QString `Exiv2Version ()`  
*Return a string version of Exiv2 release in format "major.minor.patch".*

### GPS manipulation methods

- class `MetaEnginePreviews`
- bool `initializeGPSInfo ()`  
*Make sure all static required GPS EXIF and XMP tags exist.*
- bool `getGPSInfo (double &altitude, double &latitude, double &longitude) const`  
*Get all GPS location information set in item.*
- QString `getGPSLatitudeString () const`  
*Get GPS location information set in the item, in the GPSCoordinate format as described in the XMP specification.*
- QString `getGPSLongitudeString () const`
- bool `getGPSLatitudeNumber (double *const latitude) const`  
*Get GPS location information set in the item, as a double floating point number as in degrees where the sign determines the direction ref (North + / South - ; East + / West -).*
- bool `getGPSLongitudeNumber (double *const longitude) const`
- bool `getGPSAltitude (double *const altitude) const`  
*Get GPS altitude information, in meters, relative to sea level (positive sign above sea level)*
- bool `setGPSInfo (const double altitude, const double latitude, const double longitude)`  
*Set all GPS location information into item.*
- bool `setGPSInfo (const double *const altitude, const double latitude, const double longitude)`  
*Set all GPS location information into item.*
- bool `setGPSInfo (const double altitude, const QString &latitude, const QString &longitude)`  
*Set all GPS location information into item.*
- bool `removeGPSInfo ()`  
*Remove all Exif tags relevant of GPS location information.*
- static void `convertToRational (const double number, long int *const numerator, long int *const denominator, const int rounding)`  
*This method converts 'number' to a rational value, returned in the 'numerator' and 'denominator' parameters.*
- static void `convertToRationalSmallDenominator (const double number, long int *const numerator, long int *const denominator)`  
*This method convert a 'number' to a rational value, returned in 'numerator' and 'denominator' parameters.*
- static double `convertDegreeAngleToDouble (double degrees, double minutes, double seconds)`  
*Converts degrees values as a double representation.*
- static QString `convertToGPSCoordinateString (const long int numeratorDegrees, const long int denominatorDegrees, const long int numeratorMinutes, const long int denominatorMinutes, const long int numeratorSeconds, const long int denominatorSeconds, const char directionReference)`



Converts a GPS position stored as rationals in Exif to the form described as GPSCoordinate in the XMP specification, either in the form "256,45,34N" or "256,45.566667N".

- static QString **convertToGPSCoordinateString** (const bool isLatitude, double coordinate)
 

Converts a GPS position stored as double floating point number in degrees to the form described as GPSCoordinate in the XMP specification.
- static bool **convertFromGPSCoordinateString** (const QString &coordinate, long int \*const numeratorDegrees, long int \*const denominatorDegrees, long int \*const numeratorMinutes, long int \*const denominatorMinutes, long int \*const numeratorSeconds, long int \*const denominatorSeconds, char \*const directionReference)
 

Converts a GPSCoordinate string as defined by XMP to three rationals and the direction reference.
- static bool **convertFromGPSCoordinateString** (const QString &gpsString, double \*const coordinate)
 

Convert a GPSCoordinate string as defined by XMP to a double floating point number in degrees where the sign determines the direction ref (North + / South - ; East + / West -).
- static bool **convertToUserPresentableNumbers** (const QString &coordinate, int \*const degrees, int \*const minutes, double \*const seconds, char \*const directionReference)
 

Converts a GPSCoordinate string to user presentable numbers, integer degrees and minutes and double floating point seconds, and a direction reference ('N' or 'S', 'E' or 'W')
- static void **convertToUserPresentableNumbers** (const bool isLatitude, double coordinate, int \*const degrees, int \*const minutes, double \*const seconds, char \*const directionReference)
 

Converts a double floating point number to user presentable numbers, integer degrees and minutes and double floating point seconds, and a direction reference ('N' or 'S', 'E' or 'W').
- bool **setProgramId** () const
 

Set the Program Name and Program Version information in Exif and Iptc metadata.

## File I/O methods

- void **setFilePath** (const QString &path)
 

Set the file path of current item.
- QString **getFilePath** () const
 

Return the file path of current item.
- bool **load** (const QString &filePath, Backend \*backend=nullptr)
 

Load all metadata (Exif, Iptc, Xmp, and JFIF Comments) from a picture (JPEG, RAW, TIFF, PNG, DNG, etc...).
- bool **loadFromSidecarAndMerge** (const QString &filePath)
 

Load metadata from a sidecar file and merge.
- bool **save** (const QString &filePath, bool setVersion=false) const
 

Save all metadata to a file.
- bool **applyChanges** (bool setVersion=false) const
 

The same than **save()** method, but it apply on current item.
- bool **exportChanges** (const QString &exvTmpFile) const
 

Export metadata to a temporary EXV file container.
- static QString **sidecarFilePathForFile** (const QString &path)
 

Return the XMP Sidecar file path for a item file path.
- static QString **sidecarPath** (const QString &path)
 

Like **sidecarFilePathForFile()**, but works for local file path.
- static QUrl **sidecarUrl** (const QUrl &url)
 

Like **sidecarFilePathForFile()**, but works for remote URLs.
- static QUrl **sidecarUrl** (const QString &path)
 

Gives a file url for a local path.
- static bool **hasSidecar** (const QString &path)
 

Performs a QFileInfo based check if the given local file has a sidecar.
- static QString **backendName** (Backend t)
 

Return a string of backend name used to parse metadata from file.

### Comments manipulation methods

- bool **hasComments** () const  
*Return 'true' if metadata container in memory as Comments.*
- bool **clearComments** () const  
*Clear the Comments metadata container in memory.*
- QByteArray **getComments** () const  
*Return a Qt byte array copy of Comments container get from current item.*
- QString **getCommentsDecoded** () const  
*Return a Qt string object of Comments from current item decoded using the 'detectEncodingAndDecode()' method.*
- bool **setComments** (const QByteArray &data) const  
*Set the Comments data using a Qt byte array.*
- static bool **canWriteComment** (const QString &filePath)  
*Return 'true' if Comments can be written in file.*
- static QString **detectLanguageAlt** (const QString &value, QString &lang)  
*Language Alternative autodetection.*

### Exif manipulation methods

- TagsMap **getStdExifTagsList** () const  
*Return a map of all standard Exif tags supported by Exiv2.*
- TagsMap **getMakernoteTagsList** () const  
*Return a map of all non-standard Exif tags (makernotes) supported by Exiv2.*
- bool **hasExif** () const  
*Return 'true' if metadata container in memory as Exif.*
- bool **clearExif** () const  
*Clear the Exif metadata container in memory.*
- QByteArray **getExifEncoded** (bool addExifHeader=false) const  
*Returns the exif data encoded to a QByteArray in a form suitable for storage in a JPEG image.*
- bool **setExif** (const QByteArray &data) const  
*Set the Exif data using a Qt byte array.*
- QImage **getExifThumbnail** (bool fixOrientation) const  
*Return a QImage copy of Exif thumbnail image.*
- bool **rotateExifQImage** (QImage &image, ImageOrientation orientation) const  
*Fix orientation of a QImage image accordingly with Exif orientation tag.*
- bool **setExifThumbnail** (const QImage &thumb) const  
*Set the Exif Thumbnail image.*
- bool **removeExifThumbnail** () const  
*Remove the Exif Thumbnail from the item.*
- bool **setTiffThumbnail** (const QImage &thumb) const  
*Adds a JPEG thumbnail to a TIFF images.*
- QString **getExifComment** (bool readDescription=true) const  
*Return a QString copy of Exif user comments.*
- QString **getExifTagComment** (const char \*exifTagName) const  
*Return a Exif tag comment like a string.*
- bool **setExifComment** (const QString &comment, bool writeDescription=true) const  
*Set the Exif user comments from item.*
- QString **getExifTagString** (const char \*exifTagName, bool escapeCR=true) const  
*Get an Exif tags content like a string.*
- bool **setExifTagString** (const char \*exifTagName, const QString &value) const

- Set an Exif tag content using a string.*

  - bool [getExifTagLong](#) (const char \*exifTagName, long &val) const

*Get an Exif tag content like a long value.*
- bool [getExifTagLong](#) (const char \*exifTagName, long &val, int component) const

*Get an Exif tag content like a long value.*
- bool [setExifTagLong](#) (const char \*exifTagName, long val) const

*Set an Exif tag content using a long value.*
- bool [setExifTagUShort](#) (const char \*exifTagName, unsigned int val) const

*Set an Exif tag content using a unsigned short value.*
- bool [getExifTagRational](#) (const char \*exifTagName, long int &num, long int &den, int component=0) const

*Get the 'component' index of an Exif tags content like a rational value.*
- bool [setExifTagRational](#) (const char \*exifTagName, long int num, long int den) const

*Set an Exif tag content using a rational value.*
- bool [setExifTagURational](#) (const char \*exifTagName, unsigned long int num, unsigned long int den) const

*Set an Exif tag content using a unsigned rational value.*
- QByteArray [getExifTagData](#) (const char \*exifTagName) const

*Get an Exif tag content like a bytes array.*
- bool [setExifTagData](#) (const char \*exifTagName, const QByteArray &data) const

*Set an Exif tag content using a bytes array.*
- QVariant [getExifTagVariant](#) (const char \*exifTagName, bool rationalAsListOfInts=true, bool escapeCR=true, int component=0) const

*Get an Exif tags content as a QVariant.*
- bool [setExifTagVariant](#) (const char \*exifTagName, const QVariant &data, bool rationalWantSmall↔Denominator=true) const

*Set an Exif tag content using a QVariant.*
- bool [removeExifTag](#) (const char \*exifTagName) const

*Remove the Exif tag 'exifTagName' from Exif metadata.*
- QString [getExifTagTitle](#) (const char \*exifTagName)

*Return the Exif Tag title or a null string.*
- QString [getExifTagDescription](#) (const char \*exifTagName)

*Return the Exif Tag description or a null string.*
- QString [createExifUserStringFromValue](#) (const char \*exifTagName, const QVariant &val, bool escape↔CR=true)

*Takes a QVariant value as it could have been retrieved by getExifTagVariant with the given exifTagName, and returns its value properly converted to a string (including translations from Exiv2).*
- [MetaEngine::MetaDataMap](#) [getExifTagsDataList](#) (const QStringList &exifKeysFilter=QStringList(), bool invertSelection=false, bool extractBinary=true) const

*Return a map of Exif tags name/value found in metadata sorted by Exif keys given by 'exifKeysFilter'.*
- static bool [canWriteExif](#) (const QString &filePath)

*Return 'true' if Exif can be written in file.*

### IPTC manipulation methods

- [MetaEngine::TagsMap](#) [getIptcTagsList](#) () const

*Return a map of all standard Iptc tags supported by Exiv2.*
- bool [hasIptc](#) () const

*Return 'true' if metadata container in memory as Iptc.*
- bool [clearIptc](#) () const

*Clear the Iptc metadata container in memory.*
- QByteArray [getIptc](#) (bool addIrbHeader=false) const

*Return a Qt byte array copy of Iptc container get from current item.*

- bool [setIptc](#) (const QByteArray &data) const  
*Set the Iptc data using a Qt byte array.*
- QString [getIptcTagString](#) (const char \*iptcTagName, bool escapeCR=true) const  
*Get an Iptc tag content like a string.*
- bool [setIptcTagString](#) (const char \*iptcTagName, const QString &value) const  
*Set an Iptc tag content using a string.*
- QStringList [getIptcTagsStringList](#) (const char \*iptcTagName, bool escapeCR=true) const  
*Returns a strings list with of multiple Iptc tags from the item.*
- bool [setIptcTagsStringList](#) (const char \*iptcTagName, int maxSize, const QStringList &oldValues, const QStringList &newValues) const  
*Set multiple Iptc tags contents using a strings list.*
- QByteArray [getIptcTagData](#) (const char \*iptcTagName) const  
*Get an Iptc tag content as a bytes array.*
- bool [setIptcTagData](#) (const char \*iptcTagName, const QByteArray &data) const  
*Set an Iptc tag content using a bytes array.*
- bool [removeIptcTag](#) (const char \*iptcTagName) const  
*Remove the all instance of Iptc tags 'iptcTagName' from Iptc metadata.*
- QString [getIptcTagTitle](#) (const char \*iptcTagName)  
*Return the Iptc Tag title or a null string.*
- QString [getIptcTagDescription](#) (const char \*iptcTagName)  
*Return the Iptc Tag description or a null string.*
- [MetaEngine::MetaDataMap](#) [getIptcTagsDataList](#) (const QStringList &iptcKeysFilter=QStringList(), bool invertSelection=false) const  
*Return a map of Iptc tags name/value found in metadata sorted by Iptc keys given by 'iptcKeysFilter'.*
- QStringList [getIptcKeywords](#) () const  
*Return a strings list of Iptc keywords from item.*
- bool [setIptcKeywords](#) (const QStringList &oldKeywords, const QStringList &newKeywords) const  
*Set Iptc keywords using a list of strings defined by 'newKeywords' parameter.*
- QStringList [getIptcSubjects](#) () const  
*Return a strings list of Iptc subjects from item.*
- bool [setIptcSubjects](#) (const QStringList &oldSubjects, const QStringList &newSubjects) const  
*Set Iptc subjects using a list of strings defined by 'newSubjects' parameter.*
- QStringList [getIptcSubCategories](#) () const  
*Return a strings list of Iptc sub-categories from item.*
- bool [setIptcSubCategories](#) (const QStringList &oldSubCategories, const QStringList &newSubCategories) const  
*Set Iptc sub-categories using a list of strings defined by 'newSubCategories' parameter.*
- static bool [canWriteIptc](#) (const QString &filePath)  
*Return 'true' if Iptc can be written in file.*

### XMP manipulation methods

- [MetaEngine::TagsMap](#) [getXmpTagsList](#) () const  
*Return a map of all standard Xmp tags supported by Exiv2.*
- bool [hasXmp](#) () const  
*Return 'true' if metadata container in memory as Xmp.*
- bool [clearXmp](#) () const  
*Clear the Xmp metadata container in memory.*
- QByteArray [getXmp](#) () const  
*Return a Qt byte array copy of XMP container get from current item.*

- bool [setXmp](#) (const QByteArray &data) const  
*Set the Xmp data using a Qt byte array.*
- QString [getXmpTagString](#) (const char \*xmpTagName, bool escapeCR=true) const  
*Get a Xmp tag content like a string.*
- bool [setXmpTagString](#) (const char \*xmpTagName, const QString &value) const  
*Set a Xmp tag content using a string.*
- bool [setXmpTagString](#) (const char \*xmpTagName, const QString &value, [XmpTagType](#) type) const  
*Set a Xmp tag with a specific type.*
- QString [getXmpTagTitle](#) (const char \*xmpTagName)  
*Return the Xmp Tag title or a null string.*
- QString [getXmpTagDescription](#) (const char \*xmpTagName)  
*Return the Xmp Tag description or a null string.*
- [MetaEngine::MetaDataMap](#) [getXmpTagsDataList](#) (const QStringList &xmpKeysFilter=QStringList(), bool invertSelection=false) const  
*Return a map of Xmp tags name/value found in metadata sorted by Xmp keys given by 'xmpKeysFilter'.*
- [MetaEngine::AltLangMap](#) [getXmpTagStringListLangAlt](#) (const char \*xmpTagName, bool escapeCR=true) const  
*Get all redondant Alternative Language Xmp tags content like a map.*
- bool [setXmpTagStringListLangAlt](#) (const char \*xmpTagName, const [MetaEngine::AltLangMap](#) &values) const  
*Set an Alternative Language Xmp tag content using a map.*
- QString [getXmpTagStringLangAlt](#) (const char \*xmpTagName, const QString &langAlt, bool escapeCR) const  
*Get a Xmp tag content like a string set with an alternative language header 'langAlt' (like "fr-FR" for French - RFC3066 notation) If 'escapeCR' parameter is true, the CR characters will be removed.*
- bool [setXmpTagStringLangAlt](#) (const char \*xmpTagName, const QString &value, const QString &langAlt) const  
*Set a Xmp tag content using a string with an alternative language header.*
- QStringList [getXmpTagStringSeq](#) (const char \*xmpTagName, bool escapeCR=true) const  
*Get a Xmp tag content like a sequence of strings.*
- bool [setXmpTagStringSeq](#) (const char \*xmpTagName, const QStringList &seq) const  
*Set a Xmp tag content using the sequence of strings 'seq'.*
- QStringList [getXmpTagStringBag](#) (const char \*xmpTagName, bool escapeCR) const  
*Get a Xmp tag content like a bag of strings.*
- bool [setXmpTagStringBag](#) (const char \*xmpTagName, const QStringList &bag) const  
*Set a Xmp tag content using the bag of strings 'bag'.*
- bool [addToXmpTagStringBag](#) (const char \*xmpTagName, const QStringList &entriesToAdd) const  
*Set an Xmp tag content using a list of strings defined by the 'entriesToAdd' parameter.*
- bool [removeFromXmpTagStringBag](#) (const char \*xmpTagName, const QStringList &entriesToRemove) const  
*Remove those Xmp tag entries that are listed in entriesToRemove from the entries in metadata.*
- QVariant [getXmpTagVariant](#) (const char \*xmpTagName, bool rationalAsListOfInts=true, bool stringEscape↔CR=true) const  
*Get an Xmp tag content as a QVariant.*
- QStringList [getXmpKeywords](#) () const  
*Return a strings list of Xmp keywords from item.*
- bool [setXmpKeywords](#) (const QStringList &newKeywords) const  
*Set Xmp keywords using a list of strings defined by 'newKeywords' parameter.*
- bool [removeXmpKeywords](#) (const QStringList &keywordsToRemove)  
*Remove those Xmp keywords that are listed in keywordsToRemove from the keywords in metadata.*
- QStringList [getXmpSubjects](#) () const  
*Return a strings list of Xmp subjects from item.*
- bool [setXmpSubjects](#) (const QStringList &newSubjects) const  
*Set Xmp subjects using a list of strings defined by 'newSubjects' parameter.*

- bool [removeXmpSubjects](#) (const QStringList &subjectsToRemove)  
*Remove those Xmp subjects that are listed in subjectsToRemove from the subjects in metadata.*
- QStringList [getXmpSubCategories](#) () const  
*Return a strings list of Xmp sub-categories from item.*
- bool [setXmpSubCategories](#) (const QStringList &newSubCategories) const  
*Set Xmp sub-categories using a list of strings defined by 'newSubCategories' parameter.*
- bool [removeXmpSubCategories](#) (const QStringList &categoriesToRemove)  
*Remove those Xmp sub-categories that are listed in categoriesToRemove from the sub-categories in metadata.*
- bool [removeXmpTag](#) (const char \*xmpTagName, bool family=false) const  
*Remove the Xmp tag 'xmpTagName' from Xmp metadata.*
- static bool [canWriteXmp](#) (const QString &filePath)  
*Return 'true' if Xmp can be written in file.*
- static bool [registerXmpNameSpace](#) (const QString &uri, const QString &prefix)  
*Register a namespace which Exiv2 doesn't know yet.*
- static bool [unregisterXmpNameSpace](#) (const QString &uri)  
*Unregister a previously registered custom namespace.*

## 9.978.1 Member Typedef Documentation

### 9.978.1.1 AltLangMap

```
typedef QMap<QString, QString> Digikam::MetaEngine::AltLangMap
```

The map key is the language code following RFC3066 notation (like "fr-FR" for French), and the map value the text.

### 9.978.1.2 TagsMap

```
typedef QMap<QString, QStringList> Digikam::MetaEngine::TagsMap
```

- name,
- title,
- description.

## 9.978.2 Member Enumeration Documentation

### 9.978.2.1 Backend

```
enum Digikam::MetaEngine::Backend
```

#### Enumerator

Exiv2Backend	Default backend used by <a href="#">MetaEngine</a> .
LibRawBackend	<a href="#">DMetadata</a> only.
LibHeifBackend	<a href="#">DMetadata</a> only.
ImageMagickBackend	<a href="#">DMetadata</a> only.
FFMpegBackend	<a href="#">DMetadata</a> only.
ExifToolBackend	<a href="#">DMetadata</a> only.
VideoMergeBackend	<a href="#">DMetadata</a> only.
NoBackend	No backend used (aka file cannot be read).

### 9.978.2.2 MetadataWritingMode

enum `Digikam::MetaEngine::MetadataWritingMode`

See also

[MetadataWritingMode\(\)](#), [metadataWritingMode\(\)](#)

Enumerator

<code>WRITE_TO_FILE_ONLY</code>	Write metadata to item file only.
<code>WRITE_TO_SIDECAR_ONLY</code>	Write metadata to sidecar file only.
<code>WRITE_TO_SIDECAR_AND_FILE</code>	Write metadata to item and sidecar files.
<code>WRITE_TO_SIDECAR_ONLY_FOR_READ_ONLY_FILES</code>	Write metadata to sidecar file only for read only items such as RAW files for example.

## 9.978.3 Member Function Documentation

### 9.978.3.1 addToXmpTagStringBag()

```
bool Digikam::MetaEngine::addToXmpTagStringBag (
    const char * xmpTagName,
    const QStringList & entriesToAdd ) const
```

The existing entries are preserved. The method will compare all new with all already existing entries to prevent duplicates in the item. Return true if the entries have been added to metadata.

### 9.978.3.2 applyChanges()

```
bool Digikam::MetaEngine::applyChanges (
    bool setVersion = false ) const
```

Return true if metadata have been saved into file.

### 9.978.3.3 backendName()

```
QString Digikam::MetaEngine::backendName (
    Backend t ) [static]
```

See Backend enum for details.

### 9.978.3.4 convertDegreeAngleToDouble()

```
double Digikam::MetaEngine::convertDegreeAngleToDouble (
    double degrees,
    double minutes,
    double seconds ) [static]
```

This code take a care about hemisphere position.

**9.978.3.5 convertFromGPSCoordinateString() [1/2]**

```
bool Digikam::MetaEngine::convertFromGPSCoordinateString (
    const QString & coordinate,
    long int *const numeratorDegrees,
    long int *const denominatorDegrees,
    long int *const numeratorMinutes,
    long int *const denominatorMinutes,
    long int *const numeratorSeconds,
    long int *const denominatorSeconds,
    char *const directionReference ) [static]
```

Returns true if the conversion was successful. If minutes is given in the fractional form, a denominator of 1000000 for the minutes will be used.

**9.978.3.6 convertFromGPSCoordinateString() [2/2]**

```
bool Digikam::MetaEngine::convertFromGPSCoordinateString (
    const QString & gpsString,
    double *const coordinate ) [static]
```

Returns true if the conversion was successful.

**9.978.3.7 convertToGPSCoordinateString()**

```
QString Digikam::MetaEngine::convertToGPSCoordinateString (
    const long int numeratorDegrees,
    const long int denominatorDegrees,
    const long int numeratorMinutes,
    const long int denominatorMinutes,
    const long int numeratorSeconds,
    const long int denominatorSeconds,
    const char directionReference ) [static]
```

Precision: A second at sea level measures 30m for our purposes, a minute 1800m. (for more details, see [https://en.wikipedia.org/wiki/Geographic\\_coordinate\\_system](https://en.wikipedia.org/wiki/Geographic_coordinate_system)) This means with a decimal precision of 8 for minutes we get +/-0,018mm. (if I calculated correctly)

**9.978.3.8 convertToRational()**

```
void Digikam::MetaEngine::convertToRational (
    const double number,
    long int *const numerator,
    long int *const denominator,
    const int rounding ) [static]
```

Set the precision using 'rounding' parameter. Use this method if you want to retrieve a most exact rational for a number without further properties, without any requirements to the denominator.



### 9.978.3.9 convertToRationalSmallDenominator()

```
void Digikam::MetaEngine::convertToRationalSmallDenominator (
    const double number,
    long int *const numerator,
    long int *const denominator ) [static]
```

This method will be able to retrieve a rational number from a double - if you constructed your double with 1.0 / 4786.0, this method will retrieve 1 / 4786. If your number is not expected to be rational, use the method above which is just as exact with rounding = 4 and more exact with rounding > 4.

### 9.978.3.10 convertToUserPresentableNumbers()

```
void Digikam::MetaEngine::convertToUserPresentableNumbers (
    const bool isLatitude,
    double coordinate,
    int *const degrees,
    int *const minutes,
    double *const seconds,
    char *const directionReference ) [static]
```

The method needs to know for the direction reference if the latitude or the longitude is meant by the double parameter.

### 9.978.3.11 createExifUserStringFromValue()

```
QString Digikam::MetaEngine::createExifUserStringFromValue (
    const char * exifTagName,
    const QVariant & val,
    bool escapeCR = true )
```

This is equivalent to calling `getExifTagString` directly. If `escapeCR` is true CR characters will be removed from the result.

### 9.978.3.12 detectLanguageAlt()

```
QString Digikam::MetaEngine::detectLanguageAlt (
    const QString & value,
    QString & lang ) [static]
```

Return a QString without language alternative header. Header is saved into 'lang'. If no language alternative is found, value is returned as well and 'lang' is set to a null string.

### 9.978.3.13 exportChanges()

```
bool Digikam::MetaEngine::exportChanges (
    const QString & exvTmpFile ) const
```

'exvTmpFile' is the path to the temporary EXV container to create.

#### 9.978.3.14 `getComments()`

```
QByteArray Digikam::MetaEngine::getComments ( ) const
```

Comments are JFIF section of JPEG images. Look Exiv2 API for more information. Return a null Qt byte array if there is no Comments metadata in memory.

#### 9.978.3.15 `getCommentsDecoded()`

```
QString Digikam::MetaEngine::getCommentsDecoded ( ) const
```

Return a null string if there is no Comments metadata available.

#### 9.978.3.16 `getDigitizationDateTime()`

```
QDateTime Digikam::MetaEngine::getDigitizationDateTime (
    bool fallbackToCreationTime = false ) const
```

First Exif information is checked, then IPTC. If no digitization time stamp is found, [getItemDateTime\(\)](#) is called if `fallbackToCreationTime` is true, or a null `QDateTime` is returned if `fallbackToCreationTime` is false.

#### 9.978.3.17 `getExifComment()`

```
QString Digikam::MetaEngine::getExifComment (
    bool readDescription = true ) const
```

Return a null string if user comments cannot be found.

#### 9.978.3.18 `getExifEncoded()`

```
QByteArray Digikam::MetaEngine::getExifEncoded (
    bool addExifHeader = false ) const
```

Note that this encoding is a lossy operation.

Set true 'addExifHeader' parameter to add an Exif header to Exif metadata. Returns a null Qt byte array if there is no Exif metadata in memory.

#### 9.978.3.19 `getExifTagComment()`

```
QString Digikam::MetaEngine::getExifTagComment (
    const char * exifTagName ) const
```

Return a null string if user comments cannot be found.

**9.978.3.20 getExifTagData()**

```
QByteArray Digikam::MetaEngine::getExifTagData (
    const char * exifTagName ) const
```

Return an empty bytes array if Exif tag cannot be found.

**9.978.3.21 getExifTagLong() [1/2]**

```
bool Digikam::MetaEngine::getExifTagLong (
    const char * exifTagName,
    long & val ) const
```

Return true if Exif tag be found.

**9.978.3.22 getExifTagLong() [2/2]**

```
bool Digikam::MetaEngine::getExifTagLong (
    const char * exifTagName,
    long & val,
    int component ) const
```

Return true if Exif tag be found.

**9.978.3.23 getExifTagRational()**

```
bool Digikam::MetaEngine::getExifTagRational (
    const char * exifTagName,
    long int & num,
    long int & den,
    int component = 0 ) const
```

'num' and 'den' are the numerator and the denominator of the rational value. Return true if Exif tag be found.

**9.978.3.24 getExifTagsDataList()**

```
MetaEngine::MetaDataMap Digikam::MetaEngine::getExifTagsDataList (
    const QStringList & exifKeysFilter = QStringList(),
    bool invertSelection = false,
    bool extractBinary = true ) const
```

'exifKeysFilter' is a QStringList of Exif keys. For example, if you use the string list given below:

```
"Iop" "Thumbnail" "Image" "Photo"
```

List can be empty to not filter output.

... this method will return a map of all Exif tags which :

- include "Iop", or "Thumbnail", or "Image", or "Photo" in the Exif tag keys if 'invertSelection' is false.
- not include "Iop", or "Thumbnail", or "Image", or "Photo" in the Exif tag keys if 'invertSelection' is true. if 'extractBinary' is true, tags with undefined types of data are extracted (default), else contents is replaced by "Binary data ... bytes". Take a care as large binary data as original RAW data from DNG container can be huge and listing Exif tags from GUI can take a while.

### 9.978.3.25 `getExifTagString()`

```
QString Digikam::MetaEngine::getExifTagString (
    const char * exifTagName,
    bool escapeCR = true ) const
```

If 'escapeCR' parameter is true, the CR characters will be removed. If Exif tag cannot be found a null string is returned.

### 9.978.3.26 `getExifTagVariant()`

```
QVariant Digikam::MetaEngine::getExifTagVariant (
    const char * exifTagName,
    bool rationalAsListOfInts = true,
    bool escapeCR = true,
    int component = 0 ) const
```

Returns a null QVariant if the Exif tag cannot be found. For string and integer values the matching QVariant types will be used, for date and time values QVariant::DateTime. Rationals will be returned as QVariant::List with two integer QVariants (numerator, denominator) if rationalAsListOfInts is true, as double if rationalAsListOfInts is false. An exif tag of numerical type may contain more than one value; set component to the desired index.

### 9.978.3.27 `getExifThumbnail()`

```
QImage Digikam::MetaEngine::getExifThumbnail (
    bool fixOrientation ) const
```

Return a null image if thumbnail cannot be found. The 'fixOrientation' parameter will rotate automatically the thumbnail if Exif orientation tags information are attached with thumbnail.

### 9.978.3.28 `getGPSInfo()`

```
bool Digikam::MetaEngine::getGPSInfo (
    double & altitude,
    double & latitude,
    double & longitude ) const
```

Return true if all information can be found.

### 9.978.3.29 `getGPSLatitudeNumber()`

```
bool Digikam::MetaEngine::getGPSLatitudeNumber (
    double *const latitude ) const
```

Returns true if the information is available.

### 9.978.3.30 `getGPSLatitudeString()`

```
QString Digikam::MetaEngine::getGPSLatitudeString ( ) const
```

Returns a null string in the information cannot be found.

### 9.978.3.31 getIptc()

```
QByteArray Digikam::MetaEngine::getIptc (
    bool addIrbHeader = false ) const
```

Set true 'addIrbHeader' parameter to add an Irb header to Iptc metadata. Return a null Qt byte array if there is no Iptc metadata in memory.

### 9.978.3.32 getIptcKeywords()

```
QStringList Digikam::MetaEngine::getIptcKeywords ( ) const
```

Return an empty list if no keyword are set.

### 9.978.3.33 getIptcSubCategories()

```
QStringList Digikam::MetaEngine::getIptcSubCategories ( ) const
```

Return an empty list if no sub-category are set.

### 9.978.3.34 getIptcSubjects()

```
QStringList Digikam::MetaEngine::getIptcSubjects ( ) const
```

Return an empty list if no subject are set.

### 9.978.3.35 getIptcTagData()

```
QByteArray Digikam::MetaEngine::getIptcTagData (
    const char * iptcTagName ) const
```

Return an empty bytes array if Iptc tag cannot be found.

### 9.978.3.36 getIptcTagsDataList()

```
MetaEngine::MetaDataMap Digikam::MetaEngine::getIptcTagsDataList (
    const QStringList & iptcKeysFilter = QStringList(),
    bool invertSelection = false ) const
```

'IptcKeysFilter' is a QStringList of Iptc keys. For example, if you use the string list given below:

```
"Envelope" "Application2"
```

List can be empty to not filter output.

... this method will return a map of all Iptc tags which :

- include "Envelope", or "Application2" in the Iptc tag keys if 'invertSelection' is false.
- not include "Envelope", or "Application2" in the Iptc tag keys if 'invertSelection' is true.

### 9.978.3.37 `getIptcTagsStringList()`

```
QStringList Digikam::MetaEngine::getIptcTagsStringList (
    const char * iptcTagName,
    bool escapeCR = true ) const
```

Return an empty list if no tag is found. Get the values of all IPTC tags with the given tag name in a string list. (In Iptc, there can be multiple tags with the same name) If the 'escapeCR' parameter is true, the CR characters will be removed. If no tag can be found an empty list is returned.

### 9.978.3.38 `getIptcTagString()`

```
QString Digikam::MetaEngine::getIptcTagString (
    const char * iptcTagName,
    bool escapeCR = true ) const
```

If 'escapeCR' parameter is true, the CR characters will be removed. If Iptc tag cannot be found a null string is returned.

### 9.978.3.39 `getItemColorWorkSpace()`

```
MetaEngine::ImageColorWorkSpace Digikam::MetaEngine::getItemColorWorkSpace ( ) const
```

The makernotes of item are also parsed to get this information. See [ImageColorWorkSpace](#) values for details.

### 9.978.3.40 `getItemDateTime()`

```
QDateTime Digikam::MetaEngine::getItemDateTime ( ) const
```

Exif information are check in first, IPTC in second if item don't have Exif information. If no time stamp is found, a null date is returned.

### 9.978.3.41 `getItemDimensions()`

```
QSize Digikam::MetaEngine::getItemDimensions ( ) const
```

Return a null dimension if size cannot be found.

### 9.978.3.42 `getItemOrientation()`

```
MetaEngine::ImageOrientation Digikam::MetaEngine::getItemOrientation ( ) const
```

The makernotes of item are also parsed to get this information. See [ImageOrientation](#) values for details.

### 9.978.3.43 `getItemPreview()`

```
bool Digikam::MetaEngine::getItemPreview (
    QImage & preview ) const
```

Return a null item if preview cannot be found.

#### 9.978.3.44 `getMimeType()`

```
QString Digikam::MetaEngine::getMimeType ( ) const
```

The information is read from the file; see the docs for [getPixelSize\(\)](#) to know when it is available.

#### 9.978.3.45 `getPixelSize()`

```
QSize Digikam::MetaEngine::getPixelSize ( ) const
```

This information is read from the file, not from the metadata. The returned `QSize` is valid if the [MetaEngine](#) object was *constructed* by reading a file or item data; the information is not available when the object was created from [MetaEngineData](#). Note that in the Exif or XMP metadata, there may be fields describing the item size. These fields are not accessed by this method. When replacing the metadata with `setData()`, the metadata may change; this information always keeps referring to the file it was initially read from.

#### 9.978.3.46 `getXmp()`

```
QByteArray Digikam::MetaEngine::getXmp ( ) const
```

Return a null Qt byte array if there is no Xmp metadata in memory.

#### 9.978.3.47 `getXmpKeywords()`

```
QStringList Digikam::MetaEngine::getXmpKeywords ( ) const
```

Return an empty list if no keyword are set.

#### 9.978.3.48 `getXmpSubCategories()`

```
QStringList Digikam::MetaEngine::getXmpSubCategories ( ) const
```

Return an empty list if no sub-category are set.

#### 9.978.3.49 `getXmpSubjects()`

```
QStringList Digikam::MetaEngine::getXmpSubjects ( ) const
```

Return an empty list if no subject are set.

**9.978.3.50 getXmpTagsDataList()**

```
MetaEngine::MetaDataMap Digikam::MetaEngine::getXmpTagsDataList (
    const QStringList & xmpKeysFilter = QStringList(),
    bool invertSelection = false ) const
```

'xmpKeysFilter' is a QStringList of Xmp keys. For example, if you use the string list given below:

"dc" // Dublin Core schema. "xmp" // Standard Xmp schema.

List can be empty to not filter output.

... this method will return a map of all Xmp tags which :

- include "dc", or "xmp" in the Xmp tag keys if 'invertSelection' is false.
- not include "dc", or "xmp" in the Xmp tag keys if 'invertSelection' is true.

**9.978.3.51 getXmpTagString()**

```
QString Digikam::MetaEngine::getXmpTagString (
    const char * xmpTagName,
    bool escapeCR = true ) const
```

If 'escapeCR' parameter is true, the CR characters will be removed. If Xmp tag cannot be found a null string is returned.

**9.978.3.52 getXmpTagStringBag()**

```
QStringList Digikam::MetaEngine::getXmpTagStringBag (
    const char * xmpTagName,
    bool escapeCR ) const
```

If 'escapeCR' parameter is true, the CR characters will be removed from strings. If Xmp tag cannot be found a null string list is returned.

**9.978.3.53 getXmpTagStringLangAlt()**

```
QString Digikam::MetaEngine::getXmpTagStringLangAlt (
    const char * xmpTagName,
    const QString & langAlt,
    bool escapeCR ) const
```

If Xmp tag cannot be found a null string is returned.

**9.978.3.54 getXmpTagStringListLangAlt()**

```
MetaEngine::AltLangMap Digikam::MetaEngine::getXmpTagStringListLangAlt (
    const char * xmpTagName,
    bool escapeCR = true ) const
```

See AltLangMap class description for details. If 'escapeCR' parameter is true, the CR characters will be removed from strings. If Xmp tag cannot be found a null string list is returned.



### 9.978.3.55 getXmpTagStringSeq()

```
QStringList Digikam::MetaEngine::getXmpTagStringSeq (
    const char * xmpTagName,
    bool escapeCR = true ) const
```

If 'escapeCR' parameter is true, the CR characters will be removed from strings. If Xmp tag cannot be found a null string list is returned.

### 9.978.3.56 getXmpTagVariant()

```
QVariant Digikam::MetaEngine::getXmpTagVariant (
    const char * xmpTagName,
    bool rationalAsListOfInts = true,
    bool stringEscapeCR = true ) const
```

Returns a null QVariant if the Xmp tag cannot be found. For string and integer values the matching QVariant types will be used, for date and time values QVariant::DateTime. Rationals will be returned as QVariant::List with two integer QVariants (numerator, denominator) if rationalAsListOfInts is true, as double if rationalAsListOfInts is false. Arrays (ordered, unordered, alternative) are returned as type QStringList. LangAlt values will have type Map (QMap<QString, QVariant>) with the language code as key and the contents as value, of type String.

### 9.978.3.57 initializeExiv2()

```
bool Digikam::MetaEngine::initializeExiv2 ( ) [static]
```

This method must be called before using libMetaEngine with multithreading. It initialize several non re-entrancy code from Adobe XMP SDK, and register a function to cleanup automatically all XMP SDK memory allocation. See Bug #166424 for details. It cleans up memory used by Adobe XMP SDK automatically at application exit. See Bug #166424 for details.

### 9.978.3.58 load()

```
bool Digikam::MetaEngine::load (
    const QString & filePath,
    Backend * backend = nullptr )
```

Return true if metadata have been loaded successfully from file. If backend is non null, return the backend used to populate metadata (Exiv2). See Backend enum for details.

### 9.978.3.59 loadFromData()

```
bool Digikam::MetaEngine::loadFromData (
    const QByteArray & imgData )
```

Return true if metadata have been loaded successfully from item data.

### 9.978.3.60 loadFromDataAndMerge()

```
bool Digikam::MetaEngine::loadFromDataAndMerge (
    const QByteArray & imgData,
    const QStringList & exclude = QStringList() )
```

Use 'exclude' to remove Exif tags from the 'imgData' that will not be merged. Return true if metadata have been loaded and merged successfully from item data.

### 9.978.3.61 loadFromSidecarAndMerge()

```
bool Digikam::MetaEngine::loadFromSidecarAndMerge (
    const QString & filePath )
```

Return true if metadata have been loaded successfully from file.

### 9.978.3.62 metadataWritingMode()

```
int Digikam::MetaEngine::metadataWritingMode ( ) const
```

#### Returns

Metadata writing mode as defined by the [MetadataWritingMode](#) enum.

#### See also

[MetadataWritingMode](#), [setMetadataWritingMode\(\)](#)

### 9.978.3.63 registerXmpNameSpace()

```
bool Digikam::MetaEngine::registerXmpNameSpace (
    const QString & uri,
    const QString & prefix ) [static]
```

This is only needed when new Xmp properties are added manually. 'uri' is the namespace url and 'prefix' the string used to construct new Xmp key (ex. "Xmp.digiKam.tagList").

#### Note

If the Xmp metadata is read from an item, namespaces are decoded and registered by Exiv2 at the same time.

### 9.978.3.64 removeExifTag()

```
bool Digikam::MetaEngine::removeExifTag (
    const char * exifTagName ) const
```

Return true if tag is removed successfully or if no tag was present.

**9.978.3.65 removeFromXmpTagStringBag()**

```
bool Digikam::MetaEngine::removeFromXmpTagStringBag (
    const char * xmpTagName,
    const QStringList & entriesToRemove ) const
```

Return true if tag entries are no longer contained in metadata. All other entries are preserved.

**9.978.3.66 removeGPSInfo()**

```
bool Digikam::MetaEngine::removeGPSInfo ( )
```

Return true if all tags have been removed successfully in metadata. NOTE: The XMP spec does not mention Xmp.exif.GPSLongitudeRef, and Xmp.exif.GPSLatitudeRef. But because we write historically until 7.6.0 release them in [setGPSInfo\(\)](#), we should also remove them here. See bug #450982.

**9.978.3.67 removeIptcTag()**

```
bool Digikam::MetaEngine::removeIptcTag (
    const char * iptcTagName ) const
```

Return true if all tags have been removed successfully (or none were present).

**9.978.3.68 removeXmpKeywords()**

```
bool Digikam::MetaEngine::removeXmpKeywords (
    const QStringList & keywordsToRemove )
```

Return true if keywords are no longer contained in metadata.

**9.978.3.69 removeXmpSubCategories()**

```
bool Digikam::MetaEngine::removeXmpSubCategories (
    const QStringList & categoriesToRemove )
```

Return true if subjects are no longer contained in metadata.

**9.978.3.70 removeXmpSubjects()**

```
bool Digikam::MetaEngine::removeXmpSubjects (
    const QStringList & subjectsToRemove )
```

Return true if subjects are no longer contained in metadata.

### 9.978.3.71 removeXmpTag()

```
bool Digikam::MetaEngine::removeXmpTag (
    const char * xmpTagName,
    bool family = false ) const
```

Return true if tag is removed successfully or if no tag was present.

### 9.978.3.72 rotateExifQImage()

```
bool Digikam::MetaEngine::rotateExifQImage (
    QImage & image,
    ImageOrientation orientation ) const
```

Return true if image is rotated, else false.

### 9.978.3.73 save()

```
bool Digikam::MetaEngine::save (
    const QString & filePath,
    bool setVersion = false ) const
```

This one can be different than original picture to perform transfer operation Return true if metadata have been saved into file.

### 9.978.3.74 setComments()

```
bool Digikam::MetaEngine::setComments (
    const QByteArray & data ) const
```

Return true if Comments metadata have been changed in memory.

### 9.978.3.75 setExif()

```
bool Digikam::MetaEngine::setExif (
    const QByteArray & data ) const
```

Return true if Exif metadata have been changed in memory.

### 9.978.3.76 setExifComment()

```
bool Digikam::MetaEngine::setExifComment (
    const QString & comment,
    bool writeDescription = true ) const
```

Look Exif specification for more details about this tag. Return true if Exif user comments have been changed in metadata.

**9.978.3.77 setExifTagData()**

```
bool Digikam::MetaEngine::setExifTagData (
    const char * exifTagName,
    const QByteArray & data ) const
```

Return true if tag is set successfully.

**9.978.3.78 setExifTagLong()**

```
bool Digikam::MetaEngine::setExifTagLong (
    const char * exifTagName,
    long val ) const
```

Return true if tag is set successfully.

**9.978.3.79 setExifTagRational()**

```
bool Digikam::MetaEngine::setExifTagRational (
    const char * exifTagName,
    long int num,
    long int den ) const
```

'num' and 'den' are the numerator and the denominator of the rational value. Return true if tag is set successfully.

**9.978.3.80 setExifTagString()**

```
bool Digikam::MetaEngine::setExifTagString (
    const char * exifTagName,
    const QString & value ) const
```

Return true if tag is set successfully.

**9.978.3.81 setExifTagURational()**

```
bool Digikam::MetaEngine::setExifTagURational (
    const char * exifTagName,
    unsigned long int num,
    unsigned long int den ) const
```

'num' and 'den' are the numerator and the denominator of the unsigned rational value. Return true if tag is set successfully.

**9.978.3.82 setExifTagUShort()**

```
bool Digikam::MetaEngine::setExifTagUShort (
    const char * exifTagName,
    unsigned int val ) const
```

Return true if tag is set successfully.

**9.978.3.83 setExifTagVariant()**

```
bool Digikam::MetaEngine::setExifTagVariant (
    const char * exifTagName,
    const QVariant & data,
    bool rationalWantSmallDenominator = true ) const
```

Returns true if tag is set successfully. All types described for the [getExifTagVariant\(\)](#) method are supported. Calling with a QVariant of type QByteArray is equivalent to calling setExifTagData. For the meaning of rationalWantSmall↔Denominator, see the documentation of the convertToRational methods. Setting a value with multiple components is currently not supported.

**9.978.3.84 setExifThumbnail()**

```
bool Digikam::MetaEngine::setExifThumbnail (
    const QImage & thumb ) const
```

The thumbnail image must have the right dimensions before. Look Exif specification for details. Return true if thumbnail have been changed in metadata.

**9.978.3.85 setGPSInfo() [1/3]**

```
bool Digikam::MetaEngine::setGPSInfo (
    const double *altitude,
    const double latitude,
    const double longitude )
```

Return true if all information have been changed in metadata. If you do not want altitude to be set, pass a null pointer. NOTE: The XMP spec does not mention Xmp.exif.GPSLatitudeRef, because the reference is included in Xmp.exif.GPSLatitude. See bug #450982.

NOTE: The XMP spec does not mention Xmp.exif.GPSLongitudeRef, because the reference is included in Xmp.↔exif.GPSLongitude. See bug #450982.

**9.978.3.86 setGPSInfo() [2/3]**

```
bool Digikam::MetaEngine::setGPSInfo (
    const double altitude,
    const double latitude,
    const double longitude )
```

Return true if all information have been changed in metadata.

**9.978.3.87 setGPSInfo() [3/3]**

```
bool Digikam::MetaEngine::setGPSInfo (
    const double altitude,
    const QString & latitude,
    const QString & longitude )
```

Return true if all information have been changed in metadata.

### 9.978.3.88 setImageDateTime()

```
bool Digikam::MetaEngine::setImageDateTime (
    const QDateTime & dateTime,
    bool setDateDigitized = false ) const
```

If 'setDateDigitized' parameter is true, the 'Digitalized' time stamp is set, else only 'Created' time stamp is set.

### 9.978.3.89 setIptc()

```
bool Digikam::MetaEngine::setIptc (
    const QByteArray & data ) const
```

Return true if Iptc metadata have been changed in memory.

### 9.978.3.90 setIptcKeywords()

```
bool Digikam::MetaEngine::setIptcKeywords (
    const QStringList & oldKeywords,
    const QStringList & newKeywords ) const
```

Use 'getImageKeywords()' method to set 'oldKeywords' parameter with existing keywords from item. The method will compare all new keywords with all old keywords to prevent duplicate entries in item. Return true if keywords have been changed in metadata.

### 9.978.3.91 setIptcSubCategories()

```
bool Digikam::MetaEngine::setIptcSubCategories (
    const QStringList & oldSubCategories,
    const QStringList & newSubCategories ) const
```

Use 'getImageSubCategories()' method to set 'oldSubCategories' parameter with existing sub-categories from item. The method will compare all new sub-categories with all old sub-categories to prevent duplicate entries in item. Return true if sub-categories have been changed in metadata.

### 9.978.3.92 setIptcSubjects()

```
bool Digikam::MetaEngine::setIptcSubjects (
    const QStringList & oldSubjects,
    const QStringList & newSubjects ) const
```

Use 'getImageSubjects()' method to set 'oldSubjects' parameter with existing subjects from item. The method will compare all new subjects with all old subjects to prevent duplicate entries in item. Return true if subjects have been changed in metadata.

### 9.978.3.93 setIptcTagData()

```
bool Digikam::MetaEngine::setIptcTagData (
    const char * iptcTagName,
    const QByteArray & data ) const
```

Return true if tag is set successfully.

### 9.978.3.94 setIptcTagsStringList()

```
bool Digikam::MetaEngine::setIptcTagsStringList (
    const char * iptcTagName,
    int maxSize,
    const QStringList & oldValues,
    const QStringList & newValues ) const
```

'maxSize' is the max characters size of one entry. Return true if all tags have been set successfully.

### 9.978.3.95 setIptcTagString()

```
bool Digikam::MetaEngine::setIptcTagString (
    const char * iptcTagName,
    const QString & value ) const
```

Return true if tag is set successfully.

### 9.978.3.96 setItemColorWorkSpace()

```
bool Digikam::MetaEngine::setItemColorWorkSpace (
    ImageColorWorkSpace workspace ) const
```

See ImageColorWorkSpace values for details Return true if work-space have been changed in metadata.

### 9.978.3.97 setItemDimensions()

```
bool Digikam::MetaEngine::setItemDimensions (
    const QSize & size ) const
```

Return true if size have been changed in metadata.

### 9.978.3.98 setItemOrientation()

```
bool Digikam::MetaEngine::setItemOrientation (
    ImageOrientation orientation ) const
```

See ImageOrientation values for details Return true if orientation have been changed in metadata.

### 9.978.3.99 setItemPreview()

```
bool Digikam::MetaEngine::setItemPreview (
    const QImage & preview ) const
```

The thumbnail item must have the right size before (64Kb max with JPEG file, else 256Kb). Look Iptc specification for details. Return true if preview have been changed in metadata. Re-implement this method if you want to use another item file format than JPEG to save preview.



### 9.978.3.100 setItemProgramId()

```
bool Digikam::MetaEngine::setItemProgramId (
    const QString & program,
    const QString & version ) const
```

Return true if information have been changed in metadata.

### 9.978.3.101 setMetadataWritingMode()

```
void Digikam::MetaEngine::setMetadataWritingMode (
    const int mode )
```

#### Parameters

<i>mode</i>	Metadata writing mode as defined by the <a href="#">MetadataWritingMode</a> enum.
-------------	---

#### See also

[MetadataWritingMode](#), [metadataWritingMode\(\)](#)

### 9.978.3.102 setTiffThumbnail()

```
bool Digikam::MetaEngine::setTiffThumbnail (
    const QImage & thumb ) const
```

Use this instead of setExifThumbnail for TIFF images.

### 9.978.3.103 setUpdateFileTimeStamp()

```
void Digikam::MetaEngine::setUpdateFileTimeStamp (
    bool on )
```

By default files timestamp are untouched.

### 9.978.3.104 setWriteRawFiles()

```
void Digikam::MetaEngine::setWriteRawFiles (
    const bool on )
```

By default RAW files are untouched.

### 9.978.3.105 setXmp()

```
bool Digikam::MetaEngine::setXmp (
    const QByteArray & data ) const
```

Return true if Xmp metadata have been changed in memory.

### 9.978.3.106 setXmpKeywords()

```
bool Digikam::MetaEngine::setXmpKeywords (
    const QStringList & newKeywords ) const
```

The existing keywords from item are preserved. The method will compare all new keywords with all already existing keywords to prevent duplicate entries in item. Return true if keywords have been changed in metadata.

### 9.978.3.107 setXmpSubCategories()

```
bool Digikam::MetaEngine::setXmpSubCategories (
    const QStringList & newSubCategories ) const
```

The existing sub-categories from item are preserved. The method will compare all new sub-categories with all already existing sub-categories to prevent duplicate entries in item. Return true if sub-categories have been changed in metadata.

### 9.978.3.108 setXmpSubjects()

```
bool Digikam::MetaEngine::setXmpSubjects (
    const QStringList & newSubjects ) const
```

The existing subjects from item are preserved. The method will compare all new subject with all already existing subject to prevent duplicate entries in item. Return true if subjects have been changed in metadata.

### 9.978.3.109 setXmpTagString() [1/2]

```
bool Digikam::MetaEngine::setXmpTagString (
    const char * xmpTagName,
    const QString & value ) const
```

Return true if tag is set successfully.

### 9.978.3.110 setXmpTagString() [2/2]

```
bool Digikam::MetaEngine::setXmpTagString (
    const char * xmpTagName,
    const QString & value,
    MetaEngine::XmpTagType type ) const
```

Return true if tag is set successfully. This method only accept NormalTag, ArrayBagTag and StructureTag. Other XmpTagTypes do nothing

### 9.978.3.111 setXmpTagStringBag()

```
bool Digikam::MetaEngine::setXmpTagStringBag (
    const char * xmpTagName,
    const QStringList & bag ) const
```

Return true if tag is set successfully.

### 9.978.3.112 setXmpTagStringLangAlt()

```
bool Digikam::MetaEngine::setXmpTagStringLangAlt (
    const char * xmpTagName,
    const QString & value,
    const QString & langAlt ) const
```

'langAlt' contain the language alternative information (like "fr-FR" for French - RFC3066 notation) or is null to set alternative language to default settings ("x-default"). Return true if tag is set successfully.

### 9.978.3.113 setXmpTagStringListLangAlt()

```
bool Digikam::MetaEngine::setXmpTagStringListLangAlt (
    const char * xmpTagName,
    const MetaEngine::AltLangMap & values ) const
```

See AltLangMap class description for details. If tag already exist, it will be removed before. Return true if tag is set successfully.

### 9.978.3.114 setXmpTagStringSeq()

```
bool Digikam::MetaEngine::setXmpTagStringSeq (
    const char * xmpTagName,
    const QStringList & seq ) const
```

Return true if tag is set successfully.

### 9.978.3.115 sidecarFilePathForFile()

```
QString Digikam::MetaEngine::sidecarFilePathForFile (
    const QString & path ) [static]
```

If item file path do not include a file name or is empty, this function return a null string.

### 9.978.3.116 supportBmff()

```
bool Digikam::MetaEngine::supportBmff ( ) [static]
```

Note: use this function only after to call [initializeExiv2\(\)](#), else false will always returned. The function return true only if Exiv2 >= 0.27.4 compiled with BMFF support.

## 9.979 Digikam::MetaEngineData Class Reference

### Public Member Functions

- [MetaEngineData](#) (const [MetaEngineData](#) &)
- [MetaEngineData](#) & [operator=](#) (const [MetaEngineData](#) &)

**Friends**

- class **MetaEngine**

## 9.980 Digikam::MetaEngineMergeHelper< Data, Key, KeyString, KeyStringList > Class Template Reference

**Public Member Functions**

- void [exclusiveMerge](#) (const Data &src, Data &dest)  
*Merge two (Exif,IPTC,Xmp) Data packages, the result is stored in dest.*
- void [mergeAll](#) (const Data &src, Data &dest)  
*Merge two (Exif,IPTC,Xmp) Data packages, where the result is stored in dest and fields from src take precedence over existing data from dest.*
- void [mergeFields](#) (const Data &src, Data &dest)  
*Merge two (Exif,IPTC,Xmp) Data packages, the result is stored in dest.*
- [MetaEngineMergeHelper](#) & [operator](#)<< (const KeyString &key)

**Public Attributes**

- KeyStringList **keys**

**9.980.1 Member Function Documentation****9.980.1.1 exclusiveMerge()**

```
template<class Data , class Key , class KeyString , class KeyStringList = QList<KeyString>>
void Digikam::MetaEngineMergeHelper< Data, Key, KeyString, KeyStringList >::exclusiveMerge (
    const Data & src,
    Data & dest ) [inline]
```

The following steps apply only to keys in "keys": The result is determined by src. Keys must exist in src to kept in dest. Fields from src take precedence over existing data from dest.

**9.980.1.2 mergeFields()**

```
template<class Data , class Key , class KeyString , class KeyStringList = QList<KeyString>>
void Digikam::MetaEngineMergeHelper< Data, Key, KeyString, KeyStringList >::mergeFields (
    const Data & src,
    Data & dest ) [inline]
```

Only keys in keys are considered for merging. Fields from src take precedence over existing data from dest.

## 9.981 Digikam::MetaEnginePreviews Class Reference

### Public Member Functions

- **MetaEnginePreviews** (const QByteArray &imgData)  
*Open the given image data and scan the image for embedded preview images.*
- **MetaEnginePreviews** (const QString &filePath)  
*Open the given file and scan for embedded preview images.*
- int **count** () const  
*Returns how many embedded previews are available.*
- QByteArray **data** (int index=0)  
*Retrieve the image data for the specified embedded preview image.*
- int **dataSize** (int index=0)  
*For each contained preview image, return the size of the image data in bytes, width and height of the preview, the mimeType and the file extension.*
- QString **fileExtension** (int index=0)
- int **height** (int index=0)
- QImage **image** (int index=0)  
*Loads the data of the specified preview and creates a QImage from this data.*
- bool **isEmpty** ()  
*Returns if there are any preview images available.*
- QString **mimeType** (int index=0)
- QString **originalMimeType** () const  
*Returns the mimeType of the original image, detected from the file's content.*
- QSize **originalSize** () const  
*Returns the pixel size of the original image, as read from the file (not the metadata).*
- int **size** () const
- int **width** (int index=0)

### 9.981.1 Member Function Documentation

#### 9.981.1.1 dataSize()

```
int Digikam::MetaEnginePreviews::dataSize (
    int index = 0 )
```

Ensure that index < [count\(\)](#). Previews are sorted by width\*height, largest first.

#### 9.981.1.2 image()

```
QImage Digikam::MetaEnginePreviews::image (
    int index = 0 )
```

Returns a null QImage if the loading failed.

## 9.982 Digikam::MetaEngineRotation Class Reference

### Public Types

- enum [TransformationAction](#) {  
[NoTransformation](#) = 0 , [FlipHorizontal](#) = 1 , [FlipVertical](#) = 2 , [Rotate90](#) = 5 ,  
[Rotate180](#) = 6 , [Rotate270](#) = 7 }

*This describes single transform primitives.*

### Public Member Functions

- **MetaEngineRotation** ()  
*Constructs the identity matrix (the matrix describing no transformation)*
- **MetaEngineRotation** (int m11, int m12, int m21, int m22)
- **MetaEngineRotation** ([MetaEngine::ImageOrientation](#) exifOrientation)  
*Returns the matrix corresponding to the given TransformationAction.*
- **MetaEngineRotation** ([TransformationAction](#) action)  
*Returns the matrix corresponding to the given TransformationAction.*
- [MetaEngine::ImageOrientation](#) exifOrientation () const  
*Returns the Exif orientation flag describing this matrix.*
- bool **isNoTransform** () const  
*Returns true if this matrix describes no transformation (is the identity matrix)*
- bool **operator!=** (const [MetaEngineRotation](#) &ma) const
- [MetaEngineRotation](#) & **operator\*=** (const [MetaEngineRotation](#) &ma)
- [MetaEngineRotation](#) & **operator\*=** (const [QList](#)< [TransformationAction](#) > &actions)  
*Applies the given transform actions to this matrix.*
- [MetaEngineRotation](#) & **operator\*=** ([MetaEngine::ImageOrientation](#) exifOrientation)  
*Applies the given Exif orientation flag to this matrix.*
- [MetaEngineRotation](#) & **operator\*=** ([TransformationAction](#) action)  
*Applies the given transform to this matrix.*
- bool **operator==** (const [MetaEngineRotation](#) &ma) const
- [QTransform](#) **toTransform** () const  
*Returns a QTransform representing this matrix.*
- [QList](#)< [TransformationAction](#) > **transformations** () const  
*Returns the actions described by this matrix.*

### Static Public Member Functions

- static [QTransform](#) **toTransform** ([MetaEngine::ImageOrientation](#) orientation)  
*Returns a QTransform for the given Exif orientation.*

### Protected Member Functions

- void **set** (int m11, int m12, int m21, int m22)

### Protected Attributes

- int **m** [2][2]

## 9.982.1 Member Enumeration Documentation

### 9.982.1.1 TransformationAction

enum `Digikam::MetaEngineRotation::TransformationAction`

Note some of the defined Exif rotation flags combine two of these actions. The enum values correspond to those defined as JXFORM\_CODE in the often used the JPEG tool transupp.h.

## Enumerator

NoTransformation	no transformation
FlipHorizontal	horizontal flip
FlipVertical	vertical flip
Rotate90	90-degree clockwise rotation
Rotate180	180-degree rotation
Rotate270	270-degree clockwise (or 90 ccw)

## 9.982.2 Member Function Documentation

### 9.982.2.1 exifOrientation()

```
MetaEngine::ImageOrientation Digikam::MetaEngineRotation::exifOrientation ( ) const
```

Returns ORIENTATION\_UNSPECIFIED if no flag matches this matrix.

### 9.982.2.2 transformations()

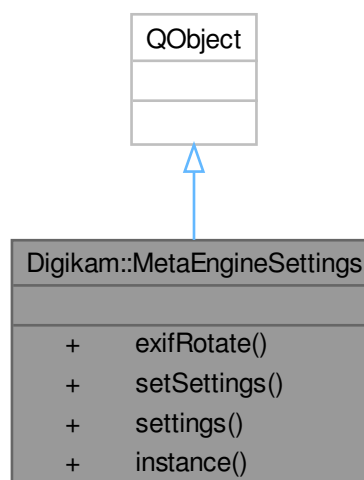
```
QList< MetaEngineRotation::TransformationAction > Digikam::MetaEngineRotation::transformations ( ) const
```

Converts the mathematically correct description into the primitive operations that can be carried out losslessly.

The order matters. Not all possible matrices are supported, but all those that can be combined by Exif rotation flags and the transform actions above. If `isNoTransform()` or the matrix is not supported returns an empty list.

## 9.983 Digikam::MetaEngineSettings Class Reference

Inheritance diagram for Digikam::MetaEngineSettings:





## Signals

- void **signalMetaEngineSettingsChanged** (const [MetaEngineSettingsContainer](#) &current, const [MetaEngineSettingsContainer](#) &previous)
- void **signalSettingsChanged** ()

## Public Member Functions

- bool **exifRotate** () const  
*Shortcut to get exif rotation settings from container.*
- void **setSettings** (const [MetaEngineSettingsContainer](#) &settings)  
*Sets the current Metadata settings and writes them to config.*
- [MetaEngineSettingsContainer](#) **settings** () const  
*Returns the current Metadata settings.*

## Static Public Member Functions

- static [MetaEngineSettings](#) \* **instance** ()  
*Global container for Metadata settings.*

## Friends

- class **MetaEngineSettingsCreator**

## 9.983.1 Member Function Documentation

### 9.983.1.1 instance()

```
MetaEngineSettings * Digikam::MetaEngineSettings::instance ( ) [static]
```

All accessor methods are thread-safe.

## 9.984 Digikam::MetaEngineSettingsContainer Class Reference

The class [MetaEngineSettingsContainer](#) encapsulates all metadata related settings.

## Public Types

- enum **AlbumDateSource** {  
  **NewestItemDate** = 0 , **OldestItemDate** , **AverageDate** , **FolderDate** ,  
  **IgnoreDate** }
- enum [RotationBehaviorFlag](#) {  
  **NoRotation** = 0 , **RotateByInternalFlag** = 1 << 0 , **RotateByMetadataFlag** = 1 << 1 , **RotateBy↔**  
  **LosslessRotation** = 1 << 2 ,  
  **RotateByLossyRotation** = 1 << 3 , **RotatingFlags** = RotateByInternalFlag | RotateByMetadataFlag ,  
  **RotatingPixels** = RotateByLosslessRotation | RotateByLossyRotation }
- typedef QFlags< [RotationBehaviorFlag](#) > **RotationBehaviorFlags**  
*Describes the allowed and desired operation when rotating a picture.*

## Public Member Functions

- QStringList **defaultExifToolSearchPaths** () const
- void **readFromConfig** (const KConfigGroup &group)
- void **writeToConfig** (KConfigGroup &group) const

## Public Attributes

- AlbumDataSource **albumDateFrom** = OldestItemDate
- bool **exifRotate** = true
- bool **exifSetOrientation** = true
- QString **exifToolPath**
- [MetaEngine::MetadataWritingMode](#) **metadataWritingMode** = [MetaEngine::WRITE\\_TO\\_FILE\\_ONLY](#)
- bool **readWithExifTool** = false
- bool **rescanImagelfModified** = false
- RotationBehaviorFlags **rotationBehavior** = RotationBehaviorFlags(RotatingFlags | RotateByLossless↔ Rotation)
- bool **saveColorLabel** = false
- bool **saveComments** = false
- bool **saveDateTime** = false
- bool **saveFaceTags** = false
- bool **savePickLabel** = false
- bool **savePosition** = false
- bool **saveRating** = false
- bool **saveTags** = false
- bool **saveTemplate** = false
- QStringList **sidecarExtensions**
- bool **updateFileTimeStamp** = true
- bool **useCompatibleFileName** = false
- bool **useFastScan** = false
- bool **useLazySync** = false
- bool **useXMPSidecar4Reading** = false
- bool **writeDngFiles** = false
- bool **writeRawFiles** = false
- bool **writeWithExifTool** = false

### 9.984.1 Detailed Description

#### Note

this allows supply changed arguments to [MetadataHub](#) without changing the global settings.

### 9.984.2 Member Enumeration Documentation

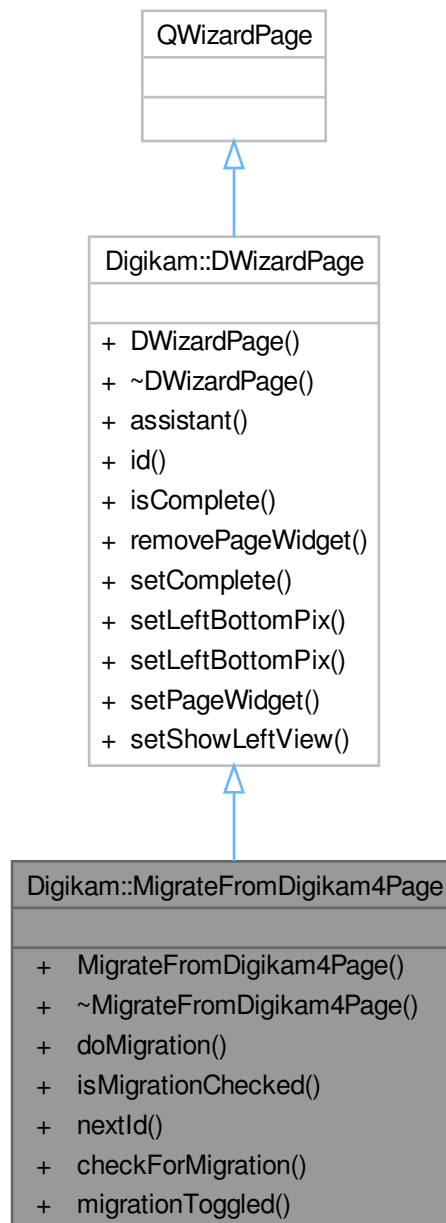
#### 9.984.2.1 RotationBehaviorFlag

```
enum Digikam::MetaEngineSettingsContainer::RotationBehaviorFlag
```

The modes are in escalating order and describe if an operation is allowed. What is actually done will be governed by what is possible: 1) RAW files cannot be rotated by content, setting the metadata may be problematic 2) Read-Only files cannot be edited, neither content nor metadata 3) Writable files will have lossy compression 4) Only JPEG and PGF offer lossless rotation Using a contents-based rotation always implies resetting the flag.

## 9.985 Digikam::MigrateFromDigikam4Page Class Reference

Inheritance diagram for Digikam::MigrateFromDigikam4Page:



### Public Slots

- void **migrationToggled** (bool b)

### Public Member Functions

- **MigrateFromDigikam4Page** (QWizard \*const dlg)
- void **doMigration** ()
- bool **isMigrationChecked** () const  
*Returns true if the user selected to do a migration.*
- int **nextId** () const override

### Public Member Functions inherited from [Digikam::DWizardPage](#)

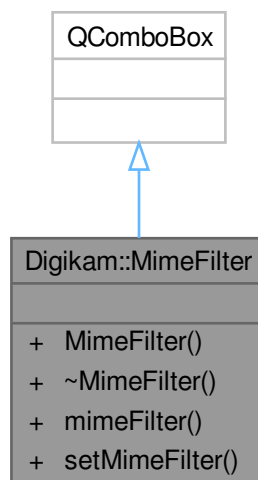
- **DWizardPage** (QWizard \*const dlg, const QString &title)
- QWizard \* **assistant** () const
- int **id** () const
- bool **isComplete** () const override
- void **removePageWidget** (QWidget \*const w)
- void **setComplete** (bool b)
- void **setLeftBottomPix** (const QIcon &icon)
- void **setLeftBottomPix** (const QPixmap &pix)
- void **setPageWidget** (QWidget \*const w)
- void **setShowLeftView** (bool v)

### Static Public Member Functions

- static bool **checkForMigration** ()  
*Return true if migration data are available on the system.*

## 9.986 Digikam::MimeFilter Class Reference

Inheritance diagram for Digikam::MimeFilter:



## Public Types

- enum [TypeMimeFilter](#) {  
**AllFiles** = 0 , **ImageFiles** , **NoRAWFiles** , **JPGFiles** ,  
**JPEG2000Files** , **JPEGXLFiles** , **WEBPFiles** , **PNGFiles** ,  
**TIFFFiles** , **PGFFiles** , **HEIFFiles** , **AVIFFiles** ,  
**DNGFiles** , **RAWFiles** , **MoviesFiles** , **AudioFiles** ,  
[RasterGraphics](#) }

## Public Member Functions

- **MimeFilter** (QWidget \*const parent)
- int **mimeFilter** ()
- void **setMimeFilter** (int filter)

## 9.986.1 Member Enumeration Documentation

### 9.986.1.1 TypeMimeFilter

enum [Digikam::MimeFilter::TypeMimeFilter](#)

#### Enumerator

HEIFFiles	HEVC H265 compression based containers.
RAWFiles	All Raw file formats such as nef, cr2, arw, pef, etc..
RasterGraphics	PSD, XCF, etc...

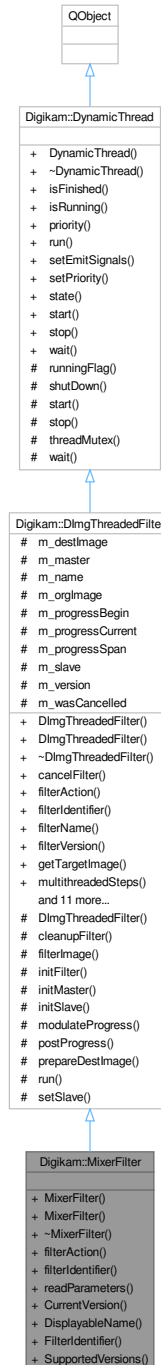
## 9.987 Digikam::MixerContainer Class Reference

### Public Attributes

- double **blackBlueGain** = 0.0
- double **blackGreenGain** = 0.0
- double **blackRedGain** = 1.0
- double **blueBlueGain** = 1.0
- double **blueGreenGain** = 0.0
- double **blueRedGain** = 0.0
- bool **bMonochrome** = false
- bool **bPreserveLum** = true
- double **greenBlueGain** = 0.0
- double **greenGreenGain** = 1.0
- double **greenRedGain** = 0.0
- double **redBlueGain** = 0.0
- double **redGreenGain** = 0.0
- double **redRedGain** = 1.0

## 9.988 Digikam::MixerFilter Class Reference

Inheritance diagram for Digikam::MixerFilter:



### Public Member Functions

- **MixerFilter** ([DImg](#) \*const orgImage, [QObject](#) \*const parent=nullptr, const [MixerContainer](#) &settings=[MixerContainer](#)())
- **MixerFilter** ([QObject](#) \*const parent=nullptr)

- [FilterAction filterAction](#) () override  
*Returns the action description corresponding to currently set options.*
- [QString filterIdentifier](#) () const override  
*Return the identifier for this filter in the image history.*
- void [readParameters](#) (const [FilterAction](#) &action) override

## Public Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImg](#) \*const orgImage, [QObject](#) \*const parent, const [QString](#) &name=[QString](#)())  
*Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter](#) ([QObject](#) \*const parent=nullptr, const [QString](#) &name=[QString](#)())  
*Constructs a filter without argument.*
- virtual void [cancelFilter](#) ()  
*Cancel the threaded computation.*
- const [QString](#) & [filterName](#) ()
- int [filterVersion](#) () const
- [DImg](#) [getTargetImage](#) ()
- [QList](#)< int > [multithreadedSteps](#) (int stop, int start=0) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool [parametersSuccessfullyRead](#) () const  
*Optional: error handling for readParameters.*
- virtual [QString](#) [readParametersError](#) (const [FilterAction](#) &actionThatFailed) const
- void [setFilterName](#) (const [QString](#) &name)
- void [setFilterVersion](#) (int version)  
*Replaying a filter action: Set the filter version.*
- void [setOriginalImage](#) (const [DImg](#) &orgImage)
- void [setupAndStartDirectly](#) (const [DImg](#) &orgImage, [DImgThreadedFilter](#) \*const master, int progress←Begin=0, int progressEnd=100)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void [setupFilter](#) (const [DImg](#) &orgImage)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void [startFilter](#) ()  
*Start the threaded computation.*
- virtual void [startFilterDirectly](#) ()  
*Start computation of this filter, directly in this thread.*
- virtual [QList](#)< int > [supportedVersions](#) () const

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread](#) ([QObject](#) \*const parent=nullptr)  
*This class extends [QRunnable](#), so you have to reimplement virtual void [run\(\)](#).*
- [~DynamicThread](#) () override  
*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool [isFinished](#) () const
- bool [isRunning](#) () const
- [QThread::Priority](#) [priority](#) () const
- void [setEmitSignals](#) (bool emitThem)
- void [setPriority](#) ([QThread::Priority](#) priority)  
*Sets the priority for this dynamic thread.*
- State [state](#) () const

### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*



## Protected Member Functions inherited from Digikam::DImgThreadedFilter

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from Digikam::DynamicThread

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from Digikam::DImgThreadedFilter

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.988.1 Member Function Documentation

### 9.988.1.1 filterAction()

`FilterAction` Digikam::MixerFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.988.1.2 filterIdentifier()

`QString` Digikam::MixerFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

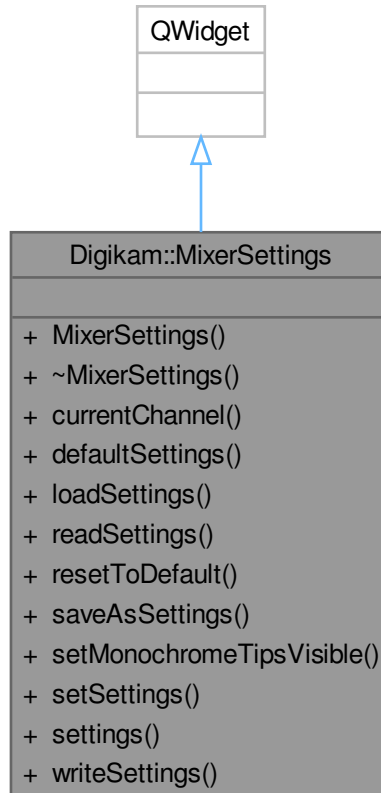
### 9.988.1.3 readParameters()

`void` Digikam::MixerFilter::readParameters (   
           const `FilterAction` & action ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

## 9.989 Digikam::MixerSettings Class Reference

Inheritance diagram for Digikam::MixerSettings:



## Signals

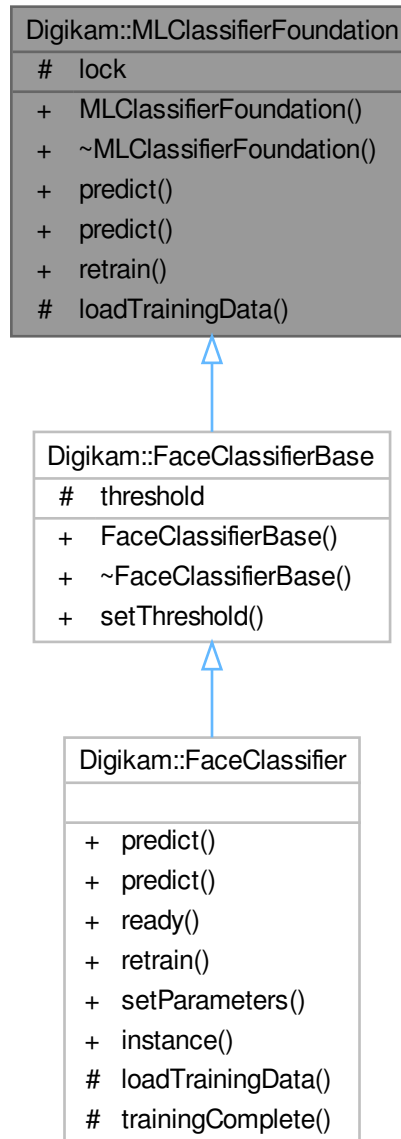
- void **signalMonochromeActivated** (bool)
- void **signalOutChannelChanged** ()
- void **signalSettingsChanged** ()

## Public Member Functions

- **MixerSettings** (QWidget \*const parent)
- int **currentChannel** () const
- [MixerContainer](#) **defaultSettings** () const
- void **loadSettings** ()
- void **readSettings** (const KConfigGroup &group)
- void **resetToDefault** ()
- void **saveAsSettings** ()
- void **setMonochromeTipsVisible** (bool b)
- void **setSettings** (const [MixerContainer](#) &settings)
- [MixerContainer](#) **settings** () const
- void **writeSettings** (KConfigGroup &group)

## 9.990 Digikam::MLClassifierFoundation Class Reference

Inheritance diagram for Digikam::MLClassifierFoundation:



### Classes

- class [VotingGroups](#)

### Public Member Functions

- virtual int **predict** (const cv::Mat &target) const =0
- virtual int **predict** (const cv::UMat &target) const =0
- virtual bool **retrain** ()=0

### Protected Member Functions

- virtual bool **loadTrainingData** ()=0

### Protected Attributes

- QReadWriteLock **lock**

## 9.991 Digikam::MLClassifierFoundation::VotingGroups Class Reference

### Classes

- struct [VoteTally](#)

### Public Types

- enum **WinnerType** { **VotesLowScore** , **VotesHighScore** , **LowScore** , **HighScore** }

### Public Member Functions

- void **addVote** (int label, float score)
- int **winner** (WinnerType winnerType)

## 9.992 Digikam::MLClassifierFoundation::VotingGroups::VoteTally Struct Reference

### Public Attributes

- int **label** = 0
- float **score** = 0.0F
- int **votes** = 0

## 9.993 Digikam::MLPipelineFoundation Class Reference

Inheritance diagram for Digikam::MLPipelineFoundation:



### Classes

- struct [\\_MLPipelinePerformanceProfile](#)

### Public Types

- enum [MLPipelineNotification](#) { [notifySkipped](#) , [notifyProcessed](#) }
- typedef struct [Digikam::MLPipelineFoundation::\\_MLPipelinePerformanceProfile](#) [MLPipelinePerformanceProfile](#)
- typedef [SharedQueue](#)< [MLPipelinePackageFoundation](#) \* > [MLPipelineQueue](#)
- enum [MLPipelineStage](#) { [Finder](#) , [Loader](#) , [Extractor](#) , [Classifier](#) , [Trainer](#) , [Writer](#) , [None](#) }

## Signals

- void **finished** ()  
*Emitted when the last package has finished processing.*
- void **processed** (const MLPipelinePackageNotify::Ptr &package)  
*Emitted when one package has finished processing.*
- void **processing** (const MLPipelinePackageNotify::Ptr &package)  
*Emitted when one package begins processing.*
- void **progressValueChanged** (float progress)
- void **scheduled** ()  
*Emitted when processing is scheduled.*
- void **signalAddMoreWorkers** ()
- void **signalUpdateItemCount** (const qlonglong itemCount)
- void **skipped** (const MLPipelinePackageNotify::Ptr &package)  
*Emitted when one or several packages were skipped, usually because they have already been scanned.*
- void **started** (const QString &message)  
*Emitted when processing has started.*

## Public Member Functions

- virtual void **cancel** ()
- bool **hasFinished** () const
- virtual bool **start** ()

## Protected Member Functions

- virtual void **addMoreWorkers** ()=0
- bool **addWorker** (const MLPipelineStage &stage)
- bool **checkMoreWorkers** (int totalItemCount, int currentItemCount, bool useFullCpu)
- virtual bool **classifier** ()=0
- void **clearAllQueues** ()
- void **clearQueue** (MLPipelineQueue \*thisQueue)
- virtual MLPipelinePackageFoundation \* **dequeue** (MLPipelineQueue \*thisQueue)
- virtual bool **enqueue** (MLPipelineQueue \*thisQueue, MLPipelinePackageFoundation \*package)
- virtual bool **extractor** ()=0
- virtual bool **finder** ()=0
- virtual bool **loader** ()=0
- void **notify** (MLPipelineNotification notification, const QString &\_name, const QString &\_path, int \_processed, const DImg &\_thumbnail)
- void **notify** (MLPipelineNotification notification, const QString &\_name, const QString &\_path, int \_processed, const QIcon &\_thumbnail)
- void **notify** (MLPipelineNotification notification, const QString &\_name, const QString &\_path, int \_processed, const QImage &\_thumbnail)
- void **pipelinePerformanceEnd** (const MLPipelineStage &stage, int totalItemCount, QElapsedTimer &timer)
- void **pipelinePerformanceEnd** (const MLPipelineStage &stage, QElapsedTimer &timer)
- void **pipelinePerformanceStart** (const MLPipelineStage &stage, QElapsedTimer &timer)
- MLPipelinePackageFoundation \* **queueEndSignal** () const
- void **showPipelinePerformance** () const
- void **stageEnd** (MLPipelineStage thisStage, MLPipelineStage nextStage)
- void **stageStart** (QThread::Priority threadPriority, MLPipelineStage thisStage, MLPipelineStage nextStage, MLPipelineQueue \*&thisQueue, MLPipelineQueue \*&nextQueue)
- virtual bool **trainer** ()=0
- void **waitForStart** ()
- virtual bool **writer** ()=0

## Protected Attributes

- bool **cancelled** = false
- QAtomicInteger< int > **itemsProcessed** = 0
- quint64 **maxBufferSize** = 2147483648  
*2 GB default*
- QMutex **mutex**
- QMap< MLPipelineStage, MLPipelinePerformanceProfile > **performanceProfileList**
- QMap< MLPipelineStage, MLPipelineQueue \* > **queues**
- QThreadPool \* **threadPool** = nullptr
- QMutex **threadStageMutex**
- QAtomicInteger< int > **totalItemCount** = 0
- quint64 **usedBufferSize** = 0
- QList< QFutureWatcher< bool > \* > **watchList**

## 9.993.1 Member Enumeration Documentation

### 9.993.1.1 MLPipelineStage

enum Digikam::MLPipelineFoundation::MLPipelineStage

#### Enumerator

Finder	Finder stage finds the data for the pipeline.
Loader	Loader stage loads and prepares the data for extraction.
Extractor	Extractor stage pulls the features from the data.
Classifier	Classifier stage adds a label (face, autotag, etc) to an extracted object.
Trainer	Classifier stage adds a label (face, autotag, etc) to an extracted object.
Writer	Writer stage saves the data to the DB.
None	Empty stage.

## 9.993.2 Member Function Documentation

### 9.993.2.1 cancel()

```
void Digikam::MLPipelineFoundation::cancel ( ) [virtual]
```

worker threads can be in 1 of 3 states when cancel is called

1. waiting for a new package
2. processing a package
3. waiting to push a package

handle all 3 cases so the worker thread sees the cancel signal



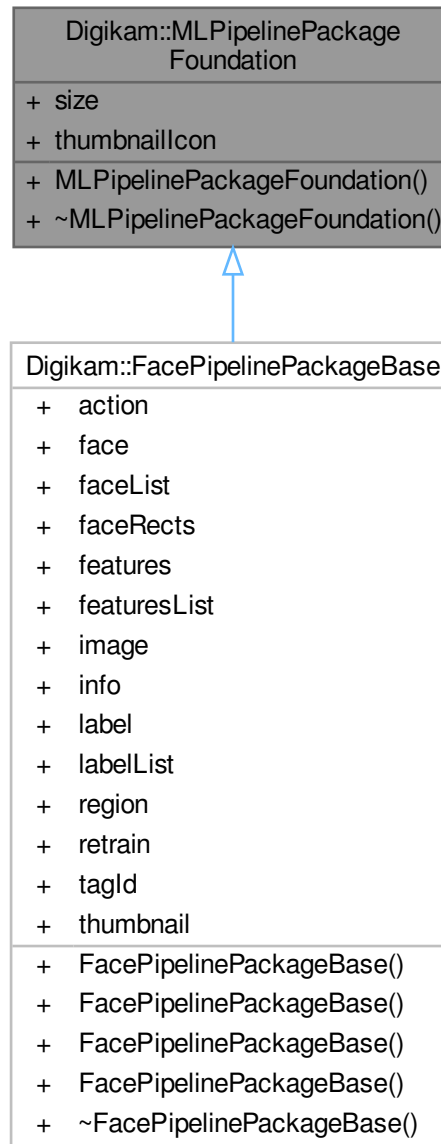
## 9.994 Digikam::MLPipelineFoundation::\_MLPipelinePerformanceProfile Struct Reference

### Public Attributes

- QAtomicInteger< int > **currentThreadCount**
- int **elapsedTime** = 0
- int **itemCount** = 0
- int **maxElapsedTime** = 0
- int **maxQueueCount** = 0
- QAtomicInteger< int > **maxThreadCount**

## 9.995 Digikam::MLPipelinePackageFoundation Class Reference

Inheritance diagram for Digikam::MLPipelinePackageFoundation:

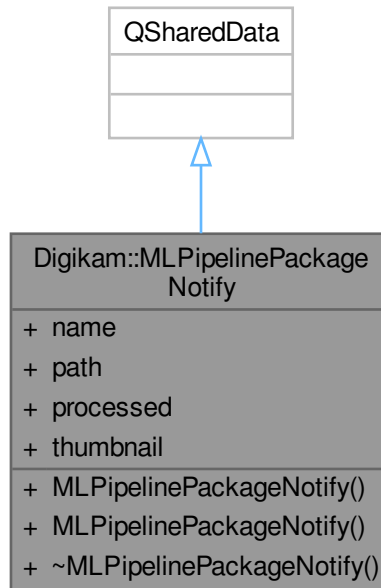


### Public Attributes

- quint64 **size** = 0
- QIcon **thumbnailIcon**

## 9.996 Digikam::MLPipelinePackageNotify Class Reference

Inheritance diagram for Digikam::MLPipelinePackageNotify:



### Public Types

- typedef `QExplicitlySharedDataPointer< MLPipelinePackageNotify >` **Ptr**

### Public Member Functions

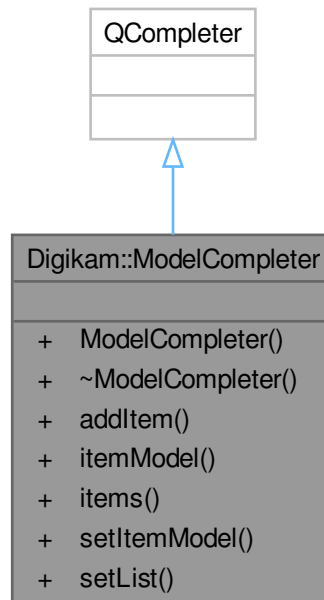
- **MLPipelinePackageNotify** (const QString &\_name, const QString &\_path, int \_processed, const [DImg](#) &↔\_thumbnail)
- **MLPipelinePackageNotify** (const QString &\_name, const QString &\_path, int \_processed, const [QIcon](#) &↔\_thumbnail)

### Public Attributes

- const QString **name**
- const QString **path**
- int **processed** = 0
- [QIcon](#) **thumbnail**

## 9.997 Digikam::ModelCompleter Class Reference

Inheritance diagram for Digikam::ModelCompleter:



### Signals

- void **signalActivated** ()
- void **signalHighlighted** (int albumId)

### Public Member Functions

- **ModelCompleter** (QObject \*const parent=nullptr)
- void **addItem** (const QString &item)
- QAbstractItemModel \* **itemModel** () const
- QStringList **items** () const
- void **setItemModel** (QAbstractItemModel \*const model, int uniqueIdRole, int displayRole=Qt::DisplayRole)  
*If the given model is != null, the model is used to populate the completion for this text field.*
- void **setList** (const QStringList &list)

### 9.997.1 Member Function Documentation

#### 9.997.1.1 setItemModel()

```

void Digikam::ModelCompleter::setItemModel (
    QAbstractItemModel *const model,
    int uniqueIdRole,
    int displayRole = Qt::DisplayRole )
  
```

## Parameters

<i>model</i>	to fill from or null for manual mode
<i>uniqueIdRole</i>	a role for which the model will return a unique integer for each entry
<i>displayRole</i>	the role to retrieve the text for completion, default is Qt::DisplayRole.

## 9.998 Digikam::ModelIndexBasedComboBox Class Reference

Inheritance diagram for Digikam::ModelIndexBasedComboBox:



### Public Member Functions

- [ModelIndexBasedComboBox](#) (QWidget \*const parent=nullptr)  
*QComboBox has a current index based on a single integer.*

- QModelIndex **currentIndex** () const
- void **hidePopup** () override
- void **setCurrentIndex** (const QModelIndex &index)
- void **showPopup** () override

### Protected Attributes

- QPersistentModelIndex **m\_currentIndex**

## 9.998.1 Constructor & Destructor Documentation

### 9.998.1.1 QModelIndexBasedComboBox()

```
Digikam::ModelIndexBasedComboBox::ModelIndexBasedComboBox (  
    QWidget *const parent = nullptr ) [explicit]
```

This is not sufficient for more complex models. This class is a combo box that stores a current index based on QModelIndex.

## 9.999 Digikam::ModelMenu Class Reference

A QMenu that is dynamically populated from a QAbstractItemModel.

Inheritance diagram for Digikam::ModelMenu:



### Signals

- void **activated** (const QModelIndex &index)
- void **hovered** (const QString &text)

### Public Member Functions

- **ModelMenu** (QWidget \*const parent=nullptr)



- int **firstSeparator** () const
- int **hoverRole** () const
- QAction \* **makeAction** (const QIcon &icon, const QString &text, QObject \*const parent)
- int **maxRows** () const
- QAbstractItemModel \* **model** () const
- QModelIndex **rootIndex** () const
- int **separatorRole** () const
- void **setFirstSeparator** (int offset)
- void **setHoverRole** (int role)
- void **setMaxRows** (int max)
- void **setModel** (QAbstractItemModel \*model)
- void **setRootIndex** (const QModelIndex &index)
- void **setSeparatorRole** (int role)

### Protected Member Functions

- void **createMenu** (const QModelIndex &parent, int max, QMenu \*parentMenu=nullptr, QMenu \*menu=nullptr)
  - put all of the children of parent into menu up to max*
- virtual void **postPopulated** ()
  - add any actions after the tree*
- virtual bool **prePopulated** ()
  - add any actions before the tree, return true if any actions are added.*

## 9.999.1 Member Function Documentation

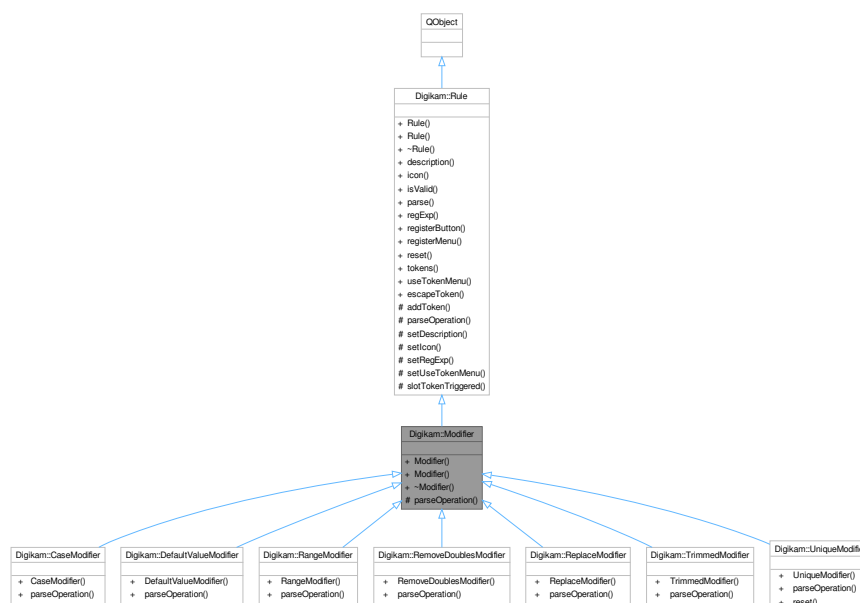
### 9.999.1.1 prePopulated()

```
bool Digikam::ModelMenu::prePopulated ( ) [protected], [virtual]
```

Reimplemented in [Digikam::BookmarksMenu](#).

## 9.1000 Digikam::Modifier Class Reference

Inheritance diagram for Digikam::Modifier:



## Public Member Functions

- **Modifier** (const QString &name, const QString &description)
- **Modifier** (const QString &name, const QString &description, const QString &icon)

## Public Member Functions inherited from [Digikam::Rule](#)

- **Rule** (const QString &name)
- **Rule** (const QString &name, const QString &icon)
- QString **description** () const
- QPixmap **icon** (Rule::IconType type=Rule::Action) const
- bool **isValid** () const

*Checks the validity of the parse object.*

- **ParseResults parse** ([ParseSettings](#) &settings)
- QRegularExpression & **regExp** () const
- TODO: This is probably not needed anymore.*
- QPushButton \* **registerButton** (QWidget \*parent)
- Register a button in the parent object.*
- QAction \* **registerMenu** (QMenu \*parent)
- Register a menu action in the parent object.*
- virtual void **reset** ()
- Resets the parser to its initial state.*
- TokenList & **tokens** () const
- bool **useTokenMenu** () const
- Returns true if a token menu is used.*

## Protected Member Functions

- QString **parseOperation** ([ParseSettings](#) &settings, const QRegularExpressionMatch &match) override=0
- TODO: describe me.*

## Protected Member Functions inherited from [Digikam::Rule](#)

- bool **addToken** (const QString &id, const QString &description, const QString &actionName=QString())
- add a token to the parser, every parser should at least assign one token object*
- void **setDescription** (const QString &desc)
- void **setIcon** (const QString &pixmap)
- void **setRegExp** (const QRegularExpression &regExp)
- void **setUseTokenMenu** (bool value)

*If multiple tokens have been assigned to a rule, a menu will be created.*

## Additional Inherited Members

## Public Types inherited from [Digikam::Rule](#)

- enum **IconType** { **Action** = 0 , **Dialog** }

## Signals inherited from [Digikam::Rule](#)

- void **signalTokenTriggered** (const QString &)

## Static Public Member Functions inherited from [Digikam::Rule](#)

- static QString **escapeToken** (const QString &token)  
*Escape the token characters to make them work in regular expressions.*

## Protected Slots inherited from [Digikam::Rule](#)

- virtual void **slotTokenTriggered** (const QString &)

## 9.1000.1 Member Function Documentation

### 9.1000.1.1 parseOperation()

```
QString Digikam::Modifier::parseOperation (
    ParseSettings & settings,
    const QRegularExpressionMatch & match ) [override], [protected], [pure virtual]
```

#### Parameters

<i>settings</i>	contains settings
<i>match</i>	result of the regular expression match done in <code>Option::parse()</code>

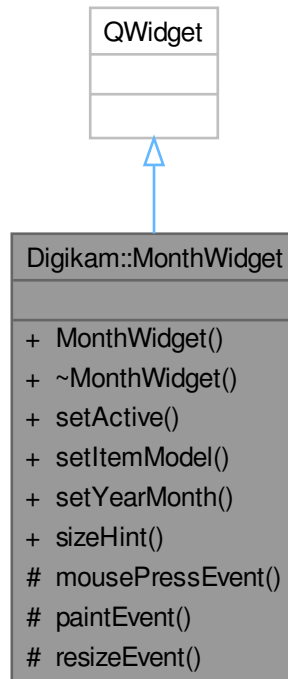
#### Returns

Implements [Digikam::Rule](#).

Implemented in [Digikam::CaseModifier](#), [Digikam::DefaultValueModifier](#), [Digikam::RangeModifier](#), [Digikam::RemoveDoublesModifier](#), [Digikam::ReplaceModifier](#), [Digikam::TrimmedModifier](#), and [Digikam::UniqueModifier](#).

## 9.1001 Digikam::MonthWidget Class Reference

Inheritance diagram for Digikam::MonthWidget:



### Public Member Functions

- **MonthWidget** (`QWidget *const parent`)
- void **setActive** (`bool val`)
- void **setItemModel** (`ItemFilterModel *const model`)
- void **setYearMonth** (`int year, int month`)
- `QSize` **sizeHint** () const override

### Protected Member Functions

- void **mousePressEvent** (`QMouseEvent *e`) override
- void **paintEvent** (`QPaintEvent *`) override
- void **resizeEvent** (`QResizeEvent *e`) override

## 9.1002 Digikam::MysqlAdminBinary Class Reference

Inheritance diagram for Digikam::MysqlAdminBinary:



### Additional Inherited Members

### Public Slots inherited from [Digikam::DBinaryIface](#)

- virtual void **slotAddPossibleSearchDirectory** (const QString &dir)

- virtual void **slotAddSearchDirectory** (const QString &dir)
- virtual void **slotNavigateAndCheck** ()

### Signals inherited from [Digikam::DBinaryIface](#)

- void **signalBinaryValid** ()
- void **signalSearchDirectoryAdded** (const QString &dir)

### Public Member Functions inherited from [Digikam::DBinaryIface](#)

- **DBinaryIface** (const QString &binaryName, const QString &minimalVersion, const QString &header, const int headerLine, const QString &projectName, const QString &url, const QString &pluginName, const QStringList &args=QStringList(), const QString &desc=QString())
- **DBinaryIface** (const QString &binaryName, const QString &projectName, const QString &url, const QString &pluginName, const QStringList &args=QStringList(), const QString &desc=QString())
- virtual QString **baseName** () const
- virtual bool **checkDir** ()
- virtual bool **checkDirForPath** (const QString &path)
- const QString & **description** () const
- bool **developmentVersion** () const
- virtual QString **directory** () const
- bool **hasError** () const
- bool **isFound** () const
- bool **isValid** () const
- virtual QString **minimalVersion** () const
- virtual QString **path** () const
- virtual QString **path** (const QString &dir) const
- virtual QString **projectName** () const
- virtual bool **recheckDirectories** ()
- virtual void **setup** (const QString &prev=QString())
- virtual QUrl **url** () const
- const QString & **version** () const
- bool **versionsRight** () const
- bool **versionsRight** (const float) const

### Static Public Member Functions inherited from [Digikam::DBinaryIface](#)

- static QString **goodBaseName** (const QString &b)

### Protected Member Functions inherited from [Digikam::DBinaryIface](#)

- QString **findHeader** (const QStringList &output, const QString &header) const
- virtual bool **parseHeader** (const QString &output)
- virtual QString **readConfig** ()
- void **setVersion** (QString &version)
- virtual void **writeConfig** ()

**Protected Attributes inherited from [Digikam::DBinaryIface](#)**

- const QStringList **m\_binaryArguments**
- const QString **m\_binaryBaseName**
- QLabel \* **m\_binaryLabel** = nullptr
- const bool **m\_checkVersion**
- const QString **m\_configGroup**
- QString **m\_description**
- bool **m\_developmentVersion** = false
- QLabel \* **m\_downloadButton** = nullptr
- bool **m\_hasError** = false
- const int **m\_headerLine**
- const QString **m\_headerStarts**
- bool **m\_isFound** = false
- QLineEdit \* **m\_lineEdit** = nullptr
- const QString **m\_minimalVersion**
- QPushButton \* **m\_pathButton** = nullptr
- QString **m\_pathDir** = QLatin1String("")
- QFrame \* **m\_pathWidget** = nullptr
- const QString **m\_projectName**
- QSet< QString > **m\_searchPaths**
- QLabel \* **m\_statusIcon** = nullptr
- const QUrl **m\_url**
- QString **m\_version** = QLatin1String("")
- QLabel \* **m\_versionLabel** = nullptr

## 9.1003 Digikam::MysqlInitBinary Class Reference

Inheritance diagram for Digikam::MysqlInitBinary:



### Additional Inherited Members

### Public Slots inherited from [Digikam::DBinaryIface](#)

- virtual void **slotAddPossibleSearchDirectory** (const QString &dir)



- virtual void **slotAddSearchDirectory** (const QString &dir)
- virtual void **slotNavigateAndCheck** ()

### Signals inherited from [Digikam::DBinaryIface](#)

- void **signalBinaryValid** ()
- void **signalSearchDirectoryAdded** (const QString &dir)

### Public Member Functions inherited from [Digikam::DBinaryIface](#)

- **DBinaryIface** (const QString &binaryName, const QString &minimalVersion, const QString &header, const int headerLine, const QString &projectName, const QString &url, const QString &pluginName, const QStringList &args=QStringList(), const QString &desc=QString())
- **DBinaryIface** (const QString &binaryName, const QString &projectName, const QString &url, const QString &pluginName, const QStringList &args=QStringList(), const QString &desc=QString())
- virtual QString **baseName** () const
- virtual bool **checkDir** ()
- virtual bool **checkDirForPath** (const QString &path)
- const QString & **description** () const
- bool **developmentVersion** () const
- virtual QString **directory** () const
- bool **hasError** () const
- bool **isFound** () const
- bool **isValid** () const
- virtual QString **minimalVersion** () const
- virtual QString **path** () const
- virtual QString **path** (const QString &dir) const
- virtual QString **projectName** () const
- virtual bool **recheckDirectories** ()
- virtual void **setup** (const QString &prev=QString())
- virtual QUrl **url** () const
- const QString & **version** () const
- bool **versionsRight** () const
- bool **versionsRight** (const float) const

### Static Public Member Functions inherited from [Digikam::DBinaryIface](#)

- static QString **goodBaseName** (const QString &b)

### Protected Member Functions inherited from [Digikam::DBinaryIface](#)

- QString **findHeader** (const QStringList &output, const QString &header) const
- virtual bool **parseHeader** (const QString &output)
- virtual QString **readConfig** ()
- void **setVersion** (QString &version)
- virtual void **writeConfig** ()

## Protected Attributes inherited from [Digikam::DBinaryIface](#)

- const QStringList **m\_binaryArguments**
- const QString **m\_binaryBaseName**
- QLabel \* **m\_binaryLabel** = nullptr
- const bool **m\_checkVersion**
- const QString **m\_configGroup**
- QString **m\_description**
- bool **m\_developmentVersion** = false
- QLabel \* **m\_downloadButton** = nullptr
- bool **m\_hasError** = false
- const int **m\_headerLine**
- const QString **m\_headerStarts**
- bool **m\_isFound** = false
- QLineEdit \* **m\_lineEdit** = nullptr
- const QString **m\_minimalVersion**
- QPushButton \* **m\_pathButton** = nullptr
- QString **m\_pathDir** = QLatin1String("")
- QFrame \* **m\_pathWidget** = nullptr
- const QString **m\_projectName**
- QSet< QString > **m\_searchPaths**
- QLabel \* **m\_statusIcon** = nullptr
- const QUrl **m\_url**
- QString **m\_version** = QLatin1String("")
- QLabel \* **m\_versionLabel** = nullptr

## 9.1004 Digikam::MysqlServerBinary Class Reference

Inheritance diagram for Digikam::MysqlServerBinary:



### Additional Inherited Members

#### Public Slots inherited from [Digikam::DBinaryIface](#)

- virtual void **slotAddPossibleSearchDirectory** (const QString &dir)

- virtual void **slotAddSearchDirectory** (const QString &dir)
- virtual void **slotNavigateAndCheck** ()

### Signals inherited from [Digikam::DBinaryIface](#)

- void **signalBinaryValid** ()
- void **signalSearchDirectoryAdded** (const QString &dir)

### Public Member Functions inherited from [Digikam::DBinaryIface](#)

- **DBinaryIface** (const QString &binaryName, const QString &minimalVersion, const QString &header, const int headerLine, const QString &projectName, const QString &url, const QString &pluginName, const QStringList &args=QStringList(), const QString &desc=QString())
- **DBinaryIface** (const QString &binaryName, const QString &projectName, const QString &url, const QString &pluginName, const QStringList &args=QStringList(), const QString &desc=QString())
- virtual QString **baseName** () const
- virtual bool **checkDir** ()
- virtual bool **checkDirForPath** (const QString &path)
- const QString & **description** () const
- bool **developmentVersion** () const
- virtual QString **directory** () const
- bool **hasError** () const
- bool **isFound** () const
- bool **isValid** () const
- virtual QString **minimalVersion** () const
- virtual QString **path** () const
- virtual QString **path** (const QString &dir) const
- virtual QString **projectName** () const
- virtual bool **recheckDirectories** ()
- virtual void **setup** (const QString &prev=QString())
- virtual QUrl **url** () const
- const QString & **version** () const
- bool **versionsRight** () const
- bool **versionsRight** (const float) const

### Static Public Member Functions inherited from [Digikam::DBinaryIface](#)

- static QString **goodBaseName** (const QString &b)

### Protected Member Functions inherited from [Digikam::DBinaryIface](#)

- QString **findHeader** (const QStringList &output, const QString &header) const
- virtual bool **parseHeader** (const QString &output)
- virtual QString **readConfig** ()
- void **setVersion** (QString &version)
- virtual void **writeConfig** ()

## Protected Attributes inherited from [Digikam::DBinaryIface](#)

- const QStringList **m\_binaryArguments**
- const QString **m\_binaryBaseName**
- QLabel \* **m\_binaryLabel** = nullptr
- const bool **m\_checkVersion**
- const QString **m\_configGroup**
- QString **m\_description**
- bool **m\_developmentVersion** = false
- QLabel \* **m\_downloadButton** = nullptr
- bool **m\_hasError** = false
- const int **m\_headerLine**
- const QString **m\_headerStarts**
- bool **m\_isFound** = false
- QLineEdit \* **m\_lineEdit** = nullptr
- const QString **m\_minimalVersion**
- QPushButton \* **m\_pathButton** = nullptr
- QString **m\_pathDir** = QLatin1String("")
- QFrame \* **m\_pathWidget** = nullptr
- const QString **m\_projectName**
- QSet< QString > **m\_searchPaths**
- QLabel \* **m\_statusIcon** = nullptr
- const QUrl **m\_url**
- QString **m\_version** = QLatin1String("")
- QLabel \* **m\_versionLabel** = nullptr

## 9.1005 Digikam::MysqlUpgradeBinary Class Reference

Inheritance diagram for Digikam::MysqlUpgradeBinary:



### Additional Inherited Members

### Public Slots inherited from [Digikam::DBinaryIface](#)

- virtual void `slotAddPossibleSearchDirectory` (const QString &dir)

- virtual void **slotAddSearchDirectory** (const QString &dir)
- virtual void **slotNavigateAndCheck** ()

### Signals inherited from [Digikam::DBinaryIface](#)

- void **signalBinaryValid** ()
- void **signalSearchDirectoryAdded** (const QString &dir)

### Public Member Functions inherited from [Digikam::DBinaryIface](#)

- **DBinaryIface** (const QString &binaryName, const QString &minimalVersion, const QString &header, const int headerLine, const QString &projectName, const QString &url, const QString &pluginName, const QStringList &args=QStringList(), const QString &desc=QString())
- **DBinaryIface** (const QString &binaryName, const QString &projectName, const QString &url, const QString &pluginName, const QStringList &args=QStringList(), const QString &desc=QString())
- virtual QString **baseName** () const
- virtual bool **checkDir** ()
- virtual bool **checkDirForPath** (const QString &path)
- const QString & **description** () const
- bool **developmentVersion** () const
- virtual QString **directory** () const
- bool **hasError** () const
- bool **isFound** () const
- bool **isValid** () const
- virtual QString **minimalVersion** () const
- virtual QString **path** () const
- virtual QString **path** (const QString &dir) const
- virtual QString **projectName** () const
- virtual bool **recheckDirectories** ()
- virtual void **setup** (const QString &prev=QString())
- virtual QUrl **url** () const
- const QString & **version** () const
- bool **versionsRight** () const
- bool **versionsRight** (const float) const

### Static Public Member Functions inherited from [Digikam::DBinaryIface](#)

- static QString **goodBaseName** (const QString &b)

### Protected Member Functions inherited from [Digikam::DBinaryIface](#)

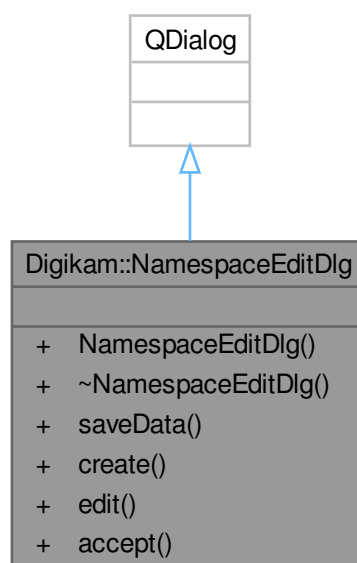
- QString **findHeader** (const QStringList &output, const QString &header) const
- virtual bool **parseHeader** (const QString &output)
- virtual QString **readConfig** ()
- void **setVersion** (QString &version)
- virtual void **writeConfig** ()

## Protected Attributes inherited from [Digikam::DBinaryIface](#)

- const QStringList **m\_binaryArguments**
- const QString **m\_binaryBaseName**
- QLabel \* **m\_binaryLabel** = nullptr
- const bool **m\_checkVersion**
- const QString **m\_configGroup**
- QString **m\_description**
- bool **m\_developmentVersion** = false
- QLabel \* **m\_downloadButton** = nullptr
- bool **m\_hasError** = false
- const int **m\_headerLine**
- const QString **m\_headerStarts**
- bool **m\_isFound** = false
- QLineEdit \* **m\_lineEdit** = nullptr
- const QString **m\_minimalVersion**
- QPushButton \* **m\_pathButton** = nullptr
- QString **m\_pathDir** = QLatin1String("")
- QFrame \* **m\_pathWidget** = nullptr
- const QString **m\_projectName**
- QSet< QString > **m\_searchPaths**
- QLabel \* **m\_statusIcon** = nullptr
- const QUrl **m\_url**
- QString **m\_version** = QLatin1String("")
- QLabel \* **m\_versionLabel** = nullptr

## 9.1006 Digikam::NamespaceEditDlg Class Reference

Inheritance diagram for Digikam::NamespaceEditDlg:





### Public Slots

- void **accept** () override

### Public Member Functions

- **NamespaceEditDlg** (bool create, [NamespaceEntry](#) &entry, QWidget \*const parent=nullptr)
- void **saveData** ([NamespaceEntry](#) &entry)

### Static Public Member Functions

- static bool **create** (QWidget \*const parent, [NamespaceEntry](#) &entry)
- static bool **edit** (QWidget \*const parent, [NamespaceEntry](#) &entry)

## 9.1007 Digikam::NamespaceEntry Class Reference

The [NamespaceEntry](#) class provide a simple container for dmetadata namespaces variables, such as names, what types of data expects and extra xml tags.

### Public Types

- enum **NamespaceType** {  
  **TAGS** = 0 , **TITLE** = 1 , **RATING** = 2 , **COMMENT** = 3 ,  
  **PICKLABEL** = 4 , **COLORLABEL** = 5 }
- enum **NsSubspace** { **EXIF** = 0 , **IPTC** = 1 , **XMP** = 2 }
- enum **SpecialOptions** {  
  **NO\_OPTS** = 0 , **COMMENT\_ALTLANG** = 1 , **COMMENT\_ATLLANGLIST** = 2 , **COMMENT\_XMP** = 3 ,  
  **COMMENT\_JPEG** = 4 , **TAG\_XMPBAG** = 5 , **TAG\_XMPSEQ** = 6 , **TAG\_ACDSEE** = 7 }
- enum **TagType** { **TAG** = 0 , **TAGPATH** = 1 }

### Public Member Functions

- **NamespaceEntry** (const [NamespaceEntry](#) &other)

### Static Public Member Functions

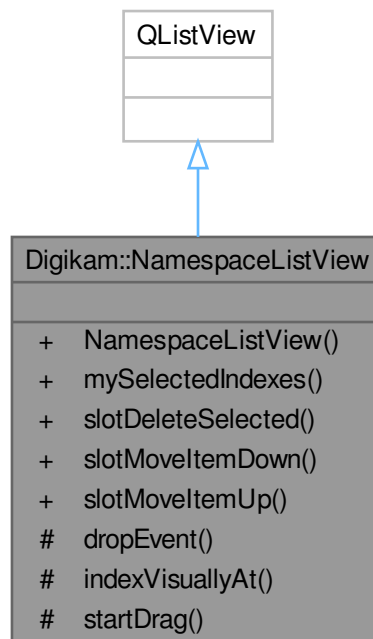
- static QString **DM\_COLORLABEL\_CONTAINER** ()
- static QString **DM\_COMMENT\_CONTAINER** ()
- static QString **DM\_PICKLABEL\_CONTAINER** ()
- static QString **DM\_RATING\_CONTAINER** ()
- static QString **DM\_TAG\_CONTAINER** ()
- static QString **DM\_TITLE\_CONTAINER** ()

**Public Attributes**

- QString **alternativeName**
- QList< int > **convertRatio**  
*Rating Options.*
- int **index** = -1
- bool **isDefault** = true
- bool **isDisabled** = false
- QString **namespaceName**  
*Tag Options.*
- NamespaceType **nsType** = TAGS
- SpecialOptions **secondNameOpts** = NO\_OPTS
- QString **separator**
- SpecialOptions **specialOpts** = NO\_OPTS
- NsSubspace **subspace** = XMP
- TagType **tagPaths** = TAGPATH

**9.1008 Digikam::NamespaceListView Class Reference**

Inheritance diagram for Digikam::NamespaceListView:

**Public Slots**

- void **slotDeleteSelected** ()  
*slotDeleteSelected - delete selected item from Quick Access List*
- void **slotMoveItemDown** ()
- void **slotMoveItemUp** ()

## Signals

- void **signalItemsChanged** ()  
*contextMenuEvent* - reimplemented method from *QListView* to handle custom context menu

## Public Member Functions

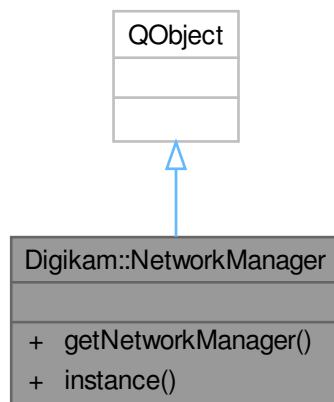
- **NamespaceListView** (QWidget \*const parent=nullptr)
- QModelIndexList **mySelectedIndexes** ()

## Protected Member Functions

- void **dropEvent** (QDropEvent \*e) override
- QModelIndex **indexVisuallyAt** (const QPoint &p)
- void **startDrag** (Qt::DropActions supportedActions) override  
*Reimplemented methods to enable custom drag-n-drop in QListView.*

## 9.1009 Digikam::NetworkManager Class Reference

Inheritance diagram for Digikam::NetworkManager:



## Public Member Functions

- QNetworkAccessManager \* **getNetworkManager** (QObject \*const object) const  
*Get the current QNetworkAccessManager or create a new QNetworkAccessManager if the passed QObject runs on a different thread.*

## Static Public Member Functions

- static NetworkManager \* **instance** ()  
*Global instance of internal network manager.*

## Friends

- class **NetworkManagerCreator**

## 9.1009.1 Member Function Documentation

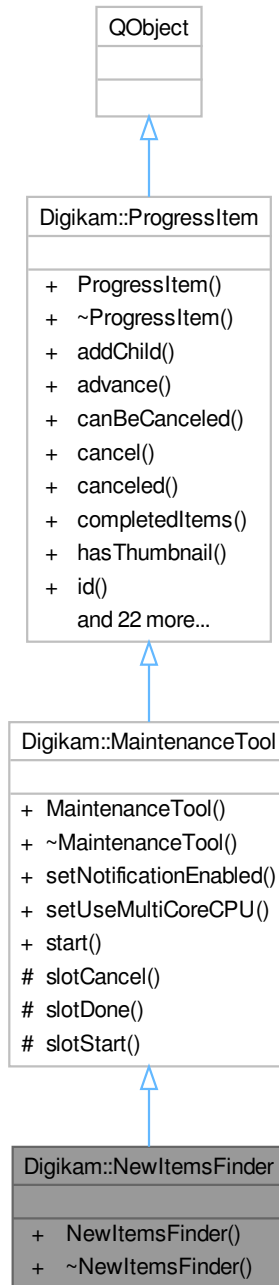
### 9.1009.1.1 instance()

```
NetworkManager * Digikam::NetworkManager::instance ( ) [static]
```

All accessor methods are thread-safe.

## 9.1010 Digikam::NewItemsFinder Class Reference

Inheritance diagram for Digikam::NewItemsFinder:



### Public Types

- enum `FinderMode` { `CompleteCollectionScan` , `ScanDeferredFiles` , `ScheduleCollectionScan` }

## Public Member Functions

- **NewItemsFinder** (const [FinderMode](#) mode=[CompleteCollectionScan](#), const QStringList &foldersToScan=QStringList(), [ProgressItem](#) \*const parent=nullptr)

## Public Member Functions inherited from [Digikam::MaintenanceTool](#)

- **MaintenanceTool** (const QString &id, [ProgressItem](#) \*const parent=nullptr)
- void **setNotificationEnabled** (bool b)
  - If true, show a notification message on desktop notification manager with time elapsed to run process.*
- virtual void **setUseMultiCoreCPU** (bool)
  - Re-implement this method if your tool is able to use multi-core CPU to process item in parallel.*

## Public Member Functions inherited from [Digikam::ProgressItem](#)

- **ProgressItem** ([ProgressItem](#) \*const parent, const QString &id, const QString &label, const QString &status, bool [canBeCanceled](#), bool hasThumb)
- void **addChild** ([ProgressItem](#) \*const kiddo)
- bool [advance](#) (unsigned int v)
  - Advance total items processed by n values and update percentage in progressbar.*
- bool [canBeCanceled](#) () const
- void **cancel** ()
- bool [canceled](#) () const
- unsigned int **completedItems** () const
- bool [hasThumbnail](#) () const
- const QString &[id](#) () const
- bool **incCompletedItems** (unsigned int v=1)
- void **incTotalItems** (unsigned int v=1)
- const QString &[label](#) () const
- [ProgressItem](#) \* [parent](#) () const
- unsigned int [progress](#) () const
- void **removeChild** ([ProgressItem](#) \*const kiddo)
- void **reset** ()
  - Reset the progress value of this item to 0 and the status string to the empty string.*
- void [setComplete](#) ()
  - Tell the item it has finished.*
- bool **setCompletedItems** (unsigned int v)
- void [setLabel](#) (const QString &v)
- void [setProgress](#) (unsigned int v)
  - Set the progress (percentage of completion) value of this item.*
- void [setShowAtStart](#) (bool [showAtStart](#))
  - Set the property to pop-up item when it's added in progress manager.*
- void [setStatus](#) (const QString &v)
  - Set the string to be used for showing this item's current status.*
- void [setThumbnail](#) (const QIcon &icon)
  - Sets whether this item has a thumbnail.*
- void **setTotalItems** (unsigned int v)
- void [setUsesBusyIndicator](#) (bool useBusyIndicator)
  - Sets whether this item uses a busy indicator instead of real progress for its progress bar.*
- bool [showAtStart](#) () const
- const QString &[status](#) () const
- bool **totalCompleted** () const
- unsigned int **totalItems** () const
- void **updateProgress** ()
  - Recalculate progress according to total/completed items and update.*
- bool [usesBusyIndicator](#) () const

## Additional Inherited Members

## Public Slots inherited from [Digikam::MaintenanceTool](#)

- void **start** ()

## Signals inherited from [Digikam::MaintenanceTool](#)

- void **signalCanceled** ()  
*Emit when process is canceled.*
- void **signalComplete** ()  
*Emit when process is done (not canceled).*

## Signals inherited from [Digikam::ProgressItem](#)

- void [progressItemAdded](#) ([ProgressItem](#) \*item)  
*Emitted when a new [ProgressItem](#) is added.*
- void [progressItemCanceled](#) ([ProgressItem](#) \*item)  
*Emitted when an item was canceled.*
- void **progressItemCanceledById** (const QString &id)
- void [progressItemCompleted](#) ([ProgressItem](#) \*item)  
*Emitted when a progress item was completed.*
- void [progressItemLabel](#) ([ProgressItem](#) \*item, const QString &label)  
*Emitted when the label of an item changed.*
- void [progressItemProgress](#) ([ProgressItem](#) \*item, unsigned int v)  
*Emitted when the progress value of an item changes.*
- void [progressItemStatus](#) ([ProgressItem](#) \*item, const QString &mess)  
*Emitted when the status message of an item changed.*
- void [progressItemThumbnail](#) ([ProgressItem](#) \*item, const QPixmap &thumb)  
*Emitted when the thumbnail data must be set in item.*
- void [progressItemUsesBusyIndicator](#) ([ProgressItem](#) \*item, bool value)  
*Emitted when the busy indicator state of an item changes.*

## Protected Slots inherited from [Digikam::MaintenanceTool](#)

- virtual void **slotCancel** ()
- virtual void **slotDone** ()
- virtual void **slotStart** ()

## 9.1010.1 Member Enumeration Documentation

### 9.1010.1.1 FinderMode

```
enum Digikam::NewItemsFinder::FinderMode
```

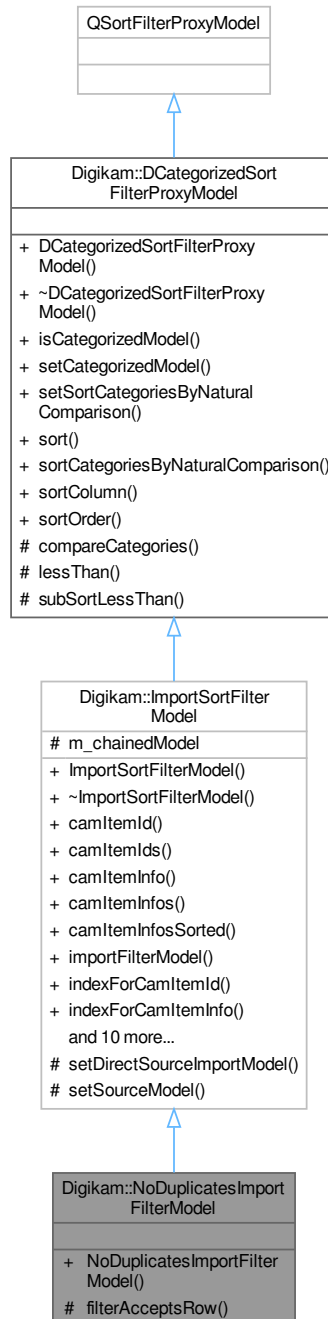
## Enumerator

CompleteCollectionScan	Scan whole collection immediately.
ScanDeferredFiles	Defer whole collection scan.
ScheduleCollectionScan	Scan immediately folders list passed in constructor.



## 9.1011 Digikam::NoDuplicatesImportFilterModel Class Reference

Inheritance diagram for Digikam::NoDuplicatesImportFilterModel:



### Public Member Functions

- **NoDuplicatesImportFilterModel** (QObject \*const parent=nullptr)

## Public Member Functions inherited from [Digikam::ImportSortFilterModel](#)

- **ImportSortFilterModel** (QObject \*const parent=nullptr)
- qlonglong **camItemId** (const QModelIndex &index) const
- QList< qlonglong > **camItemIds** (const QList< QModelIndex > &indexes) const
- [CamItemInfo](#) **camItemInfo** (const QModelIndex &index) const
- QList< [CamItemInfo](#) > **camItemInfos** (const QList< QModelIndex > &indexes) const
- QList< [CamItemInfo](#) > **camItemInfosSorted** () const
  - Returns a list of all camera infos, sorted according to this model.*
- virtual [ImportFilterModel](#) \* **importFilterModel** () const
  - Returns this, any chained [ImportFilterModel](#), or 0.*
- QModelIndex **indexForCamItemId** (qlonglong id) const
- QModelIndex **indexForCamItemInfo** (const [CamItemInfo](#) &info) const
- QModelIndex **indexForPath** (const QString &filePath) const
- QModelIndex **mapFromDirectSourceToSourceImportModel** (const QModelIndex &sourceModelIndex) const
- QModelIndex **mapFromSourceImportModel** (const QModelIndex &importModelIndex) const
- QList< QModelIndex > **mapListFromSource** (const QList< QModelIndex > &sourceIndexes) const
- QList< QModelIndex > **mapListToSource** (const QList< QModelIndex > &indexes) const
- QModelIndex **mapToSourceImportModel** (const QModelIndex &proxyIndex) const
  - Convenience methods mapped to [ImportItemModel](#).*
- void **setSourceFilterModel** ([ImportSortFilterModel](#) \*const sourceModel)
- void **setSourceImportModel** ([ImportItemModel](#) \*const sourceModel)
- [ImportSortFilterModel](#) \* **sourceFilterModel** () const
- [ImportItemModel](#) \* **sourceImportModel** () const

## Public Member Functions inherited from [Digikam::DCategorizedSortFilterProxyModel](#)

- **DCategorizedSortFilterProxyModel** (QObject \*const parent=nullptr)
- bool **isCategorizedModel** () const
- void **setCategorizedModel** (bool categorizedModel)
  - Enables or disables the categorization feature.*
- void **setSortCategoriesByNaturalComparison** (bool [sortCategoriesByNaturalComparison](#))
  - Set if the sorting using [CategorySortRole](#) will use a natural comparison in the case that strings were returned.*
- void **sort** (int column, Qt::SortOrder order=Qt::AscendingOrder) override
  - Overridden from [QSortFilterProxyModel](#).*
- bool **sortCategoriesByNaturalComparison** () const
- int **sortColumn** () const
- Qt::SortOrder **sortOrder** () const

## Protected Member Functions

- bool **filterAcceptsRow** (int source\_row, const QModelIndex &source\_parent) const override

## Protected Member Functions inherited from [Digikam::ImportSortFilterModel](#)

- virtual void **setDirectSourceImportModel** ([ImportItemModel](#) \*const sourceModel)
  - Reimplement if needed. Called only when model shall be set as (direct) sourceModel.*
- void **setSourceModel** (QAbstractItemModel \*sourceModel) override

## Protected Member Functions inherited from [Digikam::DCategorizedSortFilterProxyModel](#)

- virtual int [compareCategories](#) (const QModelIndex &left, const QModelIndex &right) const  
*This method compares the category of the `left` index with the category of the `right` index.*
- bool [lessThan](#) (const QModelIndex &left, const QModelIndex &right) const override  
*Overridden from `QSortFilterProxyModel`.*
- virtual bool [subSortLessThan](#) (const QModelIndex &left, const QModelIndex &right) const  
*This method has a similar purpose as [lessThan\(\)](#) has on `QSortFilterProxyModel`.*

## Additional Inherited Members

## Public Types inherited from [Digikam::DCategorizedSortFilterProxyModel](#)

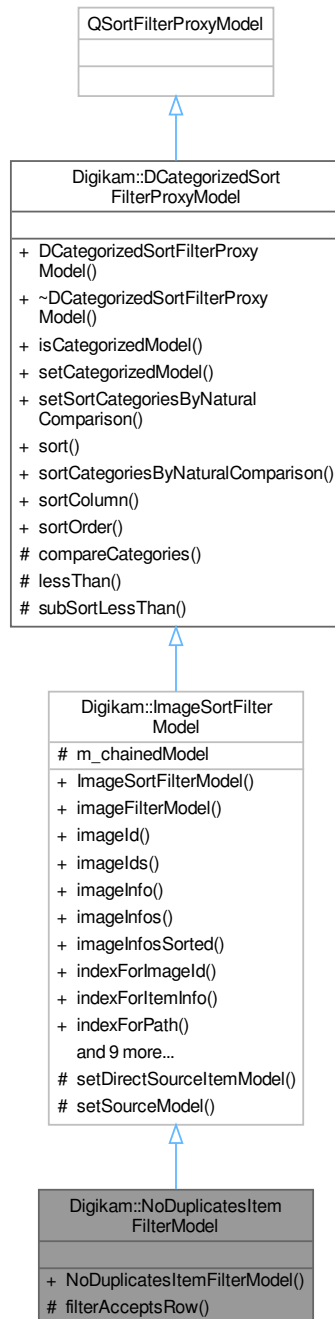
- enum [AdditionalRoles](#) { [CategoryDisplayRole](#) = 0x17CE990A , [CategorySortRole](#) = 0x27857E60 }

## Protected Attributes inherited from [Digikam::ImportSortFilterModel](#)

- [ImportSortFilterModel](#) \* [m\\_chainedModel](#) = nullptr

## 9.1012 Digikam::NoDuplicatesItemFilterModel Class Reference

Inheritance diagram for Digikam::NoDuplicatesItemFilterModel:



### Public Member Functions

- `NoDuplicatesItemFilterModel` (`QObject *const parent=nullptr`)

## Public Member Functions inherited from Digikam::ImageSortFilterModel

- **ImageSortFilterModel** (QObject \*const parent=nullptr)
- virtual **ItemFilterModel** \* **imageFilterModel** () const  
*Returns this, any chained [ItemFilterModel](#), or 0.*
- qlonglong **imageld** (const QModelIndex &index) const
- QList< qlonglong > **imagelds** (const QList< QModelIndex > &indexes) const
- **ItemInfo** **imageInfo** (const QModelIndex &index) const
- QList< **ItemInfo** > **imageInfos** (const QList< QModelIndex > &indexes) const
- QList< **ItemInfo** > **imageInfosSorted** () const  
*Returns a list of all image infos, sorted according to this model.*
- QModelIndex **indexForImageId** (qlonglong id) const
- QModelIndex **indexForItemInfo** (const **ItemInfo** &info) const
- QModelIndex **indexForPath** (const QString &filePath) const
- QModelIndex **mapFromDirectSourceToSourceItemModel** (const QModelIndex &sourceModel\_index) const
- QModelIndex **mapFromSourceItemModel** (const QModelIndex &imagemodel\_index) const
- QList< QModelIndex > **mapListFromSource** (const QList< QModelIndex > &sourceIndexes) const
- QList< QModelIndex > **mapListToSource** (const QList< QModelIndex > &indexes) const  
*Convenience methods mapped to [ItemModel](#).*
- QModelIndex **mapToSourceItemModel** (const QModelIndex &index) const
- void **setSourceFilterModel** (**ImageSortFilterModel** \*const model)
- void **setSourceItemModel** (**ItemModel** \*const model)
- **ImageSortFilterModel** \* **sourceFilterModel** () const
- **ItemModel** \* **sourceItemModel** () const

## Public Member Functions inherited from Digikam::DCategorizedSortFilterProxyModel

- **DCategorizedSortFilterProxyModel** (QObject \*const parent=nullptr)
- bool **isCategorizedModel** () const
- void **setCategorizedModel** (bool categorizedModel)  
*Enables or disables the categorization feature.*
- void **setSortCategoriesByNaturalComparison** (bool **sortCategoriesByNaturalComparison**)  
*Set if the sorting using [CategorySortRole](#) will use a natural comparison in the case that strings were returned.*
- void **sort** (int column, Qt::SortOrder order=Qt::AscendingOrder) override  
*Overridden from [QSortFilterProxyModel](#).*
- bool **sortCategoriesByNaturalComparison** () const
- int **sortColumn** () const
- Qt::SortOrder **sortOrder** () const

## Protected Member Functions

- bool **filterAcceptsRow** (int source\_row, const QModelIndex &source\_parent) const override

## Protected Member Functions inherited from Digikam::ImageSortFilterModel

- virtual void **setDirectSourceItemModel** (**ItemModel** \*const model)  
*Reimplement if needed.*
- void **setSourceModel** (QAbstractItemModel \*const model) override

### Protected Member Functions inherited from [Digikam::DCategorizedSortFilterProxyModel](#)

- virtual int [compareCategories](#) (const QModelIndex &left, const QModelIndex &right) const  
*This method compares the category of the `left` index with the category of the `right` index.*
- bool [lessThan](#) (const QModelIndex &left, const QModelIndex &right) const override  
*Overridden from `QSortFilterProxyModel`.*
- virtual bool [subSortLessThan](#) (const QModelIndex &left, const QModelIndex &right) const  
*This method has a similar purpose as `lessThan()` has on `QSortFilterProxyModel`.*

### Additional Inherited Members

### Public Types inherited from [Digikam::DCategorizedSortFilterProxyModel](#)

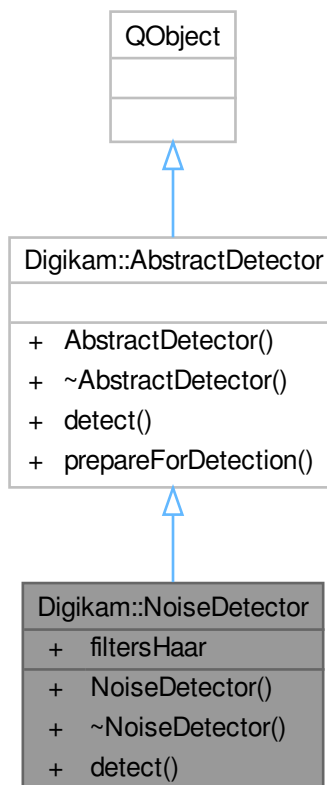
- enum [AdditionalRoles](#) { `CategoryDisplayRole` = 0x17CE990A , `CategorySortRole` = 0x27857E60 }

### Protected Attributes inherited from [Digikam::ImageSortFilterModel](#)

- [ImageSortFilterModel](#) \* `m_chainedModel` = nullptr

## 9.1013 Digikam::NoiseDetector Class Reference

Inheritance diagram for Digikam::NoiseDetector:



**Public Types**

- typedef QList< cv::Mat > **Mat3D**

**Public Member Functions**

- float [detect](#) (const cv::Mat &image) const override

**Public Member Functions inherited from [Digikam::AbstractDetector](#)**

- **AbstractDetector** (QObject \*const parent=nullptr)

**Static Public Attributes**

- static const Mat3D **filtersHaar** = initFiltersHaar()

**Additional Inherited Members****Static Public Member Functions inherited from [Digikam::AbstractDetector](#)**

- static cv::Mat **prepareForDetection** (const [DImg](#) &inputImage)

*NOTE: Maybe this function will move to `read_image()` of `imagequalityparser` in case all detectors of IQS use `cv::Mat`.*

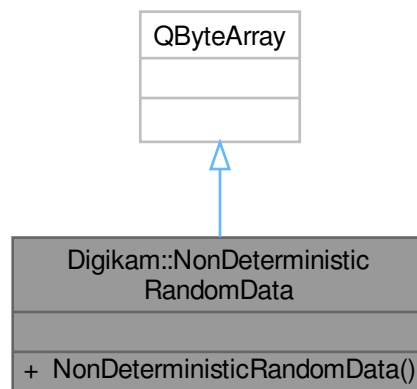
**9.1013.1 Member Function Documentation****9.1013.1.1 detect()**

```
float Digikam::NoiseDetector::detect (
    const cv::Mat & image ) const [override], [virtual]
```

Implements [Digikam::AbstractDetector](#).

**9.1014 Digikam::NonDeterministicRandomData Class Reference**

Inheritance diagram for Digikam::NonDeterministicRandomData:



## Public Member Functions

- [NonDeterministicRandomData](#) (int size)

*Constructs a QByteArray of given byte size filled with non-deterministic random data.*

## 9.1014.1 Constructor & Destructor Documentation

### 9.1014.1.1 NonDeterministicRandomData()

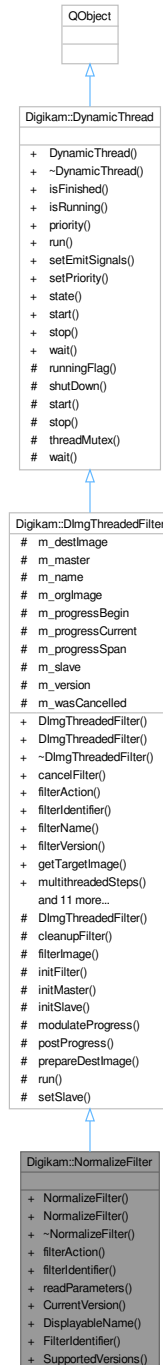
```
Digikam::NonDeterministicRandomData::NonDeterministicRandomData (  
    int size ) [explicit]
```

For larger quantities of data, prefer using a [RandomNumberGenerator](#) seeded with non-deterministic data.



## 9.1015 Digikam::NormalizeFilter Class Reference

Inheritance diagram for Digikam::NormalizeFilter:



### Public Member Functions

- **NormalizeFilter** (`Dlmg *const orgImage`, `const Dlmg *const reflImage`, `QObject *const parent=nullptr`)
- **NormalizeFilter** (`QObject *const parent=nullptr`)

- [FilterAction filterAction \(\)](#) override  
*Returns the action description corresponding to currently set options.*
- [QString filterIdentifier \(\)](#) const override  
*Return the identifier for this filter in the image history.*
- void [readParameters \(const FilterAction &action\)](#) override

## Public Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter \(DImg \\*const orgImage, QObject \\*const parent, const QString &name=QString\(\)\)](#)  
*Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter \(QObject \\*const parent=nullptr, const QString &name=QString\(\)\)](#)  
*Constructs a filter without argument.*
- virtual void [cancelFilter \(\)](#)  
*Cancel the threaded computation.*
- const [QString &filterName \(\)](#)
- int [filterVersion \(\)](#) const
- [DImg getTargetImage \(\)](#)
- [QList< int > multithreadedSteps \(int stop, int start=0\)](#) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool [parametersSuccessfullyRead \(\)](#) const  
*Optional: error handling for readParameters.*
- virtual [QString readParametersError \(const FilterAction &actionThatFailed\)](#) const
- void [setFilterName \(const QString &name\)](#)
- void [setFilterVersion \(int version\)](#)  
*Replaying a filter action: Set the filter version.*
- void [setOriginalImage \(const DImg &orgImage\)](#)
- void [setupAndStartDirectly \(const DImg &orgImage, DImgThreadedFilter \\*const master, int progress←Begin=0, int progressEnd=100\)](#)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void [setupFilter \(const DImg &orgImage\)](#)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void [startFilter \(\)](#)  
*Start the threaded computation.*
- virtual void [startFilterDirectly \(\)](#)  
*Start computation of this filter, directly in this thread.*
- virtual [QList< int > supportedVersions \(\)](#) const

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread \(QObject \\*const parent=nullptr\)](#)  
*This class extends QRunnable, so you have to reimplement virtual void [run\(\)](#).*
- [~DynamicThread \(\)](#) override  
*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool [isFinished \(\)](#) const
- bool [isRunning \(\)](#) const
- [QThread::Priority priority \(\)](#) const
- void [setEmitSignals \(bool emitThem\)](#)
- void [setPriority \(QThread::Priority priority\)](#)  
*Sets the priority for this dynamic thread.*
- State [state \(\)](#) const

### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from [Digikam::DImgThreadedFilter](#)

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.1015.1 Member Function Documentation

### 9.1015.1.1 filterAction()

`FilterAction` Digikam::NormalizeFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.1015.1.2 filterIdentifier()

`QString` Digikam::NormalizeFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.1015.1.3 readParameters()

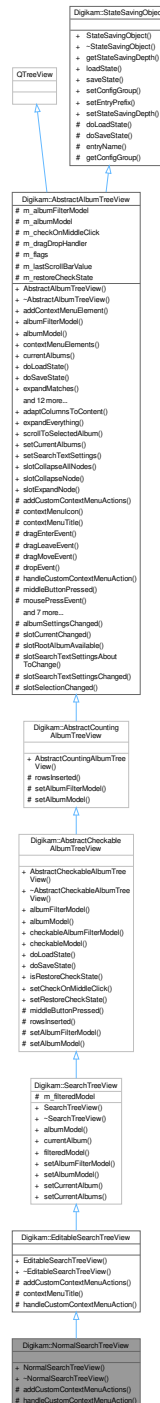
```
void Digikam::NormalizeFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

## 9.1016 Digikam::NormalSearchTreeView Class Reference

Tree view for all saved "normal" searches.

Inheritance diagram for Digikam::NormalSearchTreeView:



## Signals

- void `copySearch` (SAlbum \*album)  
Emitted if the given search shall be copied.
- void `editSearch` (SAlbum \*album)  
Emitted if the given search shall be edited.
- void `newSearch` ()  
Emitted if a new search shall be created.

## Signals inherited from Digikam::AbstractAlbumTreeView

- void **currentAlbumChanged** ([Album](#) \*currentAlbum)  
*Emitted when the currently selected album changes.*
- void **selectedAlbumsChanged** (const [QList](#)< [Album](#) \* > &selectedAlbums)  
*Emitted when the current selection changes.*

## Public Member Functions

- [NormalSearchTreeView](#) ([QWidget](#) \*const parent, [searchModel](#) \*const searchModel, [SearchModificationHelper](#) \*const searchModificationHelper)  
*Constructor.*
- ~[NormalSearchTreeView](#) () override  
*Destructor.*

## Public Member Functions inherited from Digikam::EditableSearchTreeView

- [EditableSearchTreeView](#) ([QWidget](#) \*const parent, [searchModel](#) \*const searchModel, [SearchModificationHelper](#) \*const searchModificationHelper)  
*Constructor.*
- ~[EditableSearchTreeView](#) () override  
*Destructor.*

## Public Member Functions inherited from Digikam::SearchTreeView

- [SearchTreeView](#) ([QWidget](#) \*const parent=nullptr, [Flags](#) flags=DefaultFlags)
- [searchModel](#) \* **albumModel** () const  
*Note: not filtered by search type.*
- [SAlbum](#) \* **currentAlbum** () const
- [SearchFilterModel](#) \* **filteredModel** () const  
*Contains only the searches with appropriate type - prefer to albumModel()*
- void **setAlbumFilterModel** ([SearchFilterModel](#) \*const filteredModel, [CheckableAlbumFilterModel](#) \*const model)
- void **setAlbumModel** ([searchModel](#) \*const model)

## Public Member Functions inherited from Digikam::AbstractCheckableAlbumTreeView

- [AbstractCheckableAlbumTreeView](#) ([QWidget](#) \*const parent, [Flags](#) flags)  
*Models of these view can be checkable, they need not.*
- [CheckableAlbumFilterModel](#) \* **albumFilterModel** () const
- [AbstractCheckableAlbumModel](#) \* **albumModel** () const  
*Manage check state through the model directly.*
- [CheckableAlbumFilterModel](#) \* **checkableAlbumFilterModel** () const
- [AbstractCheckableAlbumModel](#) \* **checkableModel** () const
- void **doLoadState** () override  
*Implements state loading for the album tree view in a somewhat clumsy procedure because the model may not be fully loaded when this method is called.*
- void **doSaveState** () override  
*Implement this hook method for state saving.*
- bool **isRestoreCheckState** () const  
*Tells if the check state is restored while loading / saving state.*
- void **setCheckOnMiddleClick** (bool doThat)  
*Enable checking on middle mouse button click (default: on).*
- void **setRestoreCheckState** (bool restore)  
*Set whether to restore check state or not.*

## Public Member Functions inherited from [Digikam::AbstractCountingAlbumTreeView](#)

- **AbstractCountingAlbumTreeView** (QWidget \*const parent, Flags flags)

## Public Member Functions inherited from [Digikam::AbstractAlbumTreeView](#)

- [AbstractAlbumTreeView](#) (QWidget \*const parent, Flags flags)  
*Constructs an album tree view.*
- void **addContextMenuElement** ([ContextMenuElement](#) \*const element)
- [AlbumFilterModel](#) \* **albumFilterModel** () const
- [AbstractSpecificAlbumModel](#) \* **albumModel** () const
- QList< [ContextMenuElement](#) \* > **contextMenuElements** () const
- template<class A >  
QList< A \* > **currentAlbums** ()
- bool **expandMatches** (const QModelIndex &index)  
*Ensures that every current match is visible by expanding all parent entries.*
- QModelIndex **indexVisuallyAt** (const QPoint &p)  
*This is a combination of `indexAt()` checked with `visualRect()`.*
- void **removeContextMenuElement** ([ContextMenuElement](#) \*const element)
- QList< [Album](#) \* > **selectedItems** ()  
*selectedItems()* -
- void **setAlbumManagerCurrentAlbum** (const bool setCurrentAlbum)  
*Some treeviews shall control the global current album kept by `AlbumManager`.*
- void **setContextMenuIcon** (const QPixmap &pixmap)  
*Set the context menu title and icon.*
- void **setContextMenuTitle** (const QString &title)
- void **setEnabledContextMenu** (const bool enable)  
*Determines the global decision to show a popup menu or not.*
- void **setExpandNewCurrentItem** (const bool doThat)  
*Expand an item when making it the new current item.*
- void **setExpandOnSingleClick** (const bool doThat)  
*Enable expanding of tree items on single click on the item (default: off)*
- void **setSelectAlbumOnClick** (const bool selectOnClick)  
*Sets whether to select an album on click via the album manager or not.*
- void **setSelectOnContextMenu** (const bool select)  
*Sets whether to select the album under the mouse cursor on a context menu request (so that the album is shown using the album manager) or not.*
- bool **viewportEvent** (QEvent \*event) override  
*For internal use only.*

## Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (QObject \*const host)  
*Constructor.*
- virtual ~**StateSavingObject** ()  
*Destructor.*
- [StateSavingDepth](#) **getStateSavingDepth** () const  
*Returns the depth used for state saving or loading.*
- void **loadState** ()  
*Invokes loading the class' state.*



- void **saveState** ()  
*Invokes saving the class' state.*
- virtual void **setConfigGroup** (const KConfigGroup &group)  
*Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void **setEntryPrefix** (const QString &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void **setStateSavingDepth** (const [StateSavingDepth](#) depth)  
*Sets the depth used for state saving or loading.*

### Protected Member Functions

- void **addCustomContextMenuActions** ([ContextMenuHelper](#) &cmh, [Album](#) \*album) override  
*Adds actions to delete or rename existing searches.*
- void **handleCustomContextMenuAction** (QAction \*action, const [AlbumPointer](#)< [Album](#) > &album) override  
*Handles deletion and renaming actions.*

### Protected Member Functions inherited from [Digikam::EditableSearchTreeView](#)

- QString **contextMenuTitle** () const override  
*implemented hook methods for context menus.*

### Protected Member Functions inherited from [Digikam::AbstractCheckableAlbumTreeView](#)

- void **middleButtonPressed** ([Album](#) \*a) override
- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- void **setAlbumFilterModel** ([CheckableAlbumFilterModel](#) \*const filterModel)
- void **setAlbumModel** ([AbstractCheckableAlbumModel](#) \*const model)

### Protected Member Functions inherited from [Digikam::AbstractCountingAlbumTreeView](#)

- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- void **setAlbumFilterModel** ([AlbumFilterModel](#) \*const filterModel)
- void **setAlbumModel** ([AbstractCountingAlbumModel](#) \*const model)

### Protected Member Functions inherited from [Digikam::AbstractAlbumTreeView](#)

- virtual QPixmap **contextMenuIcon** () const  
*Hook method that can be implemented to return a special icon used for the context menu.*
- void **dragEnterEvent** (QDragEnterEvent \*e) override
- void **dragLeaveEvent** (QDragLeaveEvent \*e) override
- void **dragMoveEvent** (QDragMoveEvent \*e) override
- void **dropEvent** (QDropEvent \*e) override
- void **mousePressEvent** (QMouseEvent \*e) override  
*Other helper methods.*
- virtual QPixmap  **pixmapForDrag** (const QStyleOptionViewItem &option, QList< QModelIndex > indexes)  
*TODO: Move to delegate, when we have one.*
- void **rowsAboutToBeRemoved** (const QModelIndex &parent, int start, int end) override
- void **rowsInserted** (const QModelIndex &index, int start, int end) override
- void **setAlbumFilterModel** ([AlbumFilterModel](#) \*const filterModel)
- void **setAlbumModel** ([AbstractSpecificAlbumModel](#) \*const model)
- virtual bool **showContextMenuAt** (QContextMenuEvent \*event, [Album](#) \*albumForEvent)  
*Hook method to implement that determines if a context menu shall be displayed for the given event at the position coded in the event.*
- void **startDrag** (Qt::DropActions supportedActions) override

## Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- QString [entryName](#) (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup [getConfigGroup](#) () const  
*Returns the config group that must be used for state saving and loading.*

## Additional Inherited Members

## Public Types inherited from [Digikam::AbstractAlbumTreeView](#)

- enum [Flag](#) {  
[CreateDefaultModel](#) , [CreateDefaultFilterModel](#) , [CreateDefaultDelegate](#) , [ShowCountAccordingToSettings](#) ,  
[AlwaysShowInclusiveCounts](#) , **DefaultFlags** = [CreateDefaultFilterModel](#) | [CreateDefaultDelegate](#) | Show↔  
CountAccordingToSettings }
- typedef QFlags< [Flag](#) > **Flags**

## Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }  
*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

## Public Slots inherited from [Digikam::SearchTreeView](#)

- void [setCurrentAlbum](#) (int searchId, bool selectInAlbumManager=true)
- void [setCurrentAlbums](#) (const QList< [Album](#) \* > &albums, bool selectInAlbumManager=true)

## Public Slots inherited from [Digikam::AbstractAlbumTreeView](#)

- void [adaptColumnsToContent](#) ()  
*Adapt the column sizes to the contents of the tree view.*
- void [expandEverything](#) (const QModelIndex &index)  
*Expands the complete tree under the given index.*
- void [scrollToSelectedAlbum](#) ()  
*Scrolls to the first selected album if there is one.*
- void [setCurrentAlbums](#) (const QList< [Album](#) \* > &albums, bool selectInAlbumManager=true)  
*Selects the given album.*
- void [setSearchTextSettings](#) (const [SearchTextSettings](#) &settings)
- void [slotCollapseAllNodes](#) ()  
*slotCollapseAllNodes - collapse all nodes without root node*
- void [slotCollapseNode](#) ()  
*slotCollapseNode - collapse recursively selected nodes*
- void [slotExpandNode](#) ()  
*slotExpandNode - expands recursively selected nodes*

## Protected Slots inherited from [Digikam::AbstractAlbumTreeView](#)

- void **albumSettingsChanged** ()
- void **slotCurrentChanged** ()
- virtual void **slotRootAlbumAvailable** ()  
*override if implemented behavior is not as intended*
- void **slotSearchTextSettingsAboutToChange** (bool searched, bool willSearch)
- void **slotSearchTextSettingsChanged** (bool wasSearching, bool searching)
- void **slotSelectionChanged** ()

## Protected Attributes inherited from [Digikam::SearchTreeView](#)

- [SearchFilterModel](#) \* **m\_filteredModel** = nullptr

## Protected Attributes inherited from [Digikam::AbstractAlbumTreeView](#)

- [AlbumFilterModel](#) \* **m\_albumFilterModel** = nullptr
- [AbstractSpecificAlbumModel](#) \* **m\_albumModel** = nullptr
- bool **m\_checkOnMiddleClick** = false
- [AlbumModelDragDropHandler](#) \* **m\_dragDropHandler** = nullptr
- Flags **m\_flags** = DefaultFlags
- int **m\_lastScrollBarValue** = 0
- bool **m\_restoreCheckState** = false

### 9.1016.1 Detailed Description

Allows editing and creating searches in the context menu.

Author

jwienke

### 9.1016.2 Constructor & Destructor Documentation

#### 9.1016.2.1 NormalSearchTreeView()

```
Digikam::NormalSearchTreeView::NormalSearchTreeView (
    QWidget *const parent,
    SearchModel *const searchModel,
    SearchModificationHelper *const searchModificationHelper )
```

Parameters

<i>parent</i>	qt parent
<i>searchModel</i>	the model this view should act on
<i>searchModificationHelper</i>	the modification helper object used to perform operations on the displayed searches

### 9.1016.3 Member Function Documentation

#### 9.1016.3.1 addCustomContextMenuActions()

```
void Digikam::NormalSearchTreeView::addCustomContextMenuActions (
    ContextMenuHelper & cmh,
    Album * album ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::EditableSearchTreeView](#).

#### 9.1016.3.2 copySearch

```
void Digikam::NormalSearchTreeView::copySearch (
    SAlbum * album ) [signal]
```

##### Parameters

<i>album</i>	search to copy
--------------	----------------

#### 9.1016.3.3 editSearch

```
void Digikam::NormalSearchTreeView::editSearch (
    SAlbum * album ) [signal]
```

##### Parameters

<i>album</i>	search to edit
--------------	----------------

#### 9.1016.3.4 handleCustomContextMenuAction()

```
void Digikam::NormalSearchTreeView::handleCustomContextMenuAction (
    QAction * action,
    const AlbumPointer< Album > & album ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::EditableSearchTreeView](#).

## 9.1017 Digikam::NRContainer Class Reference

### Public Attributes

- double **softness** [3] = { 0.9 }  
*Y, Cb, Cr softness.*
- double **thresholds** [3] = { 1.2 }  
*Separated values per channel.*

## 9.1017.1 Member Data Documentation

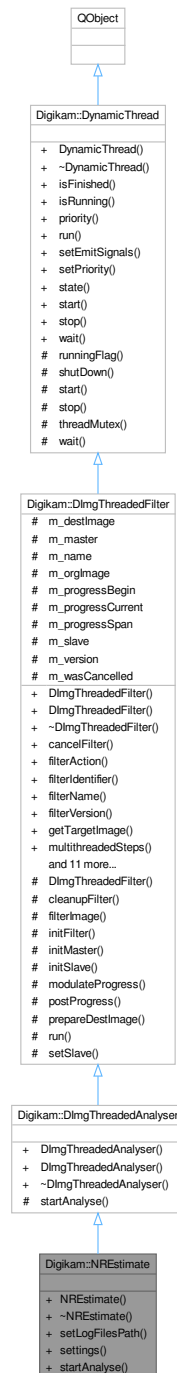
### 9.1017.1.1 thresholds

```
double Digikam::NRContainer::thresholds[3] = { 1.2 }
```

Y, Cb, Cr thresholds.

## 9.1018 Digikam::NREstimate Class Reference

Inheritance diagram for Digikam::NREstimate:



### Public Member Functions

- **NREstimate** (`Dimg *const img`, `QObject *const parent=nullptr`)

*Standard constructor with image container to parse.*

- void [setLogFilesPath](#) (const QString &path)  
*To set image path where log files will be created to host computation algorithm results, for hacking purpose.*
- [NRContainer settings](#) () const  
*Return all Wavelets noise reduction settings computed by image analys.*
- void [startAnalyse](#) () override  
*Perform estimate noise.*

### Public Member Functions inherited from [Digikam::DImgThreadedAnalyser](#)

- [DImgThreadedAnalyser](#) (DImg \*const orgImage, QObject \*const parent=nullptr, const QString &name=QString())  
*Constructs an image analyser with all arguments (ready to use).*
- [DImgThreadedAnalyser](#) (QObject \*const parent=nullptr, const QString &name=QString())  
*Constructs a filter without argument.*

### Public Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) (DImg \*const orgImage, QObject \*const parent, const QString &name=QString())  
*Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter](#) (QObject \*const parent=nullptr, const QString &name=QString())  
*Constructs a filter without argument.*
- virtual void [cancelFilter](#) ()  
*Cancel the threaded computation.*
- const QString & [filterName](#) ()
- int [filterVersion](#) () const
- [DImg](#) [getTargetImage](#) ()
- QList< int > [multithreadedSteps](#) (int stop, int start=0) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool [parametersSuccessfullyRead](#) () const  
*Optional: error handling for readParameters.*
- virtual QString [readParametersError](#) (const [FilterAction](#) &actionThatFailed) const
- void [setFilterName](#) (const QString &name)
- void [setFilterVersion](#) (int version)  
*Replaying a filter action: Set the filter version.*
- void [setOriginalImage](#) (const [DImg](#) &orgImage)
- void [setupAndStartDirectly](#) (const [DImg](#) &orgImage, [DImgThreadedFilter](#) \*const master, int progress←Begin=0, int progressEnd=100)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void [setupFilter](#) (const [DImg](#) &orgImage)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void [startFilter](#) ()  
*Start the threaded computation.*
- virtual void [startFilterDirectly](#) ()  
*Start computation of this filter, directly in this thread.*

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread](#) (QObject \*const parent=nullptr)
 

*This class extends QRunnable, so you have to reimplement virtual void [run\(\)](#).*
- [~DynamicThread](#) () override
 

*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool [isFinished](#) () const
- bool [isRunning](#) () const
- QThread::Priority [priority](#) () const
- void [setEmitSignals](#) (bool emitThem)
- void [setPriority](#) (QThread::Priority priority)
 

*Sets the priority for this dynamic thread.*
- State [state](#) () const

## Additional Inherited Members

## Public Types inherited from [Digikam::DynamicThread](#)

- enum [State](#) { [Inactive](#) , [Scheduled](#) , [Running](#) , [Deactivating](#) }

## Public Slots inherited from [Digikam::DynamicThread](#)

- void [start](#) ()
- void [stop](#) ()
 

*Stop computation, sets the running flag to false.*
- void [wait](#) ()
 

*Waits until the thread finishes.*

## Signals inherited from [Digikam::DImgThreadedFilter](#)

- void [finished](#) (bool success)
 

*Emitted when the computation has completed.*
- void [progress](#) (int progress)
 

*Emitted when progress info from the calculation is available.*
- void [started](#) ()
 

*This signal is emitted when image data is available and the computation has started.*

## Signals inherited from [Digikam::DynamicThread](#)

- void [finished](#) ()
- void [starting](#) ()
 

*Emitted if emitSignals is enabled.*



## Protected Member Functions inherited from Digikam::DImgThreadedFilter

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from Digikam::DynamicThread

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from Digikam::DImgThreadedFilter

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.1018.1 Member Function Documentation

### 9.1018.1.1 setLogFilePath()

```
void Digikam::NREstimate::setLogFilePath (
    const QString & path )
```

If path is not set, no log files will be created.

### 9.1018.1.2 startAnalyse()

```
void Digikam::NREstimate::startAnalyse ( ) [override], [virtual]
```

Implements [Digikam::DImgThreadedAnalyser](#).

## 9.1019 Digikam::NRFilter Class Reference

Inheritance diagram for Digikam::NRFilter:



### Public Member Functions

- **NRFilter** ([DImg](#) \*const orgImage, [QObject](#) \*const parent, const [NRContainer](#) &settings)
- **NRFilter** ([QObject](#) \*const parent=nullptr)

- [FilterAction filterAction \(\)](#) override  
*Returns the action description corresponding to currently set options.*
- [QString filterIdentifier \(\)](#) const override  
*Return the identifier for this filter in the image history.*
- void [readParameters \(const FilterAction &action\)](#) override

## Public Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter \(DImg \\*const orgImage, QObject \\*const parent, const QString &name=QString\(\)\)](#)  
*Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter \(QObject \\*const parent=nullptr, const QString &name=QString\(\)\)](#)  
*Constructs a filter without argument.*
- virtual void [cancelFilter \(\)](#)  
*Cancel the threaded computation.*
- const [QString &filterName \(\)](#)
- int [filterVersion \(\)](#) const
- [DImg getTargetImage \(\)](#)
- [QList< int > multithreadedSteps \(int stop, int start=0\)](#) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool [parametersSuccessfullyRead \(\)](#) const  
*Optional: error handling for readParameters.*
- virtual [QString readParametersError \(const FilterAction &actionThatFailed\)](#) const
- void [setFilterName \(const QString &name\)](#)
- void [setFilterVersion \(int version\)](#)  
*Replaying a filter action: Set the filter version.*
- void [setOriginalImage \(const DImg &orgImage\)](#)
- void [setupAndStartDirectly \(const DImg &orgImage, DImgThreadedFilter \\*const master, int progress←Begin=0, int progressEnd=100\)](#)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void [setupFilter \(const DImg &orgImage\)](#)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void [startFilter \(\)](#)  
*Start the threaded computation.*
- virtual void [startFilterDirectly \(\)](#)  
*Start computation of this filter, directly in this thread.*
- virtual [QList< int > supportedVersions \(\)](#) const

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread \(QObject \\*const parent=nullptr\)](#)  
*This class extends QRunnable, so you have to reimplement virtual void [run\(\)](#).*
- [~DynamicThread \(\)](#) override  
*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool [isFinished \(\)](#) const
- bool [isRunning \(\)](#) const
- [QThread::Priority priority \(\)](#) const
- void [setEmitSignals \(bool emitThem\)](#)
- void [setPriority \(QThread::Priority priority\)](#)  
*Sets the priority for this dynamic thread.*
- State [state \(\)](#) const

### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static void **srgb2ycbcr** (float \*\*const fimg, uint size)
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from [Digikam::DImgThreadedFilter](#)

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.1019.1 Member Function Documentation

### 9.1019.1.1 filterAction()

`FilterAction` Digikam::NRFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.1019.1.2 filterIdentifier()

`QString` Digikam::NRFilter::filterIdentifier ( ) const [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

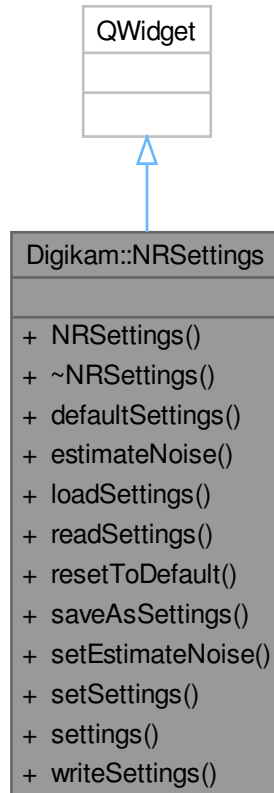
### 9.1019.1.3 readParameters()

`void` Digikam::NRFilter::readParameters (   
     const `FilterAction` & *action* ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

## 9.1020 Digikam::NRSettings Class Reference

Inheritance diagram for Digikam::NRSettings:



## Signals

- void **signalEstimateNoise** ()
- void **signalSettingsChanged** ()

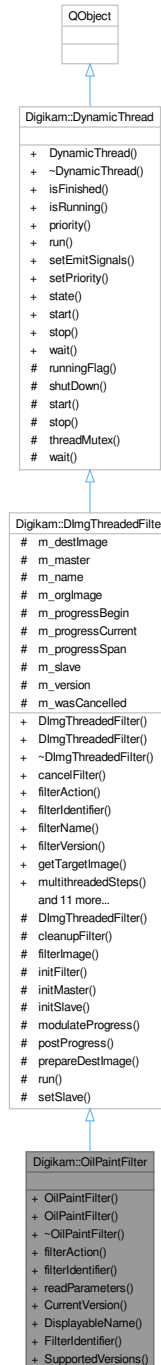
## Public Member Functions

- **NRSettings** (QWidget \*const parent)
- [NRContainer](#) **defaultSettings** () const
- bool **estimateNoise** () const
- void **loadSettings** ()
- void **readSettings** (const KConfigGroup &group)
- void **resetToDefault** ()
- void **saveAsSettings** ()
- void **setEstimateNoise** (bool b)
- void **setSettings** (const [NRContainer](#) &settings)
- [NRContainer](#) **settings** () const
- void **writeSettings** (KConfigGroup &group)



## 9.1021 Digikam::OilPaintFilter Class Reference

Inheritance diagram for Digikam::OilPaintFilter:



### Public Member Functions

- **OilPaintFilter** ([DImg](#) \*const orgImage, `QObject` \*const parent=nullptr, int brushSize=1, int smoothness=30)
- **OilPaintFilter** (`QObject` \*const parent=nullptr)

- [FilterAction filterAction \(\)](#) override  
*Returns the action description corresponding to currently set options.*
- [QString filterIdentifier \(\)](#) const override  
*Return the identifier for this filter in the image history.*
- void [readParameters \(const FilterAction &action\)](#) override

## Public Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter \(DImg \\*const orgImage, QObject \\*const parent, const QString &name=QString\(\)\)](#)  
*Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter \(QObject \\*const parent=nullptr, const QString &name=QString\(\)\)](#)  
*Constructs a filter without argument.*
- virtual void [cancelFilter \(\)](#)  
*Cancel the threaded computation.*
- const [QString &filterName \(\)](#)
- int [filterVersion \(\)](#) const
- [DImg getTargetImage \(\)](#)
- [QList< int > multithreadedSteps \(int stop, int start=0\)](#) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool [parametersSuccessfullyRead \(\)](#) const  
*Optional: error handling for readParameters.*
- virtual [QString readParametersError \(const FilterAction &actionThatFailed\)](#) const
- void [setFilterName \(const QString &name\)](#)
- void [setFilterVersion \(int version\)](#)  
*Replaying a filter action: Set the filter version.*
- void [setOriginalImage \(const DImg &orgImage\)](#)
- void [setupAndStartDirectly \(const DImg &orgImage, DImgThreadedFilter \\*const master, int progress←Begin=0, int progressEnd=100\)](#)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void [setupFilter \(const DImg &orgImage\)](#)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void [startFilter \(\)](#)  
*Start the threaded computation.*
- virtual void [startFilterDirectly \(\)](#)  
*Start computation of this filter, directly in this thread.*
- virtual [QList< int > supportedVersions \(\)](#) const

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread \(QObject \\*const parent=nullptr\)](#)  
*This class extends QRunnable, so you have to reimplement virtual void [run\(\)](#).*
- [~DynamicThread \(\)](#) override  
*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool [isFinished \(\)](#) const
- bool [isRunning \(\)](#) const
- [QThread::Priority priority \(\)](#) const
- void [setEmitSignals \(bool emitThem\)](#)
- void [setPriority \(QThread::Priority priority\)](#)  
*Sets the priority for this dynamic thread.*
- State [state \(\)](#) const

### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from [Digikam::DImgThreadedFilter](#)

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.1021.1 Member Function Documentation

### 9.1021.1.1 filterAction()

`FilterAction` Digikam::OilPaintFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.1021.1.2 filterIdentifier()

`QString` Digikam::OilPaintFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

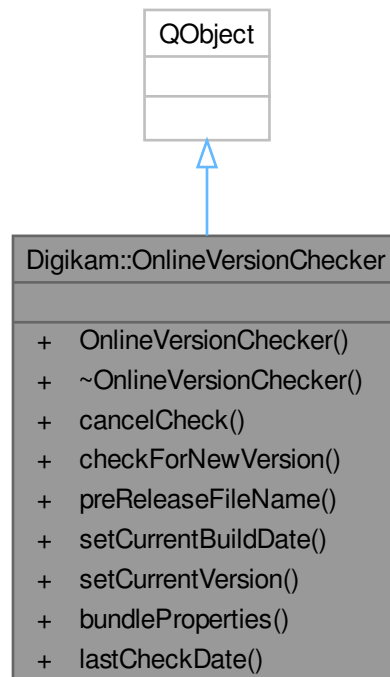
### 9.1021.1.3 readParameters()

```
void Digikam::OilPaintFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

## 9.1022 Digikam::OnlineVersionChecker Class Reference

Inheritance diagram for Digikam::OnlineVersionChecker:



## Signals

- void **signalNewVersionAvailable** (const QString &version)
- void **signalNewVersionCheckError** (const QString &error)

## Public Member Functions

- **OnlineVersionChecker** (QObject \*const parent, bool checkPreRelease=false)
- void **cancelCheck** ()
- void **checkForNewVersion** ()
- QString **preReleaseFileName** () const
- void **setCurrentBuildDate** (const QDateTime &dt)
- void **setCurrentVersion** (const QString &version)

## Static Public Member Functions

- static bool **bundleProperties** (QString &arch, QString &ext, QString &qtVersion, QString &dir)  
*Return true if the system and architecture are supported by the bundle workflow.*
- static QString **lastCheckDate** ()  
*Return the last date as string when have been performed a check for new version.*

## 9.1022.1 Member Function Documentation

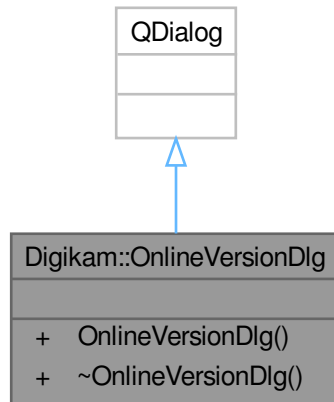
### 9.1022.1.1 bundleProperties()

```
bool Digikam::OnlineVersionChecker::bundleProperties (
    QString & arch,
    QString & ext,
    QString & qtVersion,
    QString & dir ) [static]
```

'arch' is the relevant prefix for the bundle architecture. 'ext' is the relevant bundle file extension. 'qtVersion' is the relevant version of Qt used in the bundle file-name. 'dir' is the subdirectory if any to get the bundle file.

## 9.1023 Digikam::OnlineVersionDlg Class Reference

Inheritance diagram for Digikam::OnlineVersionDlg:



### Signals

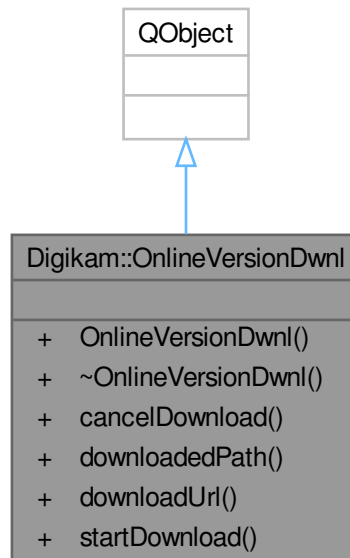
- void **signalSetupUpdate** ()

### Public Member Functions

- **OnlineVersionDlg** (QWidget \*const parent=nullptr, const QString &version=QLatin1String(digikam\_↵  
version\_short), const QDateTime &buildDt=digiKamBuildDate(), bool checkPreRelease=false, bool update↵  
WithDebug=false)

## 9.1024 Digikam::OnlineVersionDwnl Class Reference

Inheritance diagram for Digikam::OnlineVersionDwnl:



### Signals

- void **signalComputeChecksum** ()
- void **signalDownloadError** (const QString &error)
- void **signalDownloadProgress** (qint64 bytesReceived, qint64 bytesTotal)

### Public Member Functions

- **OnlineVersionDwnl** (QObject \*const parent=nullptr, bool checkPreRelease=false, bool updateWithDebug=false)
- void **cancelDownload** ()
- QString **downloadedPath** () const
- QString **downloadUrl** () const
- void **startDownload** (const QString &version)

## 9.1025 Digikam::OpenCVDNNFaceDetector Class Reference

### Public Member Functions

- **OpenCVDNNFaceDetector** ([DetectorNNModel](#) model=[DetectorNNModel::DNNDetectorYuNet](#))
- std::vector< cv::Rect > **cvDetectFaces** (const cv::Mat &inputImage, const cv::Size &paddedSize)
- QList< QRect > **detectFaces** (const cv::Mat &inputImage, const cv::Size &paddedSize)



*There is no proof that doing this will help, since face can be detected at various positions (even half, masked faces can be detected), not only frontal.*

- cv::Mat **prepareForDetection** (const [DImg](#) &inputImage, cv::Size &paddedSize) const
- cv::Mat **prepareForDetection** (const QImage &inputImage, cv::Size &paddedSize) const
- cv::Mat **prepareForDetection** (const QString &inputImagePath, cv::Size &paddedSize) const
- cv::Mat **prepareForDetectionYuNet** (cv::Mat &cvImage, cv::Size &paddedSize) const
- void **setAccuracy** (const int accuracy)
- void **setFaceDetectionSize** ([FaceScanSettings::FaceDetectionSize](#) size)

### Static Public Member Functions

- static int [recommendedImageSizeForDetection](#) ()  
*Returns the image size (one dimension).*

## 9.1025.1 Member Function Documentation

### 9.1025.1.1 detectFaces()

```
QList< QRect > Digikam::OpenCVDNNFaceDetector::detectFaces (
    const cv::Mat & inputImage,
    const cv::Size & paddedSize )
```

Effort on doing this should be questioned. TODO: Restructure and improve Face Detection module.

void OpenCVDNNFaceDetector::resizeBboxToStandardHumanFace(int& width, int& height) { Human head sizes data. [https://en.wikipedia.org/wiki/Human\\_head#Average\\_head\\_sizes](https://en.wikipedia.org/wiki/Human_head#Average_head_sizes)

```
float maxRatioFrontalFace    = 15.4 / 15.5;
float minRatioNonFrontalFace = 8.6  / 21.6;

float r = width*1.0/height, rReference;

if      ((r >= minRatioNonFrontalFace*0.9) && r <= (maxRatioFrontalFace * 1.1))
{
    rReference = r;
}
else if (r <= 0.25)
{
    rReference = r * 1.5;
}
else if (r >= 4)
{
    rReference = r / 1.5;
}
else if (r < minRatioNonFrontalFace * 0.9)
{
    rReference = minRatioNonFrontalFace;
}
else if (r > maxRatioFrontalFace * 1.1)
{
    rReference = maxRatioFrontalFace;
}

if (width > height)
{
    height = width / rReference;
}
else
{
    width = height * rReference;
}

}
```

### 9.1025.1.2 recommendedImageSizeForDetection()

```
int Digikam::OpenCVDNNFaceDetector::recommendedImageSizeForDetection ( ) [static]
```

recommended for face detection. If the image is considerably larger, it will be rescaled automatically.

## 9.1026 Digikam::OpenCVDNNFaceRecognizer Class Reference

### Public Types

- enum [Classifier](#) { [SVM](#) = 0 , [OpenCV\\_KNN](#) , [Tree](#) , [DB](#) }

### Public Member Functions

- **OpenCVDNNFaceRecognizer** ([Classifier](#) method, [FaceScanSettings::FaceRecognitionModel](#) recModel)  
*OpenCVDNNFaceRecognizer: Master class to control entire recognition using OpenFace algorithm.*
- void **clearTraining** (const QList< int > &idsToClear)  
*Clear specified trained data.*
- QVector< int > **recognize** (const QList< QPair< QImage \*, QString > > &inputImages)  
*Try to recognize a list of given images.*
- int **recognize** (const QPair< QImage \*, QString > &inputImage)  
*Try to recognize the given image.*
- bool **registerTrainingData** (const cv::Mat &preprocessedImage, int label)  
*register training data for unit test.*
- bool **remove** (const QString &hash)  
*Returns a cvMat of the extracted features from the cvinputImage, optimized for recognition.*
- void **setNbNeighbors** (int k)  
*Set K parameter of K-Nearest neighbors algorithm.*
- void **setThreshold** (int threshold)  
*Set maximum square distance of 2 vectors.*
- void **train** (const QList< QPair< QImage \*, QString > > &images, const int label)  
*Register faces corresponding to an identity.*
- int **verifyTestData** (const cv::Mat &preprocessedImage)  
*predict label of test data for unit test.*

### Static Public Member Functions

- static cv::Mat **prepareForRecognition** (const cv::Mat &cvinputImage)  
*Returns a cvMat created from the cvinputImage, optimized for recognition.*
- static cv::Mat **prepareForRecognition** (QImage &inputImage)  
*Returns a cvMat created from the inputImage, optimized for recognition.*

## 9.1026.1 Member Enumeration Documentation

### 9.1026.1.1 Classifier

```
enum Digikam::OpenCVDNNFaceRecognizer::Classifier
```

## Enumerator

SVM	Support Vector Machines ( <a href="https://docs.opencv.org/4.x/dc/dd6/ml_intro.html#ml_intro_svm">https://docs.opencv.org/4.x/dc/dd6/ml_intro.html#ml_intro_svm</a> )
OpenCV_KNN	K-Nearest Neighbors ( <a href="https://docs.opencv.org/4.x/dc/dd6/ml_intro.html#ml_intro_knn">https://docs.opencv.org/4.x/dc/dd6/ml_intro.html#ml_intro_knn</a> )
Tree	K-Nearest Neighbors Tree ( <a href="https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm">https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm</a> )
DB	Closest Neighbors Tree from the database.

## 9.1026.2 Member Function Documentation

### 9.1026.2.1 recognize() [1/2]

```
QVector< int > Digikam::OpenCVDNNFaceRecognizer::recognize (
    const QList< QPair< QImage *, QString > > & inputImages )
```

Returns a list of identity ids. If an identity cannot be recognized, returns -1.

### 9.1026.2.2 recognize() [2/2]

```
int Digikam::OpenCVDNNFaceRecognizer::recognize (
    const QPair< QImage *, QString > & inputImage )
```

Returns the identity id. If the identity cannot be recognized, returns -1. TODO: verify workflow to economize this routine.

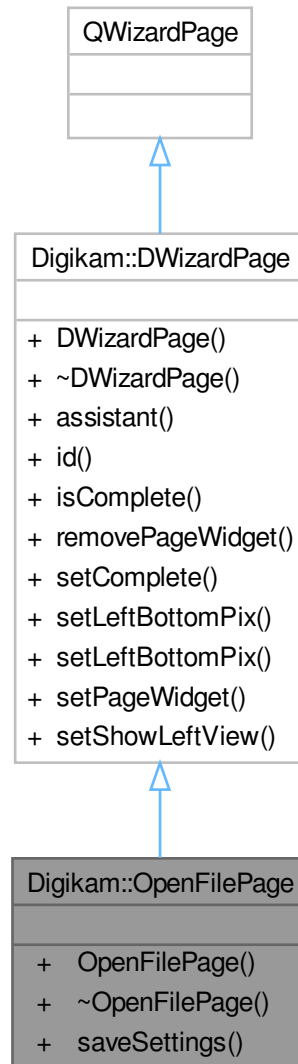
## 9.1027 Digikam::OpenfacePreprocessor Class Reference

### Public Member Functions

- bool **loadModels** ()  
*Load shapepredictor model for face alignment with 68 points of face landmark extraction.*
- cv::Mat **process** (const cv::Mat &image)

## 9.1028 Digikam::OpenFilePage Class Reference

Inheritance diagram for Digikam::OpenFilePage:



### Public Member Functions

- `OpenFilePage` (`QWizard *const dlg`)
- void `saveSettings` ()

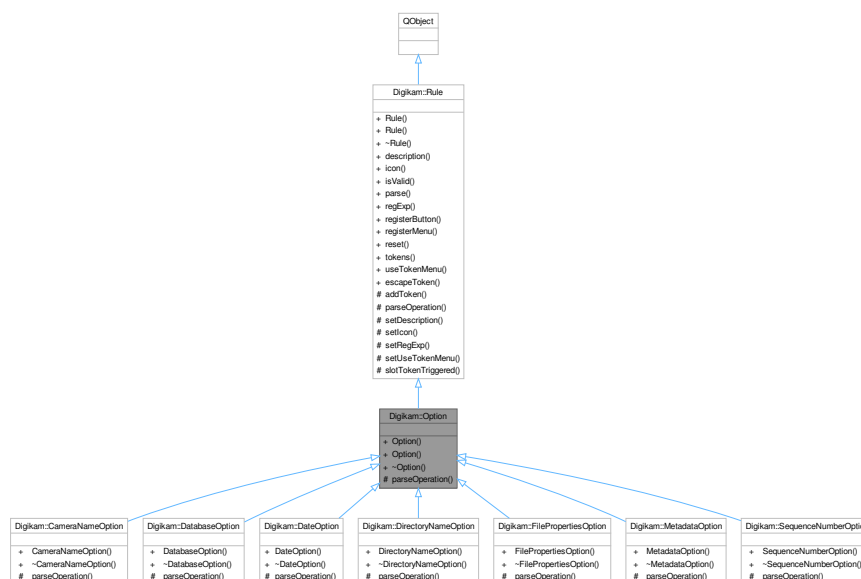
### Public Member Functions inherited from [Digikam::DWizardPage](#)

- `DWizardPage` (`QWizard *const dlg, const QString &title`)
- `QWizard * assistant` () const

- int **id** () const
- bool **isComplete** () const override
- void **removePageWidget** (QWidget \*const w)
- void **setComplete** (bool b)
- void **setLeftBottomPix** (const QIcon &icon)
- void **setLeftBottomPix** (const QPixmap &pix)
- void **setPageWidget** (QWidget \*const w)
- void **setShowLeftView** (bool v)

## 9.1029 Digikam::Option Class Reference

Inheritance diagram for Digikam::Option:



### Public Member Functions

- **Option** (const QString &name, const QString &description)
- **Option** (const QString &name, const QString &description, const QString &icon)

### Public Member Functions inherited from Digikam::Rule

- **Rule** (const QString &name)
- **Rule** (const QString &name, const QString &icon)
- QString **description** () const
- QPixmap **icon** (Rule::IconType type=Rule::Action) const
- bool **isValid** () const
- ParseResults **parse** (ParseSettings &settings)
- QRegularExpression & **regExp** () const

*Checks the validity of the parse object.*

*TODO: This is probably not needed anymore.*

- QPushButton \* [registerButton](#) (QWidget \*parent)  
*Register a button in the parent object.*
- QAction \* [registerMenu](#) (QMenu \*parent)  
*Register a menu action in the parent object.*
- virtual void [reset](#) ()  
*Resets the parser to its initial state.*
- TokenList & [tokens](#) () const
- bool [useTokenMenu](#) () const  
*Returns true if a token menu is used.*

### Protected Member Functions

- QString [parseOperation](#) (ParseSettings &settings, const QRegularExpressionMatch &match) override=0  
*TODO: describe me.*

### Protected Member Functions inherited from [Digikam::Rule](#)

- bool [addToken](#) (const QString &id, const QString &description, const QString &actionName=QString())  
*add a token to the parser, every parser should at least assign one token object*
- void [setDescription](#) (const QString &desc)
- void [setIcon](#) (const QString &pixmap)
- void [setRegExp](#) (const QRegularExpression &regExp)
- void [setUseTokenMenu](#) (bool value)  
*If multiple tokens have been assigned to a rule, a menu will be created.*

### Additional Inherited Members

### Public Types inherited from [Digikam::Rule](#)

- enum [IconType](#) { [Action](#) = 0 , [Dialog](#) }

### Signals inherited from [Digikam::Rule](#)

- void [signalTokenTriggered](#) (const QString &)

### Static Public Member Functions inherited from [Digikam::Rule](#)

- static QString [escapeToken](#) (const QString &token)  
*Escape the token characters to make them work in regular expressions.*

### Protected Slots inherited from [Digikam::Rule](#)

- virtual void [slotTokenTriggered](#) (const QString &)

## 9.1029.1 Member Function Documentation

### 9.1029.1.1 [parseOperation\(\)](#)

```
QString Digikam::Option::parseOperation (
    ParseSettings & settings,
    const QRegularExpressionMatch & match ) [override], [protected], [pure virtual]
```

**Parameters**

<i>settings</i>	contains settings
<i>match</i>	result of the regular expression match done in <code>Option::parse()</code>

**Returns**

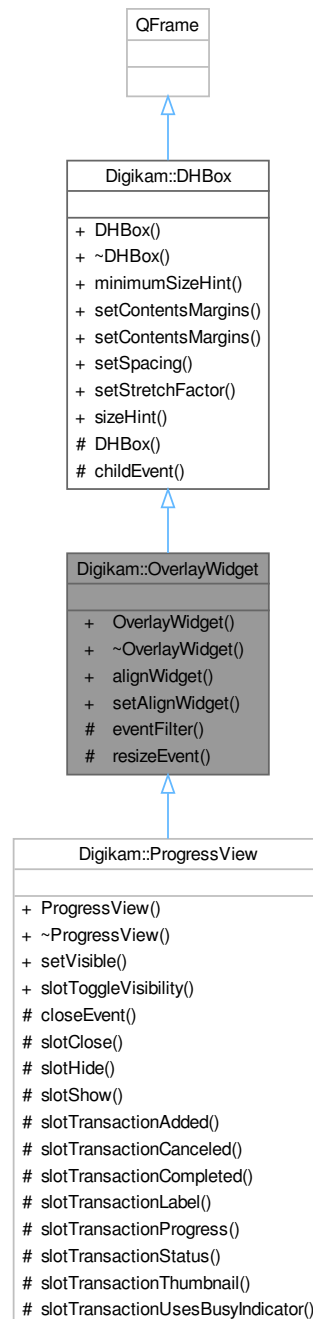
Implements [Digikam::Rule](#).

Implemented in [Digikam::CameraNameOption](#), [Digikam::DatabaseOption](#), [Digikam::DateOption](#), [Digikam::DirectoryNameOption](#), [Digikam::FilePropertiesOption](#), [Digikam::MetadataOption](#), and [Digikam::SequenceNumberOption](#).

## 9.1030 Digikam::OverlayWidget Class Reference

This is a widget that can align itself with another one, without using a layout, so that it can actually be on top of other widgets.

Inheritance diagram for Digikam::OverlayWidget:



## Public Member Functions

- **OverlayWidget** (QWidget \*const alignWidget, QWidget \*const parent, const QString &name=QString())
- QWidget \* **alignWidget** () const
- void **setAlignWidget** (QWidget \*const alignWidget)



## Public Member Functions inherited from Digikam::DHBox

- **DHBox** (QWidget \*const parent=nullptr)
- QSize **minimumSizeHint** () const override
- void **setContentsMargins** (const QMargins &margins)
- void **setContentsMargins** (int left, int top, int right, int bottom)
- void **setSpacing** (int space)
- void **setStretchFactor** (QWidget \*const widget, int stretch)
- QSize **sizeHint** () const override

## Protected Member Functions

- bool **eventFilter** (QObject \*o, QEvent \*e) override
- void **resizeEvent** (QResizeEvent \*ev) override

## Protected Member Functions inherited from Digikam::DHBox

- **DHBox** (bool vertical, QWidget \*const parent)
- void **childEvent** (QChildEvent \*e) override

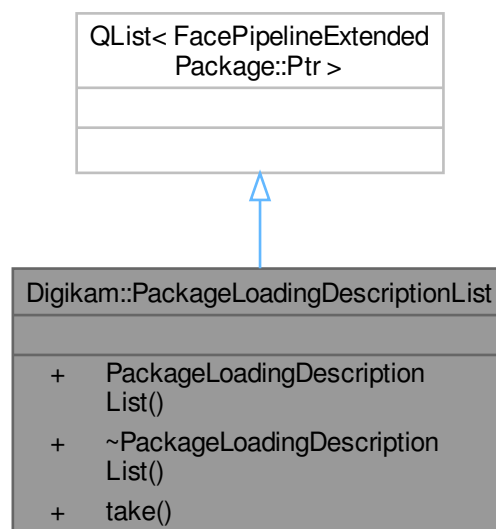
### 9.1030.1 Detailed Description

Currently the only supported type of alignment is "right aligned, on top of the other widget".

[OverlayWidget](#) inherits [DHBox](#) for convenience purposes (layout, and frame)

## 9.1031 Digikam::PackageLoadingDescriptionList Class Reference

Inheritance diagram for Digikam::PackageLoadingDescriptionList:



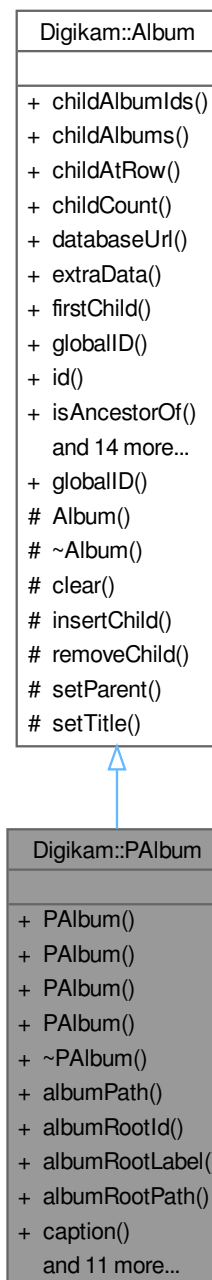
## Public Member Functions

- `FacePipelineExtendedPackage::Ptr take` (const [LoadingDescription](#) &description)

## 9.1032 Digikam::PAlbum Class Reference

A Physical [Album](#) representation.

Inheritance diagram for Digikam::PAlbum:



## Public Member Functions

- **PAlbum** (const QString &parentPath, int albumRoot)  
*Constructor for Trash album.*
- **PAlbum** (const QString &title)  
*Constructor for root album.*
- **PAlbum** (int albumRoot, const QString &label)  
*Constructor for album root albums.*
- **PAlbum** (int albumRoot, const QString &parentPath, const QString &title, int id)  
*Constructor for normal albums.*
- QString **albumPath** () const
- int **albumRootId** () const
- QString **albumRootLabel** () const
- QString **albumRootPath** () const
- QString **caption** () const
- QString **category** () const
- [CoreDbUrl databaseUrl](#) () const override
- QDate **date** () const
- QUrl **fileUrl** () const
- QString **folderPath** () const
- qlonglong **iconId** () const
- bool **isAlbumRoot** () const
- QString **prettyUrl** () const
- void **setCaption** (const QString &caption)
- void **setCategory** (const QString &category)
- void **setDate** (const QDate &date)

## Public Member Functions inherited from [Digikam::Album](#)

- QList< int > [childAlbumIds](#) (bool recursive=false)
- AlbumList [childAlbums](#) (bool recursive=false)
- [Album](#) \* [childAtRow](#) (int row) const
- int [childCount](#) () const
- void \* [extraData](#) (const void \*const key) const  
*Retrieve the associated extra data associated with key.*
- [Album](#) \* [firstChild](#) () const
- int [globalID](#) () const  
*An album ID is only unique among the set of all Albums of its Type.*
- int [id](#) () const  
*Each album has a ID uniquely identifying it in the set of Albums of a Type.*
- bool [isAncestorOf](#) ([Album](#) \*const album) const
- bool [isRoot](#) () const
- bool [isTrashAlbum](#) () const
- bool [isUsedByLabelsTree](#) () const
- [Album](#) \* [lastChild](#) () const
- [Album](#) \* [next](#) () const
- [Album](#) \* [parent](#) () const
- void **prepareForDeletion** ()  
*For secure deletion in an album model, call this function beforehand.*
- [Album](#) \* [prev](#) () const
- void [removeExtraData](#) (const void \*const key)  
*Remove the associated extra data associated with key.*

- int [rowFromAlbum](#) () const
- void [setExtraData](#) (const void \*const key, void \*const value)  
*This allows to associate some "extra" data to a [Album](#).*
- void [setUsedByLabelsTree](#) (bool isUsed)  
*Sets the property `m_usedByLabelsTree` to true if the search album was created using the Colors and labels tree view.*
- QString [title](#) () const
- [Type](#) [type](#) () const

## Friends

- class [AlbumManager](#)

## Additional Inherited Members

## Public Types inherited from [Digikam::Album](#)

- enum [Type](#) {  
    [PHYSICAL](#) = 0 , [TAG](#) , [DATE](#) , [SEARCH](#) ,  
    [FACE](#) }

## Static Public Member Functions inherited from [Digikam::Album](#)

- static int [globalID](#) ([Type](#) [type](#), int [id](#))  
*Produces the global id.*

## Protected Member Functions inherited from [Digikam::Album](#)

- [Album](#) ([Album::Type](#) [type](#), int [id](#), bool [root](#))  
*Constructor.*
- virtual [~Album](#) ()  
*Destructor.*
- void [clear](#) ()  
*Delete all child albums and also remove any associated extra data.*
- void [insertChild](#) ([Album](#) \*const [child](#))
- void [removeChild](#) ([Album](#) \*const [child](#))
- void [setParent](#) ([Album](#) \*const [parent](#))
- void [setTitle](#) (const QString &[title](#))

## 9.1032.1 Member Function Documentation

### 9.1032.1.1 [databaseUrl\(\)](#)

`CoreDbUrl` [Digikam::PAlbum::databaseUrl](#) ( ) const [override], [virtual]

#### Returns

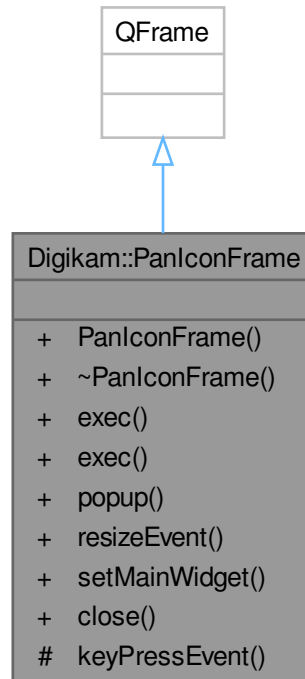
the kde url of the album

Implements [Digikam::Album](#).

## 9.1033 Digikam::PanIconFrame Class Reference

Frame with popup menu behavior to host [PanIconWidget](#).

Inheritance diagram for Digikam::PanIconFrame:



### Public Slots

- void `close` (int r)  
*Close the popup window.*

### Signals

- void `leaveModality` ()

### Public Member Functions

- **PanIconFrame** (QWidget \*const parent=nullptr)
- int **exec** (const QPoint &pos)  
*Execute the popup window.*
- int **exec** (int x, int y)  
*Execute the popup window.*
- void **popup** (const QPoint &pos)  
*Open the popup window at position pos.*
- void **resizeEvent** (QResizeEvent \*resize) override  
*The resize event.*
- void **setMainWidget** (QWidget \*const main)  
*Set the main widget.*

## Protected Member Functions

- void **keyPressEvent** (QKeyEvent \*e) override  
*Catch key press events.*

## Friends

- class **Private**

## 9.1033.1 Member Function Documentation

### 9.1033.1.1 close

```
void Digikam::PanIconFrame::close (
    int r ) [slot]
```

This is called from the main widget, usually. `r` is the result returned from [exec\(\)](#).

### 9.1033.1.2 resizeEvent()

```
void Digikam::PanIconFrame::resizeEvent (
    QResizeEvent * resize ) [override]
```

Simply resizes the main widget to the whole widgets client size.

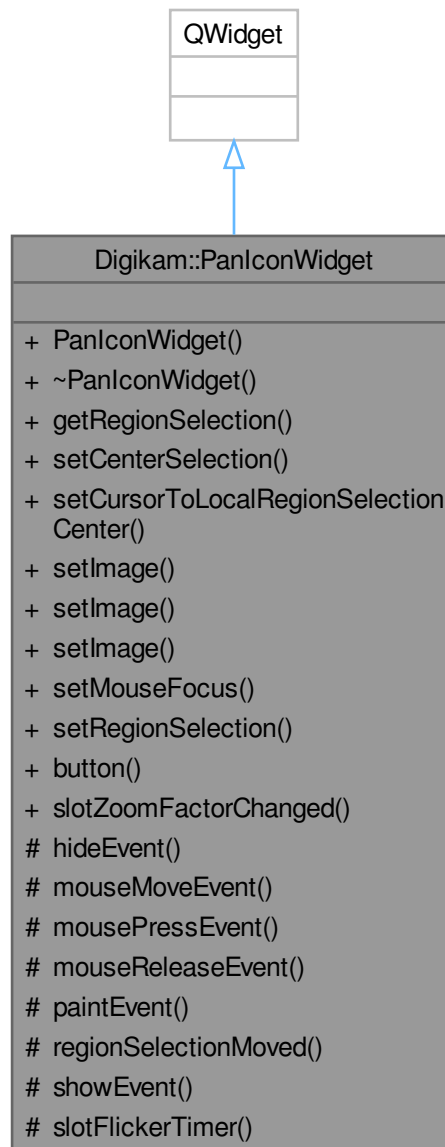
### 9.1033.1.3 setMainWidget()

```
void Digikam::PanIconFrame::setMainWidget (
    QWidget *const main )
```

You cannot set the main widget from the constructor, since it must be a child of the frame itself. Be careful: the size is set to the main widgets size. It is up to you to set the main widgets correct size before setting it as the main widget.

## 9.1034 Digikam::PanIconWidget Class Reference

Inheritance diagram for Digikam::PanIconWidget:



### Public Slots

- void **slotZoomFactorChanged** (double)

### Signals

- void **signalHidden** ()
- void **signalSelectionMoved** (const QRect &rect, bool targetDone)  
*Emitted when selection have been moved with mouse.*
- void **signalSelectionTakeFocus** ()

## Public Member Functions

- **PanIconWidget** (QWidget \*const parent=nullptr)
- QRect **getRegionSelection** () const
- void **setCenterSelection** ()
- void **setCursorToLocalRegionSelectionCenter** ()
- void **setImage** (const QImage &scaledPreviewImage, const QSize &fullImageSize)
- void **setImage** (int previewWidth, int previewHeight, const QImage &fullOriginalImage)
- void **setImage** (int previewWidth, int previewHeight, const QImage &fullOriginalImage)
- void **setMouseFocus** ()
- void **setRegionSelection** (const QRect &regionSelection)

## Static Public Member Functions

- static QPushButton \* **button** ()

## Protected Slots

- void **slotFlickerTimer** ()

## Protected Member Functions

- void **hideEvent** (QHideEvent \*) override
- void **mouseMoveEvent** (QMouseEvent \*) override
- void **mousePressEvent** (QMouseEvent \*) override
- void **mouseReleaseEvent** (QMouseEvent \*) override
- void **paintEvent** (QPaintEvent \*) override
- void **regionSelectionMoved** (bool targetDone)
  - Recalculate the target selection position and emit 'signalSelectionMoved'.*
- void **showEvent** (QShowEvent \*) override

## 9.1034.1 Member Function Documentation

### 9.1034.1.1 signalSelectionMoved

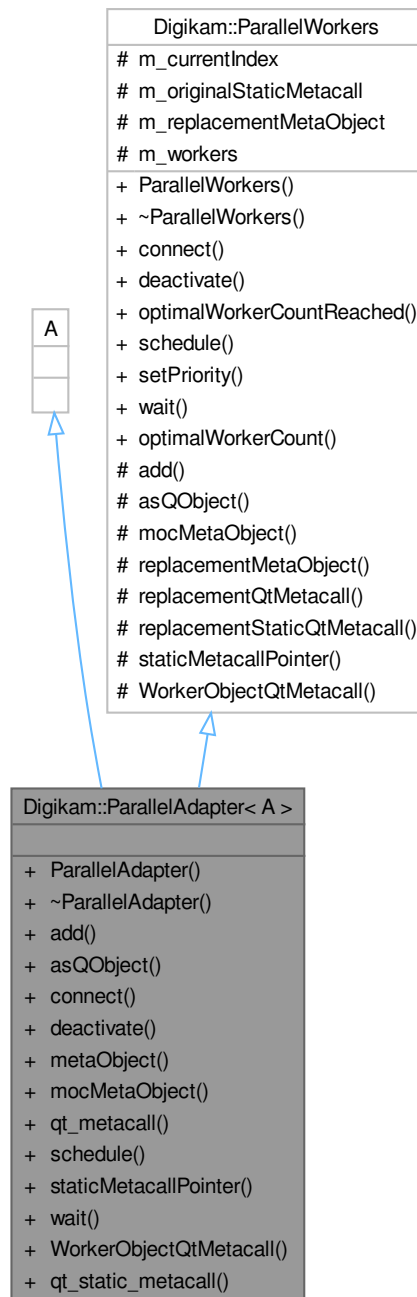
```
void Digikam::PanIconWidget::signalSelectionMoved (
    const QRect & rect,
    bool targetDone ) [signal]
```

'targetDone' boolean value is used for indicate if the mouse have been released.



## 9.1035 Digikam::ParallelAdapter< A > Class Template Reference

Inheritance diagram for Digikam::ParallelAdapter< A >:



### Public Member Functions

- [ParallelAdapter](#) ()=default

*Instead of using a single [WorkerObject](#), create a [ParallelAdapter](#) for your worker object subclass, and `add()` individual [WorkerObjects](#).*

- void **add** (A \*const worker)
- QObject \* **asQObject** () override  
*Return the target QObject (double inheritance)*
- bool **connect** (const char \*signal, const QObject \*receiver, const char \*method, Qt::ConnectionType type=Qt::AutoConnection) const
- void **deactivate** ([WorkerObject::DeactivatingMode](#) mode=[WorkerObject::FlushSignals](#))
- const QMetaObject \* **metaObject** () const override
- const QMetaObject \* **mocMetaObject** () const override  
*The moc-generated metaObject of the target object.*
- int **qt\_metacall** (QMetaObject::Call \_c, int \_id, void \*\*\_a) override
- void **schedule** ()
- StaticMetacallFunction **staticMetacallPointer** () override
- void **wait** ()
- int **WorkerObjectQtMetacall** (QMetaObject::Call \_c, int \_id, void \*\*\_a) override  
*The qt\_metacall of [WorkerObject](#), one level above the target QObject.*

## Public Member Functions inherited from [Digikam::ParallelWorkers](#)

- [ParallelWorkers](#) ()=default  
*[ParallelWorkers](#) is a helper class to distribute work over several identical workers objects.*
- bool **connect** (const char \*signal, const QObject \*receiver, const char \*method, Qt::ConnectionType type=Qt::AutoConnection) const  
*Connects signals outbound from all workers to a given receiver.*
- void **deactivate** ([WorkerObject::DeactivatingMode](#) mode=[WorkerObject::FlushSignals](#))
- bool **optimalWorkerCountReached** () const  
*Returns true if the current number of added workers has reached the [optimalWorkerCount\(\)](#)*
- void **schedule** ()  
*The corresponding methods of all added worker objects will be called.*
- void **setPriority** (QThread::Priority priority)
- void **wait** ()

## Static Public Member Functions

- static void **qt\_static\_metacall** (QObject \*o, QMetaObject::Call \_c, int \_id, void \*\*\_a)

## Static Public Member Functions inherited from [Digikam::ParallelWorkers](#)

- static int **optimalWorkerCount** ()  
*Regarding the number of logical CPUs on the current machine, returns the optimal count of concurrent workers.*

## Additional Inherited Members

## Protected Types inherited from [Digikam::ParallelWorkers](#)

- typedef void(\* **StaticMetacallFunction**) (QObject \*, QMetaObject::Call, int, void \*\*)

## Protected Member Functions inherited from [Digikam::ParallelWorkers](#)

- void **add** ([WorkerObject](#) \*const worker)
- const QMetaObject \* **replacementMetaObject** () const
- int **replacementQtMetacall** (QMetaObject::Call \_c, int \_id, void \*\*\_a)  
*Replaces slot call distribution of the target QObject.*
- int **replacementStaticQtMetacall** (QMetaObject::Call \_c, int \_id, void \*\*\_a)

## Protected Attributes inherited from [Digikam::ParallelWorkers](#)

- int **m\_currentIndex** = 0
- StaticMetacallFunction **m\_originalStaticMetacall** = nullptr
- QMetaObject \* **m\_replacementMetaObject** = nullptr
- QList< [WorkerObject](#) \* > **m\_workers**

### 9.1035.1 Constructor & Destructor Documentation

#### 9.1035.1.1 ParallelAdapter()

```
template<class A >
Digikam::ParallelAdapter< A >::ParallelAdapter ( ) [default]
```

The load will be evenly distributed. Note: unlike with [WorkerObject](#) directly, there is no need to call `schedule()`. For inbound connections (signals connected to a [WorkerObject](#)'s slot, to be processed, use a `Qt::DirectConnection` on the adapter. For outbound connections (signals emitted from the [WorkerObject](#)), use [ParallelAdapter](#)'s `connect` to have a connection from all added [WorkerObjects](#).

### 9.1035.2 Member Function Documentation

#### 9.1035.2.1 asQObject()

```
template<class A >
QObject * Digikam::ParallelAdapter< A >::asQObject ( ) [inline], [override], [virtual]
```

Implements [Digikam::ParallelWorkers](#).

#### 9.1035.2.2 mocMetaObject()

```
template<class A >
const QMetaObject * Digikam::ParallelAdapter< A >::mocMetaObject ( ) const [inline], [override], [virtual]
```

Implements [Digikam::ParallelWorkers](#).

#### 9.1035.2.3 staticMetacallPointer()

```
template<class A >
StaticMetacallFunction Digikam::ParallelAdapter< A >::staticMetacallPointer ( ) [inline], [override], [virtual]
```

Implements [Digikam::ParallelWorkers](#).

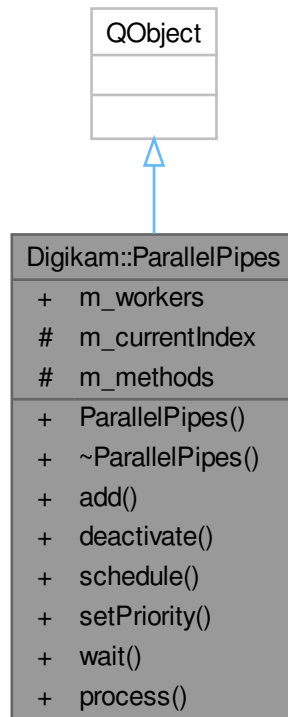
### 9.1035.2.4 WorkerObjectQtMetacall()

```
template<class A >
int Digikam::ParallelAdapter< A >::WorkerObjectQtMetacall (
    QMetaObject::Call _c,
    int _id,
    void ** _a ) [inline], [override], [virtual]
```

Implements [Digikam::ParallelWorkers](#).

## 9.1036 Digikam::ParallelPipes Class Reference

Inheritance diagram for Digikam::ParallelPipes:



### Public Slots

- void **process** (const FacePipelineExtendedPackage::Ptr &package)

### Signals

- void **processed** (const FacePipelineExtendedPackage::Ptr &package)

### Public Member Functions

- void **add** ([WorkerObject](#) \*const worker)
- void **deactivate** ([WorkerObject::DeactivatingMode](#) mode=[WorkerObject::FlushSignals](#))
- void **schedule** ()
- void **setPriority** (QThread::Priority priority)
- void **wait** ()

### Public Attributes

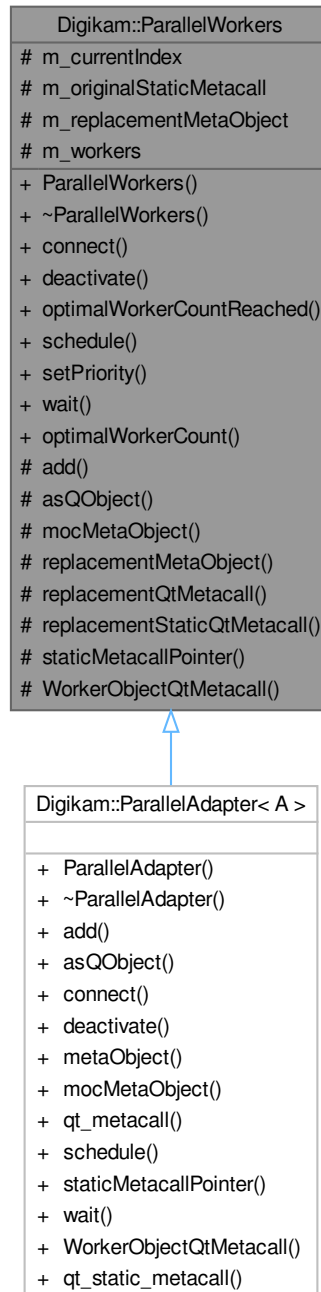
- QList< [WorkerObject](#) \* > **m\_workers**

### Protected Attributes

- int **m\_currentIndex** = 0
- QList< QMetaMethod > **m\_methods**

## 9.1037 Digikam::ParallelWorkers Class Reference

Inheritance diagram for Digikam::ParallelWorkers:



### Public Member Functions

- [ParallelWorkers](#) ()=default

*ParallelWorkers* is a helper class to distribute work over several identical workers objects.

- bool **connect** (const char \*signal, const QObject \*receiver, const char \*method, Qt::ConnectionType type=Qt::AutoConnection) const  
*Connects signals outbound from all workers to a given receiver.*
- void **deactivate** ([WorkerObject::DeactivatingMode](#) mode=[WorkerObject::FlushSignals](#))
- bool **optimalWorkerCountReached** () const  
*Returns true if the current number of added workers has reached the [optimalWorkerCount\(\)](#)*
- void **schedule** ()  
*The corresponding methods of all added worker objects will be called.*
- void **setPriority** (QThread::Priority priority)
- void **wait** ()

### Static Public Member Functions

- static int **optimalWorkerCount** ()  
*Regarding the number of logical CPUs on the current machine, returns the optimal count of concurrent workers.*

### Protected Types

- typedef void(\* **StaticMetacallFunction**) (QObject \*, QMetaObject::Call, int, void \*\*)

### Protected Member Functions

- void **add** ([WorkerObject](#) \*const worker)
- virtual QObject \* **asQObject** ()=0  
*Return the target QObject (double inheritance)*
- virtual const QMetaObject \* **mocMetaObject** () const =0  
*The moc-generated metaObject of the target object.*
- const QMetaObject \* **replacementMetaObject** () const
- int **replacementQtMetacall** (QMetaObject::Call \_c, int \_id, void \*\*\_a)  
*Replaces slot call distribution of the target QObject.*
- int **replacementStaticQtMetacall** (QMetaObject::Call \_c, int \_id, void \*\*\_a)
- virtual StaticMetacallFunction **staticMetacallPointer** ()=0
- virtual int **WorkerObjectQtMetacall** (QMetaObject::Call \_c, int \_id, void \*\*\_a)=0  
*The qt\_metacall of [WorkerObject](#), one level above the target QObject.*

### Protected Attributes

- int **m\_currentIndex** = 0
- StaticMetacallFunction **m\_originalStaticMetacall** = nullptr
- QMetaObject \* **m\_replacementMetaObject** = nullptr
- QList< [WorkerObject](#) \* > **m\_workers**

## 9.1037.1 Constructor & Destructor Documentation

### 9.1037.1.1 ParallelWorkers()

Digikam::ParallelWorkers::ParallelWorkers ( ) [default]

See [ParallelAdapter](#) for guidance how to use it.

## 9.1037.2 Member Function Documentation

### 9.1037.2.1 asQObject()

```
virtual QObject * Digikam::ParallelWorkers::asQObject ( ) [protected], [pure virtual]
```

Implemented in [Digikam::ParallelAdapter< A >](#).

### 9.1037.2.2 mocMetaObject()

```
virtual const QMetaObject * Digikam::ParallelWorkers::mocMetaObject ( ) const [protected],  
[pure virtual]
```

Implemented in [Digikam::ParallelAdapter< A >](#).

### 9.1037.2.3 WorkerObjectQtMetacall()

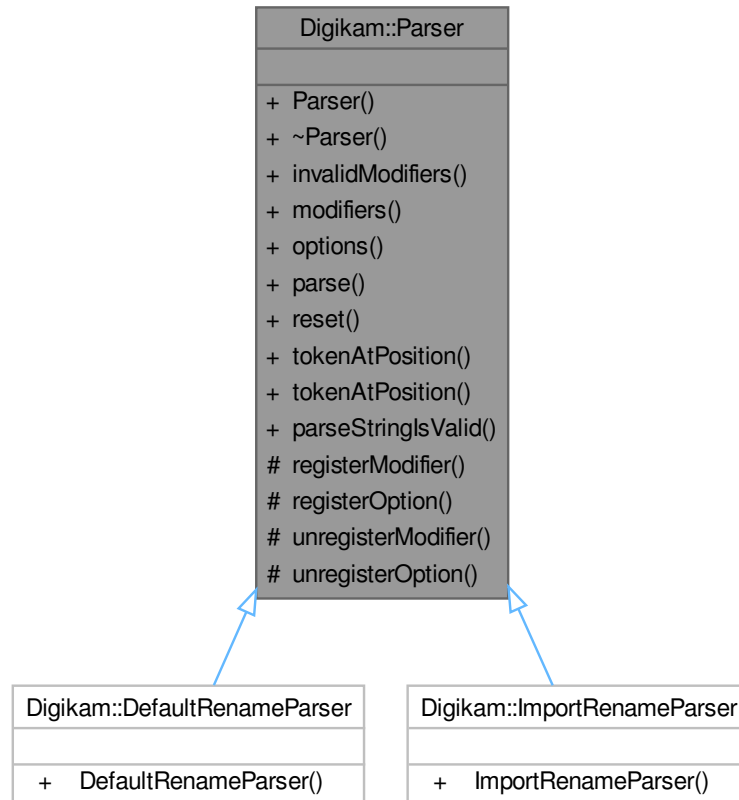
```
virtual int Digikam::ParallelWorkers::WorkerObjectQtMetacall (   
    QMetaObject::Call _c,  
    int _id,  
    void ** _a ) [protected], [pure virtual]
```

Implemented in [Digikam::ParallelAdapter< A >](#).



## 9.1038 Digikam::Parser Class Reference

Inheritance diagram for Digikam::Parser:



### Public Member Functions

- `ParseResults` **invalidModifiers** (`ParseSettings` &settings)
- `RulesList` **modifiers** () const
- `RulesList` **options** () const
- `QString` **parse** (`ParseSettings` &settings)
- void **reset** ()
- bool **tokenAtPosition** (`ParseSettings` &settings, int pos)
- bool **tokenAtPosition** (`ParseSettings` &settings, int pos, int &start, int &length)

### Static Public Member Functions

- static bool **parseStringIsValid** (const `QString` &str)  
*check if the given parse string is valid*

## Protected Member Functions

- void **registerModifier** ([Rule](#) \*modifier)
- void **registerOption** ([Rule](#) \*option)
- void **unregisterModifier** (const [Rule](#) \*modifier)
- void **unregisterOption** (const [Rule](#) \*option)

## 9.1038.1 Member Function Documentation

### 9.1038.1.1 parseStringIsValid()

```
bool Digikam::Parser::parseStringIsValid (
    const QString & str ) [static]
```

#### Parameters

<i>str</i>	the parse string
------------	------------------

#### Returns

true if valid / can be parsed

## 9.1039 Digikam::ParseResults Class Reference

### Public Types

- typedef QPair< int, int > **ResultsKey**
- typedef QMap< ResultsKey, ResultsValue > **ResultsMap**
- typedef QPair< QString, QString > **ResultsValue**

### Public Member Functions

- void **addEntry** (const ResultsKey &key, const ResultsValue &value)
- void **append** (const [ParseResults](#) &results)
- void **clear** ()
- void **debug** () const
- void **deleteEntry** (const ResultsKey &key)
- bool **hasKey** (const ResultsKey &key)
- bool **hasKeyAtApproximatePosition** (int pos) const
- bool **hasKeyAtPosition** (int pos) const
- bool **isEmpty** () const
- ResultsKey **keyAtApproximatePosition** (int pos) const
- ResultsKey **keyAtPosition** (int pos) const
- QList< ResultsKey > **keys** () const
- int **offset** (const ResultsKey &key) const
- QString **replaceTokens** (const QString &markedString) const
- QString **result** (const ResultsKey &key) const
- QString **resultValuesAsString** () const
- QString **token** (const ResultsKey &key) const
- QList< ResultsValue > **values** () const

## 9.1040 Digikam::ParseSettings Class Reference

### Public Member Functions

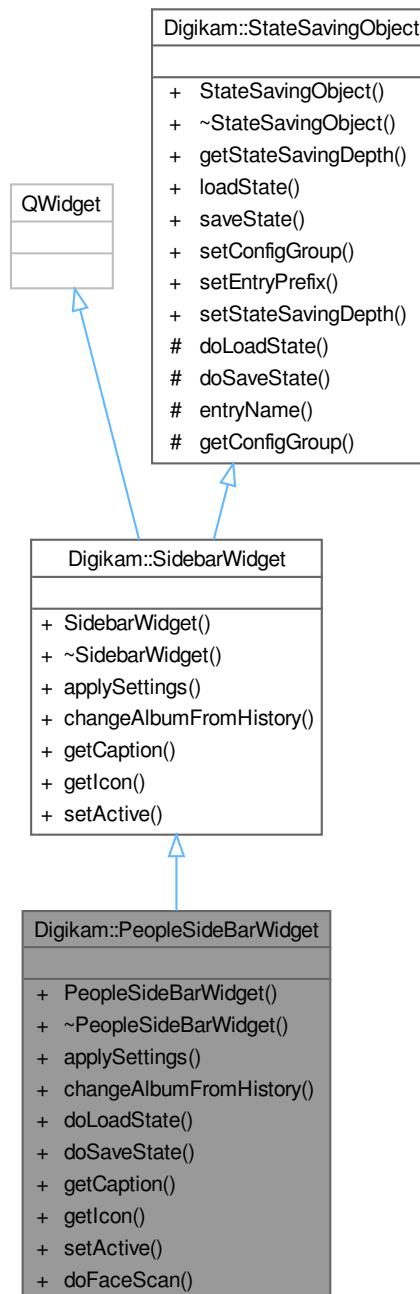
- **ParseSettings** ()  
*default constructor*
- **ParseSettings** (const [ItemInfo](#) &info)  
*ItemInfo constructor.*
- **ParseSettings** (const QString &\_parseString)
- **ParseSettings** (const QString &\_parseString, const [ItemInfo](#) &info)
- bool **isValid** () const

### Public Attributes

- QDateTime **creationTime**
- ParseResults::ResultsKey **currentResultsKey**
- int **cutFileName** = 0
- QUrl **fileUrl**
- [ParseResults](#) **invalidModifiers**
- [AdvancedRenameManager](#) \* **manager** = nullptr
- QString **parseString**
- [ParseResults](#) **results**
- int **startIndex** = 1
- QString **str2Modify**
- bool **useOriginalFileExtension** = true

## 9.1041 Digikam::PeopleSideBarWidget Class Reference

Inheritance diagram for Digikam::PeopleSideBarWidget:



### Signals

- void **requestFaceMode** (bool on)
- void **signalFindDuplicates** (const QList< [TAlbum](#) \* > &albums)

## Signals inherited from [Digikam::SidebarWidget](#)

- void **requestActiveTab** ([SidebarWidget](#) \*)  
*This signal can be emitted if this sidebar widget wants to be the one that is active.*
- void **signalNotificationError** (const QString &message, int type)  
*To dispatch error message to temporized pop-up notification widget hosted with icon-view.*

## Public Member Functions

- **PeopleSideBarWidget** (QWidget \*const parent, [TagModel](#) \*const tagModel, [SearchModificationHelper](#) \*const searchModificationHelper)
- void **applySettings** () override  
*This method is invoked when the application settings should be (re-) applied to this widget.*
- void **changeAlbumFromHistory** (const QList< [Album](#) \* > &album) override  
*This is called on this widget when the history requires to move back to the specified album.*
- void **doLoadState** () override  
*Implement this hook method for state loading.*
- void **doSaveState** () override  
*Implement this hook method for state saving.*
- const QString **getCaption** () override  
*Must be implemented to return the title of this sidebar's tab.*
- const QIcon **getIcon** () override  
*Must be implemented and return the icon that shall be visible for this sidebar widget.*
- void **setActive** (bool active) override  
*This method is called if the visible sidebar widget is changed.*

## Public Member Functions inherited from [Digikam::SidebarWidget](#)

- [SidebarWidget](#) (QWidget \*const parent)  
*Constructor.*
- **~SidebarWidget** () override=default  
*Destructor.*

## Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (QObject \*const host)  
*Constructor.*
- virtual **~StateSavingObject** ()  
*Destructor.*
- [StateSavingDepth](#) **getStateSavingDepth** () const  
*Returns the depth used for state saving or loading.*
- void **loadState** ()  
*Invokes loading the class' state.*
- void **saveState** ()  
*Invokes saving the class' state.*
- virtual void **setConfigGroup** (const KConfigGroup &group)  
*Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void **setEntryPrefix** (const QString &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void **setStateSavingDepth** (const [StateSavingDepth](#) depth)  
*Sets the depth used for state saving or loading.*

## Static Public Member Functions

- static void **doFaceScan** (const [FaceScanSettings](#) &faceScanSettings)

## Additional Inherited Members

## Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }

*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

## Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- QString [entryName](#) (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup [getConfigGroup](#) () const  
*Returns the config group that must be used for state saving and loading.*

## 9.1041.1 Member Function Documentation

### 9.1041.1.1 [applySettings\(\)](#)

```
void Digikam::PeopleSideBarWidget::applySettings ( ) [override], [virtual]
```

Implements [Digikam::SidebarWidget](#).

### 9.1041.1.2 [changeAlbumFromHistory\(\)](#)

```
void Digikam::PeopleSideBarWidget::changeAlbumFromHistory (
    const QList< Album * > & album ) [override], [virtual]
```

Implements [Digikam::SidebarWidget](#).

### 9.1041.1.3 [doLoadState\(\)](#)

```
void Digikam::PeopleSideBarWidget::doLoadState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

### 9.1041.1.4 [doSaveState\(\)](#)

```
void Digikam::PeopleSideBarWidget::doSaveState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

### 9.1041.1.5 `getCaption()`

```
const QString Digikam::PeopleSideBarWidget::getCaption ( ) [override], [virtual]
```

#### Returns

localized title string

Implements [Digikam::SidebarWidget](#).

### 9.1041.1.6 `getIcon()`

```
const QIcon Digikam::PeopleSideBarWidget::getIcon ( ) [override], [virtual]
```

#### Returns

pixmap icon

Implements [Digikam::SidebarWidget](#).

### 9.1041.1.7 `setActive()`

```
void Digikam::PeopleSideBarWidget::setActive (
    bool active ) [override], [virtual]
```

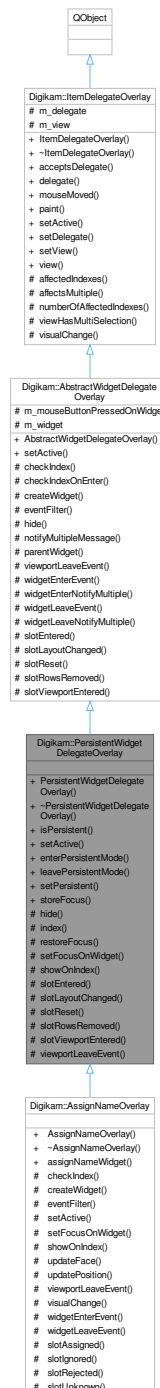
#### Parameters

<i>active</i>	if true, this widget is the new active widget, if false another widget is active
---------------	--

Implements [Digikam::SidebarWidget](#).

## 9.1042 Digikam::PersistentWidgetDelegateOverlay Class Reference

Inheritance diagram for Digikam::PersistentWidgetDelegateOverlay:



### Public Slots

- void **enterPersistentMode** ()
- void **leavePersistentMode** ()



- void **setPersistent** (bool persistent)  
*Enters persistent mode.*
- void **storeFocus** ()

### Public Member Functions

- **PersistentWidgetDelegateOverlay** (QObject \*const parent)  
*This class offers additional / modified behavior: When a "persistent" mode is entered, it will not move by mouse hover, but stay and only move on mouse click.*
- bool **isPersistent** () const
- void **setActive** (bool active) override  
*If active is true, this will call [createWidget\(\)](#), initialize the widget for use, and setup connections for the virtual slots.*

### Public Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- **AbstractWidgetDelegateOverlay** (QObject \*const parent)  
*This class provides functionality for using a widget in an overlay.*

### Public Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- **ItemDelegateOverlay** (QObject \*const parent=nullptr)
- virtual bool **acceptsDelegate** (QAbstractItemDelegate \*) const
- QAbstractItemDelegate \* **delegate** () const
- virtual void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)  
*Only these two methods are implemented as virtual methods.*
- virtual void **paint** (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index)
- void **setDelegate** (QAbstractItemDelegate \*delegate)
- void **setView** (QAbstractItemView \*view)
- QAbstractItemView \* **view** () const

### Protected Member Functions

- void **hide** () override  
*Called when the widget shall be hidden (mouse cursor left index, viewport, uninstalled etc.).*
- QModelIndex **index** () const
- void **restoreFocus** ()
- virtual void **setFocusOnWidget** ()  
*Reimplement to set the focus on the correct subwidget.*
- virtual void **showOnIndex** (const QModelIndex &index)  
*see [slotEntered\(\)](#)*
- void **slotEntered** (const QModelIndex &index) override  
*Most overlays reimplement this slot to get the starting point for repositioning a widget etc.*
- void **slotLayoutChanged** () override
- void **slotReset** () override  
*Default implementations of these three slots call [hide\(\)](#)*
- void **slotRowsRemoved** (const QModelIndex &parent, int start, int end) override
- void **slotViewportEntered** () override
- void **viewportLeaveEvent** (QObject \*obj, QEvent \*event) override  
*Called when a [QEvent::Leave](#) of the viewport is received.*

## Protected Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- virtual bool [checkIndex](#) (const QModelIndex &index) const  
*Return true here if you want to show the overlay for the given index.*
- bool [checkIndexOnEnter](#) (const QModelIndex &index) const  
*Utility method called from slotEntered.*
- virtual QWidget \* [createWidget](#) ()=0  
*Create your widget here.*
- bool [eventFilter](#) (QObject \*obj, QEvent \*event) override
- virtual QString [notifyMultipleMessage](#) (const QModelIndex &, int number)
- QWidget \* [parentWidget](#) () const  
*Returns the widget to be used as parent for your widget created in [createWidget\(\)](#)*
- virtual void [widgetEnterEvent](#) ()  
*Called when a QEvent::Enter resp.*
- void [widgetEnterNotifyMultiple](#) (const QModelIndex &index)  
*A sample implementation for above methods.*
- virtual void [widgetLeaveEvent](#) ()
- void [widgetLeaveNotifyMultiple](#) ()

## Protected Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- QList< QModelIndex > [affectedIndexes](#) (const QModelIndex &index) const
- bool [affectsMultiple](#) (const QModelIndex &index) const  
*For the context that an overlay can affect multiple items: Assuming the currently overlaid index is given.*
- int [numberOfAffectedIndexes](#) (const QModelIndex &index) const
- bool [viewHasMultiSelection](#) () const  
*Utility method.*

## Additional Inherited Members

## Signals inherited from [Digikam::ItemDelegateOverlay](#)

- void [hideNotification](#) ()
- void [requestNotification](#) (const QModelIndex &index, const QString &message)
- void [update](#) (const QModelIndex &index)

## Protected Slots inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

## Protected Slots inherited from [Digikam::ItemDelegateOverlay](#)

- virtual void [visualChange](#) ()  
*Called when any change from the delegate occurs - when the overlay is installed, when size hints, styles or fonts change.*

## Protected Attributes inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- bool [m\\_mouseButtonPressedOnWidget](#) = false
- QWidget \* [m\\_widget](#) = nullptr

## Protected Attributes inherited from [Digikam::ItemDelegateOverlay](#)

- `QAbstractItemDelegate * m_delegate = nullptr`
- `QAbstractItemView * m_view = nullptr`

### 9.1042.1 Constructor & Destructor Documentation

#### 9.1042.1.1 PersistentWidgetDelegateOverlay()

```
Digikam::PersistentWidgetDelegateOverlay::PersistentWidgetDelegateOverlay (
    QObject *const parent ) [explicit]
```

If the overlay widget had focus, it will be restored on show.

### 9.1042.2 Member Function Documentation

#### 9.1042.2.1 hide()

```
void Digikam::PersistentWidgetDelegateOverlay::hide ( ) [override], [protected], [virtual]
```

Default implementation [hide\(\)](#)s `m_widget`.

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

#### 9.1042.2.2 setActive()

```
void Digikam::PersistentWidgetDelegateOverlay::setActive (
    bool active ) [override], [virtual]
```

If active is false, this will delete the widget and disconnect all signal from model and view to this object (!)

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

#### 9.1042.2.3 setFocusOnWidget()

```
void Digikam::PersistentWidgetDelegateOverlay::setFocusOnWidget ( ) [protected], [virtual]
```

Default implementation sets focus on widget()

Reimplemented in [Digikam::AssignNameOverlay](#).

#### 9.1042.2.4 setPersistent

```
void Digikam::PersistentWidgetDelegateOverlay::setPersistent (
    bool persistent ) [slot]
```

The overlay is moved because of mouse hover.

#### 9.1042.2.5 showOnIndex()

```
void Digikam::PersistentWidgetDelegateOverlay::showOnIndex (
    const QModelIndex & index ) [protected], [virtual]
```

Reimplemented in [Digikam::AssignNameOverlay](#).

#### 9.1042.2.6 slotEntered()

```
void Digikam::PersistentWidgetDelegateOverlay::slotEntered (
    const QModelIndex & index ) [override], [protected], [virtual]
```

This class instead provides [showOnIndex\(\)](#) which you shall use for this purpose.

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

#### 9.1042.2.7 slotLayoutChanged()

```
void Digikam::PersistentWidgetDelegateOverlay::slotLayoutChanged ( ) [override], [protected],
[virtual]
```

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

#### 9.1042.2.8 slotReset()

```
void Digikam::PersistentWidgetDelegateOverlay::slotReset ( ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

#### 9.1042.2.9 slotRowsRemoved()

```
void Digikam::PersistentWidgetDelegateOverlay::slotRowsRemoved (
    const QModelIndex & parent,
    int start,
    int end ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

#### 9.1042.2.10 slotViewportEntered()

```
void Digikam::PersistentWidgetDelegateOverlay::slotViewportEntered ( ) [override], [protected],
[virtual]
```

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.1042.2.11 viewportLeaveEvent()

```
void Digikam::PersistentWidgetDelegateOverlay::viewportLeaveEvent (
    QObject * obj,
    QEvent * event ) [override], [protected], [virtual]
```

The default implementation [hide\(\)](#)s.

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

## 9.1043 Digikam::PhotoInfoContainer Class Reference

### Public Member Functions

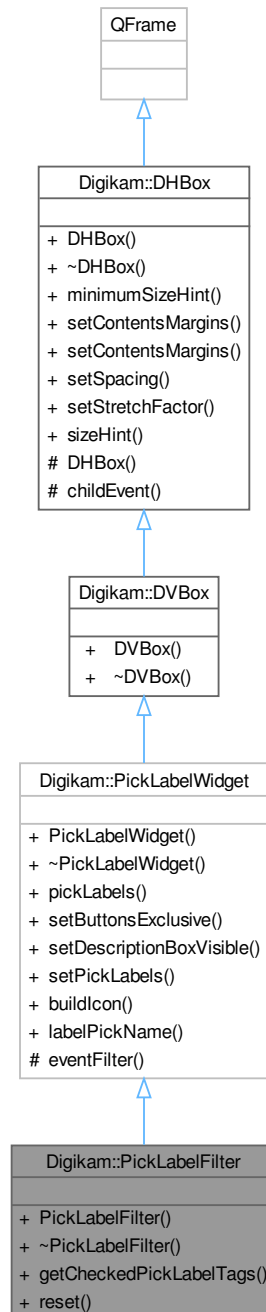
- **PhotoInfoContainer** (const [PhotoInfoContainer](#) &)=default
- bool **isEmpty** () const
- bool **isNull** () const
- [PhotoInfoContainer](#) & **operator=** (const [PhotoInfoContainer](#) &)=default
- [PhotoInfoContainer](#) & **operator=** ([PhotoInfoContainer](#) &&)=default
- bool **operator==** (const [PhotoInfoContainer](#) &t) const

### Public Attributes

- QString **aperture**
- QDateTime **dateTime**
- QString **exposureMode**
- QString **exposureProgram**
- QString **exposureTime**
- QString **flash**
- QString **focalLength**
- QString **focalLength35mm**
- bool **hasCoordinates** = false  
*true if GPS info are present*
- QString **lens**
- QString **make**
- QString **model**
- QString **sensitivity**
- QString **whiteBalance**

## 9.1044 Digikam::PickLabelFilter Class Reference

Inheritance diagram for Digikam::PickLabelFilter:



### Signals

- void **signalPickLabelSelectionChanged** (const QList< PickLabel > &)

## Signals inherited from [Digikam::PickLabelWidget](#)

- void **signalPickLabelChanged** (int)

## Public Member Functions

- **PickLabelFilter** (QWidget \*const parent=nullptr)
- QList< [TAlbum](#) \* > **getCheckedPickLabelTags** ()
- void **reset** ()

## Public Member Functions inherited from [Digikam::PickLabelWidget](#)

- **PickLabelWidget** (QWidget \*const parent=nullptr)
- QList< PickLabel > **pickLabels** () const  
*Return the list of Pick Label buttons turned on or an empty list of none.*
- void **setButtonsExclusive** (bool b)  
*Set all Color Label buttons exclusive or not.*
- void **setDescriptionBoxVisible** (bool b)  
*Show or not on the bottom view the description of label with shortcuts.*
- void **setPickLabels** (const QList< PickLabel > &list)  
*Turn on Color Label buttons using list.*

## Public Member Functions inherited from [Digikam::DVBox](#)

- **DVBox** (QWidget \*const parent=nullptr)

## Public Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (QWidget \*const parent=nullptr)
- QSize **minimumSizeHint** () const override
- void **setContentsMargins** (const QMargins &margins)
- void **setContentsMargins** (int left, int top, int right, int bottom)
- void **setSpacing** (int space)
- void **setStretchFactor** (QWidget \*const widget, int stretch)
- QSize **sizeHint** () const override

## Additional Inherited Members

## Static Public Member Functions inherited from [Digikam::PickLabelWidget](#)

- static QIcon **buildIcon** (PickLabel label)
- static QString **labelPickName** (PickLabel label)

## Protected Member Functions inherited from [Digikam::PickLabelWidget](#)

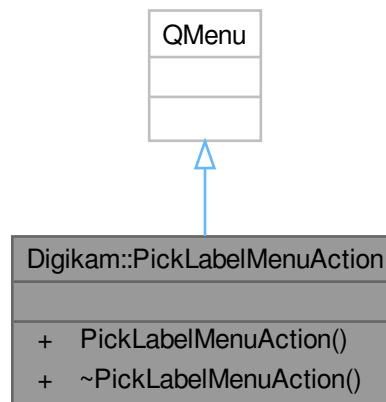
- bool **eventFilter** (QObject \*obj, QEvent \*ev) override

## Protected Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (bool vertical, QWidget \*const parent)
- void **childEvent** (QChildEvent \*e) override

## 9.1045 Digikam::PickLabelMenuAction Class Reference

Inheritance diagram for Digikam::PickLabelMenuAction:



### Signals

- void **signalPickLabelChanged** (int)

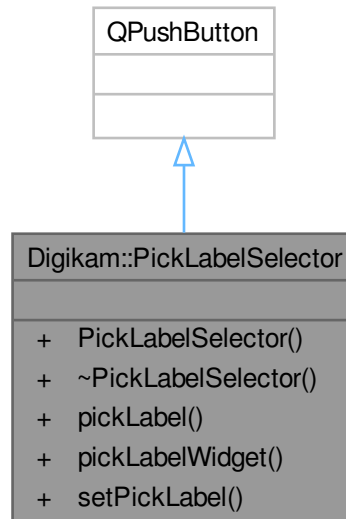
### Public Member Functions

- **PickLabelMenuAction** (QMenu \*const parent=nullptr)



## 9.1046 Digikam::PickLabelSelector Class Reference

Inheritance diagram for Digikam::PickLabelSelector:



### Signals

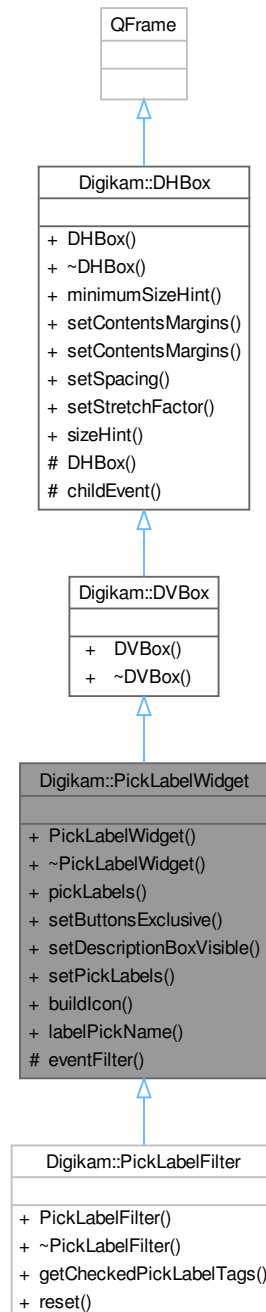
- void **signalPickLabelChanged** (int)

### Public Member Functions

- **PickLabelSelector** (QWidget \*const parent=nullptr)
- PickLabel **pickLabel** ()
- [PickLabelWidget](#) \* **pickLabelWidget** () const
- void **setPickLabel** (PickLabel label)

## 9.1047 Digikam::PickLabelWidget Class Reference

Inheritance diagram for Digikam::PickLabelWidget:



### Signals

- void **signalPickLabelChanged** (int)

## Public Member Functions

- **PickLabelWidget** (QWidget \*const parent=nullptr)
- QList< PickLabel > **pickLabels** () const  
*Return the list of Pick Label buttons turned on or an empty list of none.*
- void **setButtonsExclusive** (bool b)  
*Set all Color Label buttons exclusive or not.*
- void **setDescriptionBoxVisible** (bool b)  
*Show or not on the bottom view the description of label with shortcuts.*
- void **setPickLabels** (const QList< PickLabel > &list)  
*Turn on Color Label buttons using list.*

## Public Member Functions inherited from Digikam::DVBox

- **DVBox** (QWidget \*const parent=nullptr)

## Public Member Functions inherited from Digikam::DHBox

- **DHBox** (QWidget \*const parent=nullptr)
- QSize **minimumSizeHint** () const override
- void **setContentsMargins** (const QMargins &margins)
- void **setContentsMargins** (int left, int top, int right, int bottom)
- void **setSpacing** (int space)
- void **setStretchFactor** (QWidget \*const widget, int stretch)
- QSize **sizeHint** () const override

## Static Public Member Functions

- static QIcon **buildIcon** (PickLabel label)
- static QString **labelPickName** (PickLabel label)

## Protected Member Functions

- bool **eventFilter** (QObject \*obj, QEvent \*ev) override

## Protected Member Functions inherited from Digikam::DHBox

- **DHBox** (bool vertical, QWidget \*const parent)
- void **childEvent** (QChildEvent \*e) override

## 9.1047.1 Member Function Documentation

### 9.1047.1.1 setButtonsExclusive()

```
void Digikam::PickLabelWidget::setButtonsExclusive (
    bool b )
```

Default is true as only one can be selected. Non-exclusive mode is dedicated for Advanced Search tool.

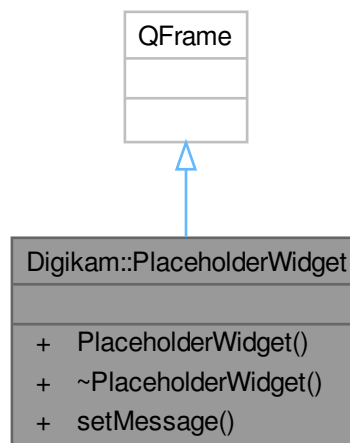
### 9.1047.1.2 setPickLabels()

```
void Digikam::PickLabelWidget::setPickLabels (
    const QList< PickLabel > & list )
```

Pass an empty list to clear all selection.

## 9.1048 Digikam::PlaceholderWidget Class Reference

Inheritance diagram for Digikam::PlaceholderWidget:



### Public Member Functions

- **PlaceholderWidget** (QWidget \*const parent=nullptr)
- void **setMessage** (const QString &message)

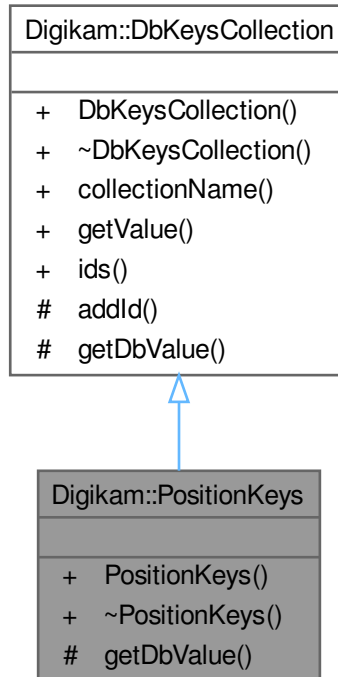
## 9.1049 Digikam::PointTransformAffine Class Reference

### Public Member Functions

- **PointTransformAffine** (const std::vector< std::vector< float > > &m\_)
- **PointTransformAffine** (const std::vector< std::vector< float > > &m\_, const std::vector< float > &b\_)
- const std::vector< float > & **get\_b** () const
- const std::vector< std::vector< float > > & **get\_m** () const
- const std::vector< float > **operator()** (const std::vector< float > &p) const

## 9.1050 Digikam::PositionKeys Class Reference

Inheritance diagram for Digikam::PositionKeys:



### Public Member Functions

- [PositionKeys \(\)](#)

### Public Member Functions inherited from [Digikam::DbKeysCollection](#)

- [DbKeysCollection](#) (const QString &n)  
*Default constructor.*
- QString [collectionName](#) () const  
*Get the name of the DbKeysCollection.*
- QString [getValue](#) (const QString &key, [ParseSettings](#) &settings)  
*Get a value from the database.*
- DbKeyIdsMap [ids](#) () const  
*Get all IDs associated with this key collection.*

### Protected Member Functions

- QString [getDbValue](#) (const QString &key, [ParseSettings](#) &settings) override  
*Abstract method for retrieving the value from the database for the given key.*

## Protected Member Functions inherited from [Digikam::DbKeysCollection](#)

- void [addId](#) (const QString &id, const QString &description)  
*Add an ID to the key collection.*

### 9.1050.1 Constructor & Destructor Documentation

#### 9.1050.1.1 PositionKeys()

```
Digikam::PositionKeys::PositionKeys ( )
```

### 9.1050.2 Member Function Documentation

#### 9.1050.2.1 getDbValue()

```
QString Digikam::PositionKeys::getDbValue (
    const QString & key,
    ParseSettings & settings ) [override], [protected], [virtual]
```

This method has to be implemented by all child classes. It is called by the [getValue\(\)](#) method.

#### Parameters

<i>key</i>	the key representing the value in the database
<i>settings</i>	the <a href="#">ParseSettings</a> object holding all relevant information about the image.

#### Returns

the value of the given database key

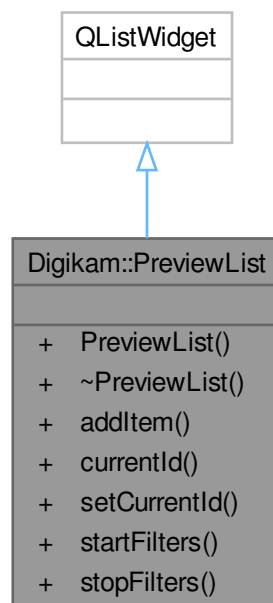
See also

[DbKeysCollection::getValue\(\)](#)

Implements [Digikam::DbKeysCollection](#).

## 9.1051 Digikam::PreviewList Class Reference

Inheritance diagram for Digikam::PreviewList:

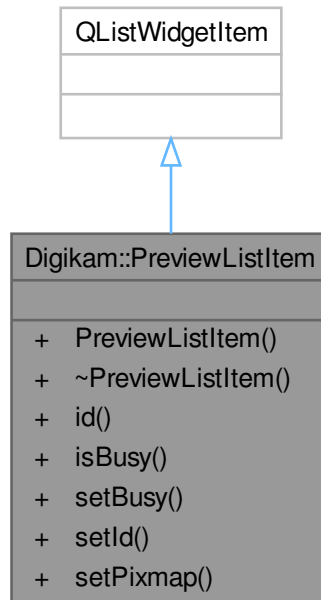


### Public Member Functions

- **PreviewList** (`QWidget *const parent=nullptr`)
- `PreviewListItem * addItem` (`DImgThreadedFilter *const filter, const QString &txt, int id`)
- `int currentId` () const
- void **setCurrentId** (int id)
- void **startFilters** ()
- void **stopFilters** ()

## 9.1052 Digikam::PreviewListItem Class Reference

Inheritance diagram for Digikam::PreviewListItem:



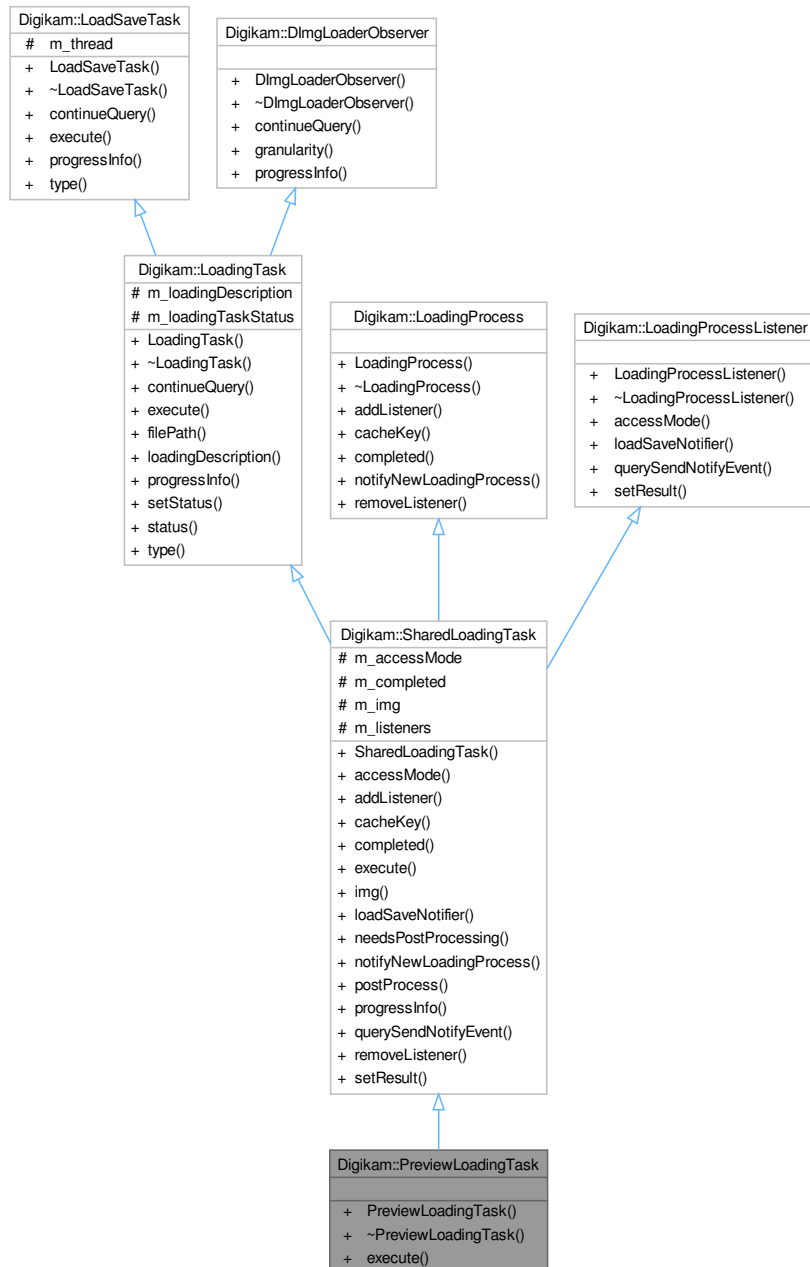
### Public Member Functions

- **PreviewListItem** (`QListWidget *const parent=nullptr`)
- `int id () const`
- `bool isBusy () const`
- `void setBusy (bool b)`
- `void setId (int id)`
- `void setPixmap (const QPixmap &pix)`



## 9.1053 Digikam::PreviewLoadingTask Class Reference

Inheritance diagram for Digikam::PreviewLoadingTask:



### Public Member Functions

- `PreviewLoadingTask` (`LoadSaveThread *const thread`, `const LoadingDescription &description`)
- `void execute ()` override

## Public Member Functions inherited from [Digikam::SharedLoadingTask](#)

- **SharedLoadingTask** ([LoadSaveThread](#) \*const thread, const [LoadingDescription](#) &description, [LoadSaveThread::AccessMode](#) mode=[LoadSaveThread::AccessModeReadWrite](#), [LoadingTaskStatus](#) loadingTaskStatus=[LoadingTaskStatusLoading](#))
- [LoadSaveThread::AccessMode](#) **accessMode** () const override
- void **addListener** ([LoadingProcessListener](#) \*const listener) override
- [QString](#) **cacheKey** () const override
- bool **completed** () const override
- void **execute** () override
- [DImg](#) **img** () const
- [LoadSaveNotifier](#) \* **loadSaveNotifier** () const override
- bool **needsPostProcessing** () const
- void **notifyNewLoadingProcess** ([LoadingProcess](#) \*const process, const [LoadingDescription](#) &description) override
- virtual void **postProcess** ()
- void **progressInfo** (float progress) override
- bool **querySendNotifyEvent** () const override
- void **removeListener** ([LoadingProcessListener](#) \*const listener) override
- void **setResult** (const [LoadingDescription](#) &loadingDescription, const [DImg](#) &img) override

## Public Member Functions inherited from [Digikam::LoadingTask](#)

- **LoadingTask** ([LoadSaveThread](#) \*const thread, const [LoadingDescription](#) &description, [LoadingTaskStatus](#) loadingTaskStatus=[LoadingTaskStatusLoading](#))
- bool **continueQuery** () override
- [QString](#) **filePath** () const
- const [LoadingDescription](#) & **loadingDescription** () const
- void **setStatus** ([LoadingTaskStatus](#) status)
- [LoadingTaskStatus](#) **status** () const
- [TaskType](#) **type** () override

## Public Member Functions inherited from [Digikam::LoadSaveTask](#)

- **LoadSaveTask** ([LoadSaveThread](#) \*const thread)

## Public Member Functions inherited from [Digikam::DImgLoaderObserver](#)

- virtual float **granularity** ()  
*Return a relative value which determines the granularity, the frequency with which the [DImgLoaderObserver](#) is checked and progress is posted.*

## Additional Inherited Members

## Public Types inherited from [Digikam::LoadingTask](#)

- enum **LoadingTaskStatus** { [LoadingTaskStatusLoading](#) , [LoadingTaskStatusPreloading](#) , [LoadingTaskStatusStopping](#) }

## Public Types inherited from [Digikam::LoadSaveTask](#)

- enum `TaskType` { `TaskTypeLoading` , `TaskTypeSaving` }

## Protected Attributes inherited from [Digikam::SharedLoadingTask](#)

- [LoadSaveThread::AccessMode](#) `m_accessMode` = `LoadSaveThread::AccessModeReadWrite`
- volatile bool `m_completed` = false
- [DImg](#) `m_img`
- `QList< LoadingProcessListener * >` `m_listeners`

## Protected Attributes inherited from [Digikam::LoadingTask](#)

- [LoadingDescription](#) `m_loadingDescription`
- volatile `LoadingTaskStatus` `m_loadingTaskStatus` = `LoadingTaskStatusLoading`

## Protected Attributes inherited from [Digikam::LoadSaveTask](#)

- [LoadSaveThread](#) \* `m_thread` = nullptr

## 9.1053.1 Member Function Documentation

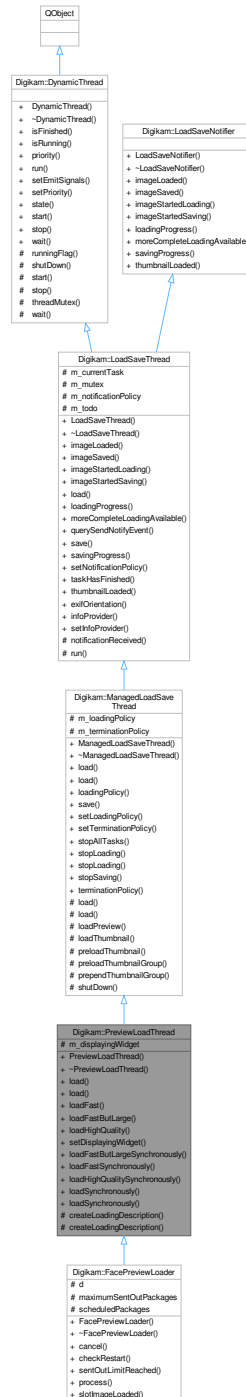
### 9.1053.1.1 `execute()`

```
void Digikam::PreviewLoadingTask::execute ( ) [override], [virtual]
```

Reimplemented from [Digikam::LoadingTask](#).

## 9.1054 Digikam::PreviewLoadThread Class Reference

Inheritance diagram for Digikam::PreviewLoadThread:



### Public Member Functions

- [PreviewLoadThread](#) (QObject \*const parent=nullptr)  
*Creates a preview load thread.*

- void **load** (const [LoadingDescription](#) &description)  
*Load a preview.*
- void **load** (const QString &filePath, const [PreviewSettings](#) &settings, int size=0)  
*Load a preview.*
- void **loadFast** (const QString &filePath, int size)  
*Load a preview that is optimized for fast loading.*
- void **loadFastButLarge** (const QString &filePath, int minimumSize)  
*Load a preview that is as large as possible without sacrificing speed for performance.*
- void **loadHighQuality** (const QString &filePath, [PreviewSettings::RawLoading](#) rawLoadingMode=[PreviewSettings::RawPreviewAutomatic](#))  
*Load a preview with higher resolution, trading more quality for less speed.*
- void **setDisplayingWidget** (QWidget \*const widget)  
*Optionally, set the displaying widget for color management.*

## Public Member Functions inherited from [Digikam::ManagedLoadSaveThread](#)

- **ManagedLoadSaveThread** (QObject \*const parent=nullptr)  
*Termination is controlled by setting the TerminationPolicy Default is TerminationPolicyTerminateLoading.*
- void **load** (const [LoadingDescription](#) &description)  
*Append a task to load the given file to the task list.*
- void **load** (const [LoadingDescription](#) &description, [LoadingPolicy](#) policy)
- [LoadingPolicy](#) **loadingPolicy** () const
- void **save** (const [DImg](#) &image, const QString &filePath, const QString &format)  
*Append a task to save the image to the task list.*
- void **setLoadingPolicy** ([LoadingPolicy](#) policy)  
*Set the loading policy.*
- void **setTerminationPolicy** ([TerminationPolicy](#) terminationPolicy)
- void **stopAllTasks** ()
- void **stopLoading** (const [LoadingDescription](#) &desc, [LoadingTaskFilter](#) filter=[LoadingTaskFilterAll](#))  
*Same than previous method, but Stop and remove tasks filtered by LoadingDescription.*
- void **stopLoading** (const QString &filePath=QString(), [LoadingTaskFilter](#) filter=[LoadingTaskFilterAll](#))  
*Stop and remove tasks filtered by filePath and policy.*
- void **stopSaving** (const QString &filePath=QString())  
*Stop and remove saving tasks filtered by filePath.*
- [TerminationPolicy](#) **terminationPolicy** () const

## Public Member Functions inherited from [Digikam::LoadSaveThread](#)

- **LoadSaveThread** (QObject \*const parent=nullptr)
- **~LoadSaveThread** () override  
*Destructor: The thread will execute all pending tasks and wait for this upon destruction.*
- void **imageLoaded** (const [LoadingDescription](#) &loadingDescription, const [DImg](#) &img) override
- void **imageSaved** (const QString &filePath, bool success) override
- void **imageStartedLoading** (const [LoadingDescription](#) &loadingDescription) override
- void **imageStartedSaving** (const QString &filePath) override
- void **load** (const [LoadingDescription](#) &description)  
*Append a task to load the given file to the task list.*
- void **loadingProgress** (const [LoadingDescription](#) &loadingDescription, float progress) override
- void **moreCompleteLoadingAvailable** (const [LoadingDescription](#) &oldLoadingDescription, const [LoadingDescription](#) &newLoadingDescription) override

- virtual bool **querySendNotifyEvent** () const
- void **save** (const [DImg](#) &image, const QString &filePath, const QString &format)
  - *Append a task to save the image to the task list.*
- void **savingProgress** (const QString &filePath, float progress) override
- void **setNotificationPolicy** ([NotificationPolicy](#) notificationPolicy)
- virtual void **taskHasFinished** ()
- void **thumbnailLoaded** (const [LoadingDescription](#) &loadingDescription, const QImage &img) override

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread](#) (QObject \*const parent=nullptr)
  - *This class extends [QRunnable](#), so you have to reimplement virtual void [run\(\)](#).*
- [~DynamicThread](#) () override
  - *The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool **isFinished** () const
- bool **isRunning** () const
- [QThread::Priority](#) **priority** () const
- void **setEmitSignals** (bool emitThem)
- void **setPriority** ([QThread::Priority](#) priority)
  - *Sets the priority for this dynamic thread.*
- State **state** () const

## Static Public Member Functions

- static [DImg](#) **loadFastButLargeSynchronously** (const QString &filePath, int minimumSize, const [IccProfile](#) &profile=[IccProfile](#)())
- static [DImg](#) **loadFastSynchronously** (const QString &filePath, int size, const [IccProfile](#) &profile=[IccProfile](#)())
  - *Synchronous versions of the above methods.*
- static [DImg](#) **loadHighQualitySynchronously** (const QString &filePath, [PreviewSettings::RawLoading](#) raw↔  
LoadingMode=[PreviewSettings::RawPreviewAutomatic](#), const [IccProfile](#) &profile=[IccProfile](#)())
- static [DImg](#) **loadSynchronously** (const [LoadingDescription](#) &description)
- static [DImg](#) **loadSynchronously** (const QString &filePath, const [PreviewSettings](#) &previewSettings, int size, const [IccProfile](#) &profile=[IccProfile](#)())

## Static Public Member Functions inherited from [Digikam::LoadSaveThread](#)

- static int **exifOrientation** (const QString &filePath, const [DMetadata](#) &metadata, bool isRaw, bool fromRaw↔  
EmbeddedPreview)
  - *Retrieves the Exif orientation, either from the info provider if available, or from the metadata.*
- static [LoadSaveFileInfoProvider](#) \* **infoProvider** ()
- static void **setInfoProvider** ([LoadSaveFileInfoProvider](#) \*const infoProvider)

## Protected Member Functions

- [LoadingDescription](#) **createLoadingDescription** (const QString &filePath, const [PreviewSettings](#) &settings, int size)

## Protected Member Functions inherited from [Digikam::ManagedLoadSaveThread](#)

- void **load** (const [LoadingDescription](#) &description, [LoadingMode](#) loadingMode, [AccessMode](#) mode=[AccessModeReadWrite](#))
- void **load** (const [LoadingDescription](#) &description, [LoadingMode](#) loadingMode, [LoadingPolicy](#) policy, [AccessMode](#) mode=[AccessModeReadWrite](#))
- void **loadPreview** (const [LoadingDescription](#) &description, [LoadingPolicy](#) policy)
- void **loadThumbnail** (const [LoadingDescription](#) &description)
- void **preloadThumbnail** (const [LoadingDescription](#) &description)
- void **preloadThumbnailGroup** (const QList< [LoadingDescription](#) > &descriptions)
- void **prependThumbnailGroup** (const QList< [LoadingDescription](#) > &descriptions)
- void **shutDown** ()

## Protected Member Functions inherited from [Digikam::LoadSaveThread](#)

- void **notificationReceived** ()
- void **run** () override

*Implement this pure virtual function in your subclass.*

## Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool **runningFlag** () const volatile  
*In you [run\(\)](#) method, you shall regularly check for [runningFlag\(\)](#) and cleanup and return if false.*
- void **shutDown** ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call [stop\(\)](#) and [wait\(\)](#), knowing that nothing will call [start\(\)](#) anymore after this 3) Be sure the thread will never be running at destruction.*
- void **start** (QMutexLocker< QMutex > &locker)  
*Doing the same as [start\(\)](#), [stop\(\)](#) and [wait](#) above, provide it with a locked QMutexLocker on mutex().*
- void **stop** (const QMutexLocker< QMutex > &locker)
- QMutex \* **threadMutex** () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void **wait** (QMutexLocker< QMutex > &locker)

## Static Protected Member Functions

- static [LoadingDescription](#) **createLoadingDescription** (const QString &filePath, const [PreviewSettings](#) &settings, int size, const [IccProfile](#) &profile)

## Protected Attributes

- QWidget \* **m\_displayingWidget** = nullptr

## Protected Attributes inherited from [Digikam::ManagedLoadSaveThread](#)

- [LoadingPolicy](#) **m\_loadingPolicy** = [LoadingPolicyAppend](#)
- [TerminationPolicy](#) **m\_terminationPolicy** = [TerminationPolicyTerminateLoading](#)

## Protected Attributes inherited from [Digikam::LoadSaveThread](#)

- [LoadSaveTask](#) \* `m_currentTask` = nullptr
- QMutex `m_mutex`
- [NotificationPolicy](#) `m_notificationPolicy` = [NotificationPolicyTimeLimited](#)
- QList< [LoadSaveTask](#) \* > `m_todo`

## Additional Inherited Members

## Public Types inherited from [Digikam::ManagedLoadSaveThread](#)

- enum [LoadingMode](#) { [LoadingModeNormal](#) , [LoadingModeShared](#) }  
*used by [SharedLoadSaveThread](#) only*
- enum [LoadingPolicy](#) { [LoadingPolicyFirstRemovePrevious](#) , [LoadingPolicyPrepend](#) , [LoadingPolicySimplePrepend](#) , [LoadingPolicyAppend](#) , [LoadingPolicySimpleAppend](#) , [LoadingPolicyPreload](#) }
- enum [LoadingTaskFilter](#) { [LoadingTaskFilterAll](#) , [LoadingTaskFilterPreloading](#) }
- enum [TerminationPolicy](#) { [TerminationPolicyTerminateLoading](#) , [TerminationPolicyTerminatePreloading](#) , [TerminationPolicyWait](#) , [TerminationPolicyTerminateAll](#) }

## Public Types inherited from [Digikam::LoadSaveThread](#)

- enum [AccessMode](#) { [AccessModeRead](#) , [AccessModeReadWrite](#) }  
*used by [SharedLoadSaveThread](#) only*
- enum [NotificationPolicy](#) { [NotificationPolicyDirect](#) , [NotificationPolicyTimeLimited](#) }

## Public Types inherited from [Digikam::DynamicThread](#)

- enum [State](#) { [Inactive](#) , [Scheduled](#) , [Running](#) , [Deactivating](#) }

## Public Slots inherited from [Digikam::DynamicThread](#)

- void [start](#) ()
- void [stop](#) ()  
*Stop computation, sets the running flag to false.*
- void [wait](#) ()  
*Waits until the thread finishes.*

## Signals inherited from [Digikam::LoadSaveThread](#)

- void [signalImageLoaded](#) (const [LoadingDescription](#) &loadingDescription, const [DImg](#) &img)  
*This signal is emitted when the loading process has finished.*
- void [signalImageSaved](#) (const QString &filePath, bool success)
- void [signalImageStartedLoading](#) (const [LoadingDescription](#) &loadingDescription)  
*All signals are delivered to the thread from where the [LoadSaveThread](#) object has been created.*
- void [signalImageStartedSaving](#) (const QString &filePath)
- void [signalLoadingProgress](#) (const [LoadingDescription](#) &loadingDescription, float progress)  
*This signal is emitted whenever new progress info is available and the notification policy allows emitting the signal.*
- void [signalMoreCompleteLoadingAvailable](#) (const [LoadingDescription](#) &oldLoadingDescription, const [LoadingDescription](#) &newLoadingDescription)  
*This signal is emitted if.*
- void [signalSavingProgress](#) (const QString &filePath, float progress)
- void [signalThumbnailLoaded](#) (const [LoadingDescription](#) &loadingDescription, const [QImage](#) &img)



## Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()

*Emitted if emitSignals is enabled.*

### 9.1054.1 Constructor & Destructor Documentation

#### 9.1054.1.1 PreviewLoadThread()

```
Digikam::PreviewLoadThread::PreviewLoadThread (  
    QObject *const parent = nullptr ) [explicit]
```

Provides three flavors of preview loading. The default loading policy, for the typical usage in a preview widget, always stops any previous tasks and loads the new task as soon as possible.

### 9.1054.2 Member Function Documentation

#### 9.1054.2.1 load() [1/2]

```
void Digikam::PreviewLoadThread::load (  
    const LoadingDescription & description )
```

Loading description will not be touched.

#### 9.1054.2.2 load() [2/2]

```
void Digikam::PreviewLoadThread::load (  
    const QString & filePath,  
    const PreviewSettings & settings,  
    int size = 0 )
```

Settings determine the loading mode. For fast loading, size is preview area size. For fast-but-large loading, it serves as a minimum size. For high quality loading, it is ignored

#### 9.1054.2.3 loadFast()

```
void Digikam::PreviewLoadThread::loadFast (  
    const QString & filePath,  
    int size )
```

Raw decoding and color management settings will be adjusted.

#### 9.1054.2.4 loadFastButLarge()

```
void Digikam::PreviewLoadThread::loadFastButLarge (  
    const QString & filePath,  
    int minimumSize )
```

Especially, raw previews are taken if larger than the given size. Raw decoding and color management settings will be adjusted.

### 9.1054.2.5 loadFastSynchronously()

```
DImg Digikam::PreviewLoadThread::loadFastSynchronously (
    const QString & filePath,
    int size,
    const IccProfile & profile = IccProfile() ) [static]
```

These are safe to call from the non-UI thread, as the [IccProfile](#) either passed or deduced independent from a displaying widget

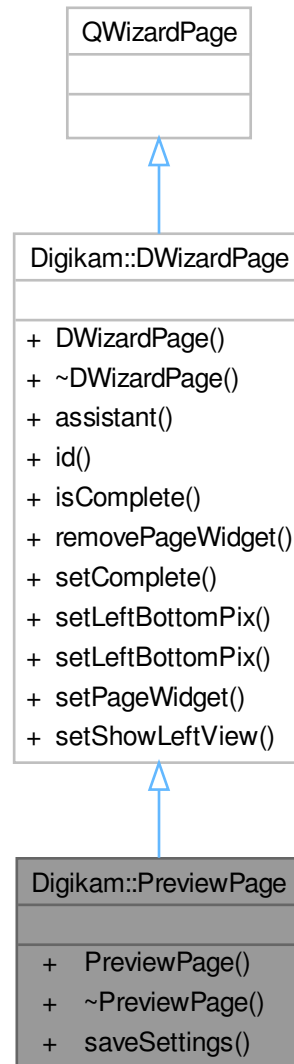
### 9.1054.2.6 loadHighQuality()

```
void Digikam::PreviewLoadThread::loadHighQuality (
    const QString & filePath,
    PreviewSettings::RawLoading rawLoadingMode = PreviewSettings::RawPreviewAutomatic
)
```

Raw decoding and color management settings will be adjusted.

## 9.1055 Digikam::PreviewPage Class Reference

Inheritance diagram for Digikam::PreviewPage:



### Public Member Functions

- **PreviewPage** (`QWizard *const dlg`)
- void **saveSettings** ()

### Public Member Functions inherited from [Digikam::DWizardPage](#)

- **DWizardPage** (`QWizard *const dlg, const QString &title`)
- `QWizard * assistant () const`

- int **id** () const
- bool **isComplete** () const override
- void **removePageWidget** (QWidget \*const w)
- void **setComplete** (bool b)
- void **setLeftBottomPix** (const QIcon &icon)
- void **setLeftBottomPix** (const QPixmap &pix)
- void **setPageWidget** (QWidget \*const w)
- void **setShowLeftView** (bool v)

## 9.1056 Digikam::PreviewSettings Class Reference

### Public Types

- enum [Quality](#) { [FastPreview](#) , [FastButLargePreview](#) , [HighQualityPreview](#) }
- enum [RawLoading](#) { [RawPreviewAutomatic](#) , [RawPreviewFromEmbeddedPreview](#) , [RawPreviewFromRawHalfSize](#) , [RawPreviewFromRawFullSize](#) }

### Public Member Functions

- [PreviewSettings](#) ([Quality](#) quality=[HighQualityPreview](#), [RawLoading](#) rawLoading=[RawPreviewAutomatic](#))
- bool **operator==** (const [PreviewSettings](#) &other) const

### Static Public Member Functions

- static [PreviewSettings](#) **fastPreview** ()
- static [PreviewSettings](#) **highQualityPreview** ()

### Public Attributes

- bool **convertToEightBit** = false
- [Quality](#) **quality**
- [RawLoading](#) **rawLoading**

## 9.1056.1 Member Enumeration Documentation

### 9.1056.1.1 Quality

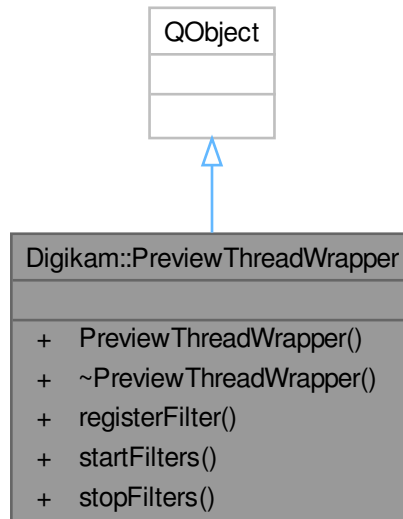
enum [Digikam::PreviewSettings::Quality](#)

#### Enumerator

FastPreview	A preview were loading time is most important. Preview can be reduced in size. Additionally specifying the size of the preview area may be appropriate
FastButLargePreview	Load a preview that is as large as possible without sacrificing speed for performance. Especially, raw previews are taken if larger than the given size. Raw decoding and color management settings will be adjusted.
HighQualityPreview	Load a high quality additional image. For normal images, loads the full data. For RAW, the additional settings below are taken into account

## 9.1057 Digikam::PreviewThreadWrapper Class Reference

Inheritance diagram for Digikam::PreviewThreadWrapper:



### Signals

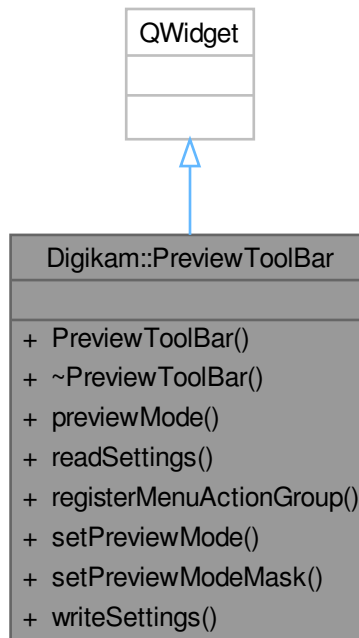
- void **signalFilterFinished** (int, const QPixmap &)
- void **signalFilterStarted** (int)

### Public Member Functions

- **PreviewThreadWrapper** (QObject \*const parent=nullptr)
- void **registerFilter** (int id, [DImgThreadedFilter](#) \*const filter)
- void **startFilters** ()
- void **stopFilters** ()

## 9.1058 Digikam::PreviewToolBar Class Reference

Inheritance diagram for Digikam::PreviewToolBar:



### Public Types

- enum `PreviewMode` {  
`PreviewOriginalImage` = 0x00000001 , `PreviewBothImagesHorz` = 0x00000002 , `PreviewBothImagesVert` = 0x00000004 , `PreviewBothImagesHorzCont` = 0x00000008 ,  
`PreviewBothImagesVertCont` = 0x00000010 , `PreviewTargetImage` = 0x00000020 , `PreviewToggleOnMouseOver` = 0x00000040 , `NoPreviewMode` = 0x00000080 ,  
`AllPreviewModes` , `UnSplitPreviewModes` = `PreviewOriginalImage` | `PreviewTargetImage` | `PreviewToggleOnMouseOver` }

### Signals

- void `signalPreviewModeChanged` (int)

### Public Member Functions

- `PreviewToolBar` (`QWidget *const parent=nullptr`)
- `PreviewMode previewMode` () const
- void `readSettings` (const `KConfigGroup &group`)
- void `registerMenuActionGroup` (`EditorWindow *const editor`)
- void `setPreviewMode` (`PreviewMode mode`)
- void `setPreviewModeMask` (int mask)
- void `writeSettings` (`KConfigGroup &group`)

## 9.1058.1 Member Enumeration Documentation

### 9.1058.1.1 PreviewMode

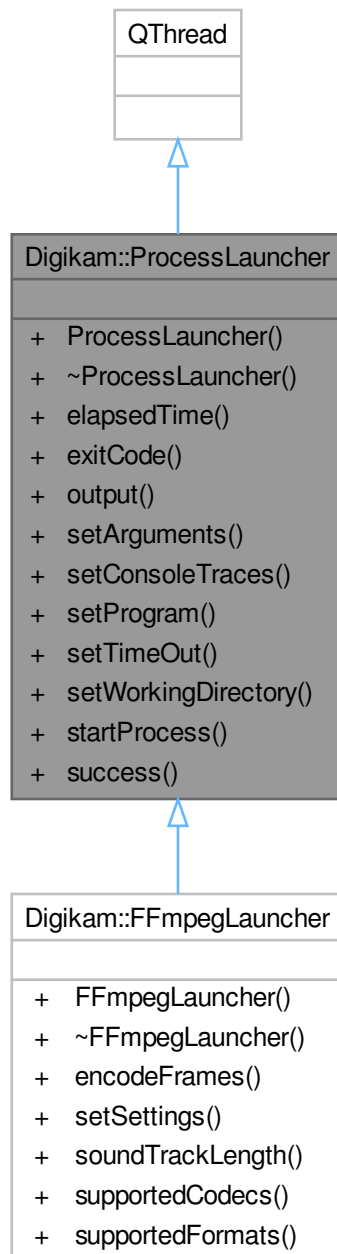
enum `Digikam::PreviewToolBar::PreviewMode`

#### Enumerator

<code>PreviewOriginalImage</code>	Original image only.
<code>PreviewBothImagesHorz</code>	Horizontal with original and target duplicated.
<code>PreviewBothImagesVert</code>	Vertical with original and target duplicated.
<code>PreviewBothImagesHorzCont</code>	Horizontal with original and target in contiguous.
<code>PreviewBothImagesVertCont</code>	Vertical with original and target in contiguous.
<code>PreviewTargetImage</code>	Target image only.
<code>PreviewToggleOnMouseOver</code>	Original image if mouse is over image area, else target image.
<code>NoPreviewMode</code>	Target image only without information displayed.

## 9.1059 Digikam::ProcessLauncher Class Reference

Inheritance diagram for Digikam::ProcessLauncher:



### Signals

- void **signalComplete** (bool `success`, int `exitCode`)

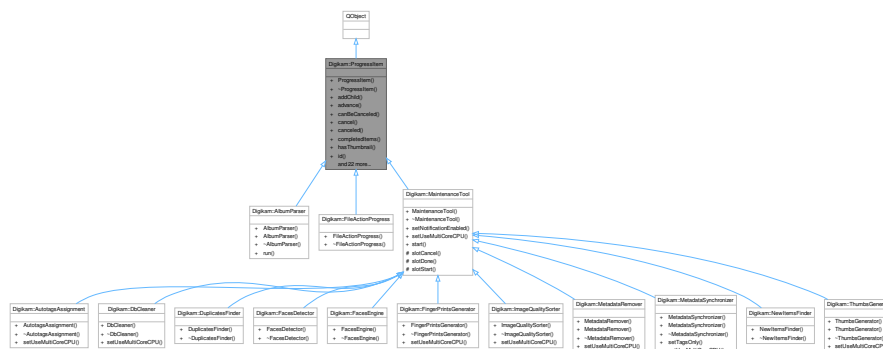


## Public Member Functions

- **ProcessLauncher** (QObject \*const parent=nullptr)
- qint64 **elapsedTime** () const  
*Return the elapsed time in ms to run the process.*
- int **exitCode** () const  
*Return the exit code from the process.*
- QString **output** () const  
*Return the process output as string.*
- void **setArguments** (const QStringList &args)
- void **setConsoleTraces** (bool b)  
*If turned on, all traces from the process are printed on the console.*
- void **setProgram** (const QString &prog)
- void **setTimeout** (int msec)
- void **setWorkingDirectory** (const QString &dir)
- void **startProcess** ()  
*Start the process.*
- bool **success** () const  
*Return true if the process is started and completed without error.*

## 9.1060 Digikam::ProgressItem Class Reference

Inheritance diagram for Digikam::ProgressItem:



## Signals

- void **progressItemAdded** (ProgressItem \*item)  
*Emitted when a new ProgressItem is added.*
- void **progressItemCanceled** (ProgressItem \*item)  
*Emitted when an item was canceled.*
- void **progressItemCanceledById** (const QString &id)
- void **progressItemCompleted** (ProgressItem \*item)  
*Emitted when a progress item was completed.*
- void **progressItemLabel** (ProgressItem \*item, const QString &label)  
*Emitted when the label of an item changed.*
- void **progressItemProgress** (ProgressItem \*item, unsigned int v)  
*Emitted when the progress value of an item changes.*

- void [progressItemStatus](#) ([ProgressItem](#) \*item, const QString &mess)  
*Emitted when the status message of an item changed.*
- void [progressItemThumbnail](#) ([ProgressItem](#) \*item, const QPixmap &thumb)  
*Emitted when the thumbnail data must be set in item.*
- void [progressItemUsesBusyIndicator](#) ([ProgressItem](#) \*item, bool value)  
*Emitted when the busy indicator state of an item changes.*

## Public Member Functions

- **ProgressItem** ([ProgressItem](#) \*const parent, const QString &id, const QString &label, const QString &status, bool canBeCanceled, bool hasThumb)
- void **addChild** ([ProgressItem](#) \*const kiddo)
- bool **advance** (unsigned int v)  
*Advance total items processed by n values and update percentage in progressbar.*
- bool **canBeCanceled** () const
- void **cancel** ()
- bool **canceled** () const
- unsigned int **completedItems** () const
- bool **hasThumbnail** () const
- const QString & **id** () const
- bool **incCompletedItems** (unsigned int v=1)
- void **incTotalItems** (unsigned int v=1)
- const QString & **label** () const
- [ProgressItem](#) \* **parent** () const
- unsigned int **progress** () const
- void **removeChild** ([ProgressItem](#) \*const kiddo)
- void **reset** ()  
*Reset the progress value of this item to 0 and the status string to the empty string.*
- void **setComplete** ()  
*Tell the item it has finished.*
- bool **setCompletedItems** (unsigned int v)
- void **setLabel** (const QString &v)
- void **setProgress** (unsigned int v)  
*Set the progress (percentage of completion) value of this item.*
- void **setShowAtStart** (bool showAtStart)  
*Set the property to pop-up item when it's added in progress manager.*
- void **setStatus** (const QString &v)  
*Set the string to be used for showing this item's current status.*
- void **setThumbnail** (const QIcon &icon)  
*Sets whether this item has a thumbnail.*
- void **setTotalItems** (unsigned int v)
- void **setUsesBusyIndicator** (bool useBusyIndicator)  
*Sets whether this item uses a busy indicator instead of real progress for its progress bar.*
- bool **showAtStart** () const
- const QString & **status** () const
- bool **totalCompleted** () const
- unsigned int **totalItems** () const
- void **updateProgress** ()  
*Recalculate progress according to total/completed items and update.*
- bool **usesBusyIndicator** () const

## 9.1060.1 Member Function Documentation

### 9.1060.1.1 advance()

```
bool Digikam::ProgressItem::advance (
    unsigned int v )
```

#### Parameters

<code>v</code>	The value to advance.
----------------	-----------------------

#### Returns

true if totalCompleted()

### 9.1060.1.2 canBeCanceled()

```
bool Digikam::ProgressItem::canBeCanceled ( ) const
```

#### Returns

Whether this item can be canceled.

### 9.1060.1.3 hasThumbnail()

```
bool Digikam::ProgressItem::hasThumbnail ( ) const
```

#### Returns

whether this item has a thumbnail.

### 9.1060.1.4 id()

```
const QString & Digikam::ProgressItem::id ( ) const
```

#### Returns

The id string which uniquely identifies the operation represented by this item.

### 9.1060.1.5 label()

```
const QString & Digikam::ProgressItem::label ( ) const
```

#### Returns

The user visible string to be used to represent this item.

### 9.1060.1.6 parent()

```
ProgressItem * Digikam::ProgressItem::parent ( ) const
```

#### Returns

The parent item of this one, if there is one.

### 9.1060.1.7 progress()

```
unsigned int Digikam::ProgressItem::progress ( ) const
```

#### Returns

The current progress value of this item in percent.

### 9.1060.1.8 progressItemAdded

```
void Digikam::ProgressItem::progressItemAdded (
    ProgressItem * item ) [signal]
```

#### Parameters

<i>item</i>	The <a href="#">ProgressItem</a> that was added.
-------------	--

### 9.1060.1.9 progressItemCanceled

```
void Digikam::ProgressItem::progressItemCanceled (
    ProgressItem * item ) [signal]
```

It will *not* go away immediately, only when the owner sets it complete, which will usually happen. Can be used to visually indicate the canceled status of an item. Should be used by the owner of the item to make sure it is set completed even if it is canceled. There is a [ProgressManager::slotStandardCancelHandler](#) which simply sets the item completed and can be used if no other work needs to be done on cancel.

#### Parameters

<i>item</i>	The canceled item;
-------------	--------------------

### 9.1060.1.10 progressItemCompleted

```
void Digikam::ProgressItem::progressItemCompleted (
    ProgressItem * item ) [signal]
```

The item will be deleted afterwards, so slots connected to this are the last chance to work with this item.

## Parameters

<i>item</i>	The completed item.
-------------	---------------------

**9.1060.1.11 progressItemLabel**

```
void Digikam::ProgressItem::progressItemLabel (
    ProgressItem * item,
    const QString & label ) [signal]
```

Should be used by progress dialogs to update the label of an item.

## Parameters

<i>item</i>	The updated item.
<i>label</i>	The new label.

**9.1060.1.12 progressItemProgress**

```
void Digikam::ProgressItem::progressItemProgress (
    ProgressItem * item,
    unsigned int v ) [signal]
```

## Parameters

<i>item</i>	The item which got a new value.
<i>v</i>	The value, for convenience.

**9.1060.1.13 progressItemStatus**

```
void Digikam::ProgressItem::progressItemStatus (
    ProgressItem * item,
    const QString & mess ) [signal]
```

Should be used by progress dialogs to update the status message for an item.

## Parameters

<i>item</i>	The updated item.
<i>mess</i>	The new message.

**9.1060.1.14 progressItemThumbnail**

```
void Digikam::ProgressItem::progressItemThumbnail (
    ProgressItem * item,
    const QPixmap & thumb ) [signal]
```

## Parameters

<i>item</i>	The updated item
<i>thumb</i>	thumbnail data

**9.1060.1.15 progressItemUsesBusyIndicator**

```
void Digikam::ProgressItem::progressItemUsesBusyIndicator (
    ProgressItem * item,
    bool value ) [signal]
```

Should be used by progress dialogs so that they can adjust the display of the progress bar to the new mode.

## Parameters

<i>item</i>	The updated item
<i>value</i>	True if the item uses a busy indicator now, false otherwise

**9.1060.1.16 setComplete()**

```
void Digikam::ProgressItem::setComplete ( )
```

This will emit [progressItemCompleted\(\)](#) result in the destruction of the item after all slots connected to this signal have executed. This is the only way to get rid of an item and needs to be called even if the item is canceled. Don't use the item after this has been called on it.

**9.1060.1.17 setLabel()**

```
void Digikam::ProgressItem::setLabel (
    const QString & v )
```

## Parameters

<i>v</i>	Set the user visible string identifying this item.
----------	--

**9.1060.1.18 setProgress()**

```
void Digikam::ProgressItem::setProgress (
    unsigned int v )
```

## Parameters

<i>v</i>	The percentage value.
----------	-----------------------

### 9.1060.1.19 setShowAtStart()

```
void Digikam::ProgressItem::setShowAtStart (
    bool showAtStart )
```

Use this method if you consider that item is important to be notified to end-user.

#### Parameters

<i>showAtStart</i>	The flag to turn on this property.
--------------------	------------------------------------

### 9.1060.1.20 setStatus()

```
void Digikam::ProgressItem::setStatus (
    const QString & v )
```

#### Parameters

<i>v</i>	The status string.
----------	--------------------

### 9.1060.1.21 setThumbnail()

```
void Digikam::ProgressItem::setThumbnail (
    const QIcon & icon )
```

#### Parameters

<i>icon</i>	The icon to use as thumbnail.
-------------	-------------------------------

### 9.1060.1.22 setUsesBusyIndicator()

```
void Digikam::ProgressItem::setUsesBusyIndicator (
    bool useBusyIndicator )
```

If it uses a busy indicator, you are still responsible for calling [setProgress\(\)](#) from time to time to update the busy indicator.

param *useBusyIndicator* The flag to indicate busy state.

### 9.1060.1.23 showAtStart()

```
bool Digikam::ProgressItem::showAtStart ( ) const
```

#### Returns

true if item must be pop-up when it's added in progress manager.

**9.1060.1.24 status()**

```
const QString & Digikam::ProgressItem::status ( ) const
```

**Returns**

The string to be used for showing this item's current status.

**9.1060.1.25 usesBusyIndicator()**

```
bool Digikam::ProgressItem::usesBusyIndicator ( ) const
```

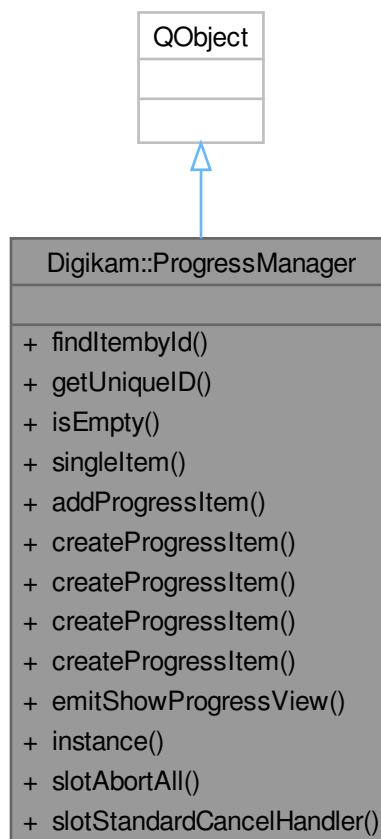
**Returns**

whether this item uses a busy indicator instead of real progress display

**9.1061 Digikam::ProgressManager Class Reference**

The [ProgressManager](#) singleton keeps track of all ongoing transactions and notifies observers (progress dialogs) when their progress percent value changes, when they are completed (by their owner), and when they are canceled.

Inheritance diagram for Digikam::ProgressManager:





## Public Slots

- void [slotAbortAll](#) ()  
*Aborts all running jobs.*
- void [slotStandardCancelHandler](#) ([ProgressItem](#) \*item)  
*Calls `setCompleted()` on the item, to make sure it goes away.*

## Signals

- void **completeTransactionDeferred** ([ProgressItem](#) \*item)
- void [progressItemAdded](#) ([ProgressItem](#) \*)
- void [progressItemCanceled](#) ([ProgressItem](#) \*)
- void [progressItemCompleted](#) ([ProgressItem](#) \*)
- void [progressItemLabel](#) ([ProgressItem](#) \*, const [QString](#) &)
- void [progressItemProgress](#) ([ProgressItem](#) \*, unsigned int)
- void [progressItemStatus](#) ([ProgressItem](#) \*, const [QString](#) &)
- void [progressItemThumbnail](#) ([ProgressItem](#) \*, const [QPixmap](#) &)
- void [progressItemUsesBusyIndicator](#) ([ProgressItem](#) \*, bool)
- void [showProgressView](#) ()  
*Emitted when an operation requests the listeners to be shown.*

## Public Member Functions

- [ProgressItem](#) \* [findItemById](#) (const [QString](#) &id) const
- [QString](#) [getUniqueID](#) ()  
*Use this to acquire a unique id number which can be used to discern an operation from all others going on at the same time.*
- bool [isEmpty](#) () const
- [ProgressItem](#) \* [singleItem](#) () const

## Static Public Member Functions

- static bool [addProgressItem](#) ([ProgressItem](#) \*const t, [ProgressItem](#) \*const parent=nullptr)  
*Add a created progressItem outside manager with the given parent.*
- static [ProgressItem](#) \* [createProgressItem](#) (const [QString](#) &id, const [QString](#) &label, const [QString](#) &status=[QString](#)(), bool canBeCanceled=true, bool hasThumb=false)  
*Use this version if you have the id string of the parent but without the parent instance.*
- static [ProgressItem](#) \* [createProgressItem](#) (const [QString](#) &label, const [QString](#) &status=[QString](#)(), bool canBeCanceled=true, bool hasThumb=false)  
*Creates a [ProgressItem](#) with a unique id and the given label.*
- static [ProgressItem](#) \* [createProgressItem](#) (const [QString](#) &parent, const [QString](#) &id, const [QString](#) &label, const [QString](#) &status=[QString](#)(), bool canBeCanceled=true, bool hasThumb=false)  
*Use this version if you have the id string of the parent and want to add a subjob to it.*
- static [ProgressItem](#) \* [createProgressItem](#) ([ProgressItem](#) \*const parent, const [QString](#) &id, const [QString](#) &label, const [QString](#) &status=[QString](#)(), bool canBeCanceled=true, bool hasThumb=false)  
*Creates a new progressItem with the given parent, id, label and initial status.*
- static void **emitShowProgressView** ()  
*Ask all listeners to show the progress dialog, because there is something that wants to be shown.*
- static [ProgressManager](#) \* [instance](#) ()

## Friends

- class **ProgressManagerCreator**

## 9.1061.1 Detailed Description

Each [ProgressItem](#) emits those signals individually and the singleton broadcasts them. Use the [createProgressItem\(\)](#) statics to acquire an item and then call `->setProgress( int percent )` on it every time you want to update the item and `->setComplete()` when the operation is done. This will delete the item. Connect to the item's [progressItemCanceled\(\)](#) signal to be notified when the user cancels the transaction using one of the observing progress dialogs or by calling `item->cancel()` in some other way. The owner is responsible for calling `setComplete()` on the item, even if it is canceled. Use the `standardCancelHandler()` slot if that is all you want to do on cancel.

### Note

if you request an item with a certain id and there is already one with that id, there will not be a new one created but the existing one will be returned. This is convenient for accessing items that are needed regularly without the to store a pointer to them or to add child items to parents by id.

## 9.1061.2 Member Function Documentation

### 9.1061.2.1 addProgressItem()

```
bool Digikam::ProgressManager::addProgressItem (
    ProgressItem *const t,
    ProgressItem *const parent = nullptr ) [static]
```

#### Parameters

<i>t</i>	The process to add on manager.
<i>parent</i>	Specify an already existing item as the parent of this one (can be null).

#### Returns

true if [ProgressItem](#) have been added to manager, else false.

### 9.1061.2.2 createProgressItem() [1/4]

```
ProgressItem * Digikam::ProgressManager::createProgressItem (
    const QString & id,
    const QString & label,
    const QString & status = QString(),
    bool canBeCanceled = true,
    bool hasThumb = false ) [static]
```

#### Parameters

<i>id</i>	Used to identify this operation for cancel and progress info.
<i>label</i>	The text to be displayed by progress handlers

## Parameters

<i>status</i>	Additional text to be displayed for the item.
<i>canBeCanceled</i>	can the user cancel this operation? Cancelling the parent will cancel the children as well (if they can be canceled) and ongoing children prevent parents from finishing.
<i>hasThumb</i>	flag to indicate if progress item has a thumbnail.

## Returns

The [ProgressItem](#) representing the operation.

**9.1061.2.3 createProgressItem() [2/4]**

```
ProgressItem * Digikam::ProgressManager::createProgressItem (
    const QString & label,
    const QString & status = QString(),
    bool canBeCanceled = true,
    bool hasThumb = false ) [static]
```

This is the simplest way to acquire a progress item. It will not have a parent.

## Parameters

<i>label</i>	The text to be displayed by progress handlers
<i>status</i>	Additional text to be displayed for the item.
<i>canBeCanceled</i>	Can the user cancel this operation? Cancelling the parent will cancel the children as well (if they can be canceled) and ongoing children prevent parents from finishing.
<i>hasThumb</i>	flag to indicate if progress item has a thumbnail.

## Returns

The [ProgressItem](#) representing the operation.

**9.1061.2.4 createProgressItem() [3/4]**

```
ProgressItem * Digikam::ProgressManager::createProgressItem (
    const QString & parent,
    const QString & id,
    const QString & label,
    const QString & status = QString(),
    bool canBeCanceled = true,
    bool hasThumb = false ) [static]
```

## Parameters

<i>parent</i>	Specify an already existing item as the parent of this one.
<i>id</i>	Used to identify this operation for cancel and progress info.
<i>label</i>	The text to be displayed by progress handlers
<i>status</i>	Additional text to be displayed for the item.

## Parameters

<i>canBeCanceled</i>	can the user cancel this operation? Cancelling the parent will cancel the children as well (if they can be canceled) and ongoing children prevent parents from finishing.
<i>hasThumb</i>	flag to indicate if progress item has a thumbnail.

## Returns

The [ProgressItem](#) representing the operation.

**9.1061.2.5 createProgressItem() [4/4]**

```
ProgressItem * Digikam::ProgressManager::createProgressItem (
    ProgressItem *const parent,
    const QString & id,
    const QString & label,
    const QString & status = QString(),
    bool canBeCanceled = true,
    bool hasThumb = false ) [static]
```

## Parameters

<i>parent</i>	Specify an already existing item as the parent of this one.
<i>id</i>	Used to identify this operation for cancel and progress info.
<i>label</i>	The text to be displayed by progress handlers
<i>status</i>	Additional text to be displayed for the item.
<i>canBeCanceled</i>	can the user cancel this operation? Cancelling the parent will cancel the children as well (if they can be canceled) and ongoing children prevent parents from finishing.
<i>hasThumb</i>	flag to indicate if progress item has a thumbnail.

## Returns

The [ProgressItem](#) representing the operation.

**9.1061.2.6 findItembyId()**

```
ProgressItem * Digikam::ProgressManager::findItembyId (
    const QString & id ) const
```

## Returns

the progressitem for this

## Parameters

<i>id</i>	if it exist, else null.
-----------	-------------------------

### 9.1061.2.7 `getUniqueId()`

```
QString Digikam::ProgressManager::getUniqueId ( )
```

Use that number as the id string for your progressItem to ensure it is unique.

#### Returns

The string with the unique ID number.

### 9.1061.2.8 `instance()`

```
ProgressManager * Digikam::ProgressManager::instance ( ) [static]
```

#### Returns

The singleton instance of this class.

### 9.1061.2.9 `isEmpty()`

```
bool Digikam::ProgressManager::isEmpty ( ) const
```

#### Returns

true when there are no more progress items.

### 9.1061.2.10 `progressItemAdded`

```
void Digikam::ProgressManager::progressItemAdded (
    ProgressItem * ) [signal]
```

#### See also

[ProgressItem::progressItemAdded\(\)](#)

### 9.1061.2.11 `progressItemCanceled`

```
void Digikam::ProgressManager::progressItemCanceled (
    ProgressItem * ) [signal]
```

#### See also

[ProgressItem::progressItemCanceled\(\)](#)

### 9.1061.2.12 progressItemCompleted

```
void Digikam::ProgressManager::progressItemCompleted (
    ProgressItem * ) [signal]
```

See also

[ProgressItem::progressItemCompleted\(\)](#)

### 9.1061.2.13 progressItemLabel

```
void Digikam::ProgressManager::progressItemLabel (
    ProgressItem * ,
    const QString & ) [signal]
```

See also

[ProgressItem::progressItemLabel\(\)](#)

### 9.1061.2.14 progressItemProgress

```
void Digikam::ProgressManager::progressItemProgress (
    ProgressItem * ,
    unsigned int ) [signal]
```

See also

[ProgressItem::progressItemProgress\(\)](#)

### 9.1061.2.15 progressItemStatus

```
void Digikam::ProgressManager::progressItemStatus (
    ProgressItem * ,
    const QString & ) [signal]
```

See also

[ProgressItem::progressItemStatus\(\)](#)

### 9.1061.2.16 progressItemThumbnail

```
void Digikam::ProgressManager::progressItemThumbnail (
    ProgressItem * ,
    const QPixmap & ) [signal]
```

See also

[ProgressItem::progressItemThumbnail](#)

### 9.1061.2.17 progressItemUsesBusyIndicator

```
void Digikam::ProgressManager::progressItemUsesBusyIndicator (
    ProgressItem * ,
    bool ) [signal]
```

See also

[ProgressItem::progressItemUsesBusyIndicator](#)

### 9.1061.2.18 showProgressView

```
void Digikam::ProgressManager::showProgressView ( ) [signal]
```

Use [emitShowProgressView\(\)](#) to trigger it.

### 9.1061.2.19 singleItem()

```
ProgressItem * Digikam::ProgressManager::singleItem ( ) const
```

Returns

the only top level progressitem when there's only one. Returns 0 if there is no item, or more than one top level item. Since this is used to calculate the overall progress, it will also return 0 if there is an item which uses a busy indicator, since that will invalidate the overall progress.

### 9.1061.2.20 slotAbortAll

```
void Digikam::ProgressManager::slotAbortAll ( ) [slot]
```

Bound to "Esc"

### 9.1061.2.21 slotStandardCancelHandler

```
void Digikam::ProgressManager::slotStandardCancelHandler (
    ProgressItem * item ) [slot]
```

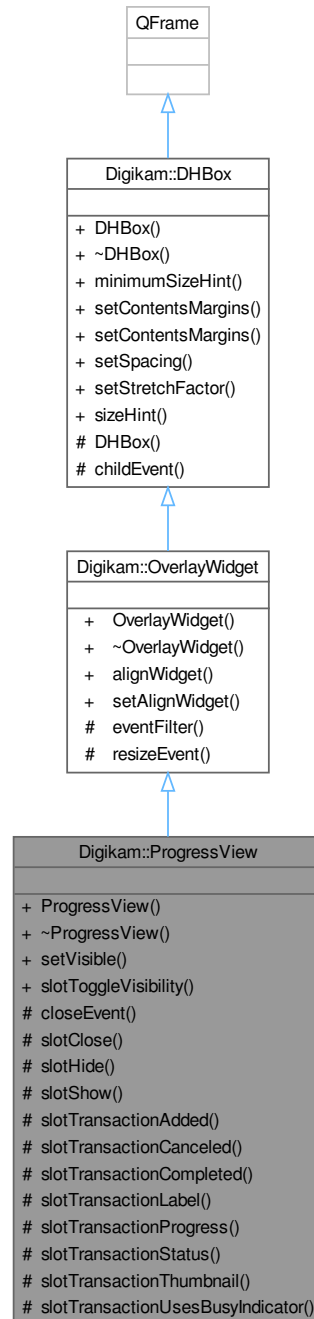
Provided for convenience.

Parameters

<i>item</i>	the canceled item.
-------------	--------------------

## 9.1062 Digikam::ProgressView Class Reference

Inheritance diagram for Digikam::ProgressView:



### Public Slots

- void **slotToggleVisibility** ()



## Signals

- void **visibilityChanged** (bool)

## Public Member Functions

- **ProgressView** (QWidget \*const alignWidget, QWidget \*const parent, const QString &name=QString())
- void **setVisible** (bool b) override

## Public Member Functions inherited from [Digikam::OverlayWidget](#)

- **OverlayWidget** (QWidget \*const alignWidget, QWidget \*const parent, const QString &name=QString())
- QWidget \* **alignWidget** () const
- void **setAlignWidget** (QWidget \*const alignWidget)

## Public Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (QWidget \*const parent=nullptr)
- QSize **minimumSizeHint** () const override
- void **setContentsMargins** (const QMargins &margins)
- void **setContentsMargins** (int left, int top, int right, int bottom)
- void **setSpacing** (int space)
- void **setStretchFactor** (QWidget \*const widget, int stretch)
- QSize **sizeHint** () const override

## Protected Slots

- void **slotClose** ()
- void **slotHide** ()
- void **slotShow** ()
- void **slotTransactionAdded** ([ProgressItem](#) \*)
- void **slotTransactionCanceled** ([ProgressItem](#) \*)
- void **slotTransactionCompleted** ([ProgressItem](#) \*)
- void **slotTransactionLabel** ([ProgressItem](#) \*, const QString &)
- void **slotTransactionProgress** ([ProgressItem](#) \*, unsigned int progress)
- void **slotTransactionStatus** ([ProgressItem](#) \*, const QString &)
- void **slotTransactionThumbnail** ([ProgressItem](#) \*, const QPixmap &)
- void **slotTransactionUsesBusyIndicator** ([ProgressItem](#) \*, bool)

## Protected Member Functions

- void **closeEvent** (QCloseEvent \*) override

## Protected Member Functions inherited from [Digikam::OverlayWidget](#)

- bool **eventFilter** (QObject \*o, QEvent \*e) override
- void **resizeEvent** (QResizeEvent \*ev) override

## Protected Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (bool vertical, QWidget \*const parent)
- void **childEvent** (QChildEvent \*e) override

## 9.1063 Digikam::ProxyClickLineEdit Class Reference

Inheritance diagram for Digikam::ProxyClickLineEdit:



## Signals

- void **leftClicked** ()

## Signals inherited from [Digikam::ProxyLineEdit](#)

- void **signalClearButtonPressed** ()

## Public Member Functions

- [ProxyClickLineEdit](#) (QWidget \*const parent=nullptr)  
*A [ProxyLineEdit](#) that emits leftClicked() on mouse press event.*

## Public Member Functions inherited from [Digikam::ProxyLineEdit](#)

- [ProxyLineEdit](#) (QWidget \*const parent=nullptr)  
*This class will not act as a QLineEdit at all, but present another widget (any kind of widget) instead in the space assigned to the QLineEdit.*
- void **setClearButtonShown** (bool show)
- virtual void **setWidget** (QWidget \*widget)  
*After constructing, set the actual widget here.*

## Protected Member Functions

- void **mouseReleaseEvent** (QMouseEvent \*event) override

## Protected Member Functions inherited from [Digikam::ProxyLineEdit](#)

- void **changeEvent** (QEvent \*event) override
- void **contextMenuEvent** (QContextMenuEvent \*event) override
- void **dragEnterEvent** (QDragEnterEvent \*event) override
- void **dragLeaveEvent** (QDragLeaveEvent \*e) override
- void **dragMoveEvent** (QDragMoveEvent \*e) override
- void **dropEvent** (QDropEvent \*event) override
- void **focusInEvent** (QFocusEvent \*event) override
- void **focusOutEvent** (QFocusEvent \*event) override
- void **inputMethodEvent** (QInputMethodEvent \*event) override
- void **keyPressEvent** (QKeyEvent \*event) override
- QSize **minimumSizeHint** () const override
- void **mouseDoubleClickEvent** (QMouseEvent \*event) override
- void **mouseMoveEvent** (QMouseEvent \*event) override  
*We just re-implement all relevant QWidget event handlers and call the QWidget implementation, not the QLineEdit one.*
- void **mousePressEvent** (QMouseEvent \*event) override  
*NOTE: see bug #326718: We need to use QLineEdit parent class with these methods to have clear button working fine.*
- void **mouseReleaseEvent** (QMouseEvent \*event) override
- void **paintEvent** (QPaintEvent \*event) override
- QSize **sizeHint** () const override

## Additional Inherited Members

### Protected Attributes inherited from [Digikam::ProxyLineEdit](#)

- `QVBoxLayout * m_layout = nullptr`
- `QWidget * m_widget = nullptr`

## 9.1063.1 Constructor & Destructor Documentation

### 9.1063.1.1 ProxyClickLineEdit()

```
Digikam::ProxyClickLineEdit::ProxyClickLineEdit (  
    QWidget *const parent = nullptr ) [explicit]
```

Press on the held widget will result in the signal if the widget does not accept() them.

## 9.1064 Digikam::ProxyLineEdit Class Reference

Inheritance diagram for Digikam::ProxyLineEdit:



### Signals

- void **signalClearButtonPressed** ()

## Public Member Functions

- [ProxyLineEdit](#) (QWidget \*const parent=nullptr)  
*This class will not act as a QLineEdit at all, but present another widget (any kind of widget) instead in the space assigned to the QLineEdit.*
- void **setClearButtonShown** (bool show)
- virtual void **setWidget** (QWidget \*widget)  
*After constructing, set the actual widget here.*

## Protected Member Functions

- void **changeEvent** (QEvent \*event) override
- void **contextMenuEvent** (QContextMenuEvent \*event) override
- void **dragEnterEvent** (QDragEnterEvent \*event) override
- void **dragLeaveEvent** (QDragLeaveEvent \*e) override
- void **dragMoveEvent** (QDragMoveEvent \*e) override
- void **dropEvent** (QDropEvent \*event) override
- void **focusInEvent** (QFocusEvent \*event) override
- void **focusOutEvent** (QFocusEvent \*event) override
- void **inputMethodEvent** (QInputMethodEvent \*event) override
- void **keyPressEvent** (QKeyEvent \*event) override
- QSize **minimumSizeHint** () const override
- void **mouseDoubleClickEvent** (QMouseEvent \*event) override
- void **mouseMoveEvent** (QMouseEvent \*event) override  
*We just re-implement all relevant QWidget event handlers and call the QWidget implementation, not the QLineEdit one.*
- void **mousePressEvent** (QMouseEvent \*event) override  
*NOTE: see bug #326718: We need to use QLineEdit parent class with these methods to have clear button working fine.*
- void **mouseReleaseEvent** (QMouseEvent \*event) override
- void **paintEvent** (QPaintEvent \*event) override
- QSize **sizeHint** () const override

## Protected Attributes

- QVBoxLayout \* **m\_layout** = nullptr
- QWidget \* **m\_widget** = nullptr

## 9.1064.1 Constructor & Destructor Documentation

### 9.1064.1.1 ProxyLineEdit()

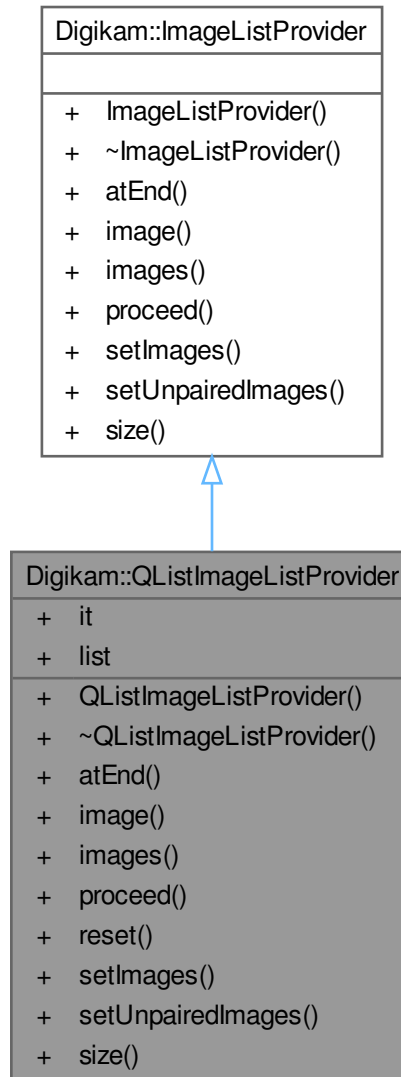
```
Digikam::ProxyLineEdit::ProxyLineEdit (
    QWidget *const parent = nullptr ) [explicit]
```

Use this class if you need to pass a QLineEdit but want actually to use a different widget.

## 9.1065 Digikam::QListImageListProvider Class Reference

A wrapper implementation for [ImageListProvider](#) if you have a QList of QImages.

Inheritance diagram for Digikam::QListImageListProvider:



### Public Member Functions

- bool `atEnd` () const override
- `QPair< QImage *, QString >` `image` () override
- `QList< QPair< QImage *, QString > >` `images` () override
- void `proceed` (int steps=1) override
- void `reset` ()
- void `setImages` (const `QList< QPair< QImage *, QString > >` &) override
- void `setUnpairedImages` (const `QList< QImage * >` &) override
- int `size` () const override

## Public Attributes

- `QList< QPair< QImage *, QString > >::const_iterator` **it**
- `QList< QPair< QImage *, QString > >` **list**

## 9.1065.1 Member Function Documentation

### 9.1065.1.1 atEnd()

```
bool Digikam::QListImageListProvider::atEnd ( ) const [override], [virtual]
```

Implements [Digikam::ImageListProvider](#).

### 9.1065.1.2 image()

```
QPair< QImage *, QString > Digikam::QListImageListProvider::image ( ) [override], [virtual]
```

Implements [Digikam::ImageListProvider](#).

### 9.1065.1.3 images()

```
QList< QPair< QImage *, QString > > Digikam::QListImageListProvider::images ( ) [override], [virtual]
```

Implements [Digikam::ImageListProvider](#).

### 9.1065.1.4 proceed()

```
void Digikam::QListImageListProvider::proceed (
    int steps = 1 ) [override], [virtual]
```

Implements [Digikam::ImageListProvider](#).

### 9.1065.1.5 setImages()

```
void Digikam::QListImageListProvider::setImages (
    const QList< QPair< QImage *, QString > > & lst ) [override], [virtual]
```

Implements [Digikam::ImageListProvider](#).

### 9.1065.1.6 setUnpairedImages()

```
void Digikam::QListImageListProvider::setUnpairedImages (
    const QList< QImage * > & lst ) [override], [virtual]
```

Implements [Digikam::ImageListProvider](#).



### 9.1065.1.7 size()

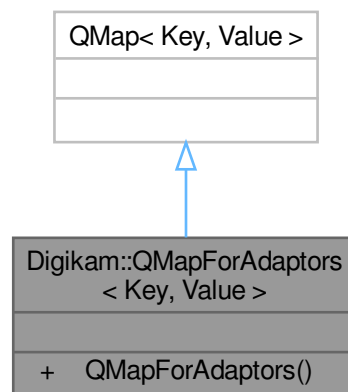
```
int Digikam::QListImageListProvider::size ( ) const [override], [virtual]
```

Implements [Digikam::ImageListProvider](#).

## 9.1066 Digikam::QMapForAdaptors< Key, Value > Class Template Reference

Adds the necessary typedefs so that `associative_property_map` accepts a `QMap`, and it can be used as a Boost Property Map.

Inheritance diagram for `Digikam::QMapForAdaptors< Key, Value >`:

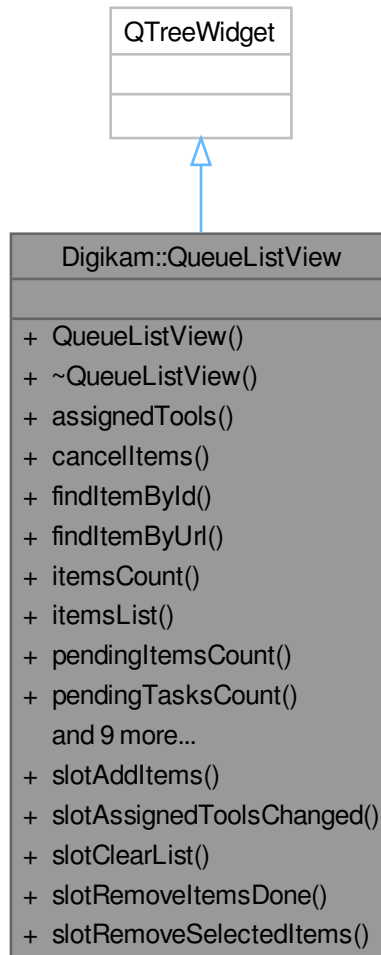


### Public Types

- typedef Value **data\_type**
- typedef Key **key\_type**
- typedef `std::pair< const Key, Value >` **value\_type**

## 9.1067 Digikam::QueueListView Class Reference

Inheritance diagram for Digikam::QueueListView:



### Public Types

- enum `ItemListType` { `Pending = 0` , `Selected` , `All` }

### Public Slots

- void `slotAddItems` (const `ItemInfoList` &)
- void `slotAssignedToolsChanged` (const `AssignedBatchTools` &)
- void `slotClearList` ()
- void `slotRemoveItemsDone` ()
- void `slotRemoveSelectedItems` ()

## Signals

- void **signalQueueContentsChanged** ()

## Public Member Functions

- **QueueListView** (QWidget \*const parent)
- **AssignedBatchTools** **assignedTools** () const
- void **cancellItems** ()
- **QueueListViewItem** \* **findItemById** (qulonglong id)
- **QueueListViewItem** \* **findItemByUrl** (const QUrl &url)
- int **itemsCount** ()
- **ItemInfoList** **itemsList** (**ItemListType** type)
- int **pendingItemsCount** ()
- int **pendingTasksCount** ()
- QPixmap **progressPixmapForIndex** (int index) const
- void **reloadThumbs** (const QUrl &url)
- void **removeItemById** (qulonglong id)
- void **removeItemByInfo** (const **ItemInfo** &info)
- void **setAssignedTools** (const **AssignedBatchTools** &tools)
- void **setEnabledToolTips** (bool val)
- void **setItemBusy** (qulonglong id)
- void **setSettings** (const **QueueSettings** &settings)
- **QueueSettings** **settings** () const

## 9.1067.1 Member Enumeration Documentation

### 9.1067.1.1 ItemListType

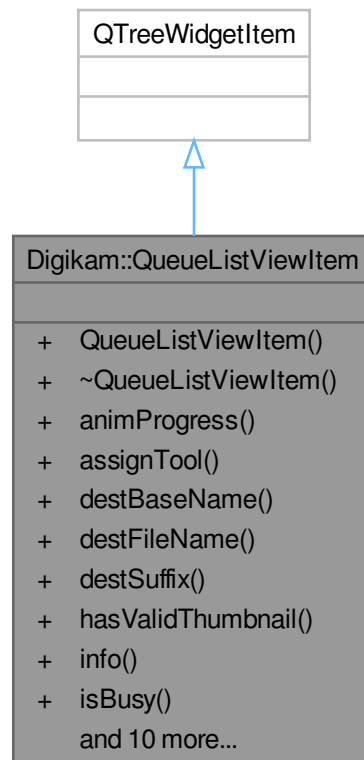
```
enum Digikam::QueueListView::ItemListType
```

#### Enumerator

Pending	Items from the list not yet processed.
Selected	Items from the list selected.
All	All items from the list.

## 9.1068 Digikam::QueueListViewItem Class Reference

Inheritance diagram for Digikam::QueueListViewItem:



### Public Member Functions

- **QueueListViewItem** (`QueueListView` \*const view, const `ItemInfo` &info)
- void **animProgress** ()
- void **assignTool** (int index, const `BatchToolSet` &set)
- `QString` **destBaseName** () const
- `QString` **destFileName** () const
- `QString` **destSuffix** () const
- bool **hasValidThumbnail** () const
- `ItemInfo` **info** () const
- bool **isBusy** () const
- bool **isDone** () const
- void **reset** ()
- void **setBusy** ()
- void **setCanceled** ()
- void **setDestFileName** (const `QString` &str)
- void **setDone** ()
- void **setFailed** ()
- void **setInfo** (const `ItemInfo` &info)
- void **setThumb** (const `QPixmap` &pix, bool hasThumb=true)
- void **unassignTool** (int index)

## 9.1069 Digikam::QueueMgrWindow Class Reference

Inheritance diagram for Digikam::QueueMgrWindow:



### Public Slots

- void **slotAssignQueueSettings** (const QString &)
- void **slotRun** ()

- void **slotRunAll** ()
- void **slotStop** ()
- void **slotUpdateQueueSettings** (const QString &)

### Signals

- void **signalBqmIsBusy** (bool)
- void **signalWindowHasMoved** ()

### Public Member Functions

- void **addNewQueue** ()
- void **applySettings** ()
- int **currentQueueId** () const
- [DInfoInterface](#) \* **infolface** ([DPluginAction](#) \*const) override  
*Return the interface instance to access to items information.*
- bool **isBusy** () const
- void **loadItemInfos** (const [ItemInfoList](#) &list, int queueId)
- void **loadItemInfosToCurrentQueue** (const [ItemInfoList](#) &list)
- void **loadItemInfosToNewQueue** (const [ItemInfoList](#) &list)
- bool **queryClose** () override
- [QueuePool](#) \* **queuePool** () const
- QMap< int, QString > **queuesMap** () const  
*Return a map of all queues available from pool (index and title).*
- void **refreshView** ()

### Public Member Functions inherited from [Digikam::DXmlGuiWindow](#)

- [DXmlGuiWindow](#) (QWidget \*const parent=nullptr, Qt::WindowFlags f=Qt::WindowFlags())
- QList< QAction \* > **allActions** () const  
*Return all actions from internal collection.*
- void **cleanupActions** ()  
*Cleanup unwanted actions from action collection.*
- QString **configGroupName** () const
- void **createFullScreenAction** (const QString &name)  
*Create Full-screen action to action collection instance from managed window set through [setManagedWindow\(\)](#).*
- void **createHelpActions** (const QString &handbookSection, bool coreOptions=true)  
*Create common actions from Help menu for all digiKam main windows.*
- void **createSettingsActions** ()  
*Create common actions to setup all digiKam main windows.*
- void **createSidebarActions** ()  
*Create common actions to handle side-bar through keyboard shortcuts.*
- bool **fullScreenIsActive** () const  
*Return true if managed window is currently in Full Screen Mode.*
- void **readFullScreenSettings** (const KConfigGroup &group)  
*Read full-screen settings from KDE config file.*
- virtual void **registerExtraPluginsActions** (QString &)
- void **registerPluginsActions** ()  
*Register all generic plugins action to this instance.*
- void **setConfigGroupName** (const QString &name)  
*Manage config group name used by window instance to get/set settings from config file.*
- void **setFullScreenOptions** (int options)  
*Set full-screen options to managed window.*
- void **unminimizeAndActivateWindow** ()

### Static Public Member Functions

- static [QueueMgrWindow](#) \* **queueManagerWindow** ()
- static bool **queueManagerWindowCreated** ()

### Static Public Member Functions inherited from [Digikam::DXmlGuiWindow](#)

- static QAction \* **buildStdAction** (StdActionType type, const QObject \*const recvr, const char \*const slot, QObject \*const parent)
- static QString **configFullScreenHideSideBarsEntry** ()
- static QString **configFullScreenHideStatusBarEntry** ()
- static QString **configFullScreenHideThumbBarEntry** ()
- static QString **configFullScreenHideToolBarsEntry** ()

*Shared with [FullScreenSettings](#).*

- static void **restoreWindowSize** (QWindow \*const win, const KConfigGroup &group)
- static void **saveWindowSize** (QWindow \*const win, KConfigGroup &group)
- static void **setGoodDefaultWindowSize** (QWindow \*const win)
- static void **setupIconTheme** ()

*If we have some local breeze icon resource, prefer it.*

### Protected Member Functions

- void **moveEvent** (QMoveEvent \*e) override

### Protected Member Functions inherited from [Digikam::DXmlGuiWindow](#)

- void **closeEvent** (QCloseEvent \*e) override
- void **editKeyboardShortcuts** (KActionCollection \*const extraac=nullptr, const QString &actitle=QString())  
*Call this method from your main window to show keyboard shortcut config dialog with an extra action collection to configure.*
- bool **eventFilter** (QObject \*obj, QEvent \*ev) override
- void **keyPressEvent** (QKeyEvent \*e) override
- QAction \* **showMenuBarAction** () const
- virtual void **showSideBars** (bool visible)  
*Re-implement this method if you want to manage sidebars visibility in full-screen mode.*
- QAction \* **showStatusBarAction** () const
- virtual void **showThumbBar** (bool visible)  
*Re-implement this method if you want to manage thumbbar visibility in full-screen mode.*
- virtual bool **thumbbarVisibility** () const  
*Re-implement this method if managed window has a thumbbar.*

### Additional Inherited Members

### Protected Slots inherited from [Digikam::DXmlGuiWindow](#)

- bool **slotClose** ()

### Protected Attributes inherited from [Digikam::DXmlGuiWindow](#)

- [DLogoAction](#) \* **m\_animLogo** = nullptr

## 9.1069.1 Member Function Documentation

### 9.1069.1.1 infoface()

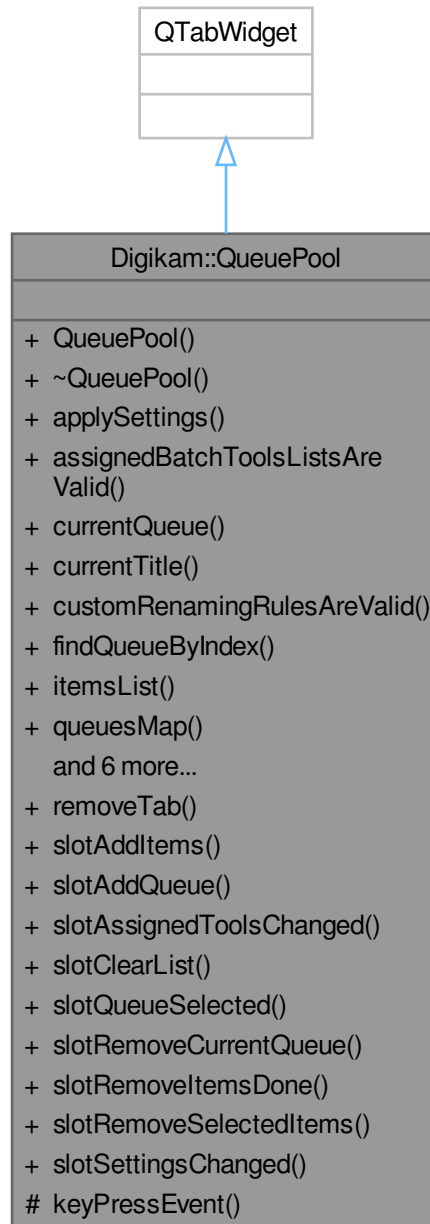
```
DInfoInterface * Digikam::QueueMgrWindow::infoIface (  
    DPluginAction * const ac ) [override], [virtual]
```

Implements [Digikam::DXmlGuiWindow](#).



## 9.1070 Digikam::QueuePool Class Reference

Inheritance diagram for Digikam::QueuePool:



### Public Slots

- void **removeTab** (int index)
- void **slotAddItems** (const [ItemInfoList](#) &, int queueId)
- void **slotAddQueue** ()

- void **slotAssignedToolsChanged** (const [AssignedBatchTools](#) &)
- void **slotClearList** ()
- void **slotQueueSelected** (int)
- void **slotRemoveCurrentQueue** ()
- void **slotRemoveItemsDone** ()
- void **slotRemoveSelectedItems** ()
- void **slotSettingsChanged** (const [QueueSettings](#) &)

## Signals

- void **signalItemSelectionChanged** ()
- void **signalQueueContentsChanged** ()
- void **signalQueuePoolChanged** ()
- void **signalQueueSelected** (int id, const [QueueSettings](#) &, const [AssignedBatchTools](#) &)

## Public Member Functions

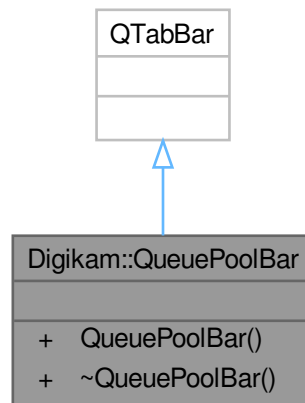
- **QueuePool** (QWidget \*const parent)
- void **applySettings** ()  
*Apply settings changes to all queues settings container when something have been changed in digiKam setup dialog.*
- bool **assignedBatchToolsListsAreValid** () const
- [QueueListView](#) \* **currentQueue** () const
- QString **currentTitle** () const
- bool **customRenamingRulesAreValid** () const
- [QueueListView](#) \* **findQueueByIndex** (int index) const
- [QueuePoolItemsList](#) **itemsList** (int index, int type) const
- QMap< int, QString > **queuesMap** () const
- QString **queueTitle** (int index) const
- bool **saveWorkflow** () const
- void **setBusy** (bool b)
- void **setItemBusy** (qulonglong id)
- int **totalPendingItems** () const
- int **totalPendingTasks** () const

## Protected Member Functions

- void **keyPressEvent** (QKeyEvent \*event) override

## 9.1071 Digikam::QueuePoolBar Class Reference

Inheritance diagram for Digikam::QueuePoolBar:



### Signals

- void **signalTestCanDecode** (const QDragMoveEvent \*, bool &)

### Public Member Functions

- **QueuePoolBar** (QWidget \*const parent)

## 9.1072 Digikam::QueueSettings Class Reference

This container host all common settings used by a queue, not including assigned batch tools.

### Public Types

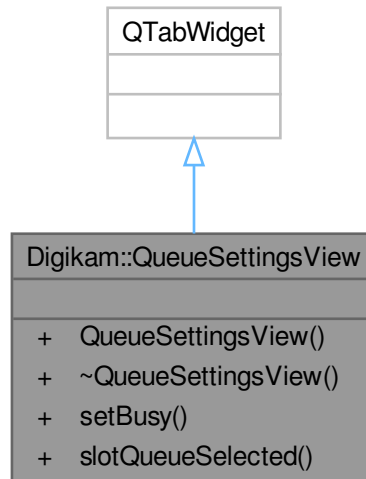
- enum **RawLoadingRule** { USEEMBEDEDJPEG = 0 , DEMOSAICING }
- enum **RenamingRule** { USEORIGINAL = 0 , CUSTOMIZE }

### Public Attributes

- FileSaveConflictBox::ConflictRule **conflictRule** = FileSaveConflictBox::DIFFNAME
- bool **exifSetOrientation** = true  
*Setting managed through Metadata control panel.*
- [IOFileSettings](#) **ioFileSettings**
- [DRawDecoderSettings](#) **rawDecodingSettings**
- RawLoadingRule **rawLoadingRule** = DEMOSAICING
- QString **renamingParser**
- RenamingRule **renamingRule** = USEORIGINAL
- bool **saveAsNewVersion** = true
- bool **useMultiCoreCPU** = false
- bool **useOrgAlbum** = true  
*If true, original file dir will be used to process queue items.*
- QUrl **workingUrl**

## 9.1073 Digikam::QueueSettingsView Class Reference

Inheritance diagram for Digikam::QueueSettingsView:



### Public Slots

- void **slotQueueSelected** (int, const [QueueSettings](#) &, const [AssignedBatchTools](#) &)

### Signals

- void **signalSettingsChanged** (const [QueueSettings](#) &)

### Public Member Functions

- **QueueSettingsView** (QWidget \*const parent=nullptr)
- void **setBusy** (bool b)

## 9.1074 Digikam::QueueToolTip Class Reference

Inheritance diagram for Digikam::QueueToolTip:



### Public Member Functions

- **QueueToolTip** ([QueueListView](#) \*const view)
- void **setQueueItem** ([QueueListViewItem](#) \*const item)

### Public Member Functions inherited from [Digikam::DItemToolTip](#)

- **DItemToolTip** (QWidget \*const parent=nullptr)

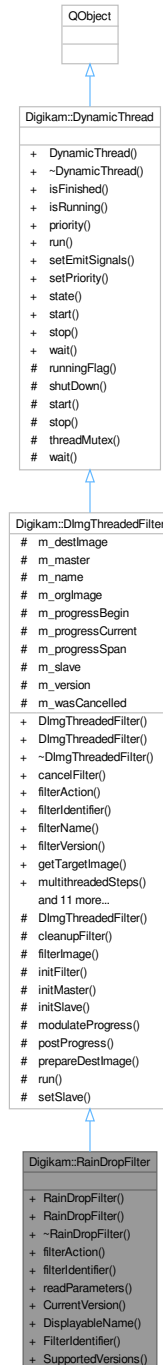
### Additional Inherited Members

### Protected Member Functions inherited from [Digikam::DItemToolTip](#)

- bool **event** (QEvent \*) override
- void **paintEvent** (QPaintEvent \*) override
- void **renderArrows** ()
- void **reposition** ()
- void **resizeEvent** (QResizeEvent \*) override
- bool **toolTipsEmpty** () const
- void **updateToolTip** ()

## 9.1075 Digikam::RainDropFilter Class Reference

Inheritance diagram for Digikam::RainDropFilter:



### Public Member Functions

- **RainDropFilter** (`DImg *const orgImage`, `QObject *const parent=nullptr`, `int drop=80`, `int amount=150`, `int coeff=30`, `const QRect &selection=QRect(0, 0, 0, 0)`)

- **RainDropFilter** (QObject \*const parent=nullptr)
- **FilterAction filterAction** () override  
*Returns the action description corresponding to currently set options.*
- QString **filterIdentifier** () const override  
*Return the identifier for this filter in the image history.*
- void **readParameters** (const **FilterAction** &action) override

## Public Member Functions inherited from **Digikam::DImgThreadedFilter**

- **DImgThreadedFilter** (DImg \*const orgImage, QObject \*const parent, const QString &name=QString())  
*Constructs a filter with all arguments (ready to use).*
- **DImgThreadedFilter** (QObject \*const parent=nullptr, const QString &name=QString())  
*Constructs a filter without argument.*
- virtual void **cancelFilter** ()  
*Cancel the threaded computation.*
- const QString & **filterName** ()
- int **filterVersion** () const
- **DImg getTargetImage** ()
- QList< int > **multithreadedSteps** (int stop, int start=0) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool **parametersSuccessfullyRead** () const  
*Optional: error handling for readParameters.*
- virtual QString **readParametersError** (const **FilterAction** &actionThatFailed) const
- void **setFilterName** (const QString &name)
- void **setFilterVersion** (int version)  
*Replaying a filter action: Set the filter version.*
- void **setOriginalImage** (const **DImg** &orgImage)
- void **setupAndStartDirectly** (const **DImg** &orgImage, **DImgThreadedFilter** \*const master, int progress←Begin=0, int progressEnd=100)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void **setupFilter** (const **DImg** &orgImage)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void **startFilter** ()  
*Start the threaded computation.*
- virtual void **startFilterDirectly** ()  
*Start computation of this filter, directly in this thread.*
- virtual QList< int > **supportedVersions** () const

## Public Member Functions inherited from **Digikam::DynamicThread**

- **DynamicThread** (QObject \*const parent=nullptr)  
*This class extends QRunnable, so you have to reimplement virtual void [run\(\)](#).*
- **~DynamicThread** () override  
*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool **isFinished** () const
- bool **isRunning** () const
- QThread::Priority **priority** () const
- void **setEmitSignals** (bool emitThem)
- void **setPriority** (QThread::Priority priority)  
*Sets the priority for this dynamic thread.*
- State **state** () const



### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from [Digikam::DImgThreadedFilter](#)

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.1075.1 Member Function Documentation

### 9.1075.1.1 filterAction()

`FilterAction` Digikam::RainDropFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.1075.1.2 filterIdentifier()

`QString` Digikam::RainDropFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.1075.1.3 readParameters()

```
void Digikam::RainDropFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

## 9.1076 Digikam::RandomNumberGenerator Class Reference

This class differs from standard pseudo random number generators (rand()) in these points:

### Public Member Functions

- [RandomNumberGenerator](#) ( )  
*Constructs a random number generator that is seeded with a constant value.*
- quint32 [currentSeed](#) ( ) const  
*Retrieves the current seed.*
- double **number** (double min, double max)  
*Returns a random double in the interval [min, max) (including min, excluding max) Warning: this method is non re-entrant.*
- int [number](#) (int min, int max)  
*Returns a random integer in the interval [min, max] (including min and max).*
- void [reseed](#) ( )  
*Seeds the generator again with the [currentSeed](#)().*
- void [seed](#) (quint32 seed)  
*Seeds the generator with the given value.*
- quint32 [seedByTime](#) ( )  
*Seeds the generator by current time.*
- quint32 [seedNonDeterministic](#) ( )  
*Seeds the generator from a non-deterministic random number generator.*
- bool **yesOrNo** (double p)  
*Returns true with a probability of p (where p shall be in the interval [0, 1]) Warning: this method is non re-entrant.*

## Static Public Member Functions

- static quint32 **nonDeterministicSeed** ()  
*Produces a non-deterministic seed, as used by [seedNonDeterministic\(\)](#)*
- static quint32 **timeSeed** ()  
*Produces a seed that includes at least the time as source of random data.*

### 9.1076.1 Detailed Description

- it uses a specified, independently implemented algorithm identical across platforms
- provides access to the used seed
- it can thus guarantee replayable sequences
- it provides convenient seeding of varying quality

### 9.1076.2 Constructor & Destructor Documentation

#### 9.1076.2.1 RandomNumberGenerator()

```
Digikam::RandomNumberGenerator::RandomNumberGenerator ( ) [explicit]
```

It is recommended to call a seed method after construction.

### 9.1076.3 Member Function Documentation

#### 9.1076.3.1 currentSeed()

```
quint32 Digikam::RandomNumberGenerator::currentSeed ( ) const
```

Can be used for [seed\(quint32\)](#) to replay the results again.

#### 9.1076.3.2 number()

```
int Digikam::RandomNumberGenerator::number (
    int min,
    int max )
```

Warning: this method is non re-entrant.

#### 9.1076.3.3 reseed()

```
void Digikam::RandomNumberGenerator::reseed ( )
```

This is not a no-op, rather, the sequence of random numbers starts again from its beginning after each re-seed. Equivalent to `seed(currentSeed())`

**9.1076.3.4 seed()**

```
void Digikam::RandomNumberGenerator::seed (
    quint32 seed )
```

This is not meant to be called with a constant value, but with a value retrieved from [currentSeed\(\)](#) on a previous run. Across platforms, the same sequence of random numbers will be generated for the same seed.

**9.1076.3.5 seedByTime()**

```
quint32 Digikam::RandomNumberGenerator::seedByTime ( )
```

This is common practice and good enough for most purposes. Returns the new [currentSeed\(\)](#).

**9.1076.3.6 seedNonDeterministic()**

```
quint32 Digikam::RandomNumberGenerator::seedNonDeterministic ( )
```

This is the most secure seeding method. Returns the new [currentSeed\(\)](#).

**9.1077 Digikam::RangeDialog Class Reference**

Inheritance diagram for Digikam::RangeDialog:



**Public Member Functions**

- **RangeDialog** ([Rule](#) \*const parent)

**Public Member Functions inherited from [Digikam::RuleDialog](#)**

- **RuleDialog** ([Rule](#) \*const parent)
- void **setSettingsWidget** (QWidget \*const settingsWidget)

**Public Attributes**

- Ui::RangeModifierDialogWidget \*const **ui** = nullptr

## 9.1078 Digikam::RangeModifier Class Reference

Inheritance diagram for Digikam::RangeModifier:



### Public Member Functions

- `QString parseOperation (ParseSettings &settings, const QRegularExpressionMatch &match)` override  
*TODO: describe me.*

## Public Member Functions inherited from [Digikam::Modifier](#)

- **Modifier** (const QString &name, const QString &description)
- **Modifier** (const QString &name, const QString &description, const QString &icon)

## Public Member Functions inherited from [Digikam::Rule](#)

- **Rule** (const QString &name)
- **Rule** (const QString &name, const QString &icon)
- QString **description** () const
- QPixmap **icon** (Rule::IconType type=Rule::Action) const
- bool **isValid** () const

*Checks the validity of the parse object.*

- **ParseResults parse** ([ParseSettings](#) &settings)
- QRegularExpression & **regExp** () const
- TODO: This is probably not needed anymore.*
- QPushButton \* **registerButton** (QWidget \*parent)
- Register a button in the parent object.*
- QAction \* **registerMenu** (QMenu \*parent)
- Register a menu action in the parent object.*
- virtual void **reset** ()
- Resets the parser to its initial state.*
- TokenList & **tokens** () const
- bool **useTokenMenu** () const
- Returns true if a token menu is used.*

## Additional Inherited Members

## Public Types inherited from [Digikam::Rule](#)

- enum **IconType** { **Action** = 0 , **Dialog** }

## Signals inherited from [Digikam::Rule](#)

- void **signalTokenTriggered** (const QString &)

## Static Public Member Functions inherited from [Digikam::Rule](#)

- static QString **escapeToken** (const QString &token)
- Escape the token characters to make them work in regular expressions.*

## Protected Slots inherited from [Digikam::Rule](#)

- virtual void **slotTokenTriggered** (const QString &)



## Protected Member Functions inherited from [Digikam::Rule](#)

- bool [addToken](#) (const QString &id, const QString &description, const QString &actionName=QString())  
*add a token to the parser, every parser should at least assign one token object*
- void [setDescription](#) (const QString &desc)
- void [setIcon](#) (const QString &pixmap)
- void [setRegExp](#) (const QRegularExpression &regExp)
- void [setUseTokenMenu](#) (bool value)  
*If multiple tokens have been assigned to a rule, a menu will be created.*

### 9.1078.1 Member Function Documentation

#### 9.1078.1.1 [parseOperation\(\)](#)

```
QString Digikam::RangeModifier::parseOperation (
    ParseSettings & settings,
    const QRegularExpressionMatch & match ) [override], [virtual]
```

##### Parameters

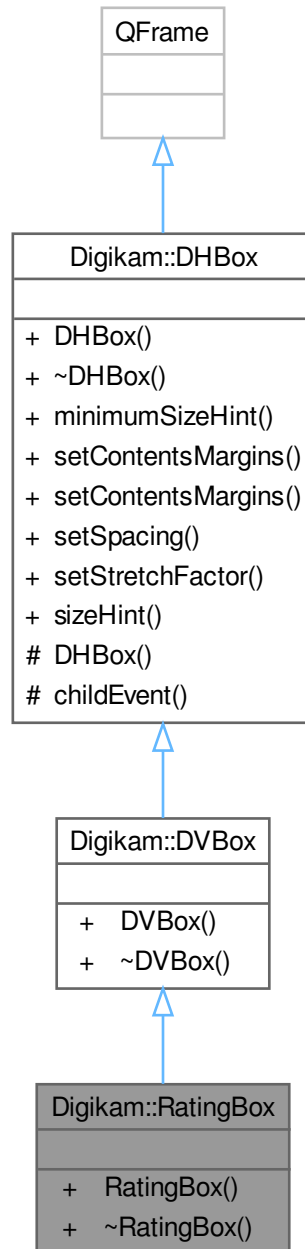
<i>settings</i>	contains settings
<i>match</i>	result of the regular expression match done in <a href="#">Option::parse()</a>

##### Returns

Implements [Digikam::Modifier](#).

## 9.1079 Digikam::RatingBox Class Reference

Inheritance diagram for Digikam::RatingBox:



### Signals

- void **signalRatingChanged** (int)

### Public Member Functions

- **RatingBox** (QWidget \*const parent)

### Public Member Functions inherited from [Digikam::DVBox](#)

- **DVBox** (QWidget \*const parent=nullptr)

### Public Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (QWidget \*const parent=nullptr)
- QSize **minimumSizeHint** () const override
- void **setContentMargins** (const QMargins &margins)
- void **setContentMargins** (int left, int top, int right, int bottom)
- void **setSpacing** (int space)
- void **setStretchFactor** (QWidget \*const widget, int stretch)
- QSize **sizeHint** () const override

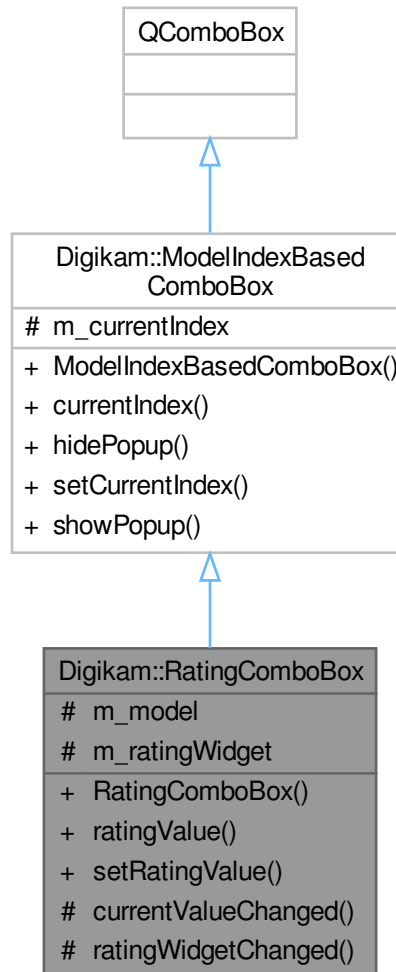
### Additional Inherited Members

### Protected Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (bool vertical, QWidget \*const parent)
- void **childEvent** (QChildEvent \*e) override

## 9.1080 Digikam::RatingComboBox Class Reference

Inheritance diagram for Digikam::RatingComboBox:



### Public Types

- enum `RatingValue` {  
`Null = -2` , `NoRating = -1` , `Rating0 = 0` , `Rating1 = 1` ,  
`Rating2 = 2` , `Rating3 = 3` , `Rating4 = 4` , `Rating5 = 5` }

*An advanced widget for entering a rating, including support for Null and NoRating values.*

### Signals

- void `ratingValueChanged` (int value)

## Public Member Functions

- **RatingComboBox** (QWidget \*const parent=nullptr)
- [RatingValue](#) **ratingValue** () const
- void **setRatingValue** ([RatingValue](#) value)

## Public Member Functions inherited from [Digikam::ModelIndexBasedComboBox](#)

- [ModelIndexBasedComboBox](#) (QWidget \*const parent=nullptr)  
*QComboBox has a current index based on a single integer.*
- QModelIndex **currentIndex** () const
- void **hidePopup** () override
- void **setCurrentIndex** (const QModelIndex &index)
- void **showPopup** () override

## Protected Slots

- void **currentValueChanged** (const QModelIndex &current, const QModelIndex &previous)
- void **ratingWidgetChanged** (int)

## Protected Attributes

- [RatingComboBoxModel](#) \* **m\_model** = nullptr
- [RatingComboBoxWidget](#) \* **m\_ratingWidget** = nullptr

## Protected Attributes inherited from [Digikam::ModelIndexBasedComboBox](#)

- QPersistentModelIndex **m\_currentIndex**

## 9.1080.1 Member Enumeration Documentation

### 9.1080.1.1 RatingValue

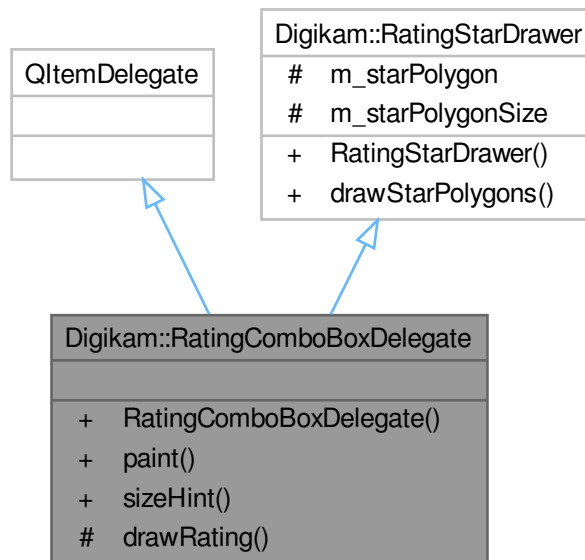
```
enum Digikam::RatingComboBox::RatingValue
```

#### Enumerator

Null	The rating value. All values except Null correspond to the integers used by the database.
------	---

## 9.1081 Digikam::RatingComboBoxDelegate Class Reference

Inheritance diagram for Digikam::RatingComboBoxDelegate:



### Public Member Functions

- **RatingComboBoxDelegate** (`QObject *const parent=nullptr`)
- void **paint** (`QPainter *painter`, `const QStyleOptionViewItem &option`, `const QModelIndex &index`) const override
- `QSize` **sizeHint** (`const QStyleOptionViewItem &option`, `const QModelIndex &index`) const override

### Public Member Functions inherited from [Digikam::RatingStarDrawer](#)

- `QRect` **drawStarPolygons** (`QPainter *p`, `int numberOfStars`) const

### Protected Member Functions

- void **drawRating** (`QPainter *painter`, `const QRect &rect`, `int rating`, `bool selectable`) const

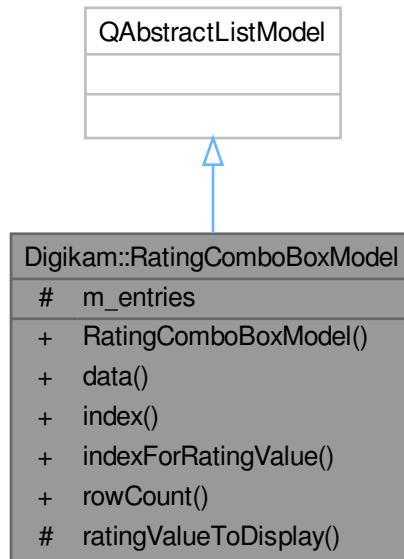
### Additional Inherited Members

### Protected Attributes inherited from [Digikam::RatingStarDrawer](#)

- `QPolygon` **m\_starPolygon** = `RatingWidget::starPolygon()`
- `QSize` **m\_starPolygonSize** = `QSize(15, 15)`

## 9.1082 Digikam::RatingComboBoxModel Class Reference

Inheritance diagram for Digikam::RatingComboBoxModel:



### Public Types

- enum **CustomRoles** { **RatingRole** = Qt::UserRole }

### Public Member Functions

- **RatingComboBoxModel** (QObject \*const parent=nullptr)
- QVariant **data** (const QModelIndex &index, int role) const override
- QModelIndex **index** (int row, int column=0, const QModelIndex &parent=QModelIndex()) const override
- QModelIndex **indexForRatingValue** ([RatingComboBox::RatingValue](#) value) const
- int **rowCount** (const QModelIndex &parent) const override

### Protected Member Functions

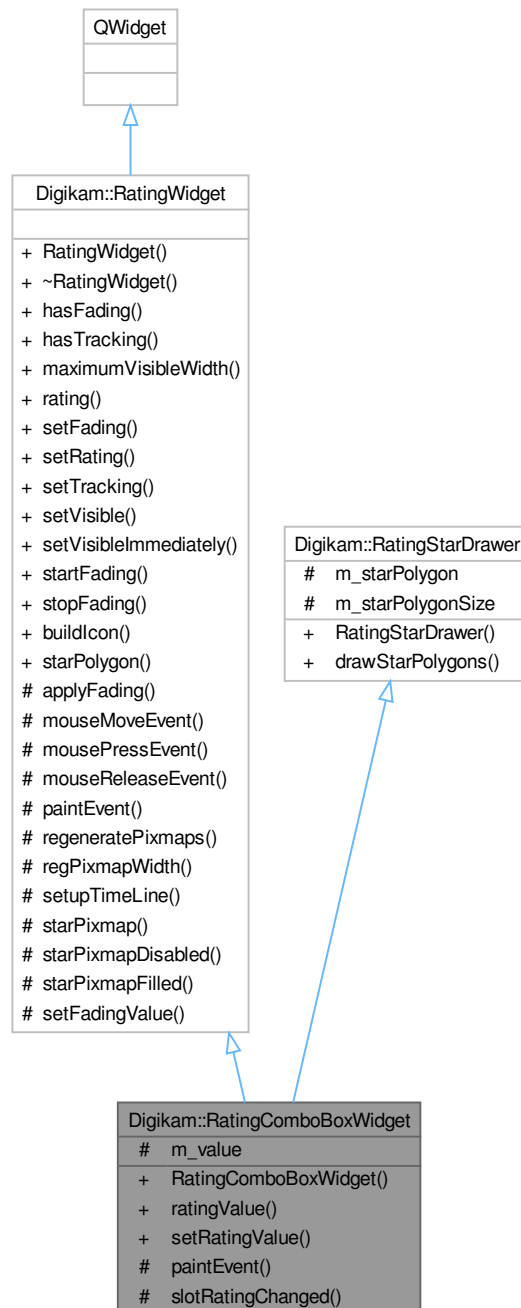
- QVariant **ratingValueToDisplay** ([RatingComboBox::RatingValue](#) value) const

### Protected Attributes

- QList< [RatingComboBox::RatingValue](#) > **m\_entries**

## 9.1083 Digikam::RatingComboBoxWidget Class Reference

Inheritance diagram for Digikam::RatingComboBoxWidget:



### Signals

- void **ratingValueChanged** (int value)



## Signals inherited from [Digikam::RatingWidget](#)

- void **signalRatingChanged** (int)
- void **signalRatingModified** (int)

*Not managed by tracking properties.*

## Public Member Functions

- **RatingComboBoxWidget** (QWidget \*const parent=nullptr)  
*Internal sub-classing the classic [RatingWidget](#), this provides support for the Null and NoRating states.*
- [RatingComboBox::RatingValue](#) **ratingValue** () const
- void **setRatingValue** ([RatingComboBox::RatingValue](#) value)

## Public Member Functions inherited from [Digikam::RatingWidget](#)

- **RatingWidget** (QWidget \*const parent)
- bool **hasFading** () const
- bool **hasTracking** () const
- int **maximumVisibleWidth** () const
- int **rating** () const
- void **setFading** (bool fading)
- void **setRating** (int val)
- void **setTracking** (bool tracking)
- void **setVisible** (bool visible) override
- void **setVisibleImmediately** ()
- void **startFading** ()
- void **stopFading** ()

## Public Member Functions inherited from [Digikam::RatingStarDrawer](#)

- QRect **drawStarPolygons** (QPainter \*p, int numberOfStars) const

## Protected Slots

- void **slotRatingChanged** (int)

## Protected Slots inherited from [Digikam::RatingWidget](#)

- void **setFadingValue** (int value)

## Protected Member Functions

- void **paintEvent** (QPaintEvent \*) override

## Protected Member Functions inherited from [Digikam::RatingWidget](#)

- void **applyFading** (QPixmap &pix)
- void **mouseMoveEvent** (QMouseEvent \*) override
- void **mousePressEvent** (QMouseEvent \*) override
- void **mouseReleaseEvent** (QMouseEvent \*) override
- void **paintEvent** (QPaintEvent \*) override
- void **regeneratePixmap** ()
- int **regPixmapWidth** () const
- void **setupTimeLine** ()
- QPixmap **starPixmap** () const
- QPixmap **starPixmapDisabled** () const
- QPixmap **starPixmapFilled** () const

## Protected Attributes

- [RatingComboBox::RatingValue](#) **m\_value** = [RatingComboBox::Null](#)

## Protected Attributes inherited from [Digikam::RatingStarDrawer](#)

- QPolygon **m\_starPolygon** = [RatingWidget::starPolygon](#)()
- QSize **m\_starPolygonSize** = QSize(15, 15)

## Additional Inherited Members

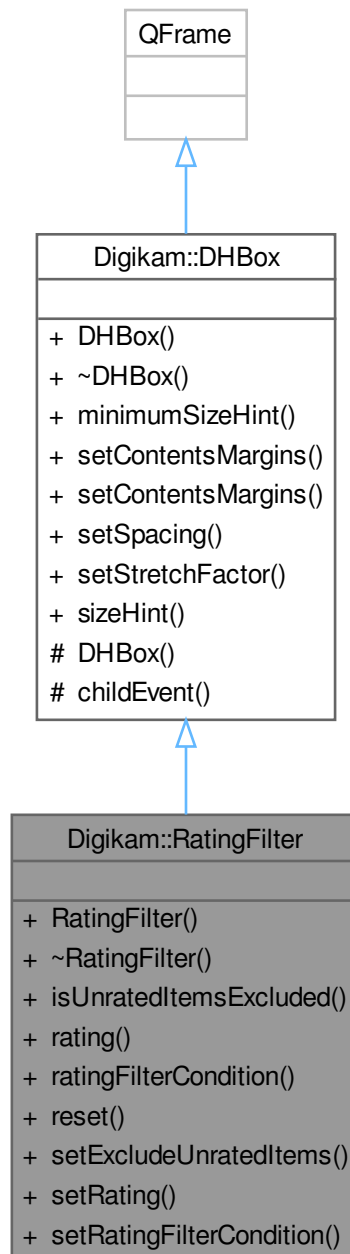
## Static Public Member Functions inherited from [Digikam::RatingWidget](#)

- static QIcon **buildIcon** (int rate, int size)
- static QPolygon **starPolygon** ()

*Pre-computed star polygon for a 15x15 pixmap.*

## 9.1084 Digikam::RatingFilter Class Reference

Inheritance diagram for Digikam::RatingFilter:



### Signals

- void **signalRatingFilterChanged** (int, [ItemFilterSettings::RatingCondition](#), bool)

### Public Member Functions

- **RatingFilter** (QWidget \*const parent)
- bool **isUnratedItemsExcluded** ()
- int **rating** () const
- [ItemFilterSettings::RatingCondition](#) **ratingFilterCondition** ()
- void **reset** ()
- void **setExcludeUnratedItems** (bool excluded)
- void **setRating** (int val)
- void **setRatingFilterCondition** ([ItemFilterSettings::RatingCondition](#) cond)

### Public Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (QWidget \*const parent=nullptr)
- QSize **minimumSizeHint** () const override
- void **setContentsMargins** (const QMargins &margins)
- void **setContentsMargins** (int left, int top, int right, int bottom)
- void **setSpacing** (int space)
- void **setStretchFactor** (QWidget \*const widget, int stretch)
- QSize **sizeHint** () const override

### Additional Inherited Members

### Protected Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (bool vertical, QWidget \*const parent)
- void **childEvent** (QChildEvent \*e) override

## 9.1085 Digikam::RatingFilterWidget Class Reference

Inheritance diagram for Digikam::RatingFilterWidget:



### Signals

- void **signalRatingFilterChanged** (int, [ItemFilterSettings::RatingCondition](#), bool)

## Signals inherited from [Digikam::RatingWidget](#)

- void **signalRatingChanged** (int)
- void **signalRatingModified** (int)

*Not managed by tracking properties.*

## Public Member Functions

- **RatingFilterWidget** (QWidget \*const parent)
- bool **isUnratedItemsExcluded** ()
- [ItemFilterSettings::RatingCondition](#) **ratingFilterCondition** ()
- void **setExcludeUnratedItems** (bool excluded)
- void **setRatingFilterCondition** ([ItemFilterSettings::RatingCondition](#) cond)

## Public Member Functions inherited from [Digikam::RatingWidget](#)

- **RatingWidget** (QWidget \*const parent)
- bool **hasFading** () const
- bool **hasTracking** () const
- int **maximumVisibleWidth** () const
- int **rating** () const
- void **setFading** (bool fading)
- void **setRating** (int val)
- void **setTracking** (bool tracking)
- void **setVisible** (bool visible) override
- void **setVisibleImmediately** ()
- void **startFading** ()
- void **stopFading** ()

## Protected Member Functions

- void **mouseMoveEvent** (QMouseEvent \*) override
- void **mousePressEvent** (QMouseEvent \*) override
- void **mouseReleaseEvent** (QMouseEvent \*) override

## Protected Member Functions inherited from [Digikam::RatingWidget](#)

- void **applyFading** (QPixmap &pix)
- void **mouseMoveEvent** (QMouseEvent \*) override
- void **mousePressEvent** (QMouseEvent \*) override
- void **mouseReleaseEvent** (QMouseEvent \*) override
- void **paintEvent** (QPaintEvent \*) override
- void **regeneratePxmmaps** ()
- int **regPixmapWidth** () const
- void **setupTimeLine** ()
- QPixmap **starPixmap** () const
- QPixmap **starPixmapDisabled** () const
- QPixmap **starPixmapFilled** () const

### Additional Inherited Members

### Static Public Member Functions inherited from [Digikam::RatingWidget](#)

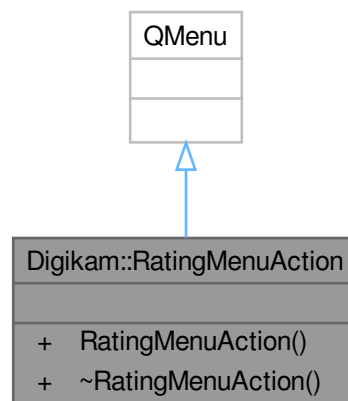
- static QIcon **buildIcon** (int rate, int size)
- static QPolygon **starPolygon** ()  
*Pre-computed star polygon for a 15x15 pixmap.*

### Protected Slots inherited from [Digikam::RatingWidget](#)

- void **setFadingValue** (int value)

## 9.1086 Digikam::RatingMenuAction Class Reference

Inheritance diagram for Digikam::RatingMenuAction:



### Signals

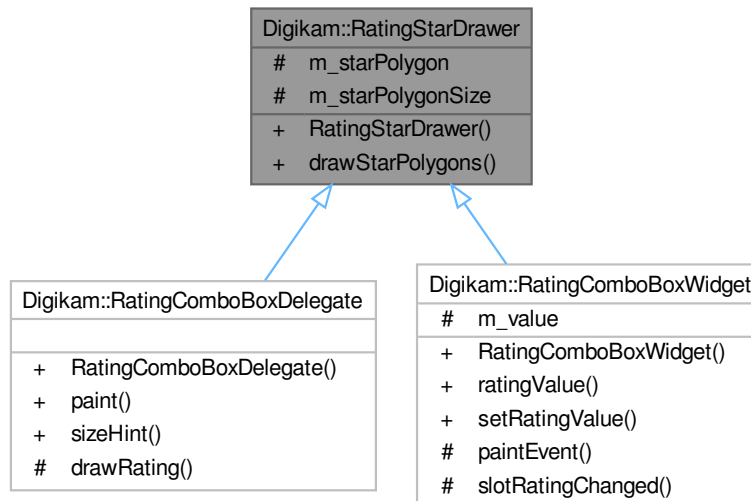
- void **signalRatingChanged** (int)

### Public Member Functions

- **RatingMenuAction** (QMenu \*const parent=nullptr)

## 9.1087 Digikam::RatingStarDrawer Class Reference

Inheritance diagram for Digikam::RatingStarDrawer:



### Public Member Functions

- `QRect drawStarPolygons (QPainter *p, int numberOfStars) const`

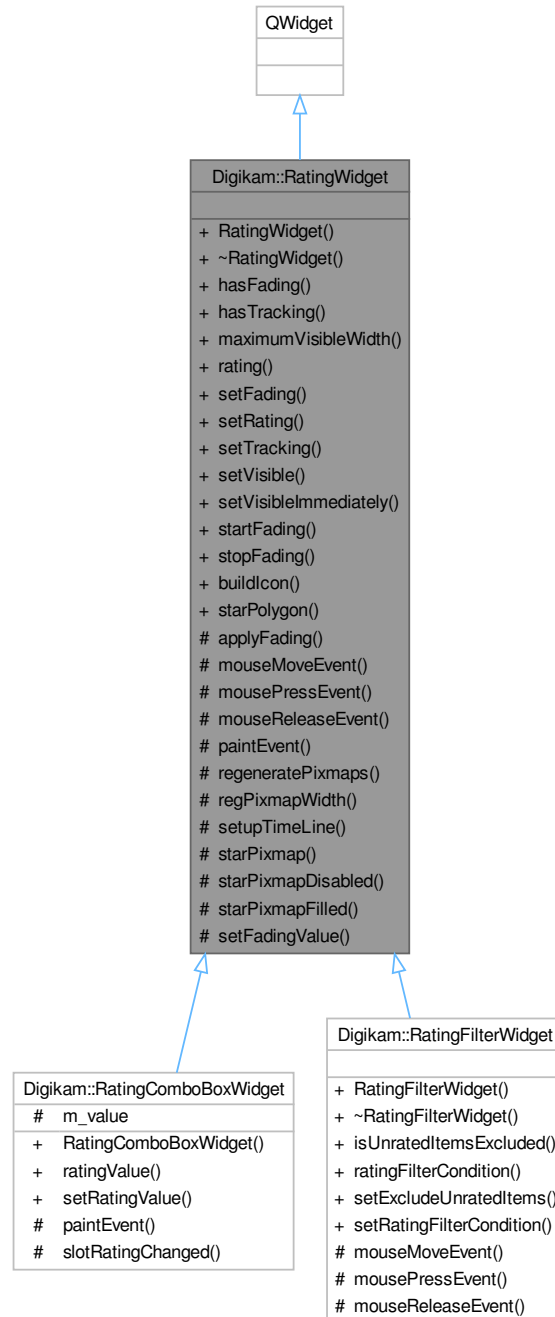
### Protected Attributes

- `QPolygon m_starPolygon = RatingWidget::starPolygon()`
- `QSize m_starPolygonSize = QSize(15, 15)`



## 9.1088 Digikam::RatingWidget Class Reference

Inheritance diagram for Digikam::RatingWidget:



### Signals

- void **signalRatingChanged** (int)
- void **signalRatingModified** (int)

*Not managed by tracking properties.*

## Public Member Functions

- **RatingWidget** (QWidget \*const parent)
- bool **hasFading** () const
- bool **hasTracking** () const
- int **maximumVisibleWidth** () const
- int **rating** () const
- void **setFading** (bool fading)
- void **setRating** (int val)
- void **setTracking** (bool tracking)
- void **setVisible** (bool visible) override
- void **setVisibleImmediately** ()
- void **startFading** ()
- void **stopFading** ()

## Static Public Member Functions

- static QIcon **buildIcon** (int rate, int size)
- static QPolygon **starPolygon** ()  
*Pre-computed star polygon for a 15x15 pixmap.*

## Protected Slots

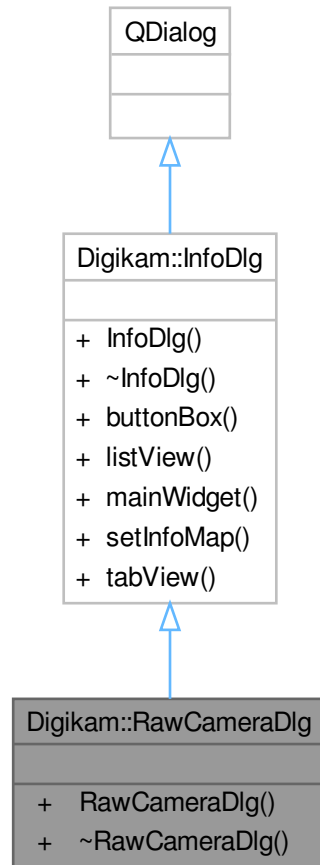
- void **setFadingValue** (int value)

## Protected Member Functions

- void **applyFading** (QPixmap &pix)
- void **mouseMoveEvent** (QMouseEvent \*) override
- void **mousePressEvent** (QMouseEvent \*) override
- void **mouseReleaseEvent** (QMouseEvent \*) override
- void **paintEvent** (QPaintEvent \*) override
- void **regeneratePxmmaps** ()
- int **regPixmapWidth** () const
- void **setupTimeLine** ()
- QPixmap **starPixmap** () const
- QPixmap **starPixmapDisabled** () const
- QPixmap **starPixmapFilled** () const

## 9.1089 Digikam::RawCameraDlg Class Reference

Inheritance diagram for Digikam::RawCameraDlg:



### Public Member Functions

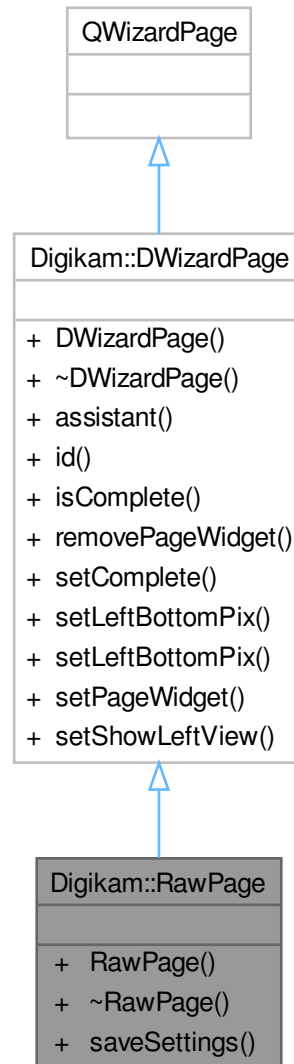
- **RawCameraDlg** (QWidget \*const parent)

### Public Member Functions inherited from [Digikam::InfoDlg](#)

- **InfoDlg** (QWidget \*const parent)
- QDialogButtonBox \* **buttonBox** () const
- QTreeWidget \* **listView** () const
- QWidget \* **mainWidget** () const
- virtual void **setInfoMap** (const QMap< QString, QString > &list)
- QTabWidget \* **tabView** () const

## 9.1090 Digikam::RawPage Class Reference

Inheritance diagram for Digikam::RawPage:



### Public Member Functions

- **RawPage** (`QWizard *const dlg`)
- void **saveSettings** ()

### Public Member Functions inherited from [Digikam::DWizardPage](#)

- **DWizardPage** (`QWizard *const dlg, const QString &title`)
- `QWizard * assistant () const`

- int **id** () const
- bool **isComplete** () const override
- void **removePageWidget** (QWidget \*const w)
- void **setComplete** (bool b)
- void **setLeftBottomPix** (const QIcon &icon)
- void **setLeftBottomPix** (const QPixmap &pix)
- void **setPageWidget** (QWidget \*const w)
- void **setShowLeftView** (bool v)

## 9.1091 Digikam::RawProcessingFilter Class Reference

This is a special filter.

Inheritance diagram for Digikam::RawProcessingFilter:



## Public Member Functions

- **RawProcessingFilter** (const [DRawDecoding](#) &settings, [DimgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const [QString](#) &name=QString())

*For use with a master filter.*

- **RawProcessingFilter** ([DImg](#) \*const orgImage, [QObject](#) \*const parent, const [DRawDecoding](#) &settings, const [QString](#) &name=QString())

- *Traditional constructor.*
- [RawProcessingFilter](#) (QObject \*const parent=nullptr)
- *Default constructor.*
- [FilterAction](#) `filterAction` () override
  - *Returns the action description corresponding to currently set options.*
- [QString](#) `filterIdentifier` () const override
  - *Return the identifier for this filter in the image history.*
- void [readParameters](#) (const [FilterAction](#) &action) override
- void [setObserver](#) ([DImgLoaderObserver](#) \*const observer, int progressBegin, int progressEnd)
  - *Normally, filters post progress and are cancelled by [DynamicThread](#) facilities.*
- void [setOutputProfile](#) (const [IccProfile](#) &profile)
  - *As additional and first post-processing step, convert the image's color space to the specified profile.*
- void [setSettings](#) (const [DRawDecoding](#) &settings)
  - *Set the raw decoding settings.*
- [DRawDecoding](#) `settings` () const

## Public Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImg](#) \*const orgImage, QObject \*const parent, const QString &name=QString())
  - *Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter](#) (QObject \*const parent=nullptr, const QString &name=QString())
  - *Constructs a filter without argument.*
- virtual void [cancelFilter](#) ()
  - *Cancel the threaded computation.*
- const QString & [filterName](#) ()
- int [filterVersion](#) () const
- [DImg](#) [getTargetImage](#) ()
- QList< int > [multithreadedSteps](#) (int stop, int start=0) const
  - *This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool [parametersSuccessfullyRead](#) () const
  - *Optional: error handling for readParameters.*
- virtual QString [readParametersError](#) (const [FilterAction](#) &actionThatFailed) const
- void [setFilterName](#) (const QString &name)
- void [setFilterVersion](#) (int version)
  - *Replaying a filter action: Set the filter version.*
- void [setOriginalImage](#) (const [DImg](#) &orgImage)
- void [setupAndStartDirectly](#) (const [DImg](#) &orgImage, [DImgThreadedFilter](#) \*const master, int progress↔Begin=0, int progressEnd=100)
  - *Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void [setupFilter](#) (const [DImg](#) &orgImage)
  - *You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void [startFilter](#) ()
  - *Start the threaded computation.*
- virtual void [startFilterDirectly](#) ()
  - *Start computation of this filter, directly in this thread.*
- virtual QList< int > [supportedVersions](#) () const

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread](#) (QObject \*const parent=nullptr)  
*This class extends QRunnable, so you have to reimplement virtual void [run\(\)](#).*
- [~DynamicThread](#) () override  
*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool [isFinished](#) () const
- bool [isRunning](#) () const
- QThread::Priority [priority](#) () const
- void [setEmitSignals](#) (bool emitThem)
- void [setPriority](#) (QThread::Priority priority)  
*Sets the priority for this dynamic thread.*
- State [state](#) () const

## Static Public Member Functions

- static int [CurrentVersion](#) ()
- static QString [DisplayableName](#) ()
- static QString [FilterIdentifier](#) ()
- static QList< int > [SupportedVersions](#) ()

## Protected Member Functions

- bool [continueQuery](#) () const
- void [filterImage](#) () override  
*Main image filter method.*
- void [postProgress](#) (int)

## Protected Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by [stopComputation\(\)](#) method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*



## Protected Member Functions inherited from Digikam::DynamicThread

- bool **runningFlag** () const volatile  
*In you [run\(\)](#) method, you shall regularly check for [runningFlag\(\)](#) and cleanup and return if false.*
- void **shutDown** ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call [stop\(\)](#) and [wait\(\)](#), knowing that nothing will call [start\(\)](#) anymore after this 3) Be sure the thread will never be running at destruction.*
- void **start** (QMutexLocker< QMutex > &locker)  
*Doing the same as [start\(\)](#), [stop\(\)](#) and [wait](#) above, provide it with a locked QMutexLocker on mutex().*
- void **stop** (const QMutexLocker< QMutex > &locker)
- QMutex \* **threadMutex** () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void **wait** (QMutexLocker< QMutex > &locker)

## Protected Attributes

- [IccProfile](#) **m\_customOutputProfile**
- [DImgLoaderObserver](#) \* **m\_observer** = nullptr
- [DRawDecoding](#) **m\_settings**

## Protected Attributes inherited from Digikam::DImgThreadedFilter

- [DImg](#) **m\_destImage**  
*Output image data.*
- [DImgThreadedFilter](#) \* **m\_master** = nullptr  
*The master of this slave filter.*
- QString **m\_name**  
*Filter name.*
- [DImg](#) **m\_orgImage**  
*Copy of original Image data.*
- int **m\_progressBegin** = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int **m\_progressCurrent** = 0  
*To prevent signals bombarding with progress indicator value in [postProgress\(\)](#).*
- int **m\_progressSpan** = 0
- [DImgThreadedFilter](#) \* **m\_slave** = nullptr  
*The current slave.*
- int **m\_version** = 1
- bool **m\_wasCancelled** = false

## Additional Inherited Members

## Public Types inherited from Digikam::DynamicThread

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

## Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()
  - Stop computation, sets the running flag to false.*
- void **wait** ()
  - Waits until the thread finishes.*

## Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)
  - Emitted when the computation has completed.*
- void **progress** (int progress)
  - Emitted when progress info from the calculation is available.*
- void **started** ()
  - This signal is emitted when image data is available and the computation has started.*

## Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()
  - Emitted if emitSignals is enabled.*

### 9.1091.1 Detailed Description

It implements RAW post processing. Additionally, it provides some facilities for use from the [DImg](#) Raw loader.

The original image shall come from RawEngine without further modification.

### 9.1091.2 Constructor & Destructor Documentation

#### 9.1091.2.1 RawProcessingFilter() [1/2]

```
Digikam::RawProcessingFilter::RawProcessingFilter (
    QObject *const parent = nullptr ) [explicit]
```

You need to call [setSettings\(\)](#) and [setOriginalImage\(\)](#) before starting the filter.

#### 9.1091.2.2 RawProcessingFilter() [2/2]

```
Digikam::RawProcessingFilter::RawProcessingFilter (
    const DRawDecoding & settings,
    DImgThreadedFilter *const master,
    const DImg & orgImage,
    const DImg & destImage,
    int progressBegin = 0,
    int progressEnd = 100,
    const QString & name = QString() )
```

Computation is started immediately.

### 9.1091.3 Member Function Documentation

#### 9.1091.3.1 filterAction()

`FilterAction` Digikam::RawProcessingFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

#### 9.1091.3.2 filterIdentifier()

`QString` Digikam::RawProcessingFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

#### 9.1091.3.3 filterImage()

`void` Digikam::RawProcessingFilter::filterImage ( ) [override], [protected], [virtual]

Override in subclass.

Implements [Digikam::DImgThreadedFilter](#).

#### 9.1091.3.4 readParameters()

```
void Digikam::RawProcessingFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

#### 9.1091.3.5 setObserver()

```
void Digikam::RawProcessingFilter::setObserver (
    DImgLoaderObserver *const observer,
    int progressBegin,
    int progressEnd )
```

Here, as an alternative, a [DImgLoaderObserver](#) is set. Its `continueQuery` is called and progress is posted in the given interval.

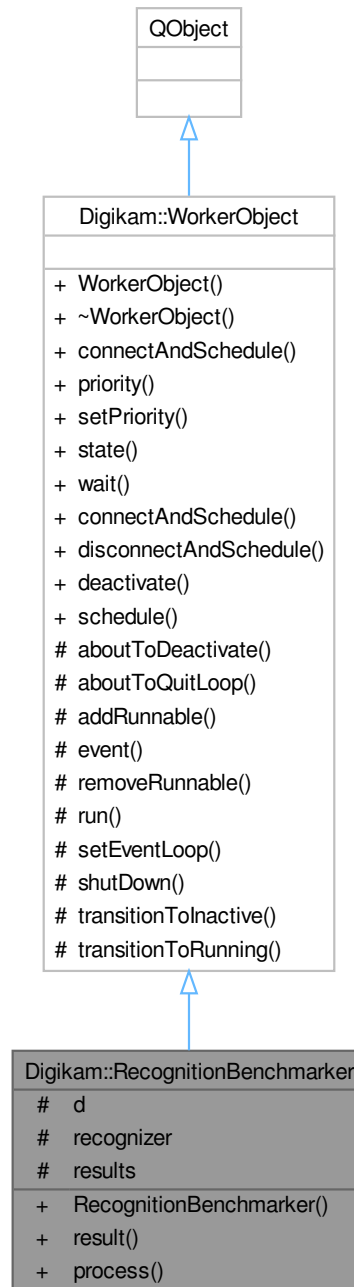
#### 9.1091.3.6 setSettings()

```
void Digikam::RawProcessingFilter::setSettings (
    const DRawDecoding & settings )
```

The post processing is carried out here, the libraw settings are needed to construct the [FilterAction](#).

## 9.1092 Digikam::RecognitionBenchmarker Class Reference

Inheritance diagram for Digikam::RecognitionBenchmarker:



### Classes

- class [Statistics](#)

## Public Slots

- void **process** (const FacePipelineExtendedPackage::Ptr &package)

## Public Slots inherited from [Digikam::WorkerObject](#)

- void [deactivate](#) ([DeactivatingMode](#) mode=[FlushSignals](#))  
*Quits execution of this worker object.*
- void **schedule** ()  
*Starts execution of this worker object: The object is moved to a thread and an event loop started, so that queued signals will be received.*

## Signals

- void **processed** (const FacePipelineExtendedPackage::Ptr &package)

## Signals inherited from [Digikam::WorkerObject](#)

- void **finished** ()
- void **started** ()

## Public Member Functions

- **RecognitionBenchmarker** (FacePipeline::Private \*const dd)
- QString [result](#) () const  
*NOTE: Bench performance code.*

## Public Member Functions inherited from [Digikam::WorkerObject](#)

- [WorkerObject](#) ()  
*Deriving from a worker object allows you to execute your slots in a thread.*
- bool [connectAndSchedule](#) (const QObject \*sender, const char \*signal, const char \*method, Qt::ConnectionType type=Qt::AutoConnection) const  
*You must normally call [schedule\(\)](#) to ensure that the object is active when you send a signal with work data.*
- QThread::Priority **priority** () const
- void [setPriority](#) (QThread::Priority priority)  
*Sets the priority for this dynamic thread.*
- State **state** () const
- void **wait** ()

## Protected Attributes

- FacePipeline::Private \*const **d** = nullptr
- [FacialRecognitionWrapper](#) **recognizer**
- QMap< int, [Statistics](#) > **results**

## Additional Inherited Members

### Public Types inherited from [Digikam::WorkerObject](#)

- enum [DeactivatingMode](#) { [FlushSignals](#) , [KeepSignals](#) , [PhaseOut](#) }
- enum [State](#) { [Inactive](#) , [Scheduled](#) , [Running](#) , [Deactivating](#) }

### Static Public Member Functions inherited from [Digikam::WorkerObject](#)

- static bool **connectAndSchedule** (const QObject \*sender, const char \*signal, const [WorkerObject](#) \*receiver, const char \*method, Qt::ConnectionType type=Qt::AutoConnection)
- static bool **disconnectAndSchedule** (const QObject \*sender, const char \*signal, const [WorkerObject](#) \*receiver, const char \*method)

### Protected Member Functions inherited from [Digikam::WorkerObject](#)

- virtual void [aboutToDeactivate](#) ()  
*Called from [deactivate\(\)](#), typically from a different thread than the worker thread, possibly the UI thread.*
- virtual void [aboutToQuitLoop](#) ()  
*Called from within thread's event loop to quit processing.*
- void **addRunnable** (WorkerObjectRunnable \*loop)
- bool **event** (QEvent \*e) override
- void **removeRunnable** (WorkerObjectRunnable \*loop)
- void **run** ()
- void **setEventLoop** (QEventLoop \*loop)
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void **transitionToInactive** ()
- bool **transitionToRunning** ()

## 9.1092.1 Member Function Documentation

### 9.1092.1.1 result()

```
QString Digikam::RecognitionBenchmarker::result ( ) const
```

No need i18n here

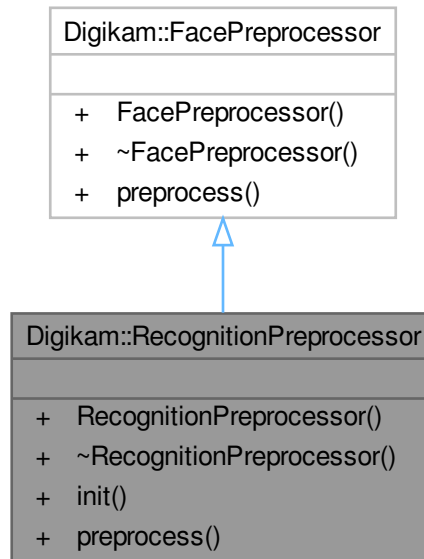
## 9.1093 Digikam::RecognitionBenchmarker::Statistics Class Reference

### Public Attributes

- int **correctlyRecognized** = 0
- int **knownFaces** = 0

## 9.1094 Digikam::RecognitionPreprocessor Class Reference

Inheritance diagram for Digikam::RecognitionPreprocessor:



### Public Member Functions

- void **init** (PreprocessorSelection mode)
- cv::Mat [preprocess](#) (const cv::Mat &image) const override

### 9.1094.1 Member Function Documentation

#### 9.1094.1.1 preprocess()

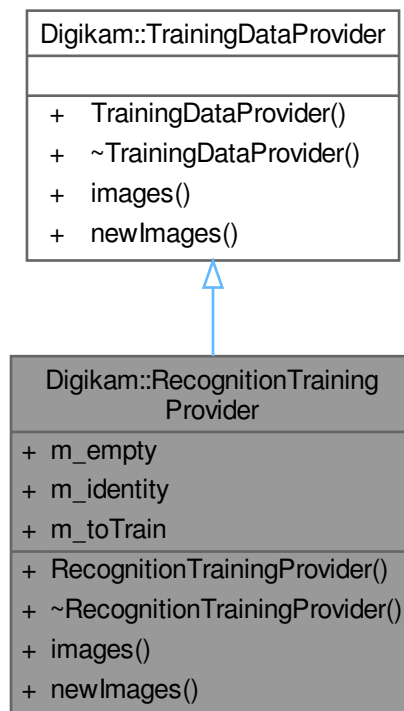
```
cv::Mat Digikam::RecognitionPreprocessor::preprocess (
    const cv::Mat & image ) const [override], [virtual]
```

Implements [Digikam::FacePreprocessor](#).

## 9.1095 Digikam::RecognitionTrainingProvider Class Reference

A simple QImage training data container used by `RecognitionDatabase::train(Identity, QImage, QString)`.

Inheritance diagram for Digikam::RecognitionTrainingProvider:



### Public Member Functions

- **RecognitionTrainingProvider** (const [Identity](#) &identity, const QList< QPair< QImage \*, QString > > &newImages)
- [ImageListProvider](#) \* `images` (const [Identity](#) &) override  
*Provides all images known for the given identity.*
- [ImageListProvider](#) \* `newImages` (const [Identity](#) &id) override  
*Provides those images for the given identity that have not yet been supplied for training.*

### Public Attributes

- [QListImageListProvider](#) `m_empty`
- [Identity](#) `m_identity`
- [QListImageListProvider](#) `m_toTrain`

## 9.1095.1 Member Function Documentation

### 9.1095.1.1 images()

```
ImageListProvider * Digikam::RecognitionTrainingProvider::images (
    const Identity & identity ) [override], [virtual]
```

Ownership of the returned object stays with the [TrainingDataProvider](#).

Implements [Digikam::TrainingDataProvider](#).



### 9.1095.1.2 newImages()

```
ImageListProvider * Digikam::RecognitionTrainingProvider::newImages (
    const Identity & identity ) [override], [virtual]
```

Ownership of the returned object stays with the [TrainingDataProvider](#).

Implements [Digikam::TrainingDataProvider](#).

## 9.1096 Digikam::RecognitionTrainingUpdateQueue Class Reference

### Public Member Functions

- QString **endSignal** ()
- QString **front** ()
- QString **pop\_front** ()
- void **push** (const QString &hash)
- void **registerReaderThread** (const QThread \*thread)
- void **unregisterReaderThread** (const QThread \*thread)

## 9.1097 Digikam::RecognitionWorker Class Reference

Inheritance diagram for Digikam::RecognitionWorker:



### Public Slots

- void **process** (const FacePipelineExtendedPackage::Ptr &package)  
*TODO: investigate this method.*

- void **setAccuracyAndModel** (int detectAccuracy, [FaceScanSettings::FaceDetectionModel](#) detectModel, [FaceScanSettings::FaceDetectionSize](#) detectSize, int recognizeAccuracy, [FaceScanSettings::FaceRecognitionModel](#) recognizeModel)
- void **setThreshold** (int threshold, bool)

### Public Slots inherited from [Digikam::WorkerObject](#)

- void **deactivate** ([DeactivatingMode](#) mode=[FlushSignals](#))  
*Quits execution of this worker object.*
- void **schedule** ()  
*Starts execution of this worker object: The object is moved to a thread and an event loop started, so that queued signals will be received.*

### Signals

- void **processed** (const [FacePipelineExtendedPackage::Ptr](#) &package)

### Signals inherited from [Digikam::WorkerObject](#)

- void **finished** ()
- void **started** ()

### Public Member Functions

- **RecognitionWorker** ([FacePipeline::Private](#) \*const dd)

### Public Member Functions inherited from [Digikam::WorkerObject](#)

- [WorkerObject](#) ()  
*Deriving from a worker object allows you to execute your slots in a thread.*
- bool **connectAndSchedule** (const [QObject](#) \*sender, const char \*signal, const char \*method, [Qt::](#)↳ [ConnectionType](#) type=[Qt::AutoConnection](#)) const  
*You must normally call [schedule\(\)](#) to ensure that the object is active when you send a signal with work data.*
- [QThread::Priority](#) **priority** () const
- void **setPriority** ([QThread::Priority](#) priority)  
*Sets the priority for this dynamic thread.*
- State **state** () const
- void **wait** ()

### Protected Member Functions

- void **aboutToDeactivate** () override  
*Called from [deactivate\(\)](#), typically from a different thread than the worker thread, possibly the UI thread.*

## Protected Member Functions inherited from [Digikam::WorkerObject](#)

- virtual void [aboutToQuitLoop](#) ()
  - Called from within thread's event loop to quit processing.*
- void **addRunnable** (WorkerObjectRunnable \*loop)
- bool **event** (QEvent \*e) override
- void **removeRunnable** (WorkerObjectRunnable \*loop)
- void **run** ()
- void **setEventLoop** (QEventLoop \*loop)
- void [shutDown](#) ()
  - If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void **transitionToInactive** ()
- bool **transitionToRunning** ()

## Protected Attributes

- FacePipeline::Private \*const **d** = nullptr
- [FaceltemRetriever](#) **imageRetriever**
- [FacialRecognitionWrapper](#) **recognizer**

## Additional Inherited Members

## Public Types inherited from [Digikam::WorkerObject](#)

- enum [DeactivatingMode](#) { [FlushSignals](#) , [KeepSignals](#) , [PhaseOut](#) }
- enum **State** { [Inactive](#) , [Scheduled](#) , [Running](#) , [Deactivating](#) }

## Static Public Member Functions inherited from [Digikam::WorkerObject](#)

- static bool **connectAndSchedule** (const QObject \*sender, const char \*signal, const [WorkerObject](#) \*receiver, const char \*method, Qt::ConnectionType type=Qt::AutoConnection)
- static bool **disconnectAndSchedule** (const QObject \*sender, const char \*signal, const [WorkerObject](#) \*receiver, const char \*method)

### 9.1097.1 Member Function Documentation

#### 9.1097.1.1 [aboutToDeactivate\(\)](#)

```
void Digikam::RecognitionWorker::aboutToDeactivate ( ) [override], [protected], [virtual]
```

You can stop any extra controlled threads here. Immediately afterwards, an event will be sent to the working thread which will cause the event loop to quit. ([aboutToQuitLoop\(\)](#))

Reimplemented from [Digikam::WorkerObject](#).

## 9.1098 Digikam::RedEye::RegressionTree Struct Reference

### Public Member Functions

- unsigned long **num\_leaves** () const
- const std::vector< float > & **operator()** (const std::vector< float > &feature\_pixel\_values, unsigned long &i) const  
*requires*

### Public Attributes

- std::vector< std::vector< float > > **leaf\_values**
- std::vector< [SplitFeature](#) > **splits**

### 9.1098.1 Member Function Documentation

#### 9.1098.1.1 operator>()

```
const std::vector< float > & Digikam::RedEye::RegressionTree::operator() (
    const std::vector< float > & feature_pixel_values,
    unsigned long & i ) const
```

- All the index values in splits are less than

#### Parameters

<i>feature_pixel_values</i>	size.
-----------------------------	-------

- leaf\_values.size() is a power of 2. (i.e. we require a tree with all the levels fully filled out.
- leaf\_values.size() == splits.size()+1 (i.e. there needs to be the right number of leaves given the number of splits in the tree) ensures runs through the tree and returns the vector at the leaf we end up in.

#### Parameters

<i>i</i>	egal the selected leaf node index.
----------	------------------------------------

## 9.1099 Digikam::RedEye::ShapePredictor Class Reference

### Public Member Functions

- unsigned long **num\_features** () const
- unsigned long **num\_parts** () const
- [FullObjectDetection](#) **operator()** (const cv::Mat &img, const cv::Rect &rect) const

### Public Attributes

- `std::vector< std::vector< unsigned long > >` **anchor\_idx**
- `std::vector< std::vector< std::vector< float > > >` **deltas**
- `std::vector< std::vector< RedEye::RegressionTree > >` **forests**
- `std::vector< float >` **initial\_shape**

## 9.1100 Digikam::RedEye::SplitFeature Struct Reference

### Public Attributes

- quint64 **idx1** = 0
- quint64 **idx2** = 0
- float **thresh** = 0.0F

## 9.1101 Digikam::RedEyeCorrectionContainer Class Reference

### Public Member Functions

- bool **isDefault** () const
- bool **operator==** (const [RedEyeCorrectionContainer](#) &other) const
- void **writeToFilterAction** ([FilterAction](#) &action, const QString &prefix=QString()) const

### Static Public Member Functions

- static [RedEyeCorrectionContainer](#) **fromFilterAction** (const [FilterAction](#) &action, const QString &prefix=QString())

### Public Attributes

- double **m\_redToAvgRatio** = 2.1

## 9.1102 Digikam::RedEyeCorrectionFilter Class Reference

Inheritance diagram for Digikam::RedEyeCorrectionFilter:



### Public Member Functions

- **RedEyeCorrectionFilter** (const [RedEyeCorrectionContainer](#) &settings, [DimgThreadedFilter](#) \*const parent↔ Filter, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100)

Constructor for slave mode: execute immediately in current thread with specified master filter.

- **RedEyeCorrectionFilter** ([DImg](#) \*const orgImage, [QObject](#) \*const parent=nullptr, const [RedEyeCorrectionContainer](#) &settings=[RedEyeCorrectionContainer](#)())
- **RedEyeCorrectionFilter** ([QObject](#) \*const parent=nullptr)
- [FilterAction](#) filterAction () override
 

Returns the action description corresponding to currently set options.
- [QString](#) filterIdentifier () const override
 

Return the identifier for this filter in the image history.

## Public Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImg](#) \*const orgImage, [QObject](#) \*const parent, const [QString](#) &name=[QString](#)())
 

Constructs a filter with all arguments (ready to use).
- [DImgThreadedFilter](#) ([QObject](#) \*const parent=nullptr, const [QString](#) &name=[QString](#)())
 

Constructs a filter without argument.
- virtual void [cancelFilter](#) ()
 

Cancel the threaded computation.
- const [QString](#) & **filterName** ()
- int **filterVersion** () const
- [DImg](#) **getTargetImage** ()
- [QList](#)< int > **multithreadedSteps** (int stop, int start=0) const
 

This method return a list of steps to process parallelized operation in filter using [QtConcurrents](#) API.
- virtual bool **parametersSuccessfullyRead** () const
 

Optional: error handling for readParameters.
- virtual [QString](#) **readParametersError** (const [FilterAction](#) &actionThatFailed) const
- void **setFilterName** (const [QString](#) &name)
- void **setFilterVersion** (int version)
 

Replaying a filter action: Set the filter version.
- void **setOriginalImage** (const [DImg](#) &orgImage)
- void **setupAndStartDirectly** (const [DImg](#) &orgImage, [DImgThreadedFilter](#) \*const master, int progress←Begin=0, int progressEnd=100)
 

Initializes the filter for use as a slave and directly starts computation (in-thread)
- void **setupFilter** (const [DImg](#) &orgImage)
 

You need to call this and then start filter of you used the constructor not setting an original image.
- virtual void **startFilter** ()
 

Start the threaded computation.
- virtual void **startFilterDirectly** ()
 

Start computation of this filter, directly in this thread.
- virtual [QList](#)< int > **supportedVersions** () const

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread](#) ([QObject](#) \*const parent=nullptr)
 

This class extends [QRunnable](#), so you have to reimplement virtual void [run\(\)](#).
- ~[DynamicThread](#) () override
 

The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.
- bool **isFinished** () const
- bool **isRunning** () const
- [QThread::Priority](#) **priority** () const
- void **setEmitSignals** (bool emitThem)
- void **setPriority** ([QThread::Priority](#) priority)
 

Sets the priority for this dynamic thread.
- State **state** () const



### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from [Digikam::DImgThreadedFilter](#)

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.1102.1 Member Function Documentation

### 9.1102.1.1 filterAction()

`FilterAction` Digikam::RedEyeCorrectionFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

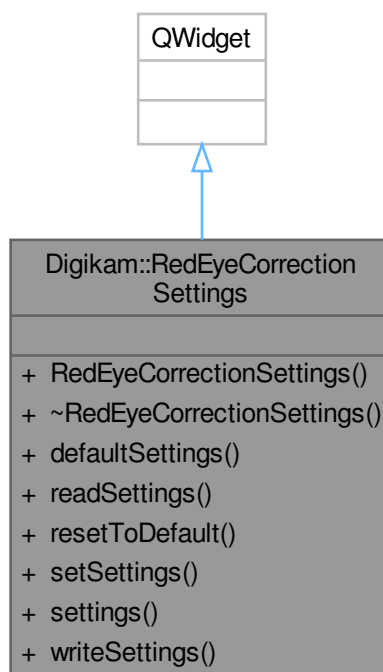
### 9.1102.1.2 filterIdentifier()

`QString` Digikam::RedEyeCorrectionFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

## 9.1103 Digikam::RedEyeCorrectionSettings Class Reference

Inheritance diagram for Digikam::RedEyeCorrectionSettings:



### Signals

- void `signalSettingsChanged` ( )

**Public Member Functions**

- **RedEyeCorrectionSettings** (QWidget \*const parent=nullptr)
- [RedEyeCorrectionContainer](#) **defaultSettings** () const
- void **readSettings** (const KConfigGroup &group)
- void **resetToDefault** ()
- void **setSettings** (const [RedEyeCorrectionContainer](#) &settings)
- [RedEyeCorrectionContainer](#) **settings** () const
- void **writeSettings** (KConfigGroup &group)

## 9.1104 Digikam::RefocusFilter Class Reference

Inheritance diagram for Digikam::RefocusFilter:



### Public Member Functions

- **RefocusFilter** (`Dlmg *const orgImage`, `QObject *const parent=nullptr`, `int matrixSize=5`, `double radius=0.9`, `double gauss=0.0`, `double correlation=0.5`, `double noise=0.01`)

- **RefocusFilter** (QObject \*const parent=nullptr)
- **FilterAction filterAction** () override  
*Returns the action description corresponding to currently set options.*
- QString **filterIdentifier** () const override  
*Return the identifier for this filter in the image history.*
- void **readParameters** (const **FilterAction** &action) override

## Public Member Functions inherited from **Digikam::DImgThreadedFilter**

- **DImgThreadedFilter** (DImg \*const orgImage, QObject \*const parent, const QString &name=QString())  
*Constructs a filter with all arguments (ready to use).*
- **DImgThreadedFilter** (QObject \*const parent=nullptr, const QString &name=QString())  
*Constructs a filter without argument.*
- virtual void **cancelFilter** ()  
*Cancel the threaded computation.*
- const QString & **filterName** ()
- int **filterVersion** () const
- **DImg getTargetImage** ()
- QList< int > **multithreadedSteps** (int stop, int start=0) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool **parametersSuccessfullyRead** () const  
*Optional: error handling for readParameters.*
- virtual QString **readParametersError** (const **FilterAction** &actionThatFailed) const
- void **setFilterName** (const QString &name)
- void **setFilterVersion** (int version)  
*Replaying a filter action: Set the filter version.*
- void **setOriginalImage** (const **DImg** &orgImage)
- void **setupAndStartDirectly** (const **DImg** &orgImage, **DImgThreadedFilter** \*const master, int progress←Begin=0, int progressEnd=100)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void **setupFilter** (const **DImg** &orgImage)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void **startFilter** ()  
*Start the threaded computation.*
- virtual void **startFilterDirectly** ()  
*Start computation of this filter, directly in this thread.*
- virtual QList< int > **supportedVersions** () const

## Public Member Functions inherited from **Digikam::DynamicThread**

- **DynamicThread** (QObject \*const parent=nullptr)  
*This class extends QRunnable, so you have to reimplement virtual void [run\(\)](#).*
- **~DynamicThread** () override  
*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool **isFinished** () const
- bool **isRunning** () const
- QThread::Priority **priority** () const
- void **setEmitSignals** (bool emitThem)
- void **setPriority** (QThread::Priority priority)  
*Sets the priority for this dynamic thread.*
- State **state** () const

### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static int **maxMatrixSize** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from [Digikam::DImgThreadedFilter](#)

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false



## 9.1104.1 Member Function Documentation

### 9.1104.1.1 filterAction()

`FilterAction` Digikam::RefocusFilter::filterAction ( ) [override], [virtual]

Implements `Digikam::DImgThreadedFilter`.

### 9.1104.1.2 filterIdentifier()

`QString` Digikam::RefocusFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements `Digikam::DImgThreadedFilter`.

### 9.1104.1.3 readParameters()

```
void Digikam::RefocusFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements `Digikam::DImgThreadedFilter`.

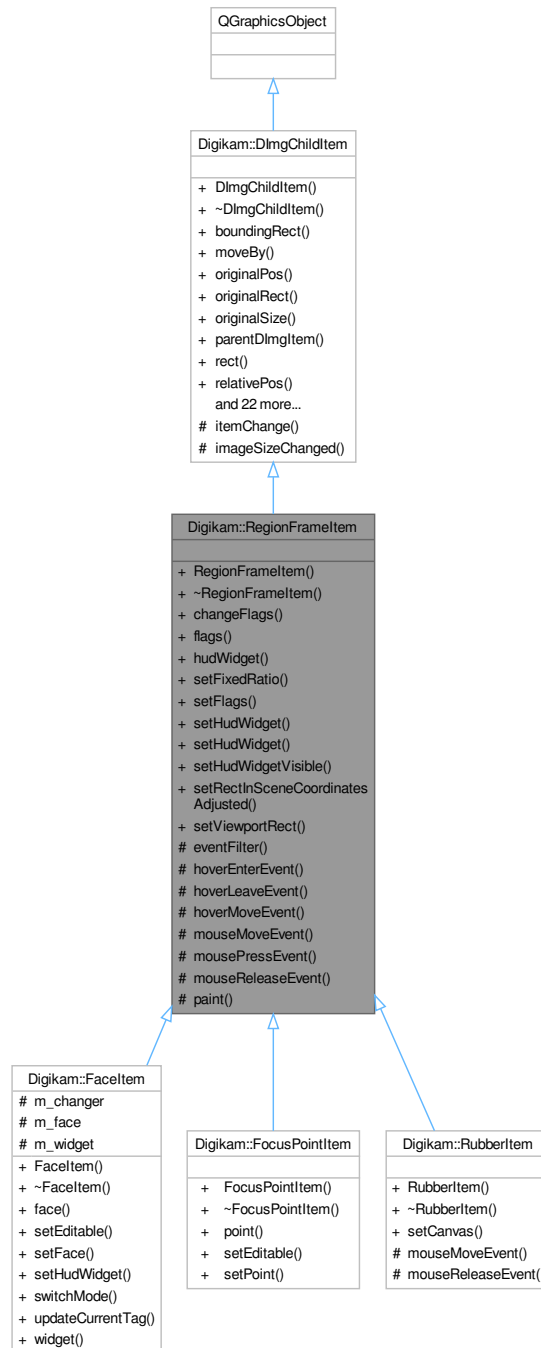
## 9.1105 Digikam::RefocusMatrix Class Reference

### Static Public Member Functions

- static double `c_mat_elt` (const `CMat` \*const mat, const int col, const int row)
- static `CMat` \* `compute_g_matrix` (const `CMat` \*const convolution, const int m, const double gamma, const double noise\_factor, const double musq, const bool symmetric)
- static void `convolve_star_mat` (const `CMat` \*const result, const `CMat` \*const mata, const `CMat` \*const matb)
- static void `fill_matrix` (`CMat` \*const matrix, const int m, double f(const int, const int, const double), const double fun\_arg)
- static void `fill_matrix2` (`CMat` \*const matrix, const int m, double f(const int, const int, const double, const double), const double fun\_arg1, const double fun\_arg2)
- static void `finish_and_free_matrix` (`Mat` \*const mat)
- static void `finish_c_mat` (`CMat` \*const mat)
- static void `finish_matrix` (`Mat` \*const mat)
- static void `init_c_mat` (`CMat` \*const mat, const int radius)
- static void `make_circle_convolution` (const double radius, `CMat` \*const convolution, const int m)
- static void `make_gaussian_convolution` (const double alpha, `CMat` \*const convolution, const int m)
- static double `mat_elt` (const `Mat` \*const mat, const int r, const int c)

## 9.1106 Digikam::RegionFrameItem Class Reference

Inheritance diagram for Digikam::RegionFrameItem:



### Public Types

- enum **Flag** { **NoFlags** = 0 , **ShowResizeHandles** = 1 << 0 , **MoveByDrag** = 1 << 1 , **GeometryEditable** = ShowResizeHandles | MoveByDrag }
- typedef `QFlags< Flag >` **Flags**

## Public Slots

- void [setViewportRect](#) (const QRectF &rect)  
*The associated HUD item is dynamically moved to be visible.*

## Signals

- void [geometryEdited](#) ()

## Signals inherited from [Digikam::DImgChildItem](#)

- void [geometryChanged](#) ()
- void [geometryOnImageChanged](#) ()
- void [positionChanged](#) ()  
*These signals are emitted in any case when the geometry changed: Either after changing the geometry relative to the original image, or when the size of the parent [GraphicsDImgItem](#) changed (zooming).*
- void [positionOnImageChanged](#) ()  
*These signals are emitted when the geometry, relative to the original image, of this item has changed.*
- void [sizeChanged](#) ()
- void [sizeOnImageChanged](#) ()

## Public Member Functions

- [RegionFrameItem](#) (QGraphicsItem \*const parent)
- void [changeFlags](#) (Flags flags, bool addOrRemove)
- Flags [flags](#) () const
- QGraphicsWidget \* [hudWidget](#) () const
- void [setFixedRatio](#) (double ratio)
- void [setFlags](#) (Flags flags)
- void [setHudWidget](#) (QGraphicsWidget \*const hudWidget)  
*Sets a widget item as HUD item.*
- void [setHudWidget](#) (QWidget \*const widget, Qt::WindowFlags wFlags=Qt::WindowFlags())
- void [setHudWidgetVisible](#) (bool visible)
- void [setRectInSceneCoordinatesAdjusted](#) (const QRectF &rect)

## Public Member Functions inherited from [Digikam::DImgChildItem](#)

- [DImgChildItem](#) (QGraphicsItem \*const parent=nullptr)  
*This is a base class for items that are positioned on top of a [GraphicsDImgItem](#), positioned in relative coordinates, i.e.*
- QRectF [boundingRect](#) () const override  
*Reimplemented.*
- void [moveBy](#) (qreal dx, qreal dy)
- QPoint [originalPos](#) () const
- QRect [originalRect](#) () const  
*Returns the position and size in coordinates of the original image.*
- QSize [originalSize](#) () const
- [GraphicsDImgItem](#) \* [parentDImgItem](#) () const  
*If the parent item is a [GraphicsDImgItem](#), return it, if the parent item is null or of a different class, returns 0.*
- QRectF [rect](#) () const  
*Returns position and size of this item, in coordinates of the parent [DImg](#) with the current zoom.*

- QPointF **relativePos** () const
- QRectF **relativeRect** () const
  - Returns the position and size relative to the **DImg** displayed in the parent item.*
- QSizeF **relativeSize** () const
- void **setOriginalPos** (const QPointF &posInOriginal)
  - Sets the position and size of this item, in coordinates of the original image.*
- void **setOriginalPos** (qreal x, qreal y)
- void **setOriginalRect** (const QRectF &rect)
- void **setOriginalRect** (qreal x, qreal y, qreal width, qreal height)
- void **setOriginalSize** (const QSizeF &sizeInOriginal)
- void **setOriginalSize** (qreal width, qreal height)
- void **setPos** (const QPointF &zoomedPos)
  - Sets the position and size of this item, in coordinates of the parent **DImg** item.*
- void **setPos** (qreal x, qreal y)
- void **setRect** (const QRectF &rect)
- void **setRect** (qreal x, qreal y, qreal width, qreal height)
- void **setRectInSceneCoordinates** (const QRectF &rect)
  - Equivalent to mapping the scene coordinates to the parent item, and calling **setRect**().*
- void **setRelativePos** (const QPointF &relativePosition)
  - Sets the position and size of this item, relative to the **DImg** displayed in the parent item.*
- void **setRelativePos** (qreal x, qreal y)
- void **setRelativeRect** (const QRectF &rect)
- void **setRelativeRect** (qreal x, qreal y, qreal width, qreal height)
- void **setRelativeSize** (const QSizeF &relativeSize)
- void **setRelativeSize** (qreal width, qreal height)
- void **setSize** (const QSizeF &zoomedSize)
- void **setSize** (qreal width, qreal height)
- QSizeF **size** () const

### Protected Member Functions

- bool **eventFilter** (QObject \*watched, QEvent \*event) override
- void **hoverEnterEvent** (QGraphicsSceneHoverEvent \*event) override
- void **hoverLeaveEvent** (QGraphicsSceneHoverEvent \*event) override
- void **hoverMoveEvent** (QGraphicsSceneHoverEvent \*event) override
- void **mouseMoveEvent** (QGraphicsSceneMouseEvent \*) override
- void **mousePressEvent** (QGraphicsSceneMouseEvent \*) override
- void **mouseReleaseEvent** (QGraphicsSceneMouseEvent \*) override
- void **paint** (QPainter \*painter, const QStyleOptionGraphicsItem \*option, QWidget \*widget=nullptr) override

### Protected Member Functions inherited from **Digikam::DImgChildItem**

- QVariant **itemChange** (GraphicsItemChange change, const QVariant &value) override

### Additional Inherited Members

### Protected Slots inherited from **Digikam::DImgChildItem**

- void **imageSizeChanged** (const QSizeF &)

## 9.1106.1 Member Function Documentation

### 9.1106.1.1 setHudWidget()

```
void Digikam::RegionFrameItem::setHudWidget (
    QGraphicsWidget *const hudWidget )
```

A HUD item will be positioned relative to this item, and repositioned on position changes or resizing. Ownership of the item is taken, and it is made a child item of this item. You can also add QWidget directly. It will be wrapped in a proxy item.

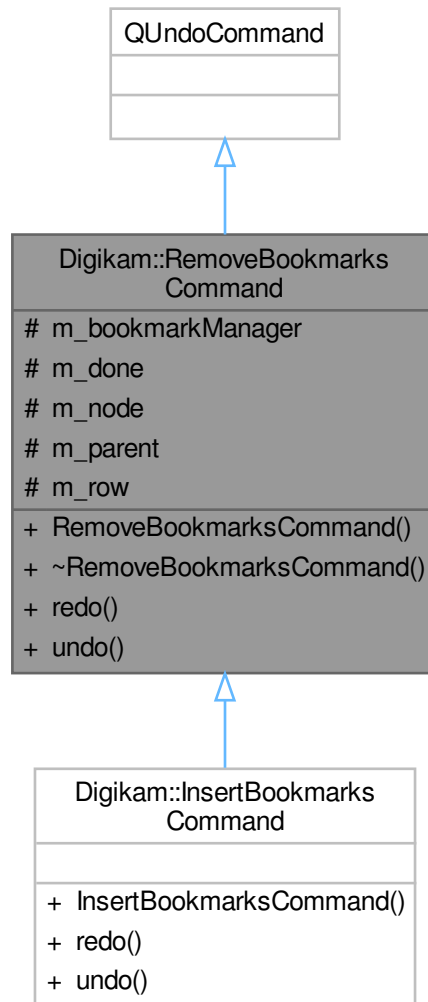
### 9.1106.1.2 setViewportRect

```
void Digikam::RegionFrameItem::setViewportRect (
    const QRectF & rect ) [slot]
```

This can only be done for *one* region at a time. Set the current primary view region of the scene by this method to dynamically reposition the HUD inside this region. The rect given is in scene coordinates.

## 9.1107 Digikam::RemoveBookmarksCommand Class Reference

Inheritance diagram for Digikam::RemoveBookmarksCommand:



### Public Member Functions

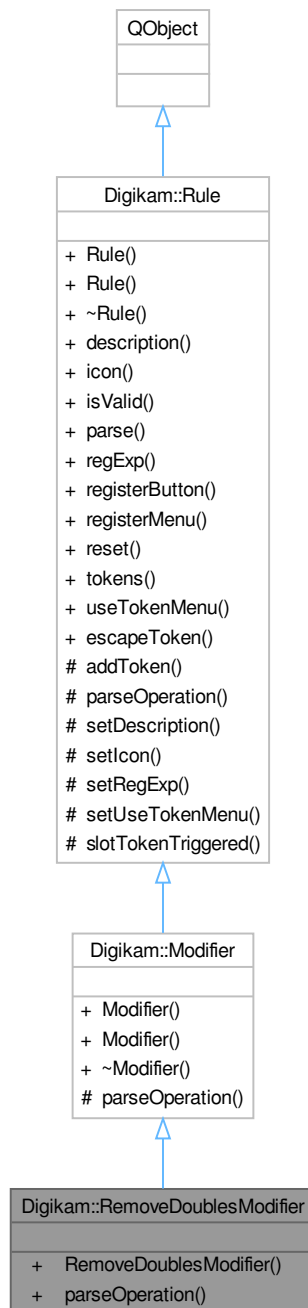
- **RemoveBookmarksCommand** ([BookmarkManager](#) \*const mgr, [BookmarkNode](#) \*const parent, int row)
- void **redo** () override
- void **undo** () override

### Protected Attributes

- [BookmarkManager](#) \* **m\_bookmarkManager** = nullptr
- bool **m\_done** = false
- [BookmarkNode](#) \* **m\_node** = nullptr
- [BookmarkNode](#) \* **m\_parent** = nullptr
- int **m\_row** = 0

## 9.1108 Digikam::RemoveDoublesModifier Class Reference

Inheritance diagram for Digikam::RemoveDoublesModifier:



### Public Member Functions

- QString [parseOperation](#) ([ParseSettings](#) &settings, const QRegularExpressionMatch &match) override  
*TODO: describe me.*

## Public Member Functions inherited from [Digikam::Modifier](#)

- **Modifier** (const QString &name, const QString &description)
- **Modifier** (const QString &name, const QString &description, const QString &icon)

## Public Member Functions inherited from [Digikam::Rule](#)

- **Rule** (const QString &name)
- **Rule** (const QString &name, const QString &icon)
- QString **description** () const
- QPixmap **icon** (Rule::IconType type=Rule::Action) const
- bool **isValid** () const

*Checks the validity of the parse object.*

- **ParseResults parse** ([ParseSettings](#) &settings)
  - QRegularExpression & **regExp** () const
- TODO: This is probably not needed anymore.*
- QPushButton \* **registerButton** (QWidget \*parent)
- Register a button in the parent object.*
- QAction \* **registerMenu** (QMenu \*parent)
- Register a menu action in the parent object.*
- virtual void **reset** ()
- Resets the parser to its initial state.*
- TokenList & **tokens** () const
  - bool **useTokenMenu** () const
- Returns true if a token menu is used.*

## Additional Inherited Members

## Public Types inherited from [Digikam::Rule](#)

- enum **IconType** { **Action** = 0 , **Dialog** }

## Signals inherited from [Digikam::Rule](#)

- void **signalTokenTriggered** (const QString &)

## Static Public Member Functions inherited from [Digikam::Rule](#)

- static QString **escapeToken** (const QString &token)
- Escape the token characters to make them work in regular expressions.*

## Protected Slots inherited from [Digikam::Rule](#)

- virtual void **slotTokenTriggered** (const QString &)



## Protected Member Functions inherited from [Digikam::Rule](#)

- bool [addToken](#) (const QString &id, const QString &description, const QString &actionName=QString())  
*add a token to the parser, every parser should at least assign one token object*
- void [setDescription](#) (const QString &desc)
- void [setIcon](#) (const QString &pixmap)
- void [setRegExp](#) (const QRegularExpression &regExp)
- void [setUseTokenMenu](#) (bool value)  
*If multiple tokens have been assigned to a rule, a menu will be created.*

### 9.1108.1 Member Function Documentation

#### 9.1108.1.1 [parseOperation\(\)](#)

```
QString Digikam::RemoveDoublesModifier::parseOperation (  
    ParseSettings & settings,  
    const QRegularExpressionMatch & match ) [override], [virtual]
```

##### Parameters

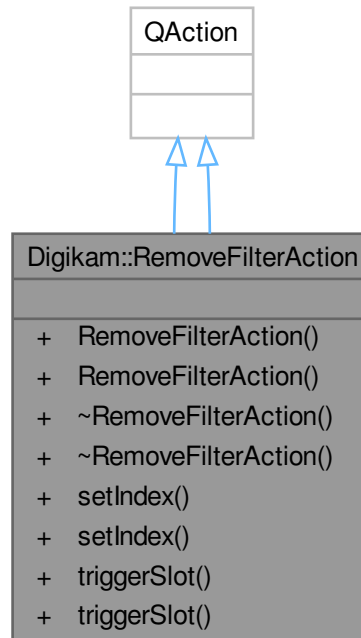
<i>settings</i>	contains settings
<i>match</i>	result of the regular expression match done in <a href="#">Option::parse()</a>

##### Returns

Implements [Digikam::Modifier](#).

## 9.1109 Digikam::RemoveFilterAction Class Reference

Inheritance diagram for Digikam::RemoveFilterAction:



### Public Slots

- void **triggerSlot** ()
- void **triggerSlot** ()

### Signals

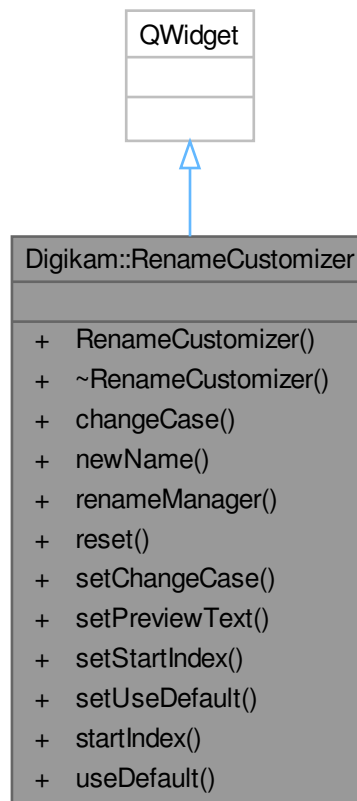
- void **actionTriggered** (QModelIndex index)
- void **actionTriggered** (QModelIndex index)

### Public Member Functions

- **RemoveFilterAction** (const QString &label, const QModelIndex &index, QObject \*const parent=nullptr)
- **RemoveFilterAction** (const QString &label, const QModelIndex &index, QObject \*const parent=nullptr)
- void **setIndex** (const QModelIndex &index)
- void **setIndex** (const QModelIndex &index)

## 9.1110 Digikam::RenameCustomizer Class Reference

Inheritance diagram for Digikam::RenameCustomizer:



### Public Types

- enum `Case` { `NONE = 0` , `UPPER` , `LOWER` }

### Signals

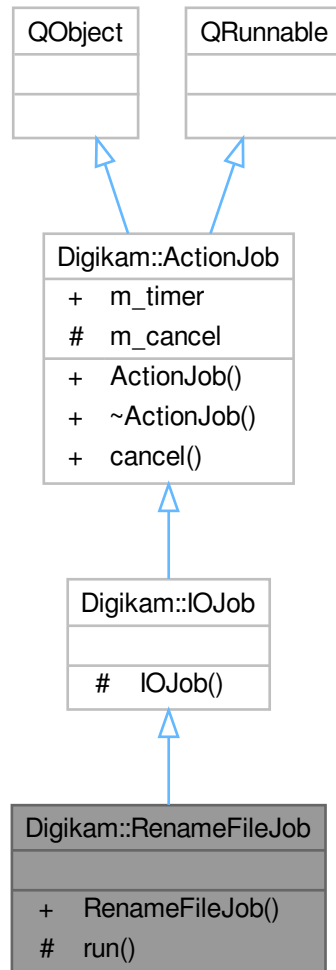
- void `signalChanged` ()

### Public Member Functions

- `RenameCustomizer` (`QWidget *const parent`, `const QString &cameraTitle`)
- `Case changeCase` () const
- `QString newName` (`const QString &fileName`) const
- `AdvancedRenameManager * renameManager` () const
- void `reset` ()
- void `setChangeCase` (`Case val`)
- void `setPreviewText` (`const QString &txt`)
- void `setStartIndex` (`int startIndex`)
- void `setDefault` (`bool val`)
- int `startIndex` () const
- bool `useDefault` () const

## 9.1111 Digikam::RenameFileJob Class Reference

Inheritance diagram for Digikam::RenameFileJob:



### Signals

- void **signalRenameFailed** (const QUrl &url)

### Signals inherited from [Digikam::IOJob](#)

- void **signalError** (const QString &errMsg)
- void **signalOneProcessed** (const QUrl &url)

## Signals inherited from [Digikam::ActionJob](#)

- void **signalDone** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job is done.*
- void **signalProgress** (int)  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job progress.*
- void **signalStarted** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager that job is started.*

## Public Member Functions

- **RenameFileJob** ([IOJobData](#) \*const data)

## Public Member Functions inherited from [Digikam::ActionJob](#)

- **ActionJob** (QObject \*const parent=nullptr)  
*Constructor which delegate deletion of QRunnable instance to [ActionThreadBase](#), not QThreadPool.*
- **~ActionJob** () override  
*Re-implement destructor in you implementation.*

## Protected Member Functions

- void **run** () override

## Additional Inherited Members

## Public Slots inherited from [Digikam::ActionJob](#)

- void **cancel** ()  
*Call this method to cancel job.*

## Public Attributes inherited from [Digikam::ActionJob](#)

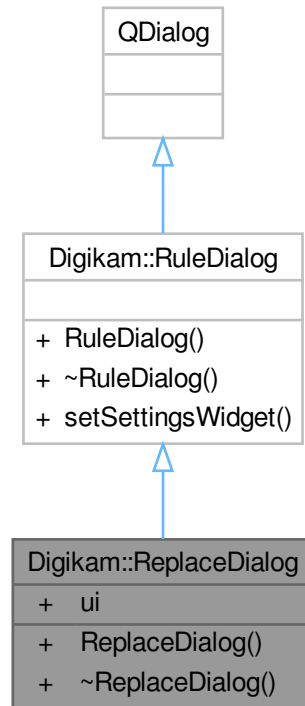
- QElapsedTimer **m\_timer**  
*Timer to determine the running time of the job.*

## Protected Attributes inherited from [Digikam::ActionJob](#)

- bool **m\_cancel** = false  
*You can use this boolean in your implementation to know if job must be canceled.*

## 9.1112 Digikam::ReplaceDialog Class Reference

Inheritance diagram for Digikam::ReplaceDialog:



### Public Member Functions

- **ReplaceDialog** ([Rule](#) \*const parent)

### Public Member Functions inherited from [Digikam::RuleDialog](#)

- **RuleDialog** ([Rule](#) \*const parent)
- void **setSettingsWidget** ([QWidget](#) \*const settingsWidget)

### Public Attributes

- `Ui::ReplaceModifierDialogWidget` \*const **ui** = nullptr

## 9.1113 Digikam::ReplaceModifier Class Reference

Inheritance diagram for Digikam::ReplaceModifier:



### Public Member Functions

- QString [parseOperation](#) ([ParseSettings](#) &settings, const QRegularExpressionMatch &match) override  
*TODO: describe me.*

## Public Member Functions inherited from [Digikam::Modifier](#)

- **Modifier** (const QString &name, const QString &description)
- **Modifier** (const QString &name, const QString &description, const QString &icon)

## Public Member Functions inherited from [Digikam::Rule](#)

- **Rule** (const QString &name)
- **Rule** (const QString &name, const QString &icon)
- QString **description** () const
- QPixmap **icon** (Rule::IconType type=Rule::Action) const
- bool **isValid** () const

*Checks the validity of the parse object.*

- **ParseResults parse** ([ParseSettings](#) &settings)
- QRegularExpression & **regExp** () const
- TODO: This is probably not needed anymore.*
- QPushButton \* **registerButton** (QWidget \*parent)
- Register a button in the parent object.*
- QAction \* **registerMenu** (QMenu \*parent)
- Register a menu action in the parent object.*
- virtual void **reset** ()
- Resets the parser to its initial state.*
- TokenList & **tokens** () const
- bool **useTokenMenu** () const
- Returns true if a token menu is used.*

## Additional Inherited Members

## Public Types inherited from [Digikam::Rule](#)

- enum **IconType** { **Action** = 0 , **Dialog** }

## Signals inherited from [Digikam::Rule](#)

- void **signalTokenTriggered** (const QString &)

## Static Public Member Functions inherited from [Digikam::Rule](#)

- static QString **escapeToken** (const QString &token)
- Escape the token characters to make them work in regular expressions.*

## Protected Slots inherited from [Digikam::Rule](#)

- virtual void **slotTokenTriggered** (const QString &)



## Protected Member Functions inherited from [Digikam::Rule](#)

- bool [addToken](#) (const QString &id, const QString &description, const QString &actionName=QString())  
*add a token to the parser, every parser should at least assign one token object*
- void [setDescription](#) (const QString &desc)
- void [setIcon](#) (const QString &pixmap)
- void [setRegExp](#) (const QRegularExpression &regExp)
- void [setUseTokenMenu](#) (bool value)  
*If multiple tokens have been assigned to a rule, a menu will be created.*

### 9.1113.1 Member Function Documentation

#### 9.1113.1.1 [parseOperation\(\)](#)

```
QString Digikam::ReplaceModifier::parseOperation (
    ParseSettings & settings,
    const QRegularExpressionMatch & match ) [override], [virtual]
```

##### Parameters

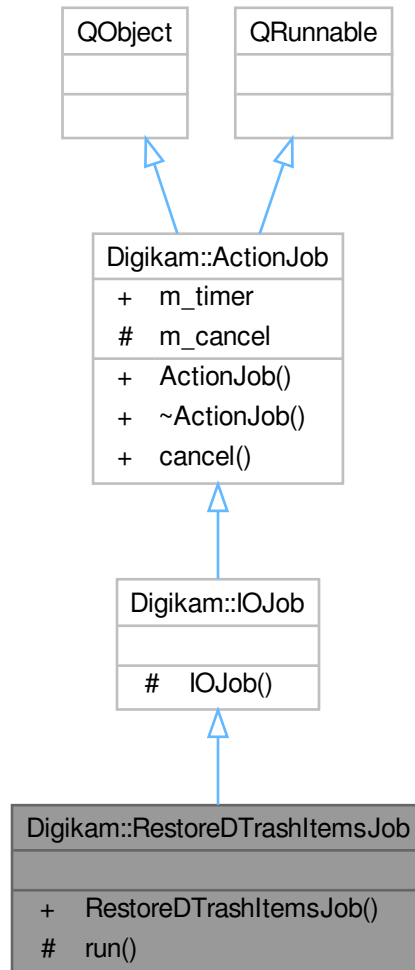
<i>settings</i>	contains settings
<i>match</i>	result of the regular expression match done in <a href="#">Option::parse()</a>

##### Returns

Implements [Digikam::Modifier](#).

## 9.1114 Digikam::RestoreDTrashItemsJob Class Reference

Inheritance diagram for Digikam::RestoreDTrashItemsJob:



### Public Member Functions

- **RestoreDTrashItemsJob** ([IOJobData](#) \*const data)

### Public Member Functions inherited from [Digikam::ActionJob](#)

- **ActionJob** ([QObject](#) \*const parent=nullptr)
  - Constructor which delegate deletion of [QRunnable](#) instance to [ActionThreadBase](#), not [QThreadPool](#).*
- **~ActionJob** () override
  - Re-implement destructor in you implementation.*

### Protected Member Functions

- void **run** () override

### Additional Inherited Members

### Public Slots inherited from [Digikam::ActionJob](#)

- void **cancel** ()  
*Call this method to cancel job.*

### Signals inherited from [Digikam::IOJob](#)

- void **signalError** (const QString &errMsg)
- void **signalOneProcessed** (const QUrl &url)

### Signals inherited from [Digikam::ActionJob](#)

- void **signalDone** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job is done.*
- void **signalProgress** (int)  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job progress.*
- void **signalStarted** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager that job is started.*

### Public Attributes inherited from [Digikam::ActionJob](#)

- QElapsedTimer **m\_timer**  
*Timer to determine the running time of the job.*

### Protected Attributes inherited from [Digikam::ActionJob](#)

- bool **m\_cancel** = false  
*You can use this boolean in your implementation to know if job must be canceled.*

## 9.1115 Digikam::RGBackend Class Reference

This class is a base class for Open Street Map and Geonames backends.

Inheritance diagram for Digikam::RGBackend:



### Signals

- void **signalRGReady** (const QList< [RGInfo](#) > &)  
*Emitted whenever some items are ready.*

### Public Member Functions

- **RGBackend** (QObject \*const parent)  
*Constructor.*
- virtual QString **backendName** ()
- virtual void **callRGBackend** (const QList< [RGInfo](#) > &, const QString &)=0
- virtual void **cancelRequests** ()=0
- virtual QString **getErrorMessage** ()

### 9.1115.1 Member Function Documentation

#### 9.1115.1.1 backendName()

```
QString Digikam::RGBackend::backendName () [virtual]
```

Reimplemented in [Digikam::BackendGeonamesRG](#), [Digikam::BackendGeonamesUSRG](#), and [Digikam::BackendOsmRG](#).

### 9.1115.1.2 callRGBBackend()

```
virtual void Digikam::RGBBackend::callRGBBackend (
    const QList< RGInfo > & ,
    const QString & ) [pure virtual]
```

Implemented in [Digikam::BackendGeonamesRG](#), [Digikam::BackendGeonamesUSRG](#), and [Digikam::BackendOsmRG](#).

### 9.1115.1.3 getErrorMessage()

```
QString Digikam::RGBBackend::getErrorMessage ( ) [virtual]
```

Reimplemented in [Digikam::BackendGeonamesRG](#), [Digikam::BackendGeonamesUSRG](#), and [Digikam::BackendOsmRG](#).

## 9.1116 Digikam::RGInfo Class Reference

This class contains data needed in reverse geocoding process.

### Public Member Functions

- **RGInfo ()**=default  
*Constructor.*
- **~RGInfo ()**=default  
*Destructor.*

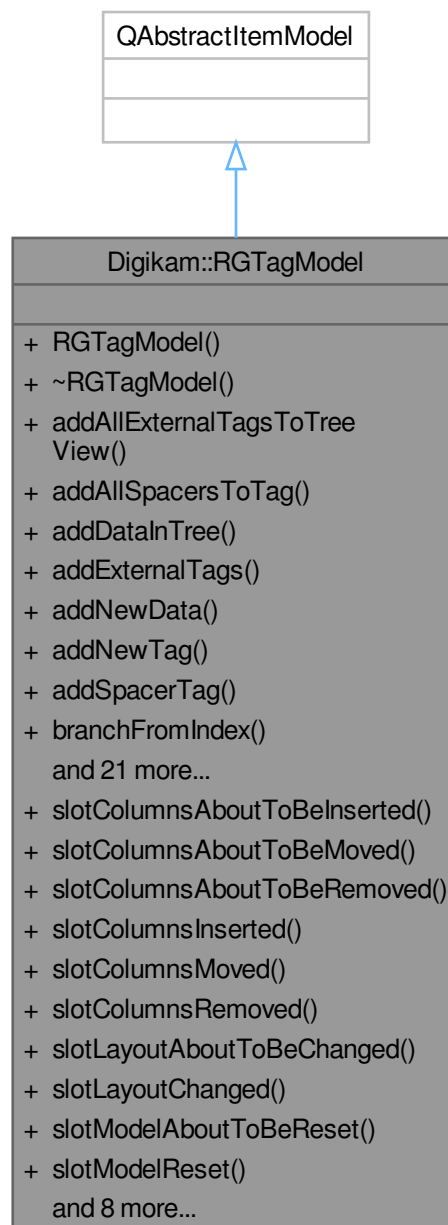
### Public Attributes

- [GeoCoordinates](#) **coordinates**  
*The coordinates of current image.*
- [QPersistentModelIndex](#) **id**  
*The image index.*
- [QMap< QString, QString >](#) **rgData**  
*The address elements and their names.*

## 9.1117 Digikam::RGTAGModel Class Reference

The model that holds data for the tag tree displayed in ReverseGeocodingWidget.

Inheritance diagram for Digikam::RGTAGModel:



### Public Slots

- void **slotColumnsAboutToBeInserted** (const QModelIndex &parent, int start, int end)

- void **slotColumnsAboutToBeMoved** (const QModelIndex &sourceParent, int sourceStart, int sourceEnd, const QModelIndex &destinationParent, int destinationColumn)
- void **slotColumnsAboutToBeRemoved** (const QModelIndex &parent, int start, int end)
- void **slotColumnsInserted** ()
- void **slotColumnsMoved** ()
- void **slotColumnsRemoved** ()
- void **slotLayoutAboutToBeChanged** ()
- void **slotLayoutChanged** ()
- void **slotModelAboutToBeReset** ()
- void **slotModelReset** ()
- void **slotRowsAboutToBeInserted** (const QModelIndex &parent, int start, int end)
- void **slotRowsAboutToBeMoved** (const QModelIndex &sourceParent, int sourceStart, int sourceEnd, const QModelIndex &destinationParent, int destinationRow)
- void **slotRowsAboutToBeRemoved** (const QModelIndex &parent, int start, int end)
- void **slotRowsInserted** ()
- void **slotRowsMoved** ()
- void **slotRowsRemoved** ()
- void **slotSourceDataChanged** (const QModelIndex &topLeft, const QModelIndex &bottomRight)
- void **slotSourceHeaderDataChanged** (const Qt::Orientation orientation, int first, int last)

### Public Member Functions

- [RGTagModel](#) (QAbstractItemModel \*const externalTagModel, QObject \*const parent=nullptr)  
*Constructor.*
- [~RGTagModel](#) () override  
*Destructor.*
- void **addAllExternalTagsToTreeView** ()  
*Add all external tags to the tag tree.*
- void **addAllSpacersToTag** (const QModelIndex &currentIndex, const QStringList &spacerList, int spacer←ListIndex)  
*Adds all spacers found in spacerList to the tag tree.*
- void **addDataInTree** ([TreeBranch](#) \*currentBranch, int currentRow, const QStringList &addressElements, const QStringList &elementsData)  
*The function starts to scan the tree starting with currentBranch.*
- void **addExternalTags** ([TreeBranch](#) \*parentBranch, int currentRow)  
*Add tags from host application to the tag tree.*
- QList< QList< [TagData](#) > > **addNewData** (const QStringList &elements, const QStringList &resultedData)  
*Add new tags to tag tree.*
- QPersistentModelIndex **addNewTag** (const QModelIndex &parent, const QString &newTagName, const QString &newElement)  
*Adds a tag containing data returned from backends.*
- void **addSpacerTag** (const QModelIndex &parent, const QString &spacerName)  
*Adds a spacer tag.*
- [TreeBranch](#) \* **branchFromIndex** (const QModelIndex &index) const  
*Returns the branch found at index.*
- void **climbTreeAndGetSpacers** (const [TreeBranch](#) \*currentBranch)  
*Gets the spacers addresses below currentBranch.*
- int **columnCount** (const QModelIndex &parent=QModelIndex()) const override  
*QAbstractItemModel.*
- QVariant **data** (const QModelIndex &index, int role=Qt::DisplayRole) const override
- void **deleteAllSpacersOrNewTags** (const QModelIndex &currentIndex, Type whatShouldRemove)  
*Deletes all spacers or all new tags.*

- void `deleteTag` (const QModelIndex &currentIndex)
  - Deletes a tag.*
- void `findAndDeleteSpacersOrNewTags` (TreeBranch \*currentBranch, int currentRow, Type whatShouldRemove)
  - Deletes all spacers or all new tags below current branch.*
- Qt::ItemFlags **flags** (const QModelIndex &index) const override
- QModelIndex `fromSourceIndex` (const QModelIndex &externalTagModelIndex) const
  - Local functions.*
- QList< TagData > `getSpacerAddress` (TreeBranch \*currentBranch)
  - Gets the address of a spacer.*
- QList< QList< TagData > > `getSpacers` ()
  - Gets all spacers.*
- QList< TagData > `getTagAddress` ()
  - Gets the address of a tag.*
- Type `getTagType` (const QModelIndex &index) const
  - Gets the type of a tag found at index.*
- QVariant **headerData** (int section, Qt::Orientation orientation, int role) const override
- QModelIndex **index** (int row, int column, const QModelIndex &parent=QModelIndex()) const override
- QModelIndex **parent** (const QModelIndex &index) const override
- void `readNewTags` (const QList< QList< TagData > > &tagAddressList)
  - Takes each tag contained in tagAddressList and adds it to the tag tree.*
- void `readTag` (TreeBranch \*&currentBranch, int currentRow, const QList< TagData > &tagAddressElements, int currentAddressElementIndex)
  - Reads new tags to tag tree.*
- int **rowCount** (const QModelIndex &parent=QModelIndex()) const override
- bool **setData** (const QModelIndex &index, const QVariant &value, int role) override
- bool **setHeaderData** (int section, Qt::Orientation orientation, const QVariant &value, int role) override
- QModelIndex `toSourceIndex` (const QModelIndex &tagModelIndex) const
  - Translates the model index from this model to host's tag model.*

### 9.1117.1 Detailed Description

The `RGTagModel` class is a wrapper above `QAbstractItemModel`. It holds data for the tag tree displayed in `ReverseGeocodingWidget`. The model gets the data from the tag model of host application and displays it in a `QTreeView`. It stores three type of tags: old tags (the tags that belong to the host's tag model), spacer tags (tags representing address elements or custom tags) and new tags (tags containing data retrieved from backend).

### 9.1117.2 Constructor & Destructor Documentation

#### 9.1117.2.1 RGTagModel()

```
Digikam::RGTagModel::RGTagModel (
    QAbstractItemModel *const externalTagModel,
    QObject *const parent = nullptr ) [explicit]
```

#### Parameters

<code>externalTagModel</code>	The tag model found in the host application.
<code>parent</code>	The parent object



### 9.1117.3 Member Function Documentation

#### 9.1117.3.1 addDataInTree()

```
void Digikam::RGTagModel::addDataInTree (
    TBranch * currentBranch,
    int currentRow,
    const QStringList & addressElements,
    const QStringList & elementsData )
```

When it finds a spacer containing an address element, it looks to see if the address element is found in addressElements list. If it's found, a new tag is added.

##### Parameters

<i>currentBranch</i>	The branch from where the scan starts.
<i>currentRow</i>	The row of the current branch.
<i>addressElements</i>	A list containing address elements. Example: {Country}, {City}...
<i>elementsData</i>	A list containing the name of each address element found in elements. Example: France, Paris...

#### 9.1117.3.2 addExternalTags()

```
void Digikam::RGTagModel::addExternalTags (
    TBranch * parentBranch,
    int currentRow )
```

##### Parameters

<i>parentBranch</i>	The branch that will be parent for the old tag.
<i>currentRow</i>	The row where this external tag will be added.

#### 9.1117.3.3 addNewData()

```
QList< QList< TagData > > Digikam::RGTagModel::addNewData (
    const QStringList & elements,
    const QStringList & resultedData )
```

The function starts to scan the tree from root level. When it finds a spacer containing an address element, it looks to see if the address element is found in elements list. If it's found, a new tag is added.

##### Parameters

<i>elements</i>	A list containing address elements. Example: {Country}, {City}...
<i>resultedData</i>	A list containing the name of each address element found in elements. Example: France, Paris...

**Returns**

A list containing new tags

**9.1117.3.4 addNewTag()**

```
QPersistentModelIndex Digikam::RGTagModel::addNewTag (
    const QModelIndex & parent,
    const QString & newTagName,
    const QString & newElement )
```

**Parameters**

<i>parent</i>	The index of the parent.
<i>newTagName</i>	The name of the new tag.
<i>newElement</i>	The new element of the tag.

**9.1117.3.5 addSpacerTag()**

```
void Digikam::RGTagModel::addSpacerTag (
    const QModelIndex & parent,
    const QString & spacerName )
```

**Parameters**

<i>parent</i>	The index of the parent. If parent == QModelIndex(), then the spacer is added to top-level
<i>spacerName</i>	The name of the spacer. If it's an address element, the address element name will have the form {addressElement}. For example: {Country}, {City}...

**9.1117.3.6 branchFromIndex()**

```
TreeBranch * Digikam::RGTagModel::branchFromIndex (
    const QModelIndex & index ) const
```

**Parameters**

<i>index</i>	Current model index.
--------------	----------------------

**Returns**

The branch for the current index.

**9.1117.3.7 climbTreeAndGetSpacers()**

```
void Digikam::RGTagModel::climbTreeAndGetSpacers (
    const TreeBranch * currentBranch )
```

Address means the path from rootTag to currentBranch.

## Parameters

<i>currentBranch</i>	The branch from where the search starts.
----------------------	--

**9.1117.3.8 deleteAllSpacersOrNewTags()**

```
void Digikam::RGTagModel::deleteAllSpacersOrNewTags (
    const QModelIndex & currentIndex,
    Type whatShouldRemove )
```

## Parameters

<i>currentIndex</i>	If <i>whatShouldRemove</i> represents a spacer, the function will remove all spacers below <i>currentIndex</i> . If <i>whatShouldRemove</i> represents a new tag, the function will delete all new tags.
<i>whatShouldRemove</i>	The tag type that should be removed. The options are: spacers or new tags.

**9.1117.3.9 deleteTag()**

```
void Digikam::RGTagModel::deleteTag (
    const QModelIndex & currentIndex )
```

## Parameters

<i>currentIndex</i>	The tag found at this index will be deleted.
---------------------	--

**9.1117.3.10 findAndDeleteSpacersOrNewTags()**

```
void Digikam::RGTagModel::findAndDeleteSpacersOrNewTags (
    TreeBranch * currentBranch,
    int currentRow,
    Type whatShouldRemove )
```

## Parameters

<i>currentBranch</i>	The tree branch from where the scan starts.
<i>currentRow</i>	The row of current branch.
<i>whatShouldRemove</i>	The tag type that should to be removed. The options are: spacers or new tags.

**9.1117.3.11 fromSourceIndex()**

```
QModelIndex Digikam::RGTagModel::fromSourceIndex (
    const QModelIndex & externalTagModelIndex ) const
```

Translates the model index from host's tag model to this model.

**Returns**

The index of current old tag.

**9.1117.3.12 getSpacerAddress()**

```
QList< TagData > Digikam::RGTAGModel::getSpacerAddress (
    TreeBranch * currentBranch )
```

Address means the path from rootTag to currentBranch

**Parameters**

<i>currentBranch</i>	The branch where the scan stops.
----------------------	----------------------------------

**Returns**

The tag address of currentBranch

**9.1117.3.13 getSpacers()**

```
QList< QList< TagData > > Digikam::RGTAGModel::getSpacers ( )
```

**Returns**

The spacer list.

**9.1117.3.14 getTagType()**

```
Type Digikam::RGTAGModel::getTagType (
    const QModelIndex & index ) const
```

**Parameters**

<i>index</i>	The index of the tag.
--------------	-----------------------

**Returns**

The type of the tag found at index.

**9.1117.3.15 readdNewTags()**

```
void Digikam::RGTAGModel::readdNewTags (
    const QList< QList< TagData > > & tagAddressList )
```

## Parameters

<i>tagAddressList</i>	A list containing new tags.
-----------------------	-----------------------------

**9.1117.3.16 readTag()**

```
void Digikam::RGTagModel::readTag (
    TreeBranch * & currentBranch,
    int currentRow,
    const QList< TagData > & tagAddressElements,
    int currentAddressElementIndex )
```

## Parameters

<i>currentBranch</i>	The branch from where the scan starts.
<i>currentRow</i>	The row of the currentBranch.
<i>tagAddressElements</i>	A list containing address elements. Example: {Country}, {City}...
<i>currentAddressElementIndex</i>	The current element in the tag address list.

## Note

tagAddressElements contains address tag: Places,Spain,Barcelona readTag climbs the tree and checks on each level if tagAddressElements[level] is found. if the tag is found, it climbs up the next level else, it recreates the new tag and climbs up that tree.

**9.1117.3.17 toSourceIndex()**

```
QModelIndex Digikam::RGTagModel::toSourceIndex (
    const QModelIndex & tagModelIndex ) const
```

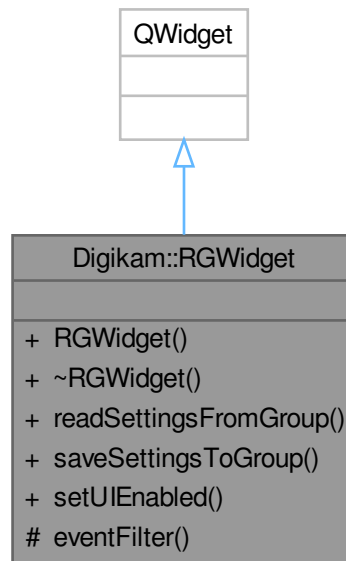
## Returns

The index of a tag in host's tag model.

**9.1118 Digikam::RGWidget Class Reference**

The [RGWidget](#) class represents the main widget for reverse geocoding.

Inheritance diagram for Digikam::RGWidget:



## Signals

- void [signalProgressChanged](#) (const int currentProgress)  
*Counts how many images were processed.*
- void **signalProgressSetup** (const int maxProgress, const QString &progressText)  
*Update the progress bar.*
- void [signalSetUIEnabled](#) (const bool enabledState)  
*This signal emits when containing widgets need to be enabled or disabled.*
- void **signalSetUIEnabled** (const bool enabledState, QObject \*const cancelObject, const QString &cancelSlot)
- void [signalUndoCommand](#) ([GPSUndoCommand](#) \*undoCommand)  
*Sends the needed data to Undo/Redo Widget.*

## Public Member Functions

- [RGWidget](#) ([GPSItemModel](#) \*const imageModel, [QItemSelectionModel](#) \*const selectionModel, [QAbstractItemModel](#) \*externTagModel, [QWidget](#) \*const parent=nullptr)  
*Constructor.*
- **~RGWidget** () override  
*Destructor.*
- void [readSettingsFromGroup](#) (const [KConfigGroup](#) \*const group)  
*Restores the settings of widgets contained in reverse geocoding widget.*
- void [saveSettingsToGroup](#) ([KConfigGroup](#) \*const group)  
*Saves the settings of widgets contained in reverse geocoding widget.*
- void [setUIEnabled](#) (const bool state)  
*Sets whether the containing widgets are enabled or disabled.*

## Protected Member Functions

- bool **eventFilter** (QObject \*watched, QEvent \*event) override  
*Here are filtered the events.*

## 9.1118.1 Constructor & Destructor Documentation

### 9.1118.1.1 RGWidget()

```
Digikam::RGWidget::RGWidget (
    GPSItemModel *const imageModel,
    QItemSelectionModel *const selectionModel,
    QAbstractItemModel * externTagModel,
    QWidget *const parent = nullptr ) [explicit]
```

#### Parameters

<i>imageModel</i>	The image model
<i>selectionModel</i>	The image selection model
<i>externTagModel</i>	The tag model
<i>parent</i>	The parent object

## 9.1118.2 Member Function Documentation

### 9.1118.2.1 readSettingsFromGroup()

```
void Digikam::RGWidget::readSettingsFromGroup (
    const KConfigGroup *const group )
```

#### Parameters

<i>group</i>	Here are stored the settings.
--------------	-------------------------------

### 9.1118.2.2 saveSettingsToGroup()

```
void Digikam::RGWidget::saveSettingsToGroup (
    KConfigGroup *const group )
```

#### Parameters

<i>group</i>	Here are stored the settings.
--------------	-------------------------------

### 9.1118.2.3 setUIEnabled()

```
void Digikam::RGWidget::setUIEnabled (
```

```
const bool state )
```

**Parameters**

<i>state</i>	If true, the controls are enabled.
--------------	------------------------------------

**9.1118.2.4 signalProgressChanged**

```
void Digikam::RGWidget::signalProgressChanged (
    const int currentProgress ) [signal]
```

**Parameters**

<i>currentProgress</i>	The number of processed images.
------------------------	---------------------------------

**9.1118.2.5 signalSetUIEnabled**

```
void Digikam::RGWidget::signalSetUIEnabled (
    const bool enabledState ) [signal]
```

**Parameters**

<i>enabledState</i>	If true, the containing widgets will be enabled. Else, they will be disabled.
---------------------	---

**9.1118.2.6 signalUndoCommand**

```
void Digikam::RGWidget::signalUndoCommand (
    GPSUndoCommand * undoCommand ) [signal]
```

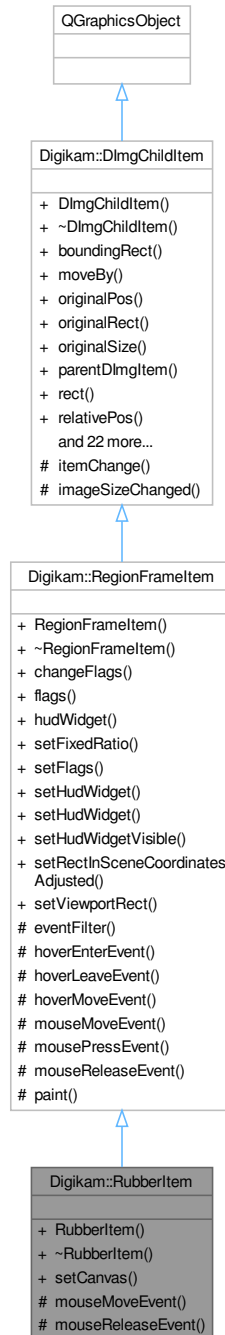
**Parameters**

<i>undoCommand</i>	Holds the data that will be used for undo or redo actions
--------------------	---



## 9.1119 Digikam::RubberItem Class Reference

Inheritance diagram for Digikam::RubberItem:



### Public Member Functions

- **RubberItem** ([ImagePreviewItem](#) \*const item)
- void **setCanvas** ([Canvas](#) \*const canvas)

## Public Member Functions inherited from [Digikam::RegionFrameItem](#)

- **RegionFrameItem** (QGraphicsItem \*const parent)
- void **changeFlags** (Flags flags, bool addOrRemove)
- Flags **flags** () const
- QGraphicsWidget \* **hudWidget** () const
- void **setFixedRatio** (double ratio)
- void **setFlags** (Flags flags)
- void **setHudWidget** (QGraphicsWidget \*const hudWidget)
  - Sets a widget item as HUD item.*
- void **setHudWidget** (QWidget \*const widget, Qt::WindowFlags wFlags=Qt::WindowFlags())
- void **setHudWidgetVisible** (bool visible)
- void **setRectInSceneCoordinatesAdjusted** (const QRectF &rect)

## Public Member Functions inherited from [Digikam::DImgChildItem](#)

- **DImgChildItem** (QGraphicsItem \*const parent=nullptr)
  - This is a base class for items that are positioned on top of a [GraphicsDImgItem](#), positioned in relative coordinates, i.e.*
- QRectF **boundingRect** () const override
  - Reimplemented.*
- void **moveBy** (qreal dx, qreal dy)
- QPoint **originalPos** () const
- QRect **originalRect** () const
  - Returns the position and size in coordinates of the original image.*
- QSize **originalSize** () const
- [GraphicsDImgItem](#) \* **parentDImgItem** () const
  - If the parent item is a [GraphicsDImgItem](#), return it, if the parent item is null or of a different class, returns 0.*
- QRectF **rect** () const
  - Returns position and size of this item, in coordinates of the parent [DImg](#) with the current zoom.*
- QPointF **relativePos** () const
- QRectF **relativeRect** () const
  - Returns the position and size relative to the [DImg](#) displayed in the parent item.*
- QSizeF **relativeSize** () const
- void **setOriginalPos** (const QPointF &posInOriginal)
  - Sets the position and size of this item, in coordinates of the original image.*
- void **setOriginalPos** (qreal x, qreal y)
- void **setOriginalRect** (const QRectF &rect)
- void **setOriginalRect** (qreal x, qreal y, qreal width, qreal height)
- void **setOriginalSize** (const QSizeF &sizeInOriginal)
- void **setOriginalSize** (qreal width, qreal height)
- void **setPos** (const QPointF &zoomedPos)
  - Sets the position and size of this item, in coordinates of the parent [DImg](#) item.*
- void **setPos** (qreal x, qreal y)
- void **setRect** (const QRectF &rect)
- void **setRect** (qreal x, qreal y, qreal width, qreal height)
- void **setRectInSceneCoordinates** (const QRectF &rect)
  - Equivalent to mapping the scene coordinates to the parent item, and calling [setRect\(\)](#).*
- void **setRelativePos** (const QPointF &relativePosition)
  - Sets the position and size of this item, relative to the [DImg](#) displayed in the parent item.*
- void **setRelativePos** (qreal x, qreal y)
- void **setRelativeRect** (const QRectF &rect)
- void **setRelativeRect** (qreal x, qreal y, qreal width, qreal height)
- void **setRelativeSize** (const QSizeF &relativeSize)
- void **setRelativeSize** (qreal width, qreal height)
- void **setSize** (const QSizeF &zoomedSize)
- void **setSize** (qreal width, qreal height)
- QSizeF **size** () const

**Protected Member Functions**

- void **mouseMoveEvent** (QGraphicsSceneMouseEvent \*event) override
- void **mouseReleaseEvent** (QGraphicsSceneMouseEvent \*event) override

**Protected Member Functions inherited from [Digikam::RegionFrameItem](#)**

- bool **eventFilter** (QObject \*watched, QEvent \*event) override
- void **hoverEnterEvent** (QGraphicsSceneHoverEvent \*event) override
- void **hoverLeaveEvent** (QGraphicsSceneHoverEvent \*event) override
- void **hoverMoveEvent** (QGraphicsSceneHoverEvent \*event) override
- void **mouseMoveEvent** (QGraphicsSceneMouseEvent \*) override
- void **mousePressEvent** (QGraphicsSceneMouseEvent \*) override
- void **mouseReleaseEvent** (QGraphicsSceneMouseEvent \*) override
- void **paint** (QPainter \*painter, const QStyleOptionGraphicsItem \*option, QWidget \*widget=nullptr) override

**Protected Member Functions inherited from [Digikam::DImgChildItem](#)**

- QVariant **itemChange** (GraphicsItemChange change, const QVariant &value) override

**Additional Inherited Members****Public Types inherited from [Digikam::RegionFrameItem](#)**

- enum **Flag** { **NoFlags** = 0 , **ShowResizeHandles** = 1 << 0 , **MoveByDrag** = 1 << 1 , **GeometryEditable** = ShowResizeHandles | MoveByDrag }
- typedef QFlags< Flag > **Flags**

**Public Slots inherited from [Digikam::RegionFrameItem](#)**

- void **setViewportRect** (const QRectF &rect)  
*The associated HUD item is dynamically moved to be visible.*

**Signals inherited from [Digikam::RegionFrameItem](#)**

- void **geometryEdited** ()

**Signals inherited from [Digikam::DImgChildItem](#)**

- void **geometryChanged** ()
- void **geometryOnImageChanged** ()
- void **positionChanged** ()  
*These signals are emitted in any case when the geometry changed: Either after changing the geometry relative to the original image, or when the size of the parent [GraphicsDImgItem](#) changed (zooming).*
- void **positionOnImageChanged** ()  
*These signals are emitted when the geometry, relative to the original image, of this item has changed.*
- void **sizeChanged** ()
- void **sizeOnImageChanged** ()

## Protected Slots inherited from [Digikam::DImgChildItem](#)

- void **imageSizeChanged** (const QSizeF &)

## 9.1120 Digikam::Rule Class Reference

Inheritance diagram for Digikam::Rule:



### Public Types

- enum **IconType** { **Action** = 0 , **Dialog** }

### Signals

- void **signalTokenTriggered** (const QString &)

### Public Member Functions

- **Rule** (const QString &name)
- **Rule** (const QString &name, const QString &icon)
- QString **description** () const
- QPixmap **icon** (Rule::IconType type=Rule::Action) const
- bool **isValid** () const  
*Checks the validity of the parse object.*
- ParseResults **parse** (ParseSettings &settings)
- QRegularExpression & **regExp** () const  
*TODO: This is probably not needed anymore.*
- QPushButton \* **registerButton** (QWidget \*parent)  
*Register a button in the parent object.*
- QAction \* **registerMenu** (QMenu \*parent)  
*Register a menu action in the parent object.*
- virtual void **reset** ()  
*Resets the parser to its initial state.*
- TokenList & **tokens** () const
- bool **useTokenMenu** () const  
*Returns true if a token menu is used.*

### Static Public Member Functions

- static QString [escapeToken](#) (const QString &token)  
*Escape the token characters to make them work in regular expressions.*

### Protected Slots

- virtual void [slotTokenTriggered](#) (const QString &)

### Protected Member Functions

- bool [addToken](#) (const QString &id, const QString &description, const QString &actionName=QString())  
*add a token to the parser, every parser should at least assign one token object*
- virtual QString [parseOperation](#) (ParseSettings &settings, const QRegularExpressionMatch &match)=0  
*TODO: describe me.*
- void [setDescription](#) (const QString &desc)
- void [setIcon](#) (const QString &pixmap)
- void [setRegExp](#) (const QRegularExpression &regExp)
- void [setUseTokenMenu](#) (bool value)  
*If multiple tokens have been assigned to a rule, a menu will be created.*

## 9.1120.1 Member Function Documentation

### 9.1120.1.1 addToken()

```
bool Digikam::Rule::addToken (
    const QString & id,
    const QString & description,
    const QString & actionName = QString() ) [protected]
```

#### Parameters

<i>id</i>	the token id string (used for parsing)
<i>description</i>	the description of the token (used for example in the tooltip)
<i>actionName</i>	[optional] the name of the token action (only used when the token menu is displayed)

#### Returns

### 9.1120.1.2 escapeToken()

```
QString Digikam::Rule::escapeToken (
    const QString & token ) [static]
```

**Parameters**

<i>token</i>	the token to be escaped
--------------	-------------------------

**Returns**

A token with escaped characters. This token can then be used in a regular expression

**9.1120.1.3 isValid()**

```
bool Digikam::Rule::isValid ( ) const
```

**Returns**

true if valid

**9.1120.1.4 parseOperation()**

```
virtual QString Digikam::Rule::parseOperation (
    ParseSettings & settings,
    const QRegularExpressionMatch & match ) [protected], [pure virtual]
```

**Parameters**

<i>settings</i>	contains settings
<i>match</i>	result of the regular expression match done in <code>Option::parse()</code>

**Returns**

Implemented in [Digikam::CaseModifier](#), [Digikam::DefaultValueModifier](#), [Digikam::RangeModifier](#), [Digikam::RemoveDoublesModifier](#), [Digikam::ReplaceModifier](#), [Digikam::TrimmedModifier](#), [Digikam::UniqueModifier](#), [Digikam::CameraNameOption](#), [Digikam::DatabaseOption](#), [Digikam::DateOption](#), [Digikam::DirectoryNameOption](#), [Digikam::FilePropertiesOption](#), [Digikam::MetadataOption](#), [Digikam::SequenceNumberOption](#), [Digikam::Modifier](#), and [Digikam::Option](#).

**9.1120.1.5 regExp()**

```
QRegularExpression & Digikam::Rule::regExp ( ) const
```

Find out. returns the currently assigned regExp object. Note that it is returned as a const ref, meaning that if you use it in your custom parse operation, the main parse method has already searched for the pattern and filled in the results of this search, so that you can use `QRegularExpressionMatch::captured()` immediately, you don't have to search on your own.

For example when implementing the [Option::parseOperation\(\)](#) method, get the regExp object with

```
const QRegularExpression& reg = regExp();
```

and immediately fetch possible matches with

```
const QString& param1 = reg.captured(1);
```

#### See also

[Option](#)

[Modifier](#)

#### Returns

a const ref to the assigned regexp object

#### 9.1120.1.6 registerButton()

```
QPushButton * Digikam::Rule::registerButton (
    QWidget * parent )
```

By calling this method, a new button for the parser object will be created and all necessary connections will be setup.

#### Parameters

<i>parent</i>	the parent object the button will be registered for
---------------	---

#### Returns

a pointer to the newly created button

#### 9.1120.1.7 registerMenu()

```
QAction * Digikam::Rule::registerMenu (
    QMenu * parent )
```

By calling this method, a new action for the parser object will be created and all necessary connections will be setup.

#### Parameters

<i>parent</i>	the parent object the action will be registered for
---------------	---

#### Returns

a pointer to the newly created action

### 9.1120.1.8 reset()

```
void Digikam::Rule::reset ( ) [virtual]
```

Reimplemented in [Digikam::UniqueModifier](#).

### 9.1120.1.9 setUseTokenMenu()

```
void Digikam::Rule::setUseTokenMenu (
    bool value ) [protected]
```

If you want to display a menu for every defined token, set this method to 'true' and re-implement the

See also

slotTokenTriggered method.

#### Parameters

<i>value</i>	boolean parameter to set token menu usage
--------------	---

### 9.1120.1.10 tokens()

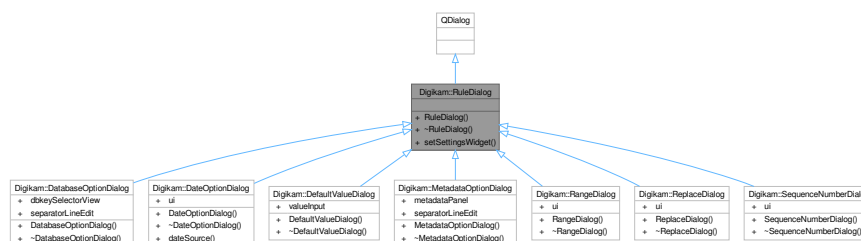
```
TokenList & Digikam::Rule::tokens ( ) const
```

Returns

a list of all registered tokens

## 9.1121 Digikam::RuleDialog Class Reference

Inheritance diagram for Digikam::RuleDialog:



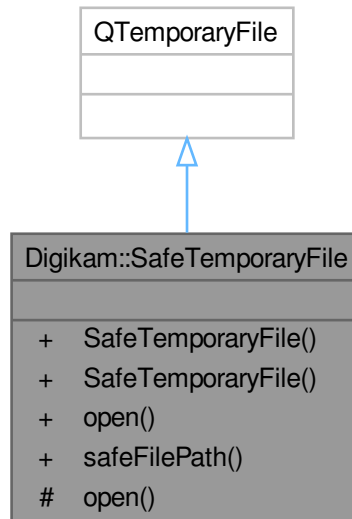
#### Public Member Functions

- **RuleDialog** ([Rule](#) \*const parent)
- void **setSettingsWidget** (QWidget \*const settingsWidget)



## 9.1122 Digikam::SafeTemporaryFile Class Reference

Inheritance diagram for Digikam::SafeTemporaryFile:



### Public Member Functions

- **SafeTemporaryFile** (const QString &templ)
- bool **open** ()
- QString **safeFilePath** () const

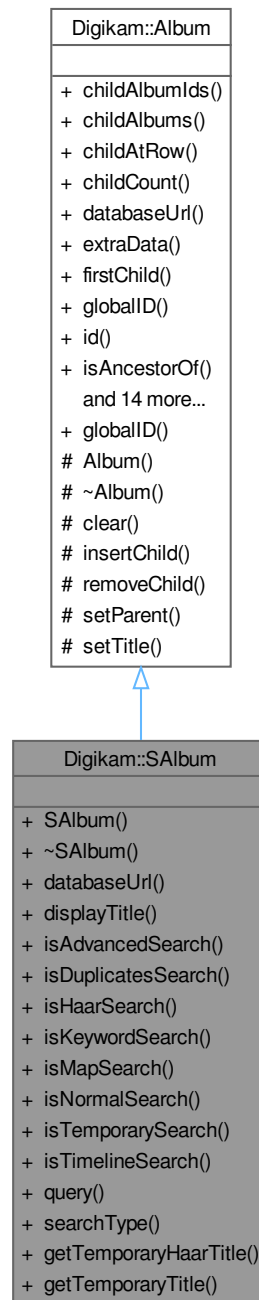
### Protected Member Functions

- bool **open** (QIODevice::OpenMode) override

## 9.1123 Digikam::SAlbum Class Reference

A Search [Album](#) representation.

Inheritance diagram for Digikam::SAlbum:



### Public Member Functions

- **SAlbum** (const QString &title, int id, bool root=false)
- **CoreDbUrl databaseUrl** () const override
- QString **displayTitle** () const
- bool **isAdvancedSearch** () const
- bool **isDuplicatesSearch** () const

- bool **isHaarSearch** () const
- bool **isKeywordSearch** () const
- bool **isMapSearch** () const
- bool **isNormalSearch** () const
- bool **isTemporarySearch** () const  
*Indicates whether this album is a temporary search or not.*
- bool **isTimelineSearch** () const
- QString **query** () const
- DatabaseSearch::Type **searchType** () const

## Public Member Functions inherited from Digikam::Album

- QList< int > **childAlbumIds** (bool recursive=false)
- AlbumList **childAlbums** (bool recursive=false)
- Album \* **childAtRow** (int row) const
- int **childCount** () const
- void \* **extraData** (const void \*const key) const  
*Retrieve the associated extra data associated with key.*
- Album \* **firstChild** () const
- int **globalID** () const  
*An album ID is only unique among the set of all Albums of its Type.*
- int **id** () const  
*Each album has a ID uniquely identifying it in the set of Albums of a Type.*
- bool **isAncestorOf** (Album \*const album) const
- bool **isRoot** () const
- bool **isTrashAlbum** () const
- bool **isUsedByLabelsTree** () const
- Album \* **lastChild** () const
- Album \* **next** () const
- Album \* **parent** () const
- void **prepareForDeletion** ()  
*For secure deletion in an album model, call this function beforehand.*
- Album \* **prev** () const
- void **removeExtraData** (const void \*const key)  
*Remove the associated extra data associated with key.*
- int **rowFromAlbum** () const
- void **setExtraData** (const void \*const key, void \*const value)  
*This allows to associate some "extra" data to a Album.*
- void **setUsedByLabelsTree** (bool isUsed)  
*Sets the property m\_usedByLabelsTree to true if the search album was created using the Colors and labels tree view.*
- QString **title** () const
- Type **type** () const

## Static Public Member Functions

- static QString **getTemporaryHaarTitle** (DatabaseSearch::HaarSearchType haarType)  
*Returns the title for a temporary haar search depending on the sub-type used for this search.*
- static QString **getTemporaryTitle** (DatabaseSearch::Type type, DatabaseSearch::HaarSearchType haarType=DatabaseSearch::HaarImageSearch)  
*Returns the title of search albums that is used to mark them as a temporary search that isn't saved officially yet and is only used for viewing purposes.*

## Static Public Member Functions inherited from [Digikam::Album](#)

- static int [globalID](#) ([Type type](#), int id)  
*Produces the global id.*

## Friends

- class [AlbumManager](#)

## Additional Inherited Members

## Public Types inherited from [Digikam::Album](#)

- enum [Type](#) {  
  [PHYSICAL](#) = 0 , [TAG](#) , [DATE](#) , [SEARCH](#) ,  
  [FACE](#) }

## Protected Member Functions inherited from [Digikam::Album](#)

- [Album](#) ([Album::Type type](#), int id, bool root)  
*Constructor.*
- virtual [~Album](#) ()  
*Destructor.*
- void [clear](#) ()  
*Delete all child albums and also remove any associated extra data.*
- void [insertChild](#) ([Album \\*const child](#))
- void [removeChild](#) ([Album \\*const child](#))
- void [setParent](#) ([Album \\*const parent](#))
- void [setTitle](#) (const [QString &title](#))

## 9.1123.1 Member Function Documentation

### 9.1123.1.1 [databaseUrl\(\)](#)

`CoreDbUrl Digikam::SAlbum::databaseUrl ( ) const [override], [virtual]`

#### Returns

the kde url of the album

Implements [Digikam::Album](#).

### 9.1123.1.2 [getTemporaryHaarTitle\(\)](#)

`QString Digikam::SAlbum::getTemporaryHaarTitle ( DatabaseSearch::HaarSearchType haarType ) [static]`

## Parameters

<i>haarType</i>	type of the haar search to get the name for
-----------------	---

## Returns

string that identifies this album uniquely as an unsaved search

**9.1123.1.3 getTemporaryTitle()**

```
QString Digikam::SAlbum::getTemporaryTitle (
    DatabaseSearch::Type type,
    DatabaseSearch::HaarSearchType haarType = DatabaseSearch::HaarImageSearch ) [static]
```

## Parameters

<i>type</i>	the type of the search to get the temporary title for
<i>haarType</i>	there are several haar searches, so that this search type needs a special handling

## Returns

string that identifies this album uniquely as an unsaved search

**9.1123.1.4 isTemporarySearch()**

```
bool Digikam::SAlbum::isTemporarySearch ( ) const
```

## Returns

true if this is a temporary search album, else false

**9.1124 Digikam::SaveProperties Class Reference****Public Attributes**

- qreal **altitude** = 0.0
- qreal **latitude** = 0.0
- qreal **longitude** = 0.0
- bool **shouldRemoveAltitude** = false
- bool **shouldRemoveCoordinates** = false
- bool **shouldWriteAltitude** = false
- bool **shouldWriteCoordinates** = false

## 9.1125 Digikam::SavingContext Class Reference

### Public Types

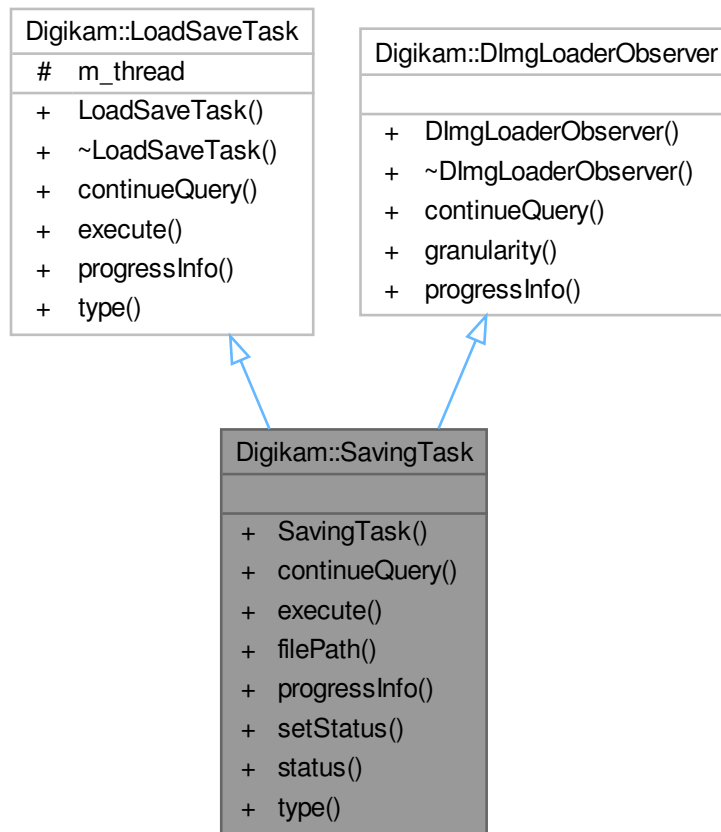
- enum **SavingState** { **SavingStateNone** , **SavingStateSave** , **SavingStateSaveAs** , **SavingStateVersion** }
- enum **SynchronizingState** { **NormalSaving** , **SynchronousSaving** }

### Public Attributes

- bool **abortingSaving** = false
- bool **destinationExisted** = false
- QUrl **destinationURL**
- SavingState **executedOperation** = SavingStateNone
- QString **format**
- QUrl **moveSrcURL**
- QString **originalFormat**
- [SafeTemporaryFile](#) \* **saveTempFile** = nullptr
- QString **saveTempFileName**
- SavingState **savingState** = SavingStateNone
- QUrl **srcURL**
- SynchronizingState **synchronizingState** = NormalSaving
- bool **synchronousSavingResult** = false
- [VersionFileOperation](#) **versionFileOperation**

## 9.1126 Digikam::SavingTask Class Reference

Inheritance diagram for Digikam::SavingTask:



### Public Types

- enum **SavingTaskStatus** { SavingTaskStatusSaving , SavingTaskStatusStopping }

### Public Types inherited from Digikam::LoadSaveTask

- enum **TaskType** { TaskTypeLoading , TaskTypeSaving }

### Public Member Functions

- **SavingTask** ([LoadSaveThread](#) \*const thread, const [DImg](#) &img, const QString &filePath, const QString &format)
- bool [continueQuery](#) () override
- void [execute](#) () override
- QString [filePath](#) () const
- void [progressInfo](#) (float progress) override
- void [setStatus](#) (SavingTaskStatus status)
- SavingTaskStatus [status](#) () const
- TaskType [type](#) () override

## Public Member Functions inherited from [Digikam::LoadSaveTask](#)

- [LoadSaveTask](#) ([LoadSaveThread](#) \*const thread)

## Public Member Functions inherited from [Digikam::DImgLoaderObserver](#)

- virtual float [granularity](#) ()

*Return a relative value which determines the granularity, the frequency with which the [DImgLoaderObserver](#) is checked and progress is posted.*

## Additional Inherited Members

## Protected Attributes inherited from [Digikam::LoadSaveTask](#)

- [LoadSaveThread](#) \* **m\_thread** = nullptr

## 9.1126.1 Member Function Documentation

### 9.1126.1.1 [continueQuery\(\)](#)

```
bool Digikam::SavingTask::continueQuery ( ) [override], [virtual]
```

Implements [Digikam::LoadSaveTask](#).

### 9.1126.1.2 [execute\(\)](#)

```
void Digikam::SavingTask::execute ( ) [override], [virtual]
```

Implements [Digikam::LoadSaveTask](#).

### 9.1126.1.3 [progressInfo\(\)](#)

```
void Digikam::SavingTask::progressInfo (
    float progress ) [override], [virtual]
```

Implements [Digikam::LoadSaveTask](#).

### 9.1126.1.4 [type\(\)](#)

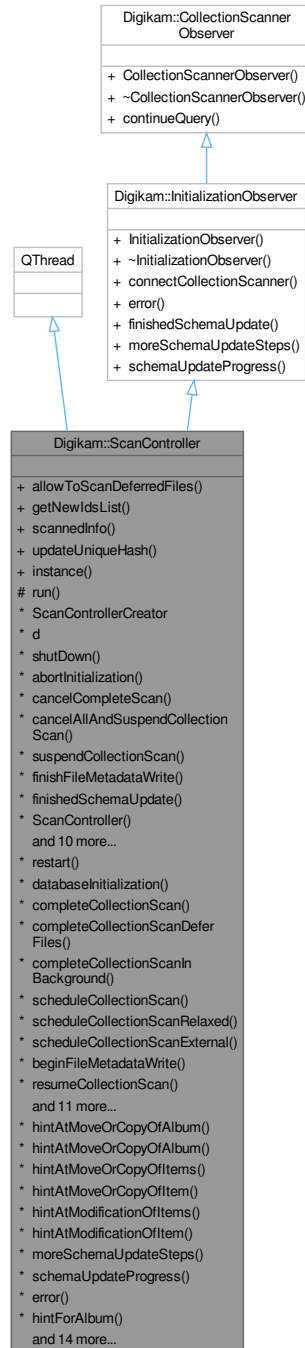
```
LoadingTask::TaskType Digikam::SavingTask::type ( ) [override], [virtual]
```

Implements [Digikam::LoadSaveTask](#).



## 9.1127 Digikam::ScanController Class Reference

Inheritance diagram for Digikam::ScanController:



### Classes

- class [FileMetadataWrite](#)

*When writing metadata to the file, the file content on disk changes, but the information is taken from the database; therefore, the resulting scanning process can be optimized.*

## Public Types

- enum **Advice** { **Success** , **ContinueWithoutDatabase** , **AbortImmediately** }

## Public Types inherited from [Digikam::InitializationObserver](#)

- enum **UpdateResult** { **UpdateSuccess** , **UpdateError** , **UpdateErrorMustAbort** }

## Public Member Functions

- void **allowToScanDeferredFiles** ()
- QList< qlonglong > **getNewIdsList** () const  
*Returns item ids from new detected items.*
- **ItemInfo scannedInfo** (const QString &filePath, [CollectionScanner::FileScanMode](#) mode=[CollectionScanner::NormalScan](#))  
*If necessary (modified or newly created, scans the file directly Returns the up-to-date ItemInfo.*
- void **updateUniqueHash** ()  
*Carries out a complete collection scan, at the same time updating the unique hash in the database and thumbnail database.*

## Static Public Member Functions

- static [ScanController](#) \* **instance** ()

## Protected Member Functions

- void **run** () override

## Stop Operations

- class **ScanControllerCreator**
- void **shutDown** ()  
*Wait for the thread to finish.*
- void **abortInitialization** ()  
*If the controller is currently processing a database update (typically after first run), cancel this hard and as soon as possible.*
- void **cancelCompleteScan** ()  
*If the controller is currently doing a complete scan (typically at startup), stop this operation.*
- void **cancelAllAndSuspendCollectionScan** ()  
*Cancels all running or scheduled operations and suspends scanning.*
- void **suspendCollectionScan** ()  
*Temporarily suspend collection scanning.*
- void **finishFileMetadataWrite** (const [ItemInfo](#) &info, bool changed)  
*Implementation of [FileMetadataWrite](#), see there.*
- void **collectionScanFinished** ()
- void **newImages** (const [ItemInfoList](#) &)
- void **partialScanDone** (const QString &path)
- void **completeScanDone** ()
- void **completeScanCanceled** ()
- void **errorFromInitialization** (const QString &)

## Start Operations

- void **restart** ()  
*Restart thread after shutdown.*
- Advice **databaseInitialization** ()  
*Calls `CoreDbAccess::checkReadyForUse()`, providing progress feedback if schema updating occurs.*
- void **completeCollectionScan** (bool defer=false)  
*Carries out a complete collection scan, providing progress feedback.*
- void **completeCollectionScanDeferFiles** ()
- void **completeCollectionScanInBackground** (bool defer, bool fastScan=true)  
*Scan Whole collection without to display a progress dialog or to manage splashscreen, as for `NewItemsFinder` tool.*
- void **scheduleCollectionScan** (const QString &path)  
*Schedules a scan of the specified part of the collection.*
- void **scheduleCollectionScanRelaxed** (const QString &path)  
*Schedules a scan of the specified part of the collection.*
- void **scheduleCollectionScanExternal** (const QString &path)  
*Schedules a scan of the specified part of the collection.*
- void **beginFileMetadataWrite** (const ItemInfo &info)  
*Implementation of `FileMetadataWrite`, see there.*
- void **resumeCollectionScan** ()  
*Resume a suspended collection scanning.*
- void **restartCollectionScan** ()  
*Restart a suspended collection scanning.*
- void **databaseInitialized** (bool success)
- void **collectionScanStarted** (const QString &message)

## Progress Operations

- void **hintAtMoveOrCopyOfAlbum** (const PAAlbum \*const album, const PAAlbum \*const dstAlbum, const QString &newAlbumName=QString())  
*Hint at the imminent copy, move or rename of an album, so that the collection scanner is informed about this.*
- void **hintAtMoveOrCopyOfAlbum** (const PAAlbum \*const album, const QString &dstPath, const QString &newAlbumName=QString())
- void **hintAtMoveOrCopyOfItems** (const QList< qlonglong > &ids, const PAAlbum \*const dstAlbum, const QStringList &itemNames)  
*Hint at the imminent copy, move or rename of items, so that the collection scanner is informed about this.*
- void **hintAtMoveOrCopyOfItem** (qlonglong id, const PAAlbum \*const dstAlbum, const QString &itemName)
- void **hintAtModificationOfItems** (const QList< qlonglong > &ids)  
*Hint at the fact that an item may have changed, although its modification date may not have changed.*
- void **hintAtModificationOfItem** (qlonglong id)
- void **totalFilesToScan** (int)
- void **startScanningAlbum** (const QString &albumRoot, const QString &album)
- void **filesScanned** (int)
- void **scanningProgress** (float progress)
- void **triggerShowProgressDialog** ()
- void **incrementProgressDialog** (int)
- void **progressFromInitialization** (const QString &, int)

## 9.1127.1 Member Function Documentation

### 9.1127.1.1 abortInitialization()

```
void Digikam::ScanController::abortInitialization ( )
```

Any progress may be lost.

### 9.1127.1.2 beginFileMetadataWrite()

```
void Digikam::ScanController::beginFileMetadataWrite (
    const ItemInfo & info )
```

Calling these methods is equivalent.

### 9.1127.1.3 cancelAllAndSuspendCollectionScan()

```
void Digikam::ScanController::cancelAllAndSuspendCollectionScan ( )
```

This method returns when all scanning has stopped. This includes a call to [suspendCollectionScan\(\)](#). Restart with [resumeCollectionScan](#).

### 9.1127.1.4 cancelCompleteScan()

```
void Digikam::ScanController::cancelCompleteScan ( )
```

It can be resumed later.

### 9.1127.1.5 completeCollectionScan()

```
void Digikam::ScanController::completeCollectionScan (
    bool defer = false )
```

Synchronous, returns when ready. The database will be locked while the scan is running. With the [DeferFiles](#) variant, deep files scanning (new files), the part which can take long, will be done during the time after the method returns, shortening the synchronous wait. After [completeCollectionScanDeferFiles](#), you need to call [allowToScanDeferredFiles\(\)](#) once to enable scanning the deferred files.

### 9.1127.1.6 databaseInitialization()

```
ScanController::Advice Digikam::ScanController::databaseInitialization ( )
```

Synchronous, returns when ready.

### 9.1127.1.7 finishFileMetadataWrite()

```
void Digikam::ScanController::finishFileMetadataWrite (
    const ItemInfo & info,
    bool changed )
```

Calling these methods is equivalent.

### 9.1127.1.8 hintAtModificationOfItems()

```
void Digikam::ScanController::hintAtModificationOfItems (
    const QList< qlonglong > & ids )
```

Note that a scan of the containing directory will need to be triggered nonetheless for the hints to take effect.

### 9.1127.1.9 hintAtMoveOrCopyOfAlbum()

```
void Digikam::ScanController::hintAtMoveOrCopyOfAlbum (
    const PAlbum *const album,
    const PAlbum *const dstAlbum,
    const QString & newAlbumName = QString() )
```

If the album is renamed, give the new name in newAlbumName. DstAlbum is the new parent album / dstPath is the new parent directory of the album, so do not include the album name to dstPath.

### 9.1127.1.10 hintAtMoveOrCopyOfItems()

```
void Digikam::ScanController::hintAtMoveOrCopyOfItems (
    const QList< qlonglong > & ids,
    const PAlbum *const dstAlbum,
    const QStringList & itemNames )
```

Give the list of existing items, specify the destination with dstAlbum, and give the names at destination in itemNames (Unless for rename, names wont usually change. Give them nevertheless.)

### 9.1127.1.11 restartCollectionScan()

```
void Digikam::ScanController::restartCollectionScan ( )
```

All scheduled scanning tasks are queued and will be done when [restartCollectionScan\(\)](#) has been called.

### 9.1127.1.12 resumeCollectionScan()

```
void Digikam::ScanController::resumeCollectionScan ( )
```

All scheduled scanning tasks are queued and will be done when [resumeCollectionScan\(\)](#) has been called. Calling these methods is recursive, you must resume as often as you called suspend.

### 9.1127.1.13 `scheduleCollectionScan()`

```
void Digikam::ScanController::scheduleCollectionScan (
    const QString & path )
```

Asynchronous, returns immediately.

### 9.1127.1.14 `scheduleCollectionScanExternal()`

```
void Digikam::ScanController::scheduleCollectionScanExternal (
    const QString & path )
```

Asynchronous, returns immediately. A very long delay with timer restart may be introduced before the actual scanning starts, so that you can call this often without checking for duplicates. This method is only for the `QFileSystem↔` `Watcher`.

### 9.1127.1.15 `scheduleCollectionScanRelaxed()`

```
void Digikam::ScanController::scheduleCollectionScanRelaxed (
    const QString & path )
```

Asynchronous, returns immediately. A small delay may be introduced before the actual scanning starts, so that you can call this often without checking for duplicates. This method must only be used from the main thread.

### 9.1127.1.16 `shutDown()`

```
void Digikam::ScanController::shutDown ( )
```

Returns after all tasks are done.

### 9.1127.1.17 `suspendCollectionScan()`

```
void Digikam::ScanController::suspendCollectionScan ( )
```

All scheduled scanning tasks are queued and will be done when `resumeCollectionScan()` has been called. Calling these methods is recursive, you must resume as often as you called suspend.

### 9.1127.1.18 `updateUniqueHash()`

```
void Digikam::ScanController::updateUniqueHash ( )
```

Synchronous, returns when ready. The database will be locked while the scan is running.

## 9.1128 `Digikam::ScanController::FileMetadataWrite` Class Reference

When writing metadata to the file, the file content on disk changes, but the information is taken from the database; therefore, the resulting scanning process can be optimized.

### Public Member Functions

- **FileMetadataWrite** (const [ItemInfo](#) &info)
- void **changed** (bool wasChanged)

### Protected Attributes

- bool **m\_changed** = false
- [ItemInfo](#) **m\_info**

## 9.1128.1 Detailed Description

Thus, if you write metadata of an [ItemInfo](#) from the database to disk, do this in the scope of a [FileMetadataWrite](#) object.

## 9.1129 Digikam::ScanStateFilter Class Reference

Inheritance diagram for Digikam::ScanStateFilter:



### Signals

- void **signalInfosToDispatch** ()



## Signals inherited from Digikam::DynamicThread

- void **finished** ()
- void **starting** ()

*Emitted if emitSignals is enabled.*

## Public Member Functions

- **ScanStateFilter** ([FacePipeline::FilterMode](#) fmode, [FacePipeline::Private](#) \*const dd)
- [FacePipelineExtendedPackage::Ptr](#) **filter** (const [ItemInfo](#) &info)
- void **process** (const [ItemInfo](#) &info)
- void **process** (const [QList](#)< [ItemInfo](#) > &infos)

## Public Member Functions inherited from Digikam::DynamicThread

- [DynamicThread](#) ([QObject](#) \*const parent=nullptr)
 

*This class extends [QRunnable](#), so you have to reimplement virtual void [run\(\)](#).*
- [~DynamicThread](#) () override
 

*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool **isFinished** () const
- bool **isRunning** () const
- [QThread::Priority](#) **priority** () const
- void **setEmitSignals** (bool emitThem)
- void **setPriority** ([QThread::Priority](#) priority)
 

*Sets the priority for this dynamic thread.*
- State **state** () const

## Public Attributes

- [FacePipeline::Private](#) \*const **d** = nullptr
- [FacePipeline::FilterMode](#) **mode** = [FacePipeline::SkipAlreadyScanned](#)
- [FacePipelineFaceTagsiface::Roles](#) **tasks**

## Protected Slots

- void **dispatch** ()

## Protected Member Functions

- void **run** () override
 

*Implement this pure virtual function in your subclass.*

## Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool **runningFlag** () const volatile  
*In you [run\(\)](#) method, you shall regularly check for [runningFlag\(\)](#) and cleanup and return if false.*
- void **shutDown** ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call [stop\(\)](#) and [wait\(\)](#), knowing that nothing will call [start\(\)](#) anymore after this 3) Be sure the thread will never be running at destruction.*
- void **start** (QMutexLocker< QMutex > &locker)  
*Doing the same as [start\(\)](#), [stop\(\)](#) and [wait](#) above, provide it with a locked QMutexLocker on mutex().*
- void **stop** (const QMutexLocker< QMutex > &locker)
- QMutex \* **threadMutex** () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void **wait** (QMutexLocker< QMutex > &locker)

## Protected Attributes

- QList< [ItemInfo](#) > **toBeSkipped**
- QList< [ItemInfo](#) > **toFilter**
- QList< [FacePipelineExtendedPackage::Ptr](#) > **toSend**

## Additional Inherited Members

## Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

## Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

## 9.1129.1 Member Function Documentation

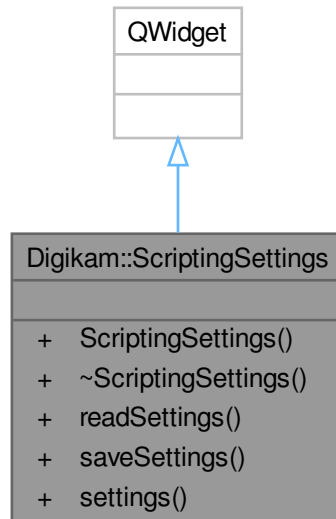
### 9.1129.1.1 run()

```
void Digikam::ScanStateFilter::run ( ) [override], [protected], [virtual]
```

Implements [Digikam::DynamicThread](#).

## 9.1130 Digikam::ScriptingSettings Class Reference

Inheritance diagram for Digikam::ScriptingSettings:



### Public Member Functions

- **ScriptingSettings** (QWidget \*const parent=nullptr)
- void **readSettings** (const KConfigGroup &group)
- void **saveSettings** (KConfigGroup &group)
- void **settings** ([DownloadSettings](#) \*const settings) const

## 9.1131 Digikam::SearchChangeset Class Reference

### Public Types

- enum **Operation** { **Unknown** , **Added** , **Deleted** , **Changed** }

### Public Member Functions

- **SearchChangeset** (int searchId, Operation operation)
- Operation **operation** () const
- [SearchChangeset](#) & **operator**<< (const QDBusArgument &argument)
- const [SearchChangeset](#) & **operator**>> (QDBusArgument &argument) const
- int **searchId** () const

## 9.1132 Digikam::SearchesDBJobInfo Class Reference

Inheritance diagram for Digikam::SearchesDBJobInfo:



### Public Member Functions

- **SearchesDBJobInfo** (QList< int > &&searchIds)
- **SearchesDBJobInfo** (QSet< qlonglong > &&imagelds, bool isAlbumUpdate, [Haarface::RefImageSelMethod](#) referenceSelectionMethod, QSet< qlonglong > &&refImagelds)

- const QSet< qlonglong > & **imagelds** () const
- bool **isAlbumUpdate** () const
- bool **isDuplicatesJob** () const
- double **maxThreshold** () const
- double **minThreshold** () const
- const QSet< qlonglong > & **refmagelds** () const
- [Haarface::ReflmageSelMethod](#) **reflimageSelectionMethod** () const
- const QList< int > & **searchIds** () const
- int **searchResultRestriction** () const
- void **setMaxThreshold** (double t)
- void **setMinThreshold** (double t)
- void **setSearchResultRestriction** (int type)

### Public Member Functions inherited from [Digikam::DBJobInfo](#)

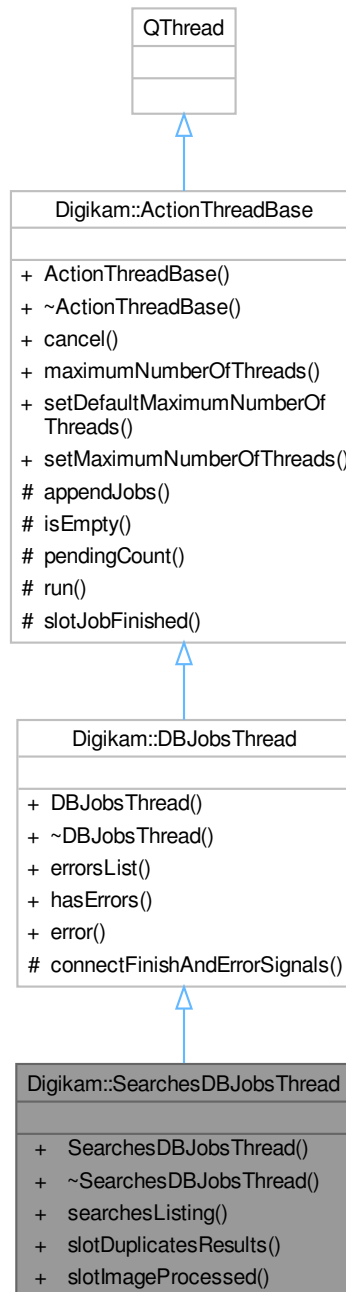
- bool **isFoldersJob** () const
- bool **isListAvailableImagesOnly** () const
- bool **isRecursive** () const
- void **setFoldersJob** ()
- void **setListAvailableImagesOnly** ()
- void **setRecursive** ()

### Public Attributes

- bool **m\_albumUpdate** = false
- bool **m\_duplicates** = false
- QSet< qlonglong > **m\_imagelds**
- double **m\_maxThreshold** = 1.0
- double **m\_minThreshold** = 0.4
- QSet< qlonglong > **m\_refmagelds**  
*Image ids of the reference images if duplicates are available.*
- [Haarface::ReflmageSelMethod](#) **m\_reflimageSelectionMethod** = [Haarface::ReflmageSelMethod::OlderOrLarger](#)
- QList< int > **m\_searchIds**
- int **m\_searchResultRestriction** = 0

## 9.1133 Digikam::SearchesDBJobsThread Class Reference

Inheritance diagram for Digikam::SearchesDBJobsThread:



### Public Slots

- void **slotDuplicatesResults** (const Haarlfacce::DuplicatesResultsMap &)
- void **slotImageProcessed** (const [ItemInfo](#) &, const QImage &, int dup)

## Public Slots inherited from [Digikam::DBJobsThread](#)

- void **error** (const QString &errString)  
*Appends the error string to m\_errorsList.*

## Signals

- void **signalProgress** (int percentage, const [ItemInfo](#) &inf, const QImage &img, int dup)

## Signals inherited from [Digikam::DBJobsThread](#)

- void **data** (const QList< [ItemListerRecord](#) > &records)
- void **finished** ()

## Public Member Functions

- **SearchesDBJobsThread** (QObject \*const parent)
- void **searchesListing** (const [SearchesDBJobInfo](#) &info)  
*Starts searches listing and scanning.*

## Public Member Functions inherited from [Digikam::DBJobsThread](#)

- **DBJobsThread** (QObject \*const parent)
- QList< QString > & **errorsList** ()  
*A method to get all errors reported from jobs.*
- bool **hasErrors** ()  
*hasErrors: a method to check for jobs errors*

## Public Member Functions inherited from [Digikam::ActionThreadBase](#)

- **ActionThreadBase** (QObject \*const parent=nullptr)
- void **cancel** (bool isCancel=true)  
*Cancel processing of current jobs under progress.*
- int **maximumNumberOfThreads** () const  
*Return the maximum number of threads used to parallelize collection of job processing.*
- void **setDefaultMaximumNumberOfThreads** ()  
*Reset maximum number of threads used to parallelize collection of job processing to max core detected on computer.*
- void **setMaximumNumberOfThreads** (int n)  
*Adjust maximum number of threads used to parallelize collection of job processing.*

## Additional Inherited Members

## Protected Slots inherited from [Digikam::ActionThreadBase](#)

- void **slotJobFinished** ()

## Protected Member Functions inherited from [Digikam::DBJobsThread](#)

- void [connectFinishAndErrorSignals](#) ([DBJob](#) \*const j)  
*Connects the signals of job to the signals of the thread.*

## Protected Member Functions inherited from [Digikam::ActionThreadBase](#)

- void [appendJobs](#) (const [ActionJobCollection](#) &jobs)  
*Append a collection of jobs to process into QThreadPool.*
- bool [isEmpty](#) () const  
*Return true if list of pending jobs to process is empty.*
- int [pendingCount](#) () const  
*Return the number of pending jobs to process.*
- void [run](#) () override  
*Main thread loop used to process jobs in todo list.*

### 9.1133.1 Member Function Documentation

#### 9.1133.1.1 [searchesListing\(\)](#)

```
void Digikam::SearchesDBJobsThread::searchesListing (  
    const SearchesDBJobInfo & info )
```

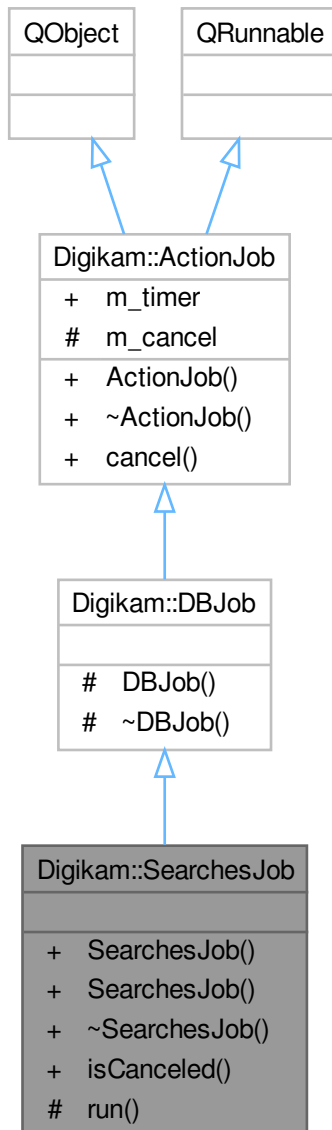
##### Parameters

<i>info</i>	represents the searches job info
-------------	----------------------------------



## 9.1134 Digikam::SearchesJob Class Reference

Inheritance diagram for Digikam::SearchesJob:



### Signals

- void **signalDuplicatesResults** (const Haarlfacce::DuplicatesResultsMap &)
- void **signalImageProcessed** (const [ItemInfo](#) &, const QImage &, int dup)

### Signals inherited from [Digikam::DBJob](#)

- void **data** (const QList< [ItemListerRecord](#) > &records)
- void **error** (const QString &err)

## Signals inherited from [Digikam::ActionJob](#)

- void **signalDone** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job is done.*
- void **signalProgress** (int)  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job progress.*
- void **signalStarted** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager that job is started.*

## Public Member Functions

- **SearchesJob** (const [SearchesDBJobInfo](#) &jobInfo)
- **SearchesJob** (const [SearchesDBJobInfo](#) &jobInfo, const QSet< qlonglong >::const\_iterator &begin, const QSet< qlonglong >::const\_iterator &end, [HaarIface](#) \*iface)
- bool **isCanceled** () const

## Public Member Functions inherited from [Digikam::ActionJob](#)

- **ActionJob** (QObject \*const parent=nullptr)  
*Constructor which delegate deletion of QRunnable instance to [ActionThreadBase](#), not QThreadPool.*
- **~ActionJob** () override  
*Re-implement destructor in you implementation.*

## Protected Member Functions

- void **run** () override

## Additional Inherited Members

## Public Slots inherited from [Digikam::ActionJob](#)

- void **cancel** ()  
*Call this method to cancel job.*

## Public Attributes inherited from [Digikam::ActionJob](#)

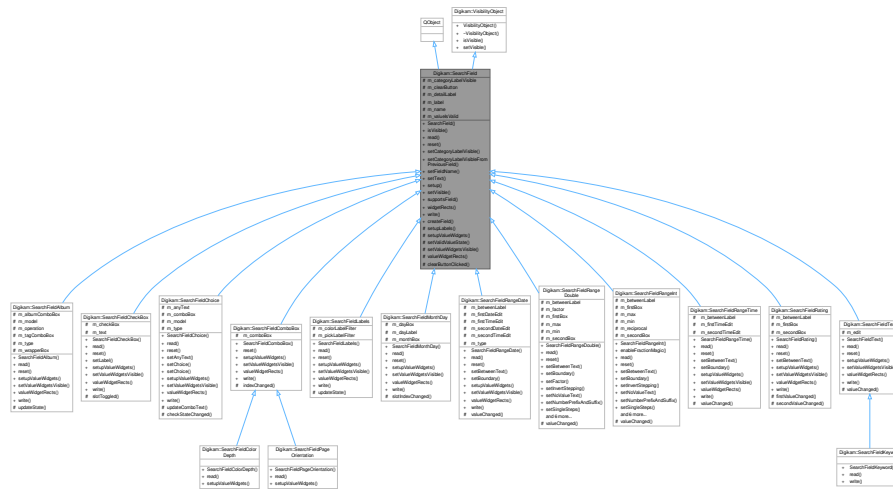
- QElapsedTimer **m\_timer**  
*Timer to determine the running time of the job.*

## Protected Attributes inherited from [Digikam::ActionJob](#)

- bool **m\_cancel** = false  
*You can use this boolean in your implementation to know if job must be canceled.*

## 9.1135 Digikam::SearchField Class Reference

Inheritance diagram for Digikam::SearchField:



### Public Types

- enum **WidgetRectType** { **LabelAndValueWidgetRects** , **ValueWidgetRectsOnly** }

### Signals

- void **signalVisibilityChanged** ()

### Public Member Functions

- **SearchField** (QObject \*const parent)
- bool **isVisible** () override
- virtual void **read** ([SearchXmlCachingReader](#) &reader)=0
- virtual void **reset** ()=0
- void **setCategoryLabelVisible** (bool visible)
- void **setCategoryLabelVisibleFromPreviousField** ([SearchField](#) \*const previousField)
- void **setFieldName** (const QString &fieldName)
- virtual void **setText** (const QString &label, const QString &detailLabel)
- void **setup** (QGridLayout \*const layout, int row=-1)
- void **setVisible** (bool visible) override
- virtual bool **supportsField** (const QString &fieldName)
- QList< QRect > **widgetRects** (WidgetRectType=ValueWidgetRectsOnly) const
- virtual void **write** ([SearchXmlWriter](#) &writer)=0

### Static Public Member Functions

- static [SearchField](#) \* **createField** (const QString &fieldName, [SearchFieldGroup](#) \*const parent)

## Protected Slots

- void **clearButtonClicked** ()

## Protected Member Functions

- virtual void **setupLabels** (QGridLayout \*layout, int line)
- virtual void **setupValueWidgets** (QGridLayout \*layout, int row, int column)=0
- void **setValidValueState** (bool valueIsValid)
- virtual void **setValueWidgetsVisible** (bool visible)=0
- virtual QList< QRect > **valueWidgetRects** () const =0

## Protected Attributes

- bool **m\_categoryLabelVisible** = true
- [AnimatedClearButton](#) \* **m\_clearButton** = nullptr
- QLabel \* **m\_detailLabel** = nullptr
- QLabel \* **m\_label** = nullptr
- QString **m\_name**
- bool **m\_valueIsValid** = false

## 9.1135.1 Member Function Documentation

### 9.1135.1.1 createField()

```
SearchField * Digikam::SearchField::createField (
    const QString & fieldName,
    SearchFieldGroup *const parent ) [static]
```

### 9.1135.1.2 isVisible()

```
bool Digikam::SearchField::isVisible ( ) [override], [virtual]
```

Implements [Digikam::VisibilityObject](#).

### 9.1135.1.3 setVisible()

```
void Digikam::SearchField::setVisible (
    bool visible ) [override], [virtual]
```

Implements [Digikam::VisibilityObject](#).

### 9.1135.1.4 write()

```
virtual void Digikam::SearchField::write (
    SearchXmlWriter & writer ) [pure virtual]
```

Implemented in [Digikam::SearchFieldRangeInt](#).

## 9.1136 Digikam::SearchFieldAlbum Class Reference

Inheritance diagram for Digikam::SearchFieldAlbum:



### Public Types

- enum **Operation** { **All** , **OneOf** , **InTree** }
- enum **Type** { **TypeAlbum** , **TypeTag** }

## Public Types inherited from [Digikam::SearchField](#)

- enum **WidgetRectType** { [LabelAndValueWidgetRects](#) , [ValueWidgetRectsOnly](#) }

## Public Member Functions

- **SearchFieldAlbum** (QObject \*const parent, Type type)
- void [read](#) ([SearchXmlCachingReader](#) &reader) override
- void [reset](#) () override
- void [setupValueWidgets](#) (QGridLayout \*layout, int row, int column) override
- void [setValueWidgetsVisible](#) (bool visible) override
- QList< QRect > [valueWidgetRects](#) () const override
- void [write](#) ([SearchXmlWriter](#) &writer) override

## Public Member Functions inherited from [Digikam::SearchField](#)

- **SearchField** (QObject \*const parent)
- bool [isVisible](#) () override
- void [setCategoryLabelVisible](#) (bool visible)
- void [setCategoryLabelVisibleFromPreviousField](#) ([SearchField](#) \*const previousField)
- void [setFieldName](#) (const QString &fieldName)
- virtual void [setText](#) (const QString &label, const QString &detailLabel)
- void [setup](#) (QGridLayout \*const layout, int row=-1)
- void [setVisible](#) (bool visible) override
- virtual bool [supportsField](#) (const QString &fieldName)
- QList< QRect > [widgetRects](#) (WidgetRectType=ValueWidgetRectsOnly) const

## Protected Slots

- void [updateState](#) ()

## Protected Slots inherited from [Digikam::SearchField](#)

- void [clearButtonClicked](#) ()

## Protected Attributes

- [AlbumTreeViewSelectComboBox](#) \* [m\\_albumComboBox](#) = nullptr
- [AbstractCheckableAlbumModel](#) \* [m\\_model](#) = nullptr
- [SqueezedComboBox](#) \* [m\\_operation](#) = nullptr
- [TagTreeViewSelectComboBox](#) \* [m\\_tagComboBox](#) = nullptr
- Type [m\\_type](#) = TypeAlbum
- QWidget \* [m\\_wrapperBox](#) = nullptr

## Protected Attributes inherited from [Digikam::SearchField](#)

- bool [m\\_categoryLabelVisible](#) = true
- [AnimatedClearButton](#) \* [m\\_clearButton](#) = nullptr
- QLabel \* [m\\_detailLabel](#) = nullptr
- QLabel \* [m\\_label](#) = nullptr
- QString [m\\_name](#)
- bool [m\\_valuesValid](#) = false

## Additional Inherited Members

## Signals inherited from [Digikam::SearchField](#)

- void `signalVisibilityChanged` ()

## Static Public Member Functions inherited from [Digikam::SearchField](#)

- static `SearchField * createField` (const QString &fieldName, `SearchFieldGroup *const` parent)

## Protected Member Functions inherited from [Digikam::SearchField](#)

- virtual void `setupLabels` (QGridLayout \*layout, int line)
- void `setValidValueState` (bool valueIsValid)

## 9.1136.1 Member Function Documentation

### 9.1136.1.1 `read()`

```
void Digikam::SearchFieldAlbum::read (
    SearchXmlCachingReader & reader ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

### 9.1136.1.2 `reset()`

```
void Digikam::SearchFieldAlbum::reset ( ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

### 9.1136.1.3 `setupValueWidgets()`

```
void Digikam::SearchFieldAlbum::setupValueWidgets (
    QGridLayout * layout,
    int row,
    int column ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

### 9.1136.1.4 `setValueWidgetsVisible()`

```
void Digikam::SearchFieldAlbum::setValueWidgetsVisible (
    bool visible ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

**9.1136.1.5 valueWidgetRects()**

```
QList< QRect > Digikam::SearchFieldAlbum::valueWidgetRects ( ) const [override], [virtual]
```

Implements [Digikam::SearchField](#).

**9.1136.1.6 write()**

```
void Digikam::SearchFieldAlbum::write (
    SearchXmlWriter & writer ) [override], [virtual]
```

Implements [Digikam::SearchField](#).



## 9.1137 Digikam::SearchFieldCheckBox Class Reference

Inheritance diagram for Digikam::SearchFieldCheckBox:



### Public Member Functions

- **SearchFieldCheckBox** (QObject \*const parent)
- void **read** (SearchXmlCachingReader &reader) override

- void [reset](#) () override
- void **setLabel** (const QString &text)
- void [setupValueWidgets](#) (QGridLayout \*layout, int row, int column) override
- void [setValueWidgetsVisible](#) (bool visible) override
- QList< QRect > [valueWidgetRects](#) () const override
- void [write](#) ([SearchXmlWriter](#) &writer) override

## Public Member Functions inherited from [Digikam::SearchField](#)

- **SearchField** (QObject \*const parent)
- bool [isVisible](#) () override
- void **setCategoryLabelVisible** (bool visible)
- void **setCategoryLabelVisibleFromPreviousField** ([SearchField](#) \*const previousField)
- void **setFieldName** (const QString &fieldName)
- virtual void **setText** (const QString &label, const QString &detailLabel)
- void **setup** (QGridLayout \*const layout, int row=-1)
- void [setVisible](#) (bool visible) override
- virtual bool **supportsField** (const QString &fieldName)
- QList< QRect > **widgetRects** (WidgetRectType=ValueWidgetRectsOnly) const

## Protected Slots

- void **slotToggled** (bool checked)

## Protected Slots inherited from [Digikam::SearchField](#)

- void **clearButtonClicked** ()

## Protected Attributes

- QCheckBox \* **m\_checkBox** = nullptr
- QString **m\_text**

## Protected Attributes inherited from [Digikam::SearchField](#)

- bool **m\_categoryLabelVisible** = true
- [AnimatedClearButton](#) \* **m\_clearButton** = nullptr
- QLabel \* **m\_detailLabel** = nullptr
- QLabel \* **m\_label** = nullptr
- QString **m\_name**
- bool **m\_valuesValid** = false

## Additional Inherited Members

## Public Types inherited from [Digikam::SearchField](#)

- enum **WidgetRectType** { **LabelAndValueWidgetRects** , **ValueWidgetRectsOnly** }

## Signals inherited from [Digikam::SearchField](#)

- void `signalVisibilityChanged` ()

## Static Public Member Functions inherited from [Digikam::SearchField](#)

- static `SearchField * createField` (const QString &fieldName, `SearchFieldGroup *const` parent)

## Protected Member Functions inherited from [Digikam::SearchField](#)

- virtual void `setupLabels` (QGridLayout \*layout, int line)
- void `setValidValueState` (bool valueIsValid)

### 9.1137.1 Member Function Documentation

#### 9.1137.1.1 `read()`

```
void Digikam::SearchFieldCheckBox::read (
    SearchXmlCachingReader & reader ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

#### 9.1137.1.2 `reset()`

```
void Digikam::SearchFieldCheckBox::reset ( ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

#### 9.1137.1.3 `setValueWidgets()`

```
void Digikam::SearchFieldCheckBox::setValueWidgets (
    QGridLayout * layout,
    int row,
    int column ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

#### 9.1137.1.4 `setValueWidgetsVisible()`

```
void Digikam::SearchFieldCheckBox::setValueWidgetsVisible (
    bool visible ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

**9.1137.1.5 valueWidgetRects()**

```
QList< QRect > Digikam::SearchFieldCheckBox::valueWidgetRects ( ) const [override], [virtual]
```

Implements [Digikam::SearchField](#).

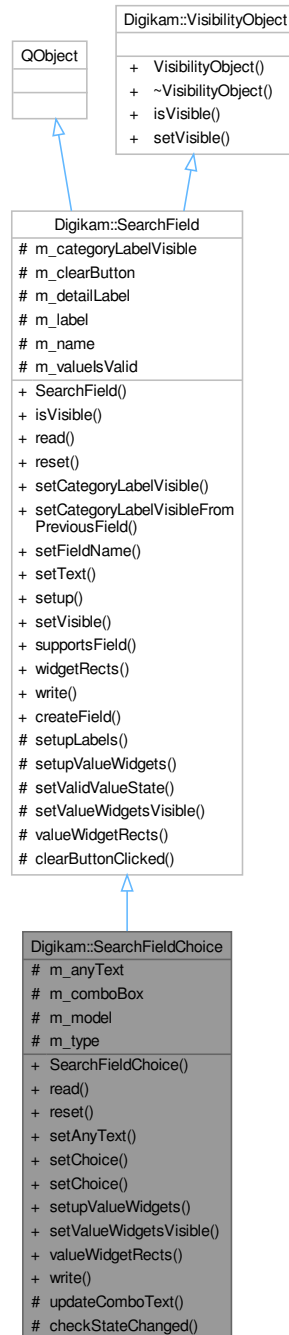
**9.1137.1.6 write()**

```
void Digikam::SearchFieldCheckBox::write (
    SearchXmlWriter & writer ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

## 9.1138 Digikam::SearchFieldChoice Class Reference

Inheritance diagram for Digikam::SearchFieldChoice:



### Public Member Functions

- **SearchFieldChoice** (QObject \*const parent)
- void `read` ([SearchXmlCachingReader](#) &reader) override

- void [reset](#) () override
- void **setAnyText** (const QString &anyText)
- void **setChoice** (const QMap< int, QString > &map)
- void **setChoice** (const QStringList &choice)
- void [setUpValueWidgets](#) (QGridLayout \*layout, int row, int column) override
- void [setValueWidgetsVisible](#) (bool visible) override
- QList< QRect > [valueWidgetRects](#) () const override
- void [write](#) ([SearchXmlWriter](#) &writer) override

### Public Member Functions inherited from [Digikam::SearchField](#)

- **SearchField** (QObject \*const parent)
- bool [isVisible](#) () override
- void **setCategoryLabelVisible** (bool visible)
- void **setCategoryLabelVisibleFromPreviousField** ([SearchField](#) \*const previousField)
- void **setFieldName** (const QString &fieldName)
- virtual void **setText** (const QString &label, const QString &detailLabel)
- void **setup** (QGridLayout \*const layout, int row=-1)
- void [setVisible](#) (bool visible) override
- virtual bool **supportsField** (const QString &fieldName)
- QList< QRect > **widgetRects** (WidgetRectType=ValueWidgetRectsOnly) const

### Protected Slots

- void **checkStateChanged** ()

### Protected Slots inherited from [Digikam::SearchField](#)

- void **clearButtonClicked** ()

### Protected Member Functions

- void **updateComboText** ()

### Protected Member Functions inherited from [Digikam::SearchField](#)

- virtual void **setupLabels** (QGridLayout \*layout, int line)
- void **setValidValueState** (bool valuesValid)

### Protected Attributes

- QString **m\_anyText**
- [ChoiceSearchComboBox](#) \* **m\_comboBox** = nullptr
- [ChoiceSearchModel](#) \* **m\_model** = nullptr
- QMetaType::Type **m\_type** = QMetaType::UnknownType

## Protected Attributes inherited from [Digikam::SearchField](#)

- bool `m_categoryLabelVisible` = true
- [AnimatedClearButton](#) \* `m_clearButton` = nullptr
- [QLabel](#) \* `m_detailLabel` = nullptr
- [QLabel](#) \* `m_label` = nullptr
- [QString](#) `m_name`
- bool `m_valuelsValid` = false

## Additional Inherited Members

## Public Types inherited from [Digikam::SearchField](#)

- enum `WidgetRectType` { `LabelAndValueWidgetRects` , `ValueWidgetRectsOnly` }

## Signals inherited from [Digikam::SearchField](#)

- void `signalVisibilityChanged` ()

## Static Public Member Functions inherited from [Digikam::SearchField](#)

- static [SearchField](#) \* `createField` (const [QString](#) &fieldName, [SearchFieldGroup](#) \*const parent)

## 9.1138.1 Member Function Documentation

### 9.1138.1.1 `read()`

```
void Digikam::SearchFieldChoice::read (
    SearchXmlCachingReader & reader ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

### 9.1138.1.2 `reset()`

```
void Digikam::SearchFieldChoice::reset ( ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

### 9.1138.1.3 `setupValueWidgets()`

```
void Digikam::SearchFieldChoice::setupValueWidgets (
    QGridLayout * layout,
    int row,
    int column ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

**9.1138.1.4 setValueWidgetsVisible()**

```
void Digikam::SearchFieldChoice::setValueWidgetsVisible (
    bool visible ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

**9.1138.1.5 valueWidgetRects()**

```
QList< QRect > Digikam::SearchFieldChoice::valueWidgetRects ( ) const [override], [virtual]
```

Implements [Digikam::SearchField](#).

**9.1138.1.6 write()**

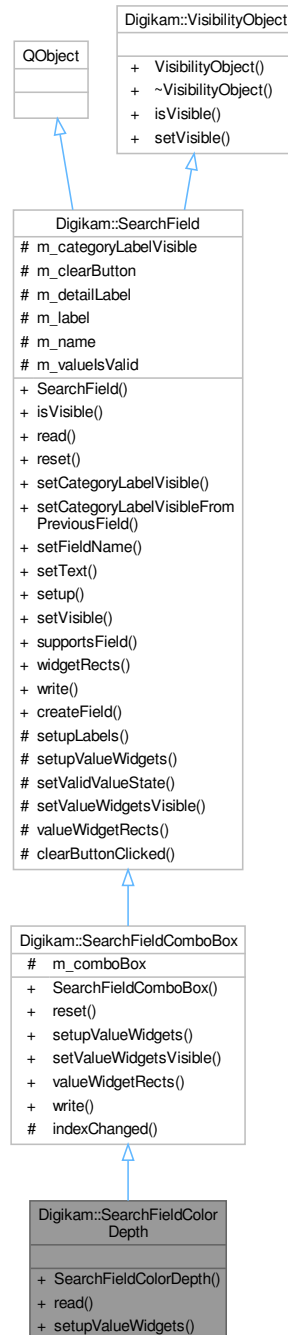
```
void Digikam::SearchFieldChoice::write (
    SearchXmlWriter & writer ) [override], [virtual]
```

Implements [Digikam::SearchField](#).



## 9.1139 Digikam::SearchFieldColorDepth Class Reference

Inheritance diagram for Digikam::SearchFieldColorDepth:



### Public Member Functions

- **SearchFieldColorDepth** (QObject \*const parent)
- void **read** (SearchXmlCachingReader &reader) override
- void **setValueWidgets** (QGridLayout \*layout, int row, int column) override

## Public Member Functions inherited from [Digikam::SearchFieldComboBox](#)

- **SearchFieldComboBox** (QObject \*const parent)
- void **reset** () override
- void **setupValueWidgets** (QGridLayout \*layout, int row, int column) override
- void **setValueWidgetsVisible** (bool visible) override
- QList< QRect > **valueWidgetRects** () const override
- void **write** ([SearchXmlWriter](#) &writer) override

## Public Member Functions inherited from [Digikam::SearchField](#)

- **SearchField** (QObject \*const parent)
- bool **isVisible** () override
- void **setCategoryLabelVisible** (bool visible)
- void **setCategoryLabelVisibleFromPreviousField** ([SearchField](#) \*const previousField)
- void **setFieldName** (const QString &fieldName)
- virtual void **setText** (const QString &label, const QString &detailLabel)
- void **setup** (QGridLayout \*const layout, int row=-1)
- void **setVisible** (bool visible) override
- virtual bool **supportsField** (const QString &fieldName)
- QList< QRect > **widgetRects** (WidgetRectType=ValueWidgetRectsOnly) const

## Additional Inherited Members

## Public Types inherited from [Digikam::SearchField](#)

- enum **WidgetRectType** { [LabelAndValueWidgetRects](#) , [ValueWidgetRectsOnly](#) }

## Signals inherited from [Digikam::SearchField](#)

- void **signalVisibilityChanged** ()

## Static Public Member Functions inherited from [Digikam::SearchField](#)

- static [SearchField](#) \* **createField** (const QString &fieldName, [SearchFieldGroup](#) \*const parent)

## Protected Slots inherited from [Digikam::SearchFieldComboBox](#)

- void **indexChanged** (int)

## Protected Slots inherited from [Digikam::SearchField](#)

- void **clearButtonClicked** ()

## Protected Member Functions inherited from [Digikam::SearchField](#)

- virtual void **setupLabels** (QGridLayout \*layout, int line)
- void **setValidValueState** (bool valueIsValid)

## Protected Attributes inherited from [Digikam::SearchFieldComboBox](#)

- `QComboBox * m_comboBox = nullptr`

## Protected Attributes inherited from [Digikam::SearchField](#)

- `bool m_categoryLabelVisible = true`
- [AnimatedClearButton](#) \* `m_clearButton = nullptr`
- `QLabel * m_detailLabel = nullptr`
- `QLabel * m_label = nullptr`
- `QString m_name`
- `bool m_valuesValid = false`

## 9.1139.1 Member Function Documentation

### 9.1139.1.1 `read()`

```
void Digikam::SearchFieldColorDepth::read (
    SearchXmlCachingReader & reader ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

### 9.1139.1.2 `setupValueWidgets()`

```
void Digikam::SearchFieldColorDepth::setupValueWidgets (
    QGridLayout * layout,
    int row,
    int column ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

## 9.1140 Digikam::SearchFieldComboBox Class Reference

Inheritance diagram for Digikam::SearchFieldComboBox:



### Public Member Functions

- **SearchFieldComboBox** (QObject \*const parent)
- void [reset](#) () override

- void [setupValueWidgets](#) (QGridLayout \*layout, int row, int column) override
- void [setValueWidgetsVisible](#) (bool visible) override
- QList< QRect > [valueWidgetRects](#) () const override
- void [write](#) (SearchXmlWriter &writer) override

## Public Member Functions inherited from Digikam::SearchField

- **SearchField** (QObject \*const parent)
- bool [isVisible](#) () override
- virtual void [read](#) (SearchXmlCachingReader &reader)=0
- void [setCategoryLabelVisible](#) (bool visible)
- void [setCategoryLabelVisibleFromPreviousField](#) (SearchField \*const previousField)
- void [setFieldName](#) (const QString &fieldName)
- virtual void [setText](#) (const QString &label, const QString &detailLabel)
- void [setup](#) (QGridLayout \*const layout, int row=-1)
- void [setVisible](#) (bool visible) override
- virtual bool [supportsField](#) (const QString &fieldName)
- QList< QRect > [widgetRects](#) (WidgetRectType=ValueWidgetRectsOnly) const

## Protected Slots

- void [indexChanged](#) (int)

## Protected Slots inherited from Digikam::SearchField

- void [clearButtonClicked](#) ()

## Protected Attributes

- QComboBox \* [m\\_comboBox](#) = nullptr

## Protected Attributes inherited from Digikam::SearchField

- bool [m\\_categoryLabelVisible](#) = true
- [AnimatedClearButton](#) \* [m\\_clearButton](#) = nullptr
- QLabel \* [m\\_detailLabel](#) = nullptr
- QLabel \* [m\\_label](#) = nullptr
- QString [m\\_name](#)
- bool [m\\_valuesValid](#) = false

## Additional Inherited Members

## Public Types inherited from Digikam::SearchField

- enum [WidgetRectType](#) { [LabelAndValueWidgetRects](#) , [ValueWidgetRectsOnly](#) }

## Signals inherited from [Digikam::SearchField](#)

- void `signalVisibilityChanged` ()

## Static Public Member Functions inherited from [Digikam::SearchField](#)

- static `SearchField * createField` (const QString &fieldName, `SearchFieldGroup *const` parent)

## Protected Member Functions inherited from [Digikam::SearchField](#)

- virtual void `setupLabels` (QGridLayout \*layout, int line)
- void `setValidValueState` (bool valueIsValid)

### 9.1140.1 Member Function Documentation

#### 9.1140.1.1 `reset()`

```
void Digikam::SearchFieldComboBox::reset ( ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

#### 9.1140.1.2 `setupValueWidgets()`

```
void Digikam::SearchFieldComboBox::setupValueWidgets (
    QGridLayout * layout,
    int row,
    int column ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

#### 9.1140.1.3 `setValueWidgetsVisible()`

```
void Digikam::SearchFieldComboBox::setValueWidgetsVisible (
    bool visible ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

#### 9.1140.1.4 `valueWidgetRects()`

```
QList< QRect > Digikam::SearchFieldComboBox::valueWidgetRects ( ) const [override], [virtual]
```

Implements [Digikam::SearchField](#).

### 9.1140.1.5 write()

```
void Digikam::SearchFieldComboBox::write (
    SearchXmlWriter & writer ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

## 9.1141 Digikam::SearchFieldGroup Class Reference

Inheritance diagram for Digikam::SearchFieldGroup:



### Public Slots

- void **setFieldsVisible** (bool visible)

**Public Member Functions**

- **SearchFieldGroup** ([SearchGroup](#) \*const parent)
- void **addField** ([SearchField](#) \*const field)
- QList< QRect > **areaOfMarkedFields** () const
- void **clearMarkedFields** ()
- [SearchField](#) \* **fieldForName** (const QString &fieldName) const
- void **markField** ([SearchField](#) \*const field)
- void **reset** ()
- void **setLabel** ([SearchFieldGroupLabel](#) \*const label)
- void **write** ([SearchXmlWriter](#) &writer)

**Protected Slots**

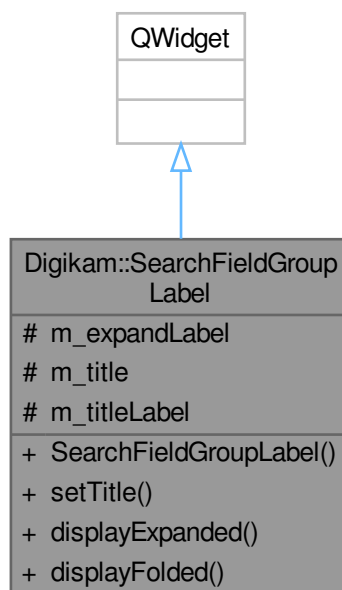
- void **slotLabelClicked** ()

**Protected Attributes**

- [VisibilityController](#) \* **m\_controller** = nullptr
- QList< [SearchField](#) \* > **m\_fields**
- [SearchFieldGroupLabel](#) \* **m\_label** = nullptr
- QGridLayout \* **m\_layout** = nullptr
- QSet< [SearchField](#) \* > **m\_markedFields**

**9.1142 Digikam::SearchFieldGroupLabel Class Reference**

Inheritance diagram for Digikam::SearchFieldGroupLabel:





### Public Slots

- void **displayExpanded** ()
- void **displayFolded** ()

### Signals

- void **clicked** ()

### Public Member Functions

- **SearchFieldGroupLabel** (QWidget \*const parent)
- void **setTitle** (const QString &title)

### Protected Attributes

- QLabel \* **m\_expandLabel** = nullptr
- QString **m\_title**
- [DClickLabel](#) \* **m\_titleLabel** = nullptr

## 9.1143 Digikam::SearchFieldKeyword Class Reference

Inheritance diagram for Digikam::SearchFieldKeyword:



### Public Member Functions

- **SearchFieldKeyword** (QObject \*const parent)
- void **read** (SearchXmlCachingReader &reader) override
- void **write** (SearchXmlWriter &writer) override

## Public Member Functions inherited from Digikam::SearchFieldText

- **SearchFieldText** (QObject \*const parent)
- void **read** ([SearchXmlCachingReader](#) &reader) override
- void **reset** () override
- void **setupValueWidgets** (QGridLayout \*layout, int row, int column) override
- void **setValueWidgetsVisible** (bool visible) override
- QList< QRect > **valueWidgetRects** () const override
- void **write** ([SearchXmlWriter](#) &writer) override

## Public Member Functions inherited from Digikam::SearchField

- **SearchField** (QObject \*const parent)
- bool **isVisible** () override
- void **setCategoryLabelVisible** (bool visible)
- void **setCategoryLabelVisibleFromPreviousField** ([SearchField](#) \*const previousField)
- void **setFieldName** (const QString &fieldName)
- virtual void **setText** (const QString &label, const QString &detailLabel)
- void **setup** (QGridLayout \*const layout, int row=-1)
- void **setVisible** (bool visible) override
- virtual bool **supportsField** (const QString &fieldName)
- QList< QRect > **widgetRects** (WidgetRectType=ValueWidgetRectsOnly) const

## Additional Inherited Members

## Public Types inherited from Digikam::SearchField

- enum **WidgetRectType** { [LabelAndValueWidgetRects](#) , [ValueWidgetRectsOnly](#) }

## Signals inherited from Digikam::SearchField

- void **signalVisibilityChanged** ()

## Static Public Member Functions inherited from Digikam::SearchField

- static [SearchField](#) \* **createField** (const QString &fieldName, [SearchFieldGroup](#) \*const parent)

## Protected Slots inherited from Digikam::SearchFieldText

- void **valueChanged** (const QString &text)

## Protected Slots inherited from Digikam::SearchField

- void **clearButtonClicked** ()

## Protected Member Functions inherited from [Digikam::SearchField](#)

- virtual void **setupLabels** (QGridLayout \*layout, int line)
- void **setValidValueState** (bool valueIsValid)

## Protected Attributes inherited from [Digikam::SearchFieldText](#)

- QLineEdit \* **m\_edit** = nullptr

## Protected Attributes inherited from [Digikam::SearchField](#)

- bool **m\_categoryLabelVisible** = true
- [AnimatedClearButton](#) \* **m\_clearButton** = nullptr
- QLabel \* **m\_detailLabel** = nullptr
- QLabel \* **m\_label** = nullptr
- QString **m\_name**
- bool **m\_valueIsValid** = false

## 9.1143.1 Member Function Documentation

### 9.1143.1.1 read()

```
void Digikam::SearchFieldKeyword::read (
    SearchXmlCachingReader & reader ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

### 9.1143.1.2 write()

```
void Digikam::SearchFieldKeyword::write (
    SearchXmlWriter & writer ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

## 9.1144 Digikam::SearchFieldLabels Class Reference

Inheritance diagram for Digikam::SearchFieldLabels:



### Public Member Functions

- **SearchFieldLabels** (QObject \*const parent)
- void **read** (SearchXmlCachingReader &reader) override

- void [reset](#) () override
- void [setValueWidgets](#) (QGridLayout \*layout, int row, int column) override
- void [setValueWidgetsVisible](#) (bool visible) override
- QList< QRect > [valueWidgetRects](#) () const override
- void [write](#) (SearchXmlWriter &writer) override

## Public Member Functions inherited from [Digikam::SearchField](#)

- **SearchField** (QObject \*const parent)
- bool [isVisible](#) () override
- void [setCategoryLabelVisible](#) (bool visible)
- void [setCategoryLabelVisibleFromPreviousField](#) ([SearchField](#) \*const previousField)
- void [setFieldName](#) (const QString &fieldName)
- virtual void [setText](#) (const QString &label, const QString &detailLabel)
- void [setup](#) (QGridLayout \*const layout, int row=-1)
- void [setVisible](#) (bool visible) override
- virtual bool [supportsField](#) (const QString &fieldName)
- QList< QRect > [widgetRects](#) (WidgetRectType=ValueWidgetRectsOnly) const

## Protected Slots

- void [updateState](#) ()

## Protected Slots inherited from [Digikam::SearchField](#)

- void [clearButtonClicked](#) ()

## Protected Attributes

- [ColorLabelFilter](#) \* [m\\_colorLabelFilter](#) = nullptr
- [PickLabelFilter](#) \* [m\\_pickLabelFilter](#) = nullptr

## Protected Attributes inherited from [Digikam::SearchField](#)

- bool [m\\_categoryLabelVisible](#) = true
- [AnimatedClearButton](#) \* [m\\_clearButton](#) = nullptr
- QLabel \* [m\\_detailLabel](#) = nullptr
- QLabel \* [m\\_label](#) = nullptr
- QString [m\\_name](#)
- bool [m\\_valueIsValid](#) = false

## Additional Inherited Members

## Public Types inherited from [Digikam::SearchField](#)

- enum [WidgetRectType](#) { [LabelAndValueWidgetRects](#) , [ValueWidgetRectsOnly](#) }

## Signals inherited from [Digikam::SearchField](#)

- void `signalVisibilityChanged` ()

## Static Public Member Functions inherited from [Digikam::SearchField](#)

- static `SearchField * createField` (const QString &fieldName, `SearchFieldGroup *const parent`)

## Protected Member Functions inherited from [Digikam::SearchField](#)

- virtual void `setupLabels` (QGridLayout \*layout, int line)
- void `setValidValueState` (bool valueIsValid)

### 9.1144.1 Member Function Documentation

#### 9.1144.1.1 `read()`

```
void Digikam::SearchFieldLabels::read (
    SearchXmlCachingReader & reader ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

#### 9.1144.1.2 `reset()`

```
void Digikam::SearchFieldLabels::reset ( ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

#### 9.1144.1.3 `setValueWidgets()`

```
void Digikam::SearchFieldLabels::setValueWidgets (
    QGridLayout * layout,
    int row,
    int column ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

#### 9.1144.1.4 `setValueWidgetsVisible()`

```
void Digikam::SearchFieldLabels::setValueWidgetsVisible (
    bool visible ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

**9.1144.1.5 valueWidgetRects()**

```
QList< QRect > Digikam::SearchFieldLabels::valueWidgetRects ( ) const [override], [virtual]
```

Implements [Digikam::SearchField](#).

**9.1144.1.6 write()**

```
void Digikam::SearchFieldLabels::write (
    SearchXmlWriter & writer ) [override], [virtual]
```

Implements [Digikam::SearchField](#).



## 9.1145 Digikam::SearchFieldMonthDay Class Reference

Inheritance diagram for Digikam::SearchFieldMonthDay:



### Public Member Functions

- **SearchFieldMonthDay** (QObject \*const parent)
- void **read** (SearchXmlCachingReader &reader) override

- void [reset](#) () override
- void [setValueWidgets](#) (QGridLayout \*layout, int row, int column) override
- void [setValueWidgetsVisible](#) (bool visible) override
- QList< QRect > [valueWidgetRects](#) () const override
- void [write](#) ([SearchXmlWriter](#) &writer) override

## Public Member Functions inherited from [Digikam::SearchField](#)

- [SearchField](#) (QObject \*const parent)
- bool [isVisible](#) () override
- void [setCategoryLabelVisible](#) (bool visible)
- void [setCategoryLabelVisibleFromPreviousField](#) ([SearchField](#) \*const previousField)
- void [setFieldName](#) (const QString &fieldName)
- virtual void [setText](#) (const QString &label, const QString &detailLabel)
- void [setup](#) (QGridLayout \*const layout, int row=-1)
- void [setVisible](#) (bool visible) override
- virtual bool [supportsField](#) (const QString &fieldName)
- QList< QRect > [widgetRects](#) (WidgetRectType=ValueWidgetRectsOnly) const

## Protected Slots

- void [slotIndexChanged](#) ()

## Protected Slots inherited from [Digikam::SearchField](#)

- void [clearButtonClicked](#) ()

## Protected Attributes

- QComboBox \* [m\\_dayBox](#) = nullptr
- QLabel \* [m\\_dayLabel](#) = nullptr
- QComboBox \* [m\\_monthBox](#) = nullptr

## Protected Attributes inherited from [Digikam::SearchField](#)

- bool [m\\_categoryLabelVisible](#) = true
- [AnimatedClearButton](#) \* [m\\_clearButton](#) = nullptr
- QLabel \* [m\\_detailLabel](#) = nullptr
- QLabel \* [m\\_label](#) = nullptr
- QString [m\\_name](#)
- bool [m\\_valuesValid](#) = false

## Additional Inherited Members

## Public Types inherited from [Digikam::SearchField](#)

- enum [WidgetRectType](#) { [LabelAndValueWidgetRects](#) , [ValueWidgetRectsOnly](#) }

## Signals inherited from [Digikam::SearchField](#)

- void `signalVisibilityChanged` ()

## Static Public Member Functions inherited from [Digikam::SearchField](#)

- static `SearchField * createField` (const QString &fieldName, `SearchFieldGroup *const` parent)

## Protected Member Functions inherited from [Digikam::SearchField](#)

- virtual void `setupLabels` (QGridLayout \*layout, int line)
- void `setValidValueState` (bool valueIsValid)

## 9.1145.1 Member Function Documentation

### 9.1145.1.1 `read()`

```
void Digikam::SearchFieldMonthDay::read (
    SearchXmlCachingReader & reader ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

### 9.1145.1.2 `reset()`

```
void Digikam::SearchFieldMonthDay::reset ( ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

### 9.1145.1.3 `setupValueWidgets()`

```
void Digikam::SearchFieldMonthDay::setupValueWidgets (
    QGridLayout * layout,
    int row,
    int column ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

### 9.1145.1.4 `setValueWidgetsVisible()`

```
void Digikam::SearchFieldMonthDay::setValueWidgetsVisible (
    bool visible ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

**9.1145.1.5 valueWidgetRects()**

```
QList< QRect > Digikam::SearchFieldMonthDay::valueWidgetRects ( ) const [override], [virtual]
```

Implements [Digikam::SearchField](#).

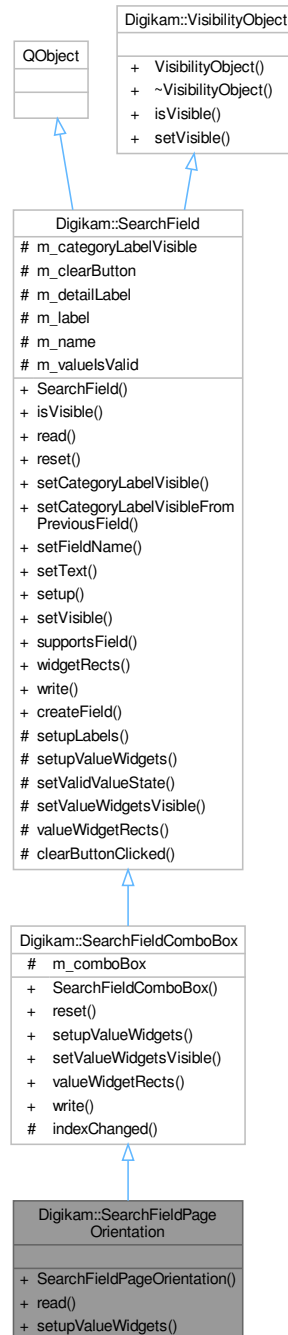
**9.1145.1.6 write()**

```
void Digikam::SearchFieldMonthDay::write (
    SearchXmlWriter & writer ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

## 9.1146 Digikam::SearchFieldPageOrientation Class Reference

Inheritance diagram for Digikam::SearchFieldPageOrientation:



### Public Member Functions

- **SearchFieldPageOrientation** (QObject \*const parent)
- void **read** (SearchXmlCachingReader &reader) override
- void **setValueWidgets** (QGridLayout \*layout, int row, int column) override

## Public Member Functions inherited from [Digikam::SearchFieldComboBox](#)

- **SearchFieldComboBox** (QObject \*const parent)
- void **reset** () override
- void **setValueWidgetsVisible** (bool visible) override
- QList< QRect > **valueWidgetRects** () const override
- void **write** ([SearchXmlWriter](#) &writer) override

## Public Member Functions inherited from [Digikam::SearchField](#)

- **SearchField** (QObject \*const parent)
- bool **isVisible** () override
- void **setCategoryLabelVisible** (bool visible)
- void **setCategoryLabelVisibleFromPreviousField** ([SearchField](#) \*const previousField)
- void **setFieldName** (const QString &fieldName)
- virtual void **setText** (const QString &label, const QString &detailLabel)
- void **setup** (QGridLayout \*const layout, int row=-1)
- void **setVisible** (bool visible) override
- virtual bool **supportsField** (const QString &fieldName)
- QList< QRect > **widgetRects** (WidgetRectType=ValueWidgetRectsOnly) const

## Additional Inherited Members

## Public Types inherited from [Digikam::SearchField](#)

- enum **WidgetRectType** { **LabelAndValueWidgetRects** , **ValueWidgetRectsOnly** }

## Signals inherited from [Digikam::SearchField](#)

- void **signalVisibilityChanged** ()

## Static Public Member Functions inherited from [Digikam::SearchField](#)

- static [SearchField](#) \* **createField** (const QString &fieldName, [SearchFieldGroup](#) \*const parent)

## Protected Slots inherited from [Digikam::SearchFieldComboBox](#)

- void **indexChanged** (int)

## Protected Slots inherited from [Digikam::SearchField](#)

- void **clearButtonClicked** ()

## Protected Member Functions inherited from [Digikam::SearchField](#)

- virtual void **setupLabels** (QGridLayout \*layout, int line)
- void **setValidValueState** (bool valueIsValid)

## Protected Attributes inherited from [Digikam::SearchFieldComboBox](#)

- `QComboBox * m_comboBox = nullptr`

## Protected Attributes inherited from [Digikam::SearchField](#)

- `bool m_categoryLabelVisible = true`
- [AnimatedClearButton](#) \* `m_clearButton = nullptr`
- `QLabel * m_detailLabel = nullptr`
- `QLabel * m_label = nullptr`
- `QString m_name`
- `bool m_valuesValid = false`

## 9.1146.1 Member Function Documentation

### 9.1146.1.1 `read()`

```
void Digikam::SearchFieldPageOrientation::read (
    SearchXmlCachingReader & reader ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

### 9.1146.1.2 `setupValueWidgets()`

```
void Digikam::SearchFieldPageOrientation::setupValueWidgets (
    QGridLayout * layout,
    int row,
    int column ) [override], [virtual]
```

Reimplemented from [Digikam::SearchFieldComboBox](#).

## 9.1147 Digikam::SearchFieldRangeDate Class Reference

Inheritance diagram for Digikam::SearchFieldRangeDate:



### Public Types

- enum **Type** { **DateOnly** , **DateTime** }



## Public Types inherited from Digikam::SearchField

- enum **WidgetRectType** { **LabelAndValueWidgetRects** , **ValueWidgetRectsOnly** }

## Public Member Functions

- **SearchFieldRangeDate** (QObject \*const parent, Type type)
- void **read** ([SearchXmlCachingReader](#) &reader) override
- void **reset** () override
- void **setBetweenText** (const QString &between)
- void **setBoundary** (const QDateTime &min, const QDateTime &max)
- void **setupValueWidgets** (QGridLayout \*layout, int row, int column) override
- void **setValueWidgetsVisible** (bool visible) override
- QList< QRect > **valueWidgetRects** () const override
- void **write** ([SearchXmlWriter](#) &writer) override

## Public Member Functions inherited from Digikam::SearchField

- **SearchField** (QObject \*const parent)
- bool **isVisible** () override
- void **setCategoryLabelVisible** (bool visible)
- void **setCategoryLabelVisibleFromPreviousField** ([SearchField](#) \*const previousField)
- void **setFieldName** (const QString &fieldName)
- virtual void **setText** (const QString &label, const QString &detailLabel)
- void **setup** (QGridLayout \*const layout, int row=-1)
- void **setVisible** (bool visible) override
- virtual bool **supportsField** (const QString &fieldName)
- QList< QRect > **widgetRects** (WidgetRectType=ValueWidgetRectsOnly) const

## Protected Slots

- void **valueChanged** ()

## Protected Slots inherited from Digikam::SearchField

- void **clearButtonClicked** ()

## Protected Attributes

- QLabel \* **m\_betweenLabel** = nullptr
- [DDateEdit](#) \* **m\_firstDateEdit** = nullptr
- [QTimeEdit](#) \* **m\_firstTimeEdit** = nullptr
- [DDateEdit](#) \* **m\_secondDateEdit** = nullptr
- [QTimeEdit](#) \* **m\_secondTimeEdit** = nullptr
- Type **m\_type** = DateOnly

## Protected Attributes inherited from [Digikam::SearchField](#)

- bool `m_categoryLabelVisible` = true
- [AnimatedClearButton](#) \* `m_clearButton` = nullptr
- QLabel \* `m_detailLabel` = nullptr
- QLabel \* `m_label` = nullptr
- QString `m_name`
- bool `m_valuesValid` = false

## Additional Inherited Members

## Signals inherited from [Digikam::SearchField](#)

- void `signalVisibilityChanged` ()

## Static Public Member Functions inherited from [Digikam::SearchField](#)

- static [SearchField](#) \* `createField` (const QString &fieldName, [SearchFieldGroup](#) \*const parent)

## Protected Member Functions inherited from [Digikam::SearchField](#)

- virtual void `setupLabels` (QGridLayout \*layout, int line)
- void `setValidValueState` (bool valuesValid)

## 9.1147.1 Member Function Documentation

### 9.1147.1.1 `read()`

```
void Digikam::SearchFieldRangeDate::read (
    SearchXmlCachingReader & reader ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

### 9.1147.1.2 `reset()`

```
void Digikam::SearchFieldRangeDate::reset ( ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

### 9.1147.1.3 `setupValueWidgets()`

```
void Digikam::SearchFieldRangeDate::setupValueWidgets (
    QGridLayout * layout,
    int row,
    int column ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

#### 9.1147.1.4 setValueWidgetsVisible()

```
void Digikam::SearchFieldRangeDate::setValueWidgetsVisible (
    bool visible ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

#### 9.1147.1.5 valueWidgetRects()

```
QList< QRect > Digikam::SearchFieldRangeDate::valueWidgetRects ( ) const [override], [virtual]
```

Implements [Digikam::SearchField](#).

#### 9.1147.1.6 write()

```
void Digikam::SearchFieldRangeDate::write (
    SearchXmlWriter & writer ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

## 9.1148 Digikam::SearchFieldRangeDouble Class Reference

Inheritance diagram for Digikam::SearchFieldRangeDouble:



### Public Member Functions

- **SearchFieldRangeDouble** (QObject \*const parent)
- void [read](#) ([SearchXmlCachingReader](#) &reader) override

- void [reset](#) () override
- void [setBetweenText](#) (const QString &text)
- void [setBoundary](#) (double min, double max, int decimals, double step)
- void [setFactor](#) (double factor)
- void [setInvertStepping](#) (bool invert)
- void [setNoValueText](#) (const QString &text)
- void [setNumberPrefixAndSuffix](#) (const QString &prefix, const QString &suffix)
- void [setSingleSteps](#) (double smaller, double larger)
- void [setSuggestedInitialValue](#) (double initialValue)
- void [setSuggestedValues](#) (const QList< double > &values)
- void [setupValueWidgets](#) (QGridLayout \*layout, int row, int column) override
- void [setValueWidgetsVisible](#) (bool visible) override
- QList< QRect > [valueWidgetRects](#) () const override
- void [write](#) ([SearchXmlWriter](#) &writer) override

### Public Member Functions inherited from [Digikam::SearchField](#)

- [SearchField](#) (QObject \*const parent)
- bool [isVisible](#) () override
- void [setCategoryLabelVisible](#) (bool visible)
- void [setCategoryLabelVisibleFromPreviousField](#) ([SearchField](#) \*const previousField)
- void [setFieldName](#) (const QString &fieldName)
- virtual void [setText](#) (const QString &label, const QString &detailLabel)
- void [setup](#) (QGridLayout \*const layout, int row=-1)
- void [setVisible](#) (bool visible) override
- virtual bool [supportsField](#) (const QString &fieldName)
- QList< QRect > [widgetRects](#) (WidgetRectType=ValueWidgetRectsOnly) const

### Protected Slots

- void [valueChanged](#) ()

### Protected Slots inherited from [Digikam::SearchField](#)

- void [clearButtonClicked](#) ()

### Protected Attributes

- QLabel \* [m\\_betweenLabel](#) = nullptr
- double [m\\_factor](#) = 1.0
- [CustomStepsDoubleSpinBox](#) \* [m\\_firstBox](#) = nullptr
- double [m\\_max](#) = 100.0
- double [m\\_min](#) = 0.0
- [CustomStepsDoubleSpinBox](#) \* [m\\_secondBox](#) = nullptr

### Protected Attributes inherited from [Digikam::SearchField](#)

- bool [m\\_categoryLabelVisible](#) = true
- [AnimatedClearButton](#) \* [m\\_clearButton](#) = nullptr
- QLabel \* [m\\_detailLabel](#) = nullptr
- QLabel \* [m\\_label](#) = nullptr
- QString [m\\_name](#)
- bool [m\\_valuesValid](#) = false

## Additional Inherited Members

## Public Types inherited from [Digikam::SearchField](#)

- enum **WidgetRectType** { [LabelAndValueWidgetRects](#) , [ValueWidgetRectsOnly](#) }

## Signals inherited from [Digikam::SearchField](#)

- void **signalVisibilityChanged** ()

## Static Public Member Functions inherited from [Digikam::SearchField](#)

- static [SearchField](#) \* **createField** (const QString &fieldName, [SearchFieldGroup](#) \*const parent)

## Protected Member Functions inherited from [Digikam::SearchField](#)

- virtual void **setupLabels** (QGridLayout \*layout, int line)
- void **setValidValueState** (bool valueIsValid)

## 9.1148.1 Member Function Documentation

### 9.1148.1.1 read()

```
void Digikam::SearchFieldRangeDouble::read (
    SearchXmlCachingReader & reader ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

### 9.1148.1.2 reset()

```
void Digikam::SearchFieldRangeDouble::reset ( ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

### 9.1148.1.3 setupValueWidgets()

```
void Digikam::SearchFieldRangeDouble::setupValueWidgets (
    QGridLayout * layout,
    int row,
    int column ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

#### 9.1148.1.4 setValueWidgetsVisible()

```
void Digikam::SearchFieldRangeDouble::setValueWidgetsVisible (
    bool visible ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

#### 9.1148.1.5 valueWidgetRects()

```
QList< QRect > Digikam::SearchFieldRangeDouble::valueWidgetRects ( ) const [override], [virtual]
```

Implements [Digikam::SearchField](#).

#### 9.1148.1.6 write()

```
void Digikam::SearchFieldRangeDouble::write (
    SearchXmlWriter & writer ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

## 9.1149 Digikam::SearchFieldRangeInt Class Reference

Inheritance diagram for Digikam::SearchFieldRangeInt:



### Public Member Functions

- **SearchFieldRangeInt** (QObject \*const parent)
- void **enableFractionMagic** (const QString &prefix)



- void [read](#) ([SearchXmlCachingReader](#) &reader) override
- void [reset](#) () override
- void [setBetweenText](#) (const QString &text)
- void [setBoundary](#) (int min, int max, int step=1)
- void [setInvertStepping](#) (bool invert)
- void [setNoValueText](#) (const QString &text)
- void [setNumberPrefixAndSuffix](#) (const QString &prefix, const QString &suffix)
- void [setSingleSteps](#) (int smaller, int larger)
- void [setSuggestedInitialValue](#) (int initialValue)
- void [setSuggestedValues](#) (const QList< int > &values)
- void [setupValueWidgets](#) (QGridLayout \*layout, int row, int column) override
- void [setValueWidgetsVisible](#) (bool visible) override
- QList< QRect > [valueWidgetRects](#) () const override
- void [write](#) ([SearchXmlWriter](#) &writer) override

## Public Member Functions inherited from [Digikam::SearchField](#)

- [SearchField](#) (QObject \*const parent)
- bool [isVisible](#) () override
- void [setCategoryLabelVisible](#) (bool visible)
- void [setCategoryLabelVisibleFromPreviousField](#) ([SearchField](#) \*const previousField)
- void [setFieldName](#) (const QString &fieldName)
- virtual void [setText](#) (const QString &label, const QString &detailLabel)
- void [setup](#) (QGridLayout \*const layout, int row=-1)
- void [setVisible](#) (bool visible) override
- virtual bool [supportsField](#) (const QString &fieldName)
- QList< QRect > [widgetRects](#) (WidgetRectType=ValueWidgetRectsOnly) const

## Protected Slots

- void [valueChanged](#) ()

## Protected Slots inherited from [Digikam::SearchField](#)

- void [clearButtonClicked](#) ()

## Protected Attributes

- QLabel \* [m\\_betweenLabel](#) = nullptr
- [CustomStepsIntSpinBox](#) \* [m\\_firstBox](#) = nullptr
- int [m\\_max](#) = 100
- int [m\\_min](#) = 0
- bool [m\\_reciprocal](#) = false
- [CustomStepsIntSpinBox](#) \* [m\\_secondBox](#) = nullptr

## Protected Attributes inherited from [Digikam::SearchField](#)

- bool [m\\_categoryLabelVisible](#) = true
- [AnimatedClearButton](#) \* [m\\_clearButton](#) = nullptr
- QLabel \* [m\\_detailLabel](#) = nullptr
- QLabel \* [m\\_label](#) = nullptr
- QString [m\\_name](#)
- bool [m\\_valuesValid](#) = false

## Additional Inherited Members

## Public Types inherited from [Digikam::SearchField](#)

- enum **WidgetRectType** { [LabelAndValueWidgetRects](#) , [ValueWidgetRectsOnly](#) }

## Signals inherited from [Digikam::SearchField](#)

- void **signalVisibilityChanged** ()

## Static Public Member Functions inherited from [Digikam::SearchField](#)

- static [SearchField](#) \* **createField** (const QString &fieldName, [SearchFieldGroup](#) \*const parent)

## Protected Member Functions inherited from [Digikam::SearchField](#)

- virtual void **setupLabels** (QGridLayout \*layout, int line)
- void **setValidValueState** (bool valueIsValid)

## 9.1149.1 Member Function Documentation

### 9.1149.1.1 read()

```
void Digikam::SearchFieldRangeInt::read (
    SearchXmlCachingReader & reader ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

### 9.1149.1.2 reset()

```
void Digikam::SearchFieldRangeInt::reset ( ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

### 9.1149.1.3 setupValueWidgets()

```
void Digikam::SearchFieldRangeInt::setupValueWidgets (
    QGridLayout * layout,
    int row,
    int column ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

#### 9.1149.1.4 setValueWidgetsVisible()

```
void Digikam::SearchFieldRangeInt::setValueWidgetsVisible (
    bool visible ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

#### 9.1149.1.5 valueWidgetRects()

```
QList< QRect > Digikam::SearchFieldRangeInt::valueWidgetRects ( ) const [override], [virtual]
```

Implements [Digikam::SearchField](#).

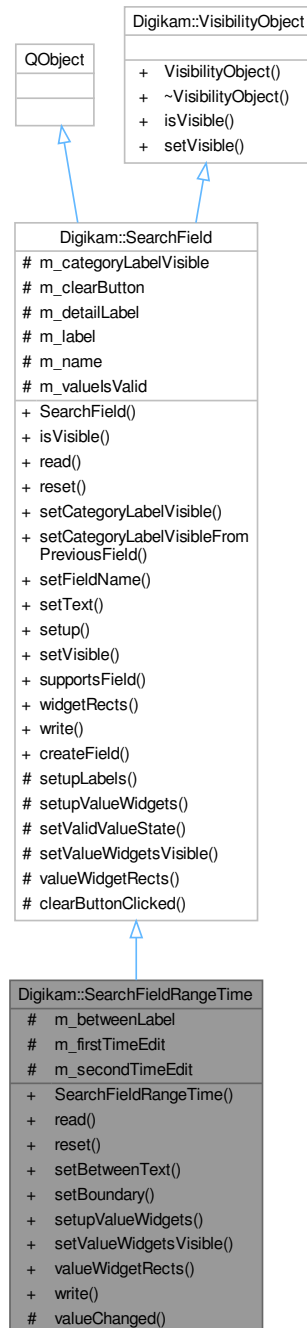
#### 9.1149.1.6 write()

```
void Digikam::SearchFieldRangeInt::write (
    SearchXmlWriter & writer ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

## 9.1150 Digikam::SearchFieldRangeTime Class Reference

Inheritance diagram for Digikam::SearchFieldRangeTime:



### Public Member Functions

- **SearchFieldRangeTime** (QObject \*const parent)
- void **read** ([SearchXmlCachingReader](#) &reader) override

- void [reset](#) () override
- void [setBetweenText](#) (const QString &between)
- void [setBoundary](#) (const QTime &min, const QTime &max)
- void [setupValueWidgets](#) (QGridLayout \*layout, int row, int column) override
- void [setValueWidgetsVisible](#) (bool visible) override
- QList< QRect > [valueWidgetRects](#) () const override
- void [write](#) ([SearchXmlWriter](#) &writer) override

## Public Member Functions inherited from [Digikam::SearchField](#)

- [SearchField](#) (QObject \*const parent)
- bool [isVisible](#) () override
- void [setCategoryLabelVisible](#) (bool visible)
- void [setCategoryLabelVisibleFromPreviousField](#) ([SearchField](#) \*const previousField)
- void [setFieldName](#) (const QString &fieldName)
- virtual void [setText](#) (const QString &label, const QString &detailLabel)
- void [setup](#) (QGridLayout \*const layout, int row=-1)
- void [setVisible](#) (bool visible) override
- virtual bool [supportsField](#) (const QString &fieldName)
- QList< QRect > [widgetRects](#) (WidgetRectType=ValueWidgetRectsOnly) const

## Protected Slots

- void [valueChanged](#) ()

## Protected Slots inherited from [Digikam::SearchField](#)

- void [clearButtonClicked](#) ()

## Protected Attributes

- QLabel \* [m\\_betweenLabel](#) = nullptr
- QTimeEdit \* [m\\_firstTimeEdit](#) = nullptr
- QTimeEdit \* [m\\_secondTimeEdit](#) = nullptr

## Protected Attributes inherited from [Digikam::SearchField](#)

- bool [m\\_categoryLabelVisible](#) = true
- [AnimatedClearButton](#) \* [m\\_clearButton](#) = nullptr
- QLabel \* [m\\_detailLabel](#) = nullptr
- QLabel \* [m\\_label](#) = nullptr
- QString [m\\_name](#)
- bool [m\\_valuesValid](#) = false

## Additional Inherited Members

## Public Types inherited from [Digikam::SearchField](#)

- enum [WidgetRectType](#) { [LabelAndValueWidgetRects](#) , [ValueWidgetRectsOnly](#) }

## Signals inherited from [Digikam::SearchField](#)

- void `signalVisibilityChanged` ()

## Static Public Member Functions inherited from [Digikam::SearchField](#)

- static `SearchField * createField` (const QString &fieldName, `SearchFieldGroup *const parent`)

## Protected Member Functions inherited from [Digikam::SearchField](#)

- virtual void `setupLabels` (QGridLayout \*layout, int line)
- void `setValidValueState` (bool valueIsValid)

### 9.1150.1 Member Function Documentation

#### 9.1150.1.1 `read()`

```
void Digikam::SearchFieldRangeTime::read (
    SearchXmlCachingReader & reader ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

#### 9.1150.1.2 `reset()`

```
void Digikam::SearchFieldRangeTime::reset ( ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

#### 9.1150.1.3 `setupValueWidgets()`

```
void Digikam::SearchFieldRangeTime::setupValueWidgets (
    QGridLayout * layout,
    int row,
    int column ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

#### 9.1150.1.4 `setValueWidgetsVisible()`

```
void Digikam::SearchFieldRangeTime::setValueWidgetsVisible (
    bool visible ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

### 9.1150.1.5 valueWidgetRects()

```
QList< QRect > Digikam::SearchFieldRangeTime::valueWidgetRects ( ) const [override], [virtual]
```

Implements [Digikam::SearchField](#).

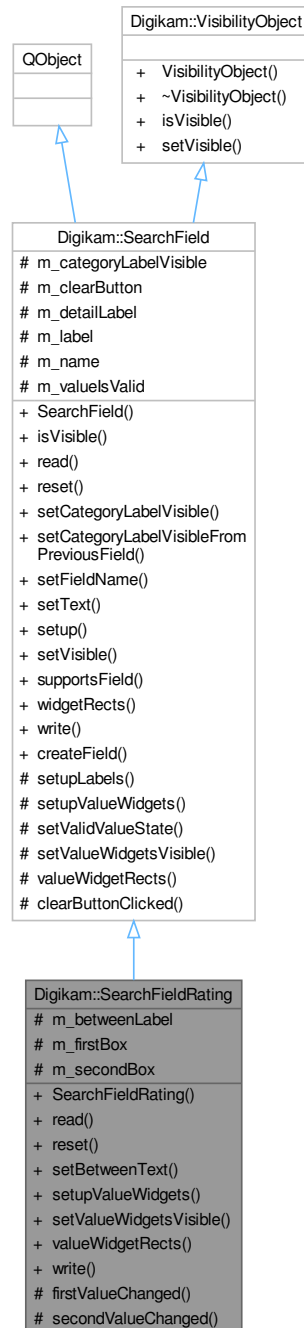
### 9.1150.1.6 write()

```
void Digikam::SearchFieldRangeTime::write (
    SearchXmlWriter & writer ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

## 9.1151 Digikam::SearchFieldRating Class Reference

Inheritance diagram for Digikam::SearchFieldRating:



### Public Member Functions

- **SearchFieldRating** (QObject \*const parent)
- void **read** ([SearchXmlCachingReader](#) &reader) override



- void [reset](#) () override
- void [setBetweenText](#) (const QString &text)
- void [setupValueWidgets](#) (QGridLayout \*layout, int row, int column) override
- void [setValueWidgetsVisible](#) (bool visible) override
- QList< QRect > [valueWidgetRects](#) () const override
- void [write](#) ([SearchXmlWriter](#) &writer) override

## Public Member Functions inherited from [Digikam::SearchField](#)

- [SearchField](#) (QObject \*const parent)
- bool [isVisible](#) () override
- void [setCategoryLabelVisible](#) (bool visible)
- void [setCategoryLabelVisibleFromPreviousField](#) ([SearchField](#) \*const previousField)
- void [setFieldName](#) (const QString &fieldName)
- virtual void [setText](#) (const QString &label, const QString &detailLabel)
- void [setup](#) (QGridLayout \*const layout, int row=-1)
- void [setVisible](#) (bool visible) override
- virtual bool [supportsField](#) (const QString &fieldName)
- QList< QRect > [widgetRects](#) (WidgetRectType=ValueWidgetRectsOnly) const

## Protected Slots

- void [firstValueChanged](#) ()
- void [secondValueChanged](#) ()

## Protected Slots inherited from [Digikam::SearchField](#)

- void [clearButtonClicked](#) ()

## Protected Attributes

- QLabel \* [m\\_betweenLabel](#) = nullptr
- [RatingComboBox](#) \* [m\\_firstBox](#) = nullptr
- [RatingComboBox](#) \* [m\\_secondBox](#) = nullptr

## Protected Attributes inherited from [Digikam::SearchField](#)

- bool [m\\_categoryLabelVisible](#) = true
- [AnimatedClearButton](#) \* [m\\_clearButton](#) = nullptr
- QLabel \* [m\\_detailLabel](#) = nullptr
- QLabel \* [m\\_label](#) = nullptr
- QString [m\\_name](#)
- bool [m\\_valuesValid](#) = false

## Additional Inherited Members

## Public Types inherited from [Digikam::SearchField](#)

- enum [WidgetRectType](#) { [LabelAndValueWidgetRects](#) , [ValueWidgetRectsOnly](#) }

## Signals inherited from [Digikam::SearchField](#)

- void `signalVisibilityChanged` ()

## Static Public Member Functions inherited from [Digikam::SearchField](#)

- static `SearchField * createField` (const QString &fieldName, `SearchFieldGroup *const parent`)

## Protected Member Functions inherited from [Digikam::SearchField](#)

- virtual void `setupLabels` (QGridLayout \*layout, int line)
- void `setValidValueState` (bool valueIsValid)

### 9.1151.1 Member Function Documentation

#### 9.1151.1.1 `read()`

```
void Digikam::SearchFieldRating::read (
    SearchXmlCachingReader & reader ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

#### 9.1151.1.2 `reset()`

```
void Digikam::SearchFieldRating::reset ( ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

#### 9.1151.1.3 `setValueWidgets()`

```
void Digikam::SearchFieldRating::setValueWidgets (
    QGridLayout * layout,
    int row,
    int column ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

#### 9.1151.1.4 `setValueWidgetsVisible()`

```
void Digikam::SearchFieldRating::setValueWidgetsVisible (
    bool visible ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

### 9.1151.1.5 valueWidgetRects()

```
QList< QRect > Digikam::SearchFieldRating::valueWidgetRects ( ) const [override], [virtual]
```

Implements [Digikam::SearchField](#).

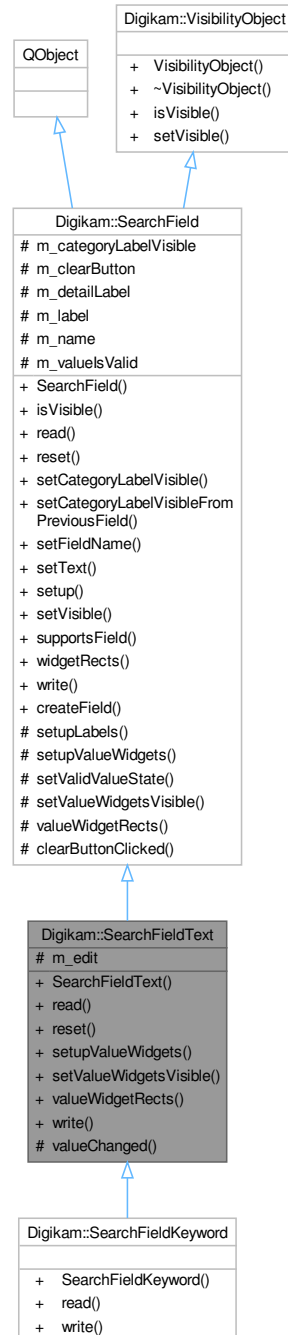
### 9.1151.1.6 write()

```
void Digikam::SearchFieldRating::write (
    SearchXmlWriter & writer ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

## 9.1152 Digikam::SearchFieldText Class Reference

Inheritance diagram for Digikam::SearchFieldText:



### Public Member Functions

- **SearchFieldText** (QObject \*const parent)
- void **read** (SearchXmlCachingReader &reader) override

- void [reset](#) () override
- void [setValueWidgets](#) (QGridLayout \*layout, int row, int column) override
- void [setValueWidgetsVisible](#) (bool visible) override
- QList< QRect > [valueWidgetRects](#) () const override
- void [write](#) (SearchXmlWriter &writer) override

## Public Member Functions inherited from Digikam::SearchField

- **SearchField** (QObject \*const parent)
- bool [isVisible](#) () override
- void [setCategoryLabelVisible](#) (bool visible)
- void [setCategoryLabelVisibleFromPreviousField](#) (SearchField \*const previousField)
- void [setFieldName](#) (const QString &fieldName)
- virtual void [setText](#) (const QString &label, const QString &detailLabel)
- void [setup](#) (QGridLayout \*const layout, int row=-1)
- void [setVisible](#) (bool visible) override
- virtual bool [supportsField](#) (const QString &fieldName)
- QList< QRect > [widgetRects](#) (WidgetRectType=ValueWidgetRectsOnly) const

## Protected Slots

- void [valueChanged](#) (const QString &text)

## Protected Slots inherited from Digikam::SearchField

- void [clearButtonClicked](#) ()

## Protected Attributes

- QLineEdit \* [m\\_edit](#) = nullptr

## Protected Attributes inherited from Digikam::SearchField

- bool [m\\_categoryLabelVisible](#) = true
- [AnimatedClearButton](#) \* [m\\_clearButton](#) = nullptr
- QLabel \* [m\\_detailLabel](#) = nullptr
- QLabel \* [m\\_label](#) = nullptr
- QString [m\\_name](#)
- bool [m\\_valuesValid](#) = false

## Additional Inherited Members

## Public Types inherited from Digikam::SearchField

- enum [WidgetRectType](#) { [LabelAndValueWidgetRects](#) , [ValueWidgetRectsOnly](#) }

## Signals inherited from [Digikam::SearchField](#)

- void `signalVisibilityChanged` ()

## Static Public Member Functions inherited from [Digikam::SearchField](#)

- static `SearchField * createField` (const QString &fieldName, `SearchFieldGroup *const parent`)

## Protected Member Functions inherited from [Digikam::SearchField](#)

- virtual void `setupLabels` (QGridLayout \*layout, int line)
- void `setValidValueState` (bool valueIsValid)

## 9.1152.1 Member Function Documentation

### 9.1152.1.1 `read()`

```
void Digikam::SearchFieldText::read (
    SearchXmlCachingReader & reader ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

### 9.1152.1.2 `reset()`

```
void Digikam::SearchFieldText::reset ( ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

### 9.1152.1.3 `setupValueWidgets()`

```
void Digikam::SearchFieldText::setupValueWidgets (
    QGridLayout * layout,
    int row,
    int column ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

### 9.1152.1.4 `setValueWidgetsVisible()`

```
void Digikam::SearchFieldText::setValueWidgetsVisible (
    bool visible ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

### 9.1152.1.5 valueWidgetRects()

```
QList< QRect > Digikam::SearchFieldText::valueWidgetRects ( ) const [override], [virtual]
```

Implements [Digikam::SearchField](#).

### 9.1152.1.6 write()

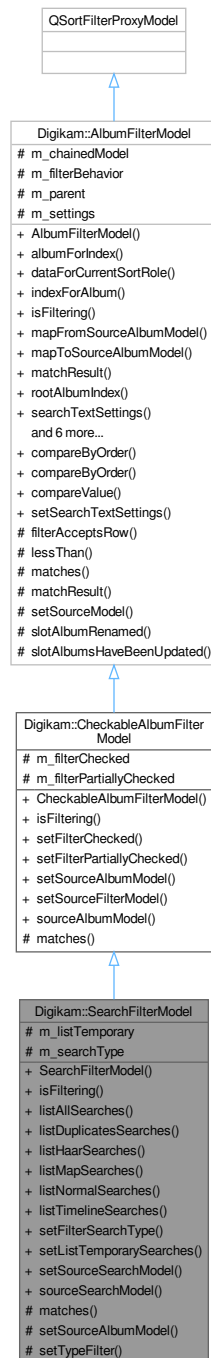
```
void Digikam::SearchFieldText::write (
    SearchXmlWriter & writer ) [override], [virtual]
```

Implements [Digikam::SearchField](#).

## 9.1153 Digikam::SearchFilterModel Class Reference

[Filter](#) model for searches that can filter by search type.

Inheritance diagram for Digikam::SearchFilterModel:



## Public Member Functions

- **SearchFilterModel** (QObject \*const parent=nullptr)
- bool **isFiltering** () const override  
*Returns if the currently applied filters will result in any filtering.*
- void **listAllSearches** ()
- void **listDuplicatesSearches** ()



- void **listHaarSearches** ()
- void **listMapSearches** ()
- void **listNormalSearches** ()
- void **listTimelineSearches** ()
- void **setFilterSearchType** (DatabaseSearch::Type)
  - Set the DatabaseSearch::Type.*
- void **setListTemporarySearches** (bool list)
  - Sets if temporary search albums shall be listed.*
- void **setSourceSearchModel** (SearchModel \*const source)
- SearchModel \* **sourceSearchModel** () const

## Public Member Functions inherited from Digikam::CheckableAlbumFilterModel

- **CheckableAlbumFilterModel** (QObject \*const parent=nullptr)
- void **setFilterChecked** (bool filter)
- void **setFilterPartiallyChecked** (bool filter)
- void **setSourceAlbumModel** (AbstractCheckableAlbumModel \*const source)
- void **setSourceFilterModel** (CheckableAlbumFilterModel \*const source)
- AbstractCheckableAlbumModel \* **sourceAlbumModel** () const

## Public Member Functions inherited from Digikam::AlbumFilterModel

- **AlbumFilterModel** (QObject \*const parent=nullptr)
- Album \* **albumForIndex** (const QModelIndex &index) const
  - Convenience methods.*
- QVariant **dataForCurrentSortRole** (Album \*album) const
- QModelIndex **indexForAlbum** (Album \*album) const
- QModelIndex **mapFromSourceAlbumModel** (const QModelIndex &index) const
- QModelIndex **mapToSourceAlbumModel** (const QModelIndex &index) const
- MatchResult **matchResult** (const QModelIndex &index) const
  - Returns the MatchResult of an index of this model.*
- QModelIndex **rootAlbumIndex** () const
- SearchTextSettings **searchTextSettings** () const
  - Returns the settings currently used for filtering.*
- void **setFilterBehavior** (FilterBehavior behavior)
  - Sets the filter behavior.*
- void **setSourceAlbumModel** (AbstractAlbumModel \*const source)
  - Sets the source model.*
- void **setSourceFilterModel** (AlbumFilterModel \*const source)
  - Sets a chained filter model.*
- AbstractAlbumModel \* **sourceAlbumModel** () const
- AlbumFilterModel \* **sourceFilterModel** () const
- void **updateFilter** ()
  - Force invalidateFilter() externally.*

## Protected Member Functions

- bool **matches** (Album \*album) const override
  - This method provides the basic match checking algorithm.*
- void **setSourceAlbumModel** (AbstractAlbumModel \*const source)
- void **setTypeFilter** (int type)

## Protected Member Functions inherited from [Digikam::AlbumFilterModel](#)

- bool **filterAcceptsRow** (int source\_row, const QModelIndex &source\_parent) const override
- bool **lessThan** (const QModelIndex &left, const QModelIndex &right) const override
- [MatchResult](#) **matchResult** ([Album](#) \*album) const  
*Returns if the filter matches this album (same logic as filterAcceptsRow).*
- void **setSourceModel** ([QAbstractItemModel](#) \*const model) override  
*Use setSourceAlbumModel.*

## Protected Attributes

- bool **m\_listTemporary** = false
- int **m\_searchType** = -1

## Protected Attributes inherited from [Digikam::CheckableAlbumFilterModel](#)

- bool **m\_filterChecked** = false
- bool **m\_filterPartiallyChecked** = false

## Protected Attributes inherited from [Digikam::AlbumFilterModel](#)

- [QPointer](#)< [AlbumFilterModel](#) > **m\_chainedModel** = nullptr
- [FilterBehavior](#) **m\_filterBehavior** = [FullFiltering](#)
- [QObject](#) \* **m\_parent** = nullptr
- [SearchTextSettings](#) **m\_settings**

## Additional Inherited Members

## Public Types inherited from [Digikam::AlbumFilterModel](#)

- enum [FilterBehavior](#) { [SimpleFiltering](#) , [FullFiltering](#) , [StrictFiltering](#) }
- enum [MatchResult](#) {  
    [NoMatch](#) = 0 , [DirectMatch](#) , [ParentMatch](#) , [ChildMatch](#) ,  
    [SpecialMatch](#) }

## Public Slots inherited from [Digikam::AlbumFilterModel](#)

- void **setSearchTextSettings** (const [SearchTextSettings](#) &settings)  
*Accepts new settings used for filtering and applies them to the model.*

## Signals inherited from [Digikam::AlbumFilterModel](#)

- void **hasSearchResult** (bool hasResult)  
*Indicates whether the newly applied filter results in a search result or not.*
- void **searchTextSettingsAboutToChange** (bool searched, bool willSearch)  
*This signal indicates that a new [SearchTextSettings](#) arrived and is about to be applied to the model.*
- void **searchTextSettingsChanged** (bool wasSearching, bool searched)  
*Indicates that new search text settings were applied.*
- void **signalFilterChanged** ()  
*Indicates that a new filter was applied to the model.*

## Static Public Member Functions inherited from [Digikam::AlbumFilterModel](#)

- `template<typename T >`  
static int **compareByOrder** (const T &a, const T &b, Qt::SortOrder sortOrder)
- static int **compareByOrder** (int compareResult, Qt::SortOrder sortOrder)  
*Takes a typical result from a compare method (0 is equal, -1 is less than, 1 is greater than) and applies the given sort order to it.*
- `template<typename T >`  
static int **compareValue** (const T &a, const T &b)  
*Returns the usual compare result of -1, 0, or 1 for lessThan, equals and greaterThan.*

## Protected Slots inherited from [Digikam::AlbumFilterModel](#)

- void **slotAlbumRenamed** ([Album](#) \*album)
- void **slotAlbumsHaveBeenUpdated** (int type)

### 9.1153.1 Member Function Documentation

#### 9.1153.1.1 `isFiltering()`

```
bool Digikam::SearchFilterModel::isFiltering ( ) const [override], [virtual]
```

##### Returns

`true` if the current selected filter could result in any filtering without checking if this really happens.

Reimplemented from [Digikam::CheckableAlbumFilterModel](#).

#### 9.1153.1.2 `matches()`

```
bool Digikam::SearchFilterModel::matches (
    Album * album ) const [override], [protected], [virtual]
```

Return true if this single album matches the current criteria. This method can be overridden to provide custom filtering.

##### Parameters

<code>album</code>	the album to tell if it matches the filter criteria or not.
--------------------	---

Reimplemented from [Digikam::CheckableAlbumFilterModel](#).

## 9.1154 Digikam::SearchGroup Class Reference

Inheritance diagram for Digikam::SearchGroup:



### Public Types

- enum **Type** { **FirstGroup** , **ChainGroup** }

## Signals

- void **removeRequested** ()

## Public Member Functions

- **SearchGroup** ([SearchView](#) \*const parent)
- Type **groupType** () const
- void **read** ([SearchXmlCachingReader](#) &reader)
- void **reset** ()
- void **setup** (Type type=FirstGroup)
- QList< QRect > **startupAnimationArea** () const
- void **write** ([SearchXmlWriter](#) &writer)

## Public Member Functions inherited from [Digikam::AbstractSearchGroupContainer](#)

- **AbstractSearchGroupContainer** (QWidget \*const parent=nullptr)  
*Abstract base class for classes that contain SearchGroups To contain common code of [SearchView](#) and [SearchGroup](#), as SearchGroups can have subgroups.*

## Protected Member Functions

- void **addGroupToLayout** ([SearchGroup](#) \*group) override  
*Re-implement: Adds a newly created group to the layout structures.*
- [SearchGroup](#) \* **createSearchGroup** () override  
*Re-implement: create and setup a search group.*

## Protected Member Functions inherited from [Digikam::AbstractSearchGroupContainer](#)

- void **finishReadingGroups** ()  
*Call when the XML part is finished.*
- void **readGroup** ([SearchXmlCachingReader](#) &reader)  
*Call when a group element is the current element.*
- void **startReadingGroups** ([SearchXmlCachingReader](#) &reader)  
*Call before reading the XML part that could contain group elements.*
- QList< QRect > **startupAnimationAreaOfGroups** () const  
*Collects the data from the same method of all contained groups (position relative to this widget)*
- void **writeGroups** ([SearchXmlWriter](#) &writer) const  
*Write contained groups to writer.*

## Protected Attributes

- QList< [SearchFieldGroup](#) \* > **m\_fieldGroups**
- QList< [SearchFieldGroupLabel](#) \* > **m\_fieldLabels**
- Type **m\_groupType** = FirstGroup
- [SearchGroupLabel](#) \* **m\_label** = nullptr
- QVBoxLayout \* **m\_layout** = nullptr
- QVBoxLayout \* **m\_subgroupLayout** = nullptr
- [SearchView](#) \* **m\_view** = nullptr

## Protected Attributes inherited from [Digikam::AbstractSearchGroupContainer](#)

- int `m_groupIndex` = 0
- `QList< SearchGroup * >` `m_groups`

## Additional Inherited Members

## Public Slots inherited from [Digikam::AbstractSearchGroupContainer](#)

- `SearchGroup * addSearchGroup ()`
- void `removeSearchGroup (SearchGroup *group)`

## Protected Slots inherited from [Digikam::AbstractSearchGroupContainer](#)

- void `removeSendingSearchGroup ()`

## 9.1154.1 Member Function Documentation

### 9.1154.1.1 addGroupToLayout()

```
void Digikam::SearchGroup::addGroupToLayout (  
    SearchGroup * group ) [override], [protected], [virtual]
```

Implements [Digikam::AbstractSearchGroupContainer](#).

### 9.1154.1.2 createSearchGroup()

```
SearchGroup * Digikam::SearchGroup::createSearchGroup ( ) [override], [protected], [virtual]
```

Implements [Digikam::AbstractSearchGroupContainer](#).

## 9.1155 Digikam::SearchGroupLabel Class Reference

Inheritance diagram for Digikam::SearchGroupLabel:



### Signals

- void **removeClicked** ()

### Public Member Functions

- **SearchGroupLabel** ([SearchViewThemedPartsCache](#) \*const cache, SearchGroup::Type type, QWidget \*const parent=nullptr)
- SearchXml::Operator **defaultFieldOperator** () const
- SearchXml::Operator **groupOperator** () const
- void **setDefaultFieldOperator** (SearchXml::Operator op)
- void **setGroupOperator** (SearchXml::Operator op)

### Protected Slots

- void **boxesToggled** ()
- void **toggleGroupOperator** ()
- void **toggleShowOptions** ()

### Protected Member Functions

- void **adjustOperatorOptions** ()
- void **paintEvent** (QPaintEvent \*) override
- void **setExtended** (bool extended)
- void **updateGroupLabel** ()

## 9.1156 Digikam::SearchInfo Class Reference

A container class for transporting search information from the database to [AlbumManager](#).

### Public Types

- typedef QList< [SearchInfo](#) > **List**

### Public Member Functions

- bool **isNull** () const
- bool **operator**< (const [SearchInfo](#) &info) const  
*needed for sorting*

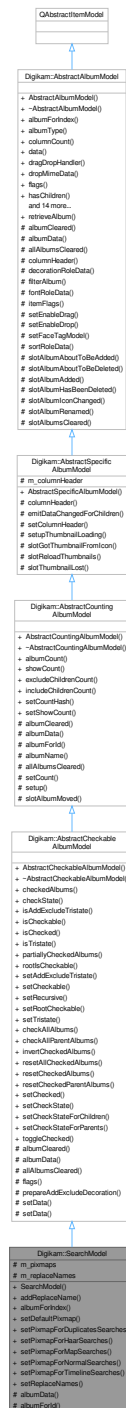
### Public Attributes

- int **id** = 0
- QString **name**
- QString **query**
- DatabaseSearch::Type **type** = DatabaseSearch::UndefinedType



## 9.1157 Digikam::SearchModel Class Reference

Inheritance diagram for Digikam::SearchModel:



### Public Member Functions

- **SearchModel** (QObject \*const parent=nullptr)

*Create a model containing searches.*

- void **addReplaceName** (const QString &technicalName, const QString &userVisibleName)
- **SAlbum \* albumForIndex** (const QModelIndex &index) const
- void **setDefaultPixmap** (const QPixmap &pix)
- void **setPixmapForDuplicatesSearches** (const QPixmap &pix)
- void **setPixmapForHaarSearches** (const QPixmap &pix)
- void **setPixmapForMapSearches** (const QPixmap &pix)
- void **setPixmapForNormalSearches** (const QPixmap &pix)  
*Set pixmaps for the DecorationRole.*
- void **setPixmapForTimelineSearches** (const QPixmap &pix)
- void **setReplaceNames** (const QHash< QString, QString > &replaceNames)  
*Set a hash of internal names (key) that shall be replaced by a user-visible string (value).*

## Public Member Functions inherited from [Digikam::AbstractCheckableAlbumModel](#)

- [AbstractCheckableAlbumModel](#) ([Album::Type](#) albumType, [Album](#) \*const rootAlbum, [RootAlbumBehavior](#) rootBehavior=[IncludeRootAlbum](#), [QObject](#) \*const parent=nullptr)  
*Abstract base class that manages the check state of Albums.*
- [QList](#)< [Album](#) \* > **checkedAlbums** () const  
*Returns a list of album with check state Checked.*
- [Qt::CheckState](#) **checkState** ([Album](#) \*album) const  
*Returns the check state of the album.*
- bool **isAddExcludeTristate** () const
- bool **isCheckable** () const
- bool **isChecked** ([Album](#) \*album) const  
*Returns if the given album has the check state Checked.*
- bool **isTristate** () const
- [QList](#)< [Album](#) \* > **partiallyCheckedAlbums** () const  
*Returns a list of album with partially check state Checked.*
- bool **rootIsCheckable** () const
- void **setAddExcludeTristate** (bool b)  
*Sets a special tristate mode, which offers the three modes "unchecked", "added" and "excluded", where "excluded" corresponds to partially checked internally, but is reflected in the treeview through the decoration only.*
- void **setCheckable** (bool isCheckable)  
*Triggers if the albums in this model are checkable.*
- void **setRecursive** (bool recursive)  
*If an item gets checked, all childs get checked as well, If an item gets unchecked, all childs get unchecked as well.*
- void **setRootCheckable** (bool rootIsCheckable)  
*Triggers if the root album is checkable.*
- void **setTristate** (bool isTristate)  
*Triggers if the albums in this model are tristate.*

## Public Member Functions inherited from [Digikam::AbstractCountingAlbumModel](#)

- [AbstractCountingAlbumModel](#) ([Album::Type](#) albumType, [Album](#) \*const rootAlbum, [RootAlbumBehavior](#) rootBehavior=[IncludeRootAlbum](#), [QObject](#) \*const parent=nullptr)  
*Supports displaying a count alongside the album name in DisplayRole.*
- virtual int **albumCount** ([Album](#) \*album) const  
*Returns the number of included items for this album.*
- bool **showCount** () const

## Public Member Functions inherited from Digikam::AbstractSpecificAlbumModel

- **AbstractSpecificAlbumModel** ([Album::Type](#) albumType, [Album](#) \*const rootAlbum, [RootAlbumBehavior](#) rootBehavior=[IncludeRootAlbum](#), [QObject](#) \*const parent=nullptr)

*Abstract base class, do not instantiate.*

## Public Member Functions inherited from Digikam::AbstractAlbumModel

- **AbstractAlbumModel** ([Album::Type](#) albumType, [Album](#) \*const rootAlbum, [RootAlbumBehavior](#) rootBehavior=[IncludeRootAlbum](#), [QObject](#) \*const parent=nullptr)
  - Create an [AbstractAlbumModel](#) object for albums with the given type.*
- [Album](#) \* **albumForIndex** (const [QModelIndex](#) &index) const
  - Returns the album object associated with the given model index.*
- [Album::Type](#) **albumType** () const
  - Returns the [Album::Type](#) of the contained albums.*
- int **columnCount** (const [QModelIndex](#) &parent=[QModelIndex](#)()) const override
- [QVariant](#) **data** (const [QModelIndex](#) &index, int role=[Qt::DisplayRole](#)) const override
- [AlbumModelDragDropHandler](#) \* **dragDropHandler** () const
  - Returns the drag drop handler, or 0 if none is installed.*
- bool **dropMimeData** (const [QMimeData](#) \*data, [Qt::DropAction](#) action, int row, int column, const [QModelIndex](#) &parent) override
- [Qt::ItemFlags](#) **flags** (const [QModelIndex](#) &index) const override
- bool **hasChildren** (const [QModelIndex](#) &parent=[QModelIndex](#)()) const override
- [QVariant](#) **headerData** (int section, [Qt::Orientation](#) orientation, int role=[Qt::DisplayRole](#)) const override
- [QModelIndex](#) **index** (int row, int column, const [QModelIndex](#) &parent=[QModelIndex](#)()) const override
- [QModelIndex](#) **indexForAlbum** ([Album](#) \*album) const
  - Return the [QModelIndex](#) for the given album, or an invalid index if the album is not contained in this model.*
- bool **isFaceTagModel** () const
  - Returns true if the album model a face tag model.*
- [QMimeData](#) \* **mimeData** (const [QModelIndexList](#) &indexes) const override
- [QStringList](#) **mimeTypes** () const override
- [QModelIndex](#) **parent** (const [QModelIndex](#) &index) const override
- [Album](#) \* **rootAlbum** () const
- [RootAlbumBehavior](#) **rootAlbumBehavior** () const
  - Returns the root album behavior set for this model.*
- [QModelIndex](#) **rootAlbumIndex** () const
  - Return the index corresponding to the root album.*
- int **rowCount** (const [QModelIndex](#) &parent=[QModelIndex](#)()) const override
- void **setDragDropHandler** ([AlbumModelDragDropHandler](#) \*handler)
  - Set a drag drop handler.*
- void **setDropIndex** (const [QModelIndex](#) &index)
  - Set current index from [QDragMoveEvent](#).*
- [Qt::DropActions](#) **supportedDropActions** () const override

## Protected Member Functions

- [QVariant](#) **albumData** ([Album](#) \*a, int role) const override
  - For subclassing convenience: A part of the implementation of data()*
- [Album](#) \* **albumForId** (int id) const override
  - need to implement in subclass*

## Protected Member Functions inherited from [Digikam::AbstractCheckableAlbumModel](#)

- void [albumCleared](#) ([Album](#) \*album) override  
*Notification when an entry is removed.*
- void [allAlbumsCleared](#) () override  
*Notification when all entries are removed.*
- Qt::ItemFlags **flags** (const QModelIndex &index) const override
- void **prepareAddExcludeDecoration** ([Album](#) \*a, QPixmap &icon) const  
*If in AddExcludeTristate mode, changes the icon as to indicate the state.*
- bool **setData** (const QModelIndex &index, const QVariant &value, int role, bool recursive)
- bool [setData](#) (const QModelIndex &index, const QVariant &value, int role=Qt::EditRole) override

## Protected Member Functions inherited from [Digikam::AbstractCountingAlbumModel](#)

- void [albumCleared](#) ([Album](#) \*album) override  
*Notification when an entry is removed.*
- virtual QString [albumName](#) ([Album](#) \*a) const  
*Can reimplement in subclass.*
- void [allAlbumsCleared](#) () override  
*Notification when all entries are removed.*
- void **setCount** ([Album](#) \*album, int count)  
*If you do not use setCountHash, excludeChildrenCount and includeChildrenCount, you can set a count here.*
- void **setup** ()  
*Call this method in children class constructors to init signal/slots connections.*

## Protected Member Functions inherited from [Digikam::AbstractSpecificAlbumModel](#)

- QString [columnHeader](#) () const override  
*For subclassing convenience: A part of the implementation of headerData()*
- void **emitDataChangedForChildren** ([Album](#) \*album)
- void **setColumnHeader** (const QString &header)
- void **setupThumbnailLoading** ()  
*You need to call this from your constructor if you intend to load the thumbnail facilities of this class.*

## Protected Member Functions inherited from [Digikam::AbstractAlbumModel](#)

- virtual QVariant [decorationRoleData](#) ([Album](#) \*a) const  
*For subclassing convenience: A part of the implementation of data()*
- virtual bool [filterAlbum](#) ([Album](#) \*album) const  
*Returns true for those and only those albums that shall be contained in this model.*
- virtual QVariant [fontRoleData](#) ([Album](#) \*a) const  
*For subclassing convenience: A part of the implementation of data()*
- virtual Qt::ItemFlags **itemFlags** ([Album](#) \*album) const  
*For subclassing convenience: A part of the implementation of itemFlags()*
- void [setEnableDrag](#) (bool enable)  
*Switch on drag and drop globally for all items.*
- void **setEnableDrop** (bool enable)
- void **setFaceTagModel** (bool enable)
- virtual QVariant [sortRoleData](#) ([Album](#) \*a) const  
*For subclassing convenience: A part of the implementation of data()*

### Protected Attributes

- QHash< int, QPixmap > **m\_pixmaps**
- QHash< QString, QString > **m\_replaceNames**

### Protected Attributes inherited from [Digikam::AbstractSpecificAlbumModel](#)

- QString **m\_columnHeader**

### Additional Inherited Members

### Public Types inherited from [Digikam::AbstractAlbumModel](#)

- enum [AlbumDataRole](#) {  
[AlbumTitleRole](#) = Qt::UserRole , [AlbumTypeRole](#) = Qt::UserRole + 1 , [AlbumPointerRole](#) = Qt::UserRole + 2  
, [AlbumIdRole](#) = Qt::UserRole + 3 ,  
[AlbumGlobalIdRole](#) = Qt::UserRole + 4 , [AlbumSortRole](#) = Qt::UserRole + 5 }
  - enum [RootAlbumBehavior](#) { [IncludeRootAlbum](#) , [IgnoreRootAlbum](#) }
- [AbstractAlbumModel](#) is the abstract base class for all models that present [Album](#) objects as managed by [AlbumManager](#).*

### Public Slots inherited from [Digikam::AbstractCheckableAlbumModel](#)

- void **checkAllAlbums** (const QModelIndex &parent=QModelIndex())  
*Checks all albums beneath the given parent.*
- void **checkAllParentAlbums** (const QModelIndex &child)  
*Checks all parent albums starting at the child, including it.*
- void **invertCheckedAlbums** (const QModelIndex &parent=QModelIndex())  
*Inverts the checked state of all albums under the given parent.*
- void **resetAllCheckedAlbums** ()  
*Resets the checked state of all albums to Qt::Unchecked.*
- void **resetCheckedAlbums** (const QModelIndex &parent=QModelIndex())  
*Resets the checked state of all albums under the given parent.*
- void **resetCheckedParentAlbums** (const QModelIndex &child)  
*Resets the checked state of all parents of the child including it.*
- void **setChecked** ([Album](#) \*album, bool isChecked)  
*Sets the check state of album to Checked or Unchecked.*
- void **setCheckState** ([Album](#) \*album, Qt::CheckState state)  
*Sets the check state of the album.*
- void **setCheckStateForChildren** ([Album](#) \*album, Qt::CheckState state)  
*Sets the checked state recursively for all children of but not for the given album.*
- void **setCheckStateForParents** ([Album](#) \*album, Qt::CheckState state)  
*Sets the checked state recursively for all parents of but not for the given album.*
- void **toggleChecked** ([Album](#) \*album)  
*Toggles the check state of album between Checked or Unchecked.*

## Public Slots inherited from [Digikam::AbstractCountingAlbumModel](#)

- void [excludeChildrenCount](#) (const QModelIndex &index)  
*Displays only the count of the album, without adding child albums' counts.*
- void [includeChildrenCount](#) (const QModelIndex &index)  
*Displays sum of the count of the album and child albums' counts.*
- void [setCountHash](#) (const QHash< int, int > &idCountHash)  
*Enable displaying the count.*
- void [setShowCount](#) (bool show)  
*Call to enable or disable showing the count. Default is false.*

## Signals inherited from [Digikam::AbstractCheckableAlbumModel](#)

- void [checkStateChanged](#) (Album \*album, Qt::CheckState checkState)  
*Emitted when the check state of an album changes.*

## Signals inherited from [Digikam::AbstractCountingAlbumModel](#)

- void [signalUpdateAlbumCount](#) (Album \*album)

## Signals inherited from [Digikam::AbstractAlbumModel](#)

- void [rootAlbumAvailable](#) ()  
*This is initialized once after creation, if the root album becomes available, if it was not already available at time of construction.*

## Static Public Member Functions inherited from [Digikam::AbstractAlbumModel](#)

- static Album \* [retrieveAlbum](#) (const QModelIndex &index)  
*Returns the album represented by the index.*

## Protected Slots inherited from [Digikam::AbstractCountingAlbumModel](#)

- void [slotAlbumMoved](#) (Album \*album)

## Protected Slots inherited from [Digikam::AbstractSpecificAlbumModel](#)

- void [slotGotThumbnailFromIcon](#) (Album \*album, const QPixmap &thumbnail)
- void [slotReloadThumbnails](#) ()
- void [slotThumbnailLost](#) (Album \*album)

## Protected Slots inherited from [Digikam::AbstractAlbumModel](#)

- void [slotAlbumAboutToBeAdded](#) (Album \*album, Album \*parent, Album \*prev)
- void [slotAlbumAboutToBeDeleted](#) (Album \*album)
- void [slotAlbumAdded](#) (Album \*)
- void [slotAlbumHasBeenDeleted](#) (Album \*album)
- void [slotAlbumIconChanged](#) (Album \*album)
- void [slotAlbumRenamed](#) (Album \*album)
- void [slotAlbumsCleared](#) ()

## 9.1157.1 Member Function Documentation

### 9.1157.1.1 albumData()

```
QVariant Digikam::SearchModel::albumData (
    Album * a,
    int role ) const [override], [protected], [virtual]
```

#### Note

these can be reimplemented in a subclass

Reimplemented from [Digikam::AbstractCheckableAlbumModel](#).

### 9.1157.1.2 albumForId()

```
Album * Digikam::SearchModel::albumForId (
    int id ) const [override], [protected], [virtual]
```

Implements [Digikam::AbstractCountingAlbumModel](#).

### 9.1157.1.3 setReplaceNames()

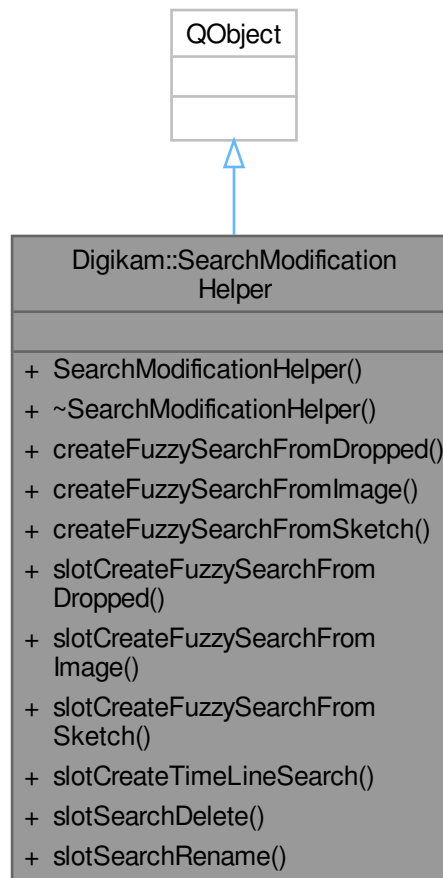
```
void Digikam::SearchModel::setReplaceNames (
    const QHash< QString, QString > & replaceNames )
```

This affects Qt::DisplayRole and AlbumTitleRole.

## 9.1158 Digikam::SearchModificationHelper Class Reference

Utility class providing methods to modify search albums ([SAlbum](#)) in a way useful to implement views.

Inheritance diagram for Digikam::SearchModificationHelper:



## Public Slots

- void [slotCreateFuzzySearchFromDropped](#) (const QString &name, const QString &filePath, float threshold, float maxThreshold, const QList< int > &targetAlbums, bool overwriteIfExisting)
 

*Creates a new fuzzy search for finding similar photos based on the file path of a photo and selects it in the album manager after creation.*
- void [slotCreateFuzzySearchFromImage](#) (const QString &name, const [ItemInfo](#) &image, float threshold, float maxThreshold, const QList< int > &targetAlbums, bool overwriteIfExisting=false)
 

*Creates a new fuzzy search for finding similar photos based on one photo and selects it in the album manager after creation.*
- void [slotCreateFuzzySearchFromSketch](#) (const QString &name, [SketchWidget](#) \*sketchWidget, unsigned int numberOfResults, const QList< int > &targetAlbums, bool overwriteIfExisting=false)
 

*Creates a new fuzzy search based on a sketch created by the user and selects it in the [AlbumManager](#) after creation.*
- [SAlbum](#) \* [slotCreateTimeLineSearch](#) (const QString &desiredName, const [DateRangeList](#) &dateRanges, bool overwriteIfExisting=false)
 

*Creates a new timeline search.*
- void [slotSearchDelete](#) ([SAlbum](#) \*searchAlbum)
 

*Deletes the given search after prompting the user.*



- void [slotSearchRename](#) (SAlbum \*searchAlbum)  
*Renames the given search via a dialog.*

## Public Member Functions

- [SearchModificationHelper](#) (QObject \*const parent, QWidget \*const dialogParent)  
*Constructor.*
- [~SearchModificationHelper](#) () override  
*Destructor.*
- SAlbum \* [createFuzzySearchFromDropped](#) (const QString &name, const QString &filePath, float threshold, float maxThreshold, const QList< int > &targetAlbums, bool overwriteIfExists=false)
- SAlbum \* [createFuzzySearchFromImage](#) (const QString &name, const [ItemInfo](#) &image, float threshold, float maxThreshold, const QList< int > &targetAlbums, bool overwriteIfExists=false)
- SAlbum \* [createFuzzySearchFromSketch](#) (const QString &name, [SketchWidget](#) \*sketchWidget, unsigned int numberOfResults, const QList< int > &targetAlbums, bool overwriteIfExists=false)

## 9.1158.1 Detailed Description

### Author

jwienke

## 9.1158.2 Constructor & Destructor Documentation

### 9.1158.2.1 SearchModificationHelper()

```
Digikam::SearchModificationHelper::SearchModificationHelper (
    QObject *const parent,
    QWidget *const dialogParent )
```

#### Parameters

<i>parent</i>	the parent for qt parent child mechanism
<i>dialogParent</i>	parent widget for dialogs displayed by this object

## 9.1158.3 Member Function Documentation

### 9.1158.3.1 createFuzzySearchFromDropped()

```
SAlbum * Digikam::SearchModificationHelper::createFuzzySearchFromDropped (
    const QString & name,
    const QString & filePath,
    float threshold,
    float maxThreshold,
    const QList< int > & targetAlbums,
    bool overwriteIfExists = false )
```

See also

[slotCreateFuzzySearchFromDropped\(\)](#)

Returns

the newly created album

### 9.1158.3.2 createFuzzySearchFromImage()

```
SAlbum * Digikam::SearchModificationHelper::createFuzzySearchFromImage (
    const QString & name,
    const ItemInfo & image,
    float threshold,
    float maxThreshold,
    const QList< int > & targetAlbums,
    bool overwriteIfExists = false )
```

See also

[slotCreateFuzzySearchFromImage\(\)](#)

Returns

the newly created album

### 9.1158.3.3 createFuzzySearchFromSketch()

```
SAlbum * Digikam::SearchModificationHelper::createFuzzySearchFromSketch (
    const QString & name,
    SketchWidget * sketchWidget,
    unsigned int numberOfResults,
    const QList< int > & targetAlbums,
    bool overwriteIfExists = false )
```

See also

[slotCreateFuzzySearchFromSketch\(\)](#)

Returns

the newly created album

### 9.1158.3.4 slotCreateFuzzySearchFromDropped

```
void Digikam::SearchModificationHelper::slotCreateFuzzySearchFromDropped (
    const QString & name,
    const QString & filePath,
    float threshold,
    float maxThreshold,
    const QList< int > & targetAlbums,
    bool overwriteIfExists ) [slot]
```

## Parameters

<i>name</i>	of the new search
<i>filePath</i>	path of the image to base this search on
<i>threshold</i>	minimum threshold for image search
<i>maxThreshold</i>	maximum threshold for image search
<i>targetAlbums</i>	The image must be in one of these albums
<i>overwriteIfExists</i>	if true, an existing search with the desired name will be overwritten without prompting the user for a new name

## 9.1158.3.5 slotCreateFuzzySearchFromImage

```
void Digikam::SearchModificationHelper::slotCreateFuzzySearchFromImage (
    const QString & name,
    const ItemInfo & image,
    float threshold,
    float maxThreshold,
    const QList< int > & targetAlbums,
    bool overwriteIfExists = false ) [slot]
```

## Parameters

<i>name</i>	of the new search
<i>image</i>	the image to base this search on
<i>threshold</i>	the threshold for image search, 0 <= threshold <= 1
<i>maxThreshold</i>	the maximum threshold of similarity.
<i>targetAlbums</i>	The image must be in one of these albums
<i>overwriteIfExists</i>	if true, an existing search with the desired name will be overwritten without prompting the user for a new name

## 9.1158.3.6 slotCreateFuzzySearchFromSketch

```
void Digikam::SearchModificationHelper::slotCreateFuzzySearchFromSketch (
    const QString & name,
    SketchWidget * sketchWidget,
    unsigned int numberOfResults,
    const QList< int > & targetAlbums,
    bool overwriteIfExists = false ) [slot]
```

## Parameters

<i>name</i>	the name of the new sketch search
<i>sketchWidget</i>	the widget containing the sketch of the user
<i>numberOfResults</i>	max number of results to display
<i>targetAlbums</i>	The image must be in one of these albums
<i>overwriteIfExists</i>	if true, an existing search with the desired name will be overwritten without prompting the user for a new name

**9.1158.3.7 slotCreateTimeLineSearch**

```
SAlbum * Digikam::SearchModificationHelper::slotCreateTimeLineSearch (
    const QString & desiredName,
    const DateRangeList & dateRanges,
    bool overwriteIfExists = false ) [slot]
```

**Parameters**

<i>desiredName</i>	desired name for the search. If this name already exists and <code>overwriteIfExists</code> is false, then the user will be prompted for a new name
<i>dateRanges</i>	date ranges to contain in this timeline search. If this is empty, no search will be created.
<i>overwriteIfExists</i>	if true, an existing search with the desired name will be overwritten without prompting the user for a new name

**9.1158.3.8 slotSearchDelete**

```
void Digikam::SearchModificationHelper::slotSearchDelete (
    SAlbum * searchAlbum ) [slot]
```

**Parameters**

<i>searchAlbum</i>	search to delete
--------------------	------------------

**9.1158.3.9 slotSearchRename**

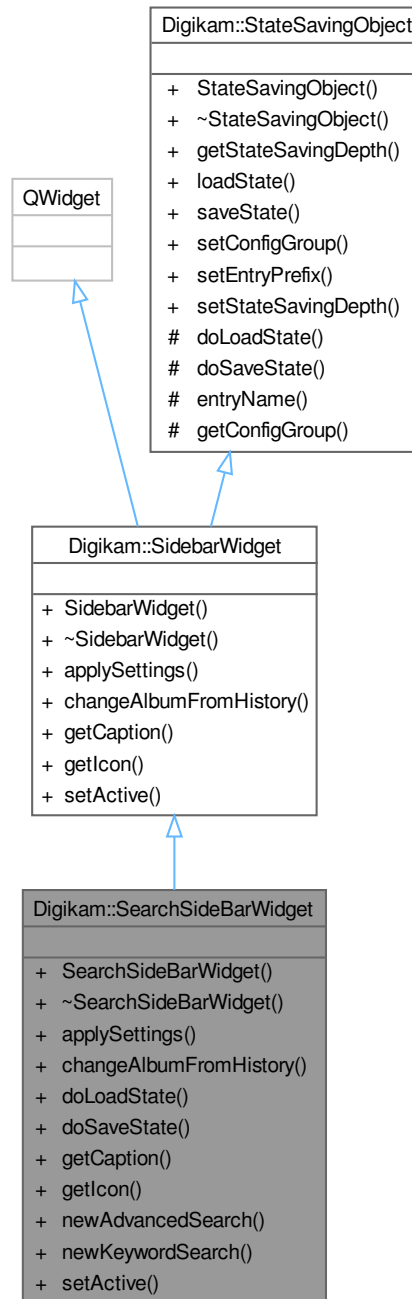
```
void Digikam::SearchModificationHelper::slotSearchRename (
    SAlbum * searchAlbum ) [slot]
```

**Parameters**

<i>searchAlbum</i>	search to rename
--------------------	------------------

## 9.1159 Digikam::SearchSideBarWidget Class Reference

Inheritance diagram for Digikam::SearchSideBarWidget:



### Public Member Functions

- **SearchSideBarWidget** (QWidget \*const parent, [SearchModel](#) \*const searchModel, [SearchModificationHelper](#) \*const searchModificationHelper)

- void [applySettings](#) () override  
*This method is invoked when the application settings should be (re-) applied to this widget.*
- void [changeAlbumFromHistory](#) (const QList< Album \* > &album) override  
*This is called on this widget when the history requires to move back to the specified album.*
- void [doLoadState](#) () override  
*Implement this hook method for state loading.*
- void [doSaveState](#) () override  
*Implement this hook method for state saving.*
- const QString [getCaption](#) () override  
*Must be implemented to return the title of this sidebar's tab.*
- const QIcon [getIcon](#) () override  
*Must be implemented and return the icon that shall be visible for this sidebar widget.*
- void **newAdvancedSearch** ()
- void **newKeywordSearch** ()
- void [setActive](#) (bool active) override  
*This method is called if the visible sidebar widget is changed.*

## Public Member Functions inherited from [Digikam::SidebarWidget](#)

- [SidebarWidget](#) (QWidget \*const parent)  
*Constructor.*
- **~SidebarWidget** () override=default  
*Destructor.*

## Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (QObject \*const host)  
*Constructor.*
- virtual **~StateSavingObject** ()  
*Destructor.*
- [StateSavingDepth](#) [getStateSavingDepth](#) () const  
*Returns the depth used for state saving or loading.*
- void **loadState** ()  
*Invokes loading the class' state.*
- void **saveState** ()  
*Invokes saving the class' state.*
- virtual void [setConfigGroup](#) (const KConfigGroup &group)  
*Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void [setEntryPrefix](#) (const QString &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void [setStateSavingDepth](#) (const [StateSavingDepth](#) depth)  
*Sets the depth used for state saving or loading.*

## Additional Inherited Members

## Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }  
*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

## Signals inherited from [Digikam::SidebarWidget](#)

- void **requestActiveTab** ([SidebarWidget](#) \*)  
*This signal can be emitted if this sidebar widget wants to be the one that is active.*
- void **signalNotificationError** (const QString &message, int type)  
*To dispatch error message to temporized pop-up notification widget hosted with icon-view.*

## Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- QString **entryName** (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup **getConfigGroup** () const  
*Returns the config group that must be used for state saving and loading.*

### 9.1159.1 Member Function Documentation

#### 9.1159.1.1 **applySettings()**

```
void Digikam::SearchSideBarWidget::applySettings ( ) [override], [virtual]
```

Implements [Digikam::SidebarWidget](#).

#### 9.1159.1.2 **changeAlbumFromHistory()**

```
void Digikam::SearchSideBarWidget::changeAlbumFromHistory (
    const QList< Album * > & album ) [override], [virtual]
```

Implements [Digikam::SidebarWidget](#).

#### 9.1159.1.3 **doLoadState()**

```
void Digikam::SearchSideBarWidget::doLoadState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

#### 9.1159.1.4 **doSaveState()**

```
void Digikam::SearchSideBarWidget::doSaveState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

### 9.1159.1.5 `getCaption()`

```
const QString Digikam::SearchSideBarWidget::getCaption ( ) [override], [virtual]
```

#### Returns

localized title string

Implements [Digikam::SidebarWidget](#).

### 9.1159.1.6 `getIcon()`

```
const QIcon Digikam::SearchSideBarWidget::getIcon ( ) [override], [virtual]
```

#### Returns

pixmap icon

Implements [Digikam::SidebarWidget](#).

### 9.1159.1.7 `setActive()`

```
void Digikam::SearchSideBarWidget::setActive (
    bool active ) [override], [virtual]
```

#### Parameters

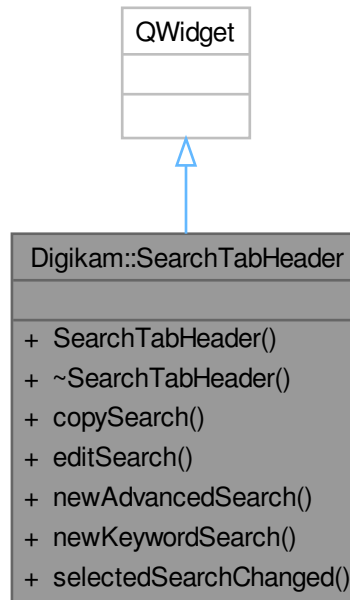
<i>active</i>	if true, this widget is the new active widget, if false another widget is active
---------------	--

Implements [Digikam::SidebarWidget](#).



## 9.1160 Digikam::SearchTabHeader Class Reference

Inheritance diagram for Digikam::SearchTabHeader:



### Public Slots

- void **copySearch** ([SAlbum](#) \*album)
- void **editSearch** ([SAlbum](#) \*album)
- void **newAdvancedSearch** ()
- void **newKeywordSearch** ()
- void **selectedSearchChanged** ([Album](#) \*album)

### Signals

- void **searchShallBeSelected** (const QList< [Album](#) \* > &albums)

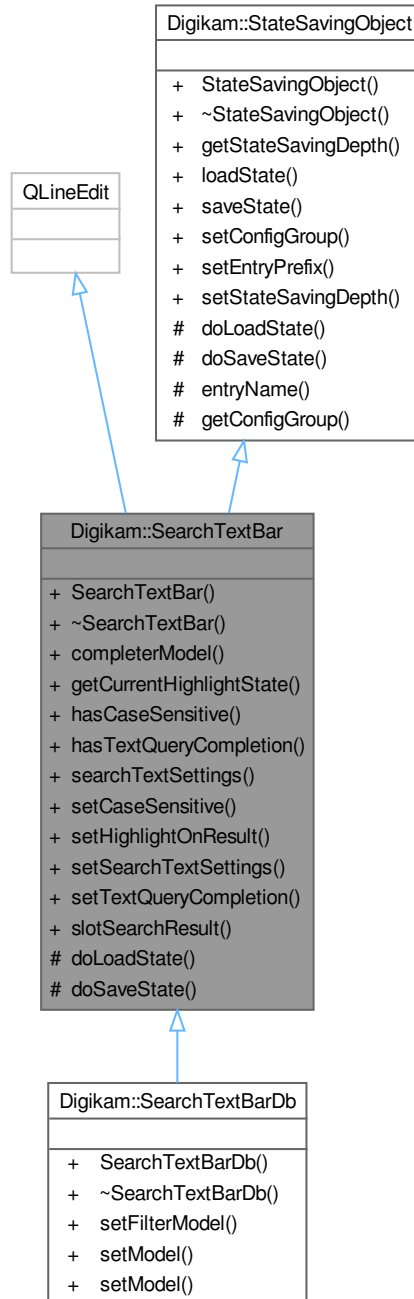
### Public Member Functions

- **SearchTabHeader** ([QWidget](#) \*const parent)

## 9.1161 Digikam::SearchTextBar Class Reference

A text input for searching entries with visual feedback.

Inheritance diagram for Digikam::SearchTextBar:



### Public Types

- enum `HighlightState` { `NEUTRAL`, `HAS_RESULT`, `NO_RESULT` }

Possible highlighting states a `SearchTextBar` can have.

## Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }

*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

## Public Slots

- void [slotSearchResult](#) (bool match)

## Signals

- void [completerActivated](#) ()
- void [completerHighlighted](#) (int albumId)
- void [signalSearchTextSettings](#) (const [SearchTextSettings](#) &settings)

## Public Member Functions

- [SearchTextBar](#) (QWidget \*const parent, const QString &name, const QString &msg=QString())
- [ModelCompleter](#) \* [completerModel](#) () const
- [HighlightState](#) [getCurrentHighlightState](#) () const  
*Tells the current highlighting state of the text input indicated via the background color.*
- bool [hasCaseSensitive](#) () const
- bool [hasTextQueryCompletion](#) () const
- [SearchTextSettings](#) [searchTextSettings](#) () const
- void [setCaseSensitive](#) (bool b)  
*Indicate whether this search text bar can be toggled to between case- sensitive and -insensitive or if always case-insensitive shall be used.*
- void [setHighlightOnResult](#) (bool highlight)  
*Tells whether highlighting for found search results shall be used or not (green and red).*
- void [setSearchTextSettings](#) (const [SearchTextSettings](#) &settings)
- void [setTextQueryCompletion](#) (bool b)

## Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (QObject \*const host)  
*Constructor.*
- virtual ~[StateSavingObject](#) ()  
*Destructor.*
- [StateSavingDepth](#) [getStateSavingDepth](#) () const  
*Returns the depth used for state saving or loading.*
- void [loadState](#) ()  
*Invokes loading the class' state.*
- void [saveState](#) ()  
*Invokes saving the class' state.*
- virtual void [setConfigGroup](#) (const KConfigGroup &group)  
*Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void [setEntryPrefix](#) (const QString &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void [setStateSavingDepth](#) (const [StateSavingDepth](#) depth)  
*Sets the depth used for state saving or loading.*

### Protected Member Functions

- void [doLoadState](#) () override  
*Implement this hook method for state loading.*
- void [doSaveState](#) () override  
*Implement this hook method for state saving.*

### Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- QString [entryName](#) (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup [getConfigGroup](#) () const  
*Returns the config group that must be used for state saving and loading.*

## 9.1161.1 Detailed Description

Can be used on QAbstractItemModels.

## 9.1161.2 Member Enumeration Documentation

### 9.1161.2.1 HighlightState

enum [Digikam::SearchTextBar::HighlightState](#)

Enumerator

NEUTRAL	No highlighting at all. Background is colored in a neutral way according to the theme.
HAS_RESULT	The background color of the text input indicates that a result was found.
NO_RESULT	The background color indicates that no result was found.

## 9.1161.3 Member Function Documentation

### 9.1161.3.1 doLoadState()

```
void Digikam::SearchTextBar::doLoadState ( ) [override], [protected], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

### 9.1161.3.2 doSaveState()

```
void Digikam::SearchTextBar::doSaveState ( ) [override], [protected], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

### 9.1161.3.3 getCurrentHighlightState()

```
SearchTextBar::HighlightState Digikam::SearchTextBar::getCurrentHighlightState ( ) const
```

#### Returns

current highlight state

### 9.1161.3.4 setCaseSensitive()

```
void Digikam::SearchTextBar::setCaseSensitive (
    bool b )
```

#### Parameters

<i>b</i>	if <code>true</code> the user can decide the toggle between case sensitivity, on <code>false</code> every search is case-insensitive
----------	--

### 9.1161.3.5 setHighlightOnResult()

```
void Digikam::SearchTextBar::setHighlightOnResult (
    bool highlight )
```

Default behavior has highlighting enabled.

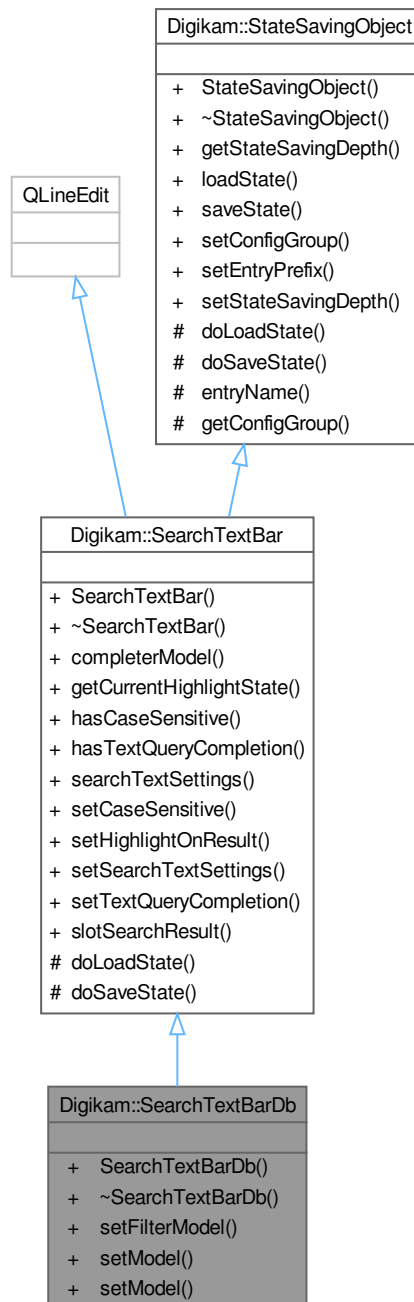
#### Parameters

<i>highlight</i>	<code>true</code> activates green and red highlighting, with <code>false</code> the normal widget background color will be displayed always
------------------	---

## 9.1162 Digikam::SearchTextBarDb Class Reference

A text input for searching entries with visual feedback.

Inheritance diagram for Digikam::SearchTextBarDb:



## Public Member Functions

- **SearchTextBarDb** (QWidget \*const parent, const QString &name, const QString &msg=QString())
- void [setFilterModel](#) (AlbumFilterModel \*const filterModel)
 

*Sets the filter model this text bar shall use to invoke filtering on and reading the result for highlighting from.*
- void [setModel](#) (AbstractAlbumModel \*const model)
 

*Sets the album model this text bar shall use to invoke filtering on and reading the result for highlighting from.*

- void [setModel](#) (QAbstractItemModel \*model, int uniqueIdRole, int displayRole=Qt::DisplayRole)  
*If the given model is != null, the model is used to populate the completion for this text field.*

### Public Member Functions inherited from [Digikam::SearchTextBar](#)

- [SearchTextBar](#) (QWidget \*const parent, const QString &name, const QString &msg=QString())
- [ModelCompleter](#) \* [completerModel](#) () const
- [HighlightState](#) [getCurrentHighlightState](#) () const  
*Tells the current highlighting state of the text input indicated via the background color.*
- bool [hasCaseSensitive](#) () const
- bool [hasTextQueryCompletion](#) () const
- [SearchTextSettings](#) [searchTextSettings](#) () const
- void [setCaseSensitive](#) (bool b)  
*Indicate whether this search text bar can be toggled to between case- sensitive and -insensitive or if always case-insensitive shall be used.*
- void [setHighlightOnResult](#) (bool highlight)  
*Tells whether highlighting for found search results shall be used or not (green and red).*
- void [setSearchTextSettings](#) (const [SearchTextSettings](#) &settings)
- void [setTextQueryCompletion](#) (bool b)

### Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (QObject \*const host)  
*Constructor.*
- virtual [~StateSavingObject](#) ()  
*Destructor.*
- [StateSavingDepth](#) [getStateSavingDepth](#) () const  
*Returns the depth used for state saving or loading.*
- void [loadState](#) ()  
*Invokes loading the class' state.*
- void [saveState](#) ()  
*Invokes saving the class' state.*
- virtual void [setConfigGroup](#) (const KConfigGroup &group)  
*Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void [setEntryPrefix](#) (const QString &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void [setStateSavingDepth](#) (const [StateSavingDepth](#) depth)  
*Sets the depth used for state saving or loading.*

### Additional Inherited Members

### Public Types inherited from [Digikam::SearchTextBar](#)

- enum [HighlightState](#) { [NEUTRAL](#) , [HAS\\_RESULT](#) , [NO\\_RESULT](#) }  
*Possible highlighting states a [SearchTextBar](#) can have.*

### Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }  
*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

## Public Slots inherited from [Digikam::SearchTextBar](#)

- void **slotSearchResult** (bool match)

## Signals inherited from [Digikam::SearchTextBar](#)

- void **completerActivated** ()
- void **completerHighlighted** (int albumId)
- void **signalSearchTextSettings** (const [SearchTextSettings](#) &settings)

## Protected Member Functions inherited from [Digikam::SearchTextBar](#)

- void [doLoadState](#) () override  
*Implement this hook method for state loading.*
- void [doSaveState](#) () override  
*Implement this hook method for state saving.*

## Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- QString [entryName](#) (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup [getConfigGroup](#) () const  
*Returns the config group that must be used for state saving and loading.*

### 9.1162.1 Detailed Description

Can be used on Database Models.

#### Author

Gilles Caulier

### 9.1162.2 Member Function Documentation

#### 9.1162.2.1 [setFilterModel\(\)](#)

```
void Digikam::SearchTextBarDb::setFilterModel (
    AlbumFilterModel *const filterModel )
```

#### Parameters

<i>filterModel</i>	filter model to use for filtering. <code>null</code> means there is no model to use and external connections need to be created with <code>signalSearchTextSettings</code> and <code>slotSearchResult</code>
--------------------	--



## 9.1162.2.2 setModel() [1/2]

```
void Digikam::SearchTextBarDb::setModel (
    AbstractAlbumModel *const model )
```

## Parameters

<i>model</i>	album model to use for filtering. <code>null</code> means there is no model to use and external connections need to be created with <code>signalSearchTextSettings</code> and <code>slotSearchResult</code>
--------------	---

## 9.1162.2.3 setModel() [2/2]

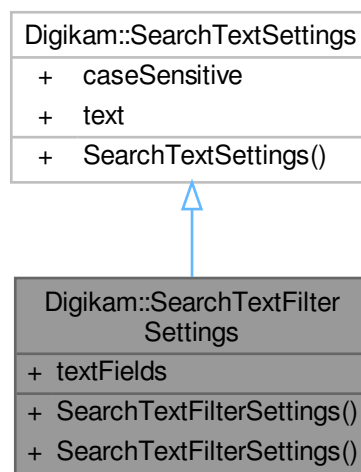
```
void Digikam::SearchTextBarDb::setModel (
    QAbstractItemModel * model,
    int uniqueIdRole,
    int displayRole = Qt::DisplayRole )
```

## Parameters

<i>model</i>	to fill from or null for manual mode
<i>uniqueIdRole</i>	a role for which the model will return a unique integer for each entry
<i>displayRole</i>	the role to retrieve the text for completion, default is <code>Qt::DisplayRole</code> .

## 9.1163 Digikam::SearchTextFilterSettings Class Reference

Inheritance diagram for Digikam::SearchTextFilterSettings:



**Public Types**

- enum **TextFilterFields** {  
**None** = 0x00 , **ImageName** = 0x01 , **ImageTitle** = 0x02 , **ImageComment** = 0x04 ,  
**TagName** = 0x08 , **AlbumName** = 0x10 , **ImageAspectRatio** = 0x20 , **ImagePixelSize** = 0x40 ,  
**All** = ImageName | ImageTitle | ImageComment | TagName | AlbumName | ImageAspectRatio | ImagePixelSize }

**Public Member Functions**

- **SearchTextFilterSettings** (const [SearchTextSettings](#) &settings)

**Public Attributes**

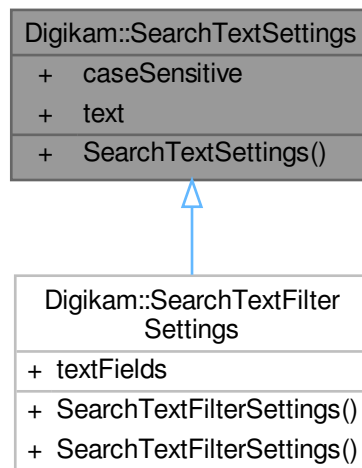
- TextFilterFields **textFields** = None

**Public Attributes inherited from [Digikam::SearchTextSettings](#)**

- Qt::CaseSensitivity **caseSensitive** = Qt::CaseInsensitive
- QString **text**

**9.1164 Digikam::SearchTextSettings Class Reference**

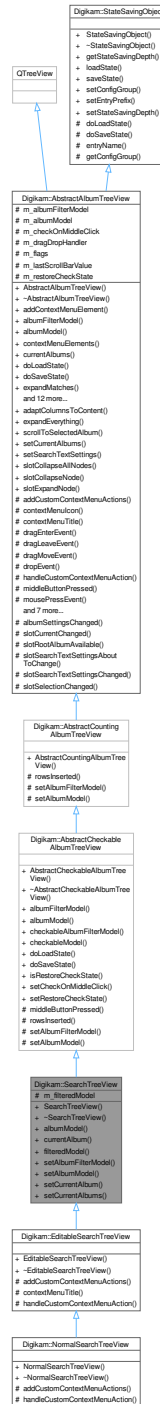
Inheritance diagram for Digikam::SearchTextSettings:

**Public Attributes**

- Qt::CaseSensitivity **caseSensitive** = Qt::CaseInsensitive
- QString **text**

## 9.1165 Digikam::SearchTreeView Class Reference

Inheritance diagram for Digikam::SearchTreeView:



### Public Slots

- void **setCurrentAlbum** (int searchId, bool selectInAlbumManager=true)
- void **setCurrentAlbums** (const QList< Album \* > &albums, bool selectInAlbumManager=true)

## Public Slots inherited from [Digikam::AbstractAlbumTreeView](#)

- void **adaptColumnsToContent** ()  
*Adapt the column sizes to the contents of the tree view.*
- void **expandEverything** (const QModelIndex &index)  
*Expands the complete tree under the given index.*
- void **scrollToSelectedAlbum** ()  
*Scrolls to the first selected album if there is one.*
- void **setCurrentAlbums** (const QList< Album \* > &albums, bool selectInAlbumManager=true)  
*Selects the given album.*
- void **setSearchTextSettings** (const SearchTextSettings &settings)
- void **slotCollapseAllNodes** ()  
*slotCollapseAllNodes - collapse all nodes without root node*
- void **slotCollapseNode** ()  
*slotCollapseNode - collapse recursively selected nodes*
- void **slotExpandNode** ()  
*slotExpandNode - expands recursively selected nodes*

## Public Member Functions

- **SearchTreeView** (QWidget \*const parent=nullptr, Flags flags=DefaultFlags)
- **SearchModel** \* **albumModel** () const  
*Note: not filtered by search type.*
- **SAlbum** \* **currentAlbum** () const
- **SearchFilterModel** \* **filteredModel** () const  
*Contains only the searches with appropriate type - prefer to [albumModel\(\)](#)*
- void **setAlbumFilterModel** (**SearchFilterModel** \*const **filteredModel**, **CheckableAlbumFilterModel** \*const model)
- void **setAlbumModel** (**SearchModel** \*const model)

## Public Member Functions inherited from [Digikam::AbstractCheckableAlbumTreeView](#)

- **AbstractCheckableAlbumTreeView** (QWidget \*const parent, Flags flags)  
*Models of these view can be checkable, they need not.*
- **CheckableAlbumFilterModel** \* **albumFilterModel** () const
- **AbstractCheckableAlbumModel** \* **albumModel** () const  
*Manage check state through the model directly.*
- **CheckableAlbumFilterModel** \* **checkableAlbumFilterModel** () const
- **AbstractCheckableAlbumModel** \* **checkableModel** () const
- void **doLoadState** () override  
*Implements state loading for the album tree view in a somewhat clumsy procedure because the model may not be fully loaded when this method is called.*
- void **doSaveState** () override  
*Implement this hook method for state saving.*
- bool **isRestoreCheckState** () const  
*Tells if the check state is restored while loading / saving state.*
- void **setCheckOnMiddleClick** (bool doThat)  
*Enable checking on middle mouse button click (default: on).*
- void **setRestoreCheckState** (bool restore)  
*Set whether to restore check state or not.*

## Public Member Functions inherited from Digikam::AbstractCountingAlbumTreeView

- **AbstractCountingAlbumTreeView** (QWidget \*const parent, Flags flags)

## Public Member Functions inherited from Digikam::AbstractAlbumTreeView

- **AbstractAlbumTreeView** (QWidget \*const parent, Flags flags)  
*Constructs an album tree view.*
- void **addContextMenuElement** (ContextMenuElement \*const element)
- AlbumFilterModel \* **albumFilterModel** () const
- AbstractSpecificAlbumModel \* **albumModel** () const
- QList< ContextMenuElement \* > **contextMenuElements** () const
- template<class A >  
QList< A \* > **currentAlbums** ()
- bool **expandMatches** (const QModelIndex &index)  
*Ensures that every current match is visible by expanding all parent entries.*
- QModelIndex **indexVisuallyAt** (const QPoint &p)  
*This is a combination of indexAt() checked with visualRect().*
- void **removeContextMenuElement** (ContextMenuElement \*const element)
- QList< Album \* > **selectedItems** ()  
*selectedItems()* -
- void **setAlbumManagerCurrentAlbum** (const bool setCurrentAlbum)  
*Some treeviews shall control the global current album kept by AlbumManager.*
- void **setContextMenuIcon** (const QPixmap &pixmap)  
*Set the context menu title and icon.*
- void **setContextMenuTitle** (const QString &title)
- void **setEnabledContextMenu** (const bool enable)  
*Determines the global decision to show a popup menu or not.*
- void **setExpandNewCurrentItem** (const bool doThat)  
*Expand an item when making it the new current item.*
- void **setExpandOnSingleClick** (const bool doThat)  
*Enable expanding of tree items on single click on the item (default: off)*
- void **setSelectAlbumOnClick** (const bool selectOnClick)  
*Sets whether to select an album on click via the album manager or not.*
- void **setSelectOnContextMenu** (const bool select)  
*Sets whether to select the album under the mouse cursor on a context menu request (so that the album is shown using the album manager) or not.*
- bool **viewportEvent** (QEvent \*event) override  
*For internal use only.*

## Public Member Functions inherited from Digikam::StateSavingObject

- **StateSavingObject** (QObject \*const host)  
*Constructor.*
- virtual ~**StateSavingObject** ()  
*Destructor.*
- **StateSavingDepth** **getStateSavingDepth** () const  
*Returns the depth used for state saving or loading.*
- void **loadState** ()  
*Invokes loading the class' state.*

- void **saveState** ()  
*Invokes saving the class' state.*
- virtual void **setConfigGroup** (const KConfigGroup &group)  
*Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void **setEntryPrefix** (const QString &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void **setStateSavingDepth** (const StateSavingDepth depth)  
*Sets the depth used for state saving or loading.*

### Protected Attributes

- [SearchFilterModel](#) \* **m\_filteredModel** = nullptr

### Protected Attributes inherited from [Digikam::AbstractAlbumTreeView](#)

- [AlbumFilterModel](#) \* **m\_albumFilterModel** = nullptr
- [AbstractSpecificAlbumModel](#) \* **m\_albumModel** = nullptr
- bool **m\_checkOnMiddleClick** = false
- [AlbumModelDragDropHandler](#) \* **m\_dragDropHandler** = nullptr
- Flags **m\_flags** = DefaultFlags
- int **m\_lastScrollBarValue** = 0
- bool **m\_restoreCheckState** = false

### Additional Inherited Members

### Public Types inherited from [Digikam::AbstractAlbumTreeView](#)

- enum **Flag** {  
[CreateDefaultModel](#) , [CreateDefaultFilterModel](#) , [CreateDefaultDelegate](#) , [ShowCountAccordingToSettings](#) ,  
[AlwaysShowInclusiveCounts](#) , **DefaultFlags** = [CreateDefaultFilterModel](#) | [CreateDefaultDelegate](#) | [ShowCountAccordingToSettings](#) }
- typedef QFlags< [Flag](#) > **Flags**

### Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }  
*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

### Signals inherited from [Digikam::AbstractAlbumTreeView](#)

- void **currentAlbumChanged** ([Album](#) \*currentAlbum)  
*Emitted when the currently selected album changes.*
- void **selectedAlbumsChanged** (const QList< [Album](#) \* > &selectedAlbums)  
*Emitted when the current selection changes.*

## Protected Slots inherited from [Digikam::AbstractAlbumTreeView](#)

- void **albumSettingsChanged** ()
- void **slotCurrentChanged** ()
- virtual void **slotRootAlbumAvailable** ()  
*override if implemented behavior is not as intended*
- void **slotSearchTextSettingsAboutToChange** (bool searched, bool willSearch)
- void **slotSearchTextSettingsChanged** (bool wasSearching, bool searching)
- void **slotSelectionChanged** ()

## Protected Member Functions inherited from [Digikam::AbstractCheckableAlbumTreeView](#)

- void **middleButtonPressed** (Album \*a) override
- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- void **setAlbumFilterModel** (CheckableAlbumFilterModel \*const filterModel)
- void **setAlbumModel** (AbstractCheckableAlbumModel \*const model)

## Protected Member Functions inherited from [Digikam::AbstractCountingAlbumTreeView](#)

- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- void **setAlbumFilterModel** (AlbumFilterModel \*const filterModel)
- void **setAlbumModel** (AbstractCountingAlbumModel \*const model)

## Protected Member Functions inherited from [Digikam::AbstractAlbumTreeView](#)

- virtual void **addCustomContextMenuActions** (ContextMenuHelper &cmh, Album \*album)  
*Hook method to add custom actions to the generated context menu.*
- virtual QPixmap **contextMenuIcon** () const  
*Hook method that can be implemented to return a special icon used for the context menu.*
- virtual QString **contextMenuTitle** () const  
*Hook method to implement that returns the title for the context menu.*
- void **dragEnterEvent** (QDragEnterEvent \*e) override
- void **dragLeaveEvent** (QDragLeaveEvent \*e) override
- void **dragMoveEvent** (QDragMoveEvent \*e) override
- void **dropEvent** (QDropEvent \*e) override
- virtual void **handleCustomContextMenuAction** (QAction \*action, const AlbumPointer< Album > &album)  
*Hook method to handle the custom context menu actions that were added with addCustomContextMenuActions.*
- void **mousePressEvent** (QMouseEvent \*e) override  
*Other helper methods.*
- virtual QPixmap  **pixmapForDrag** (const QStyleOptionViewItem &option, QList< QModelIndex > indexes)  
*TODO: Move to delegate, when we have one.*
- void **rowsAboutToBeRemoved** (const QModelIndex &parent, int start, int end) override
- void **rowsInserted** (const QModelIndex &index, int start, int end) override
- void **setAlbumFilterModel** (AlbumFilterModel \*const filterModel)
- void **setAlbumModel** (AbstractSpecificAlbumModel \*const model)
- virtual bool **showContextMenuAt** (QContextMenuEvent \*event, Album \*albumForEvent)  
*Hook method to implement that determines if a context menu shall be displayed for the given event at the position coded in the event.*
- void **startDrag** (Qt::DropActions supportedActions) override

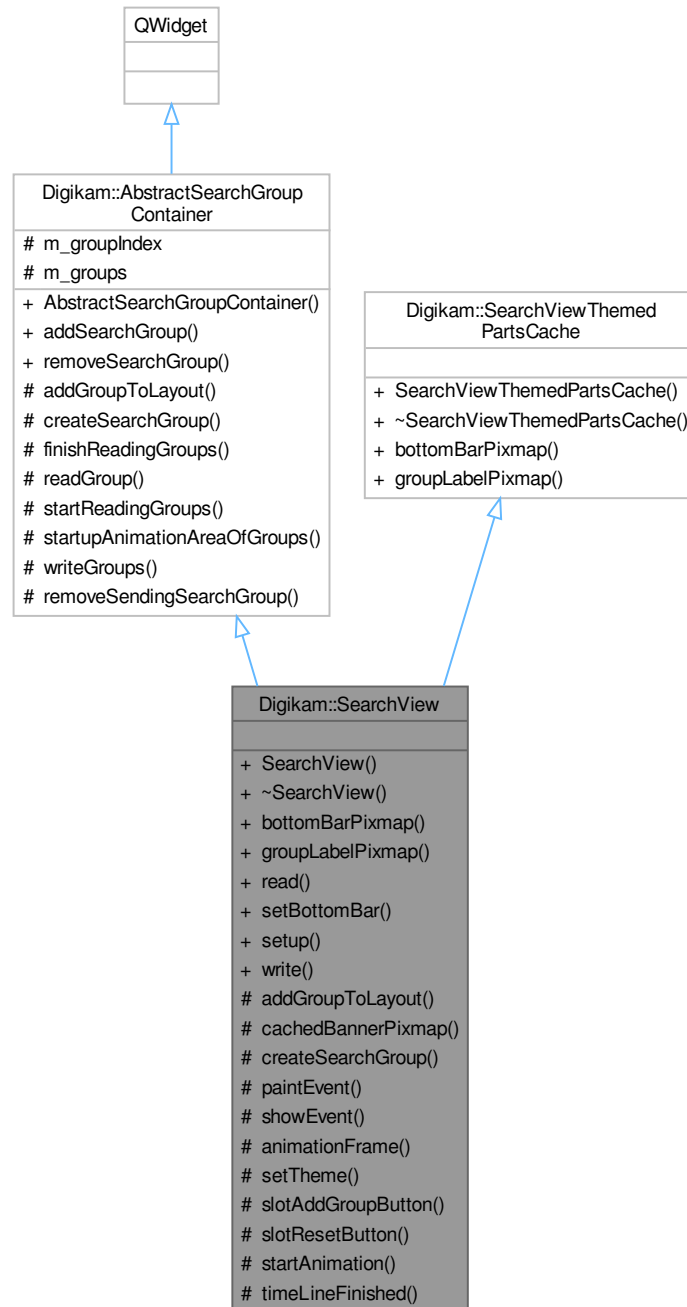
## Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- QString [entryName](#) (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup [getConfigGroup](#) () const  
*Returns the config group that must be used for state saving and loading.*



## 9.1166 Digikam::SearchView Class Reference

Inheritance diagram for Digikam::SearchView:



### Signals

- void **searchCancel** ()
- void **searchOk** ()
- void **searchTryout** ()

### Public Member Functions

- QPixmap [bottomBarPixmap](#) (int w, int h) override
- QPixmap [groupLabelPixmap](#) (int w, int h) override
- void **read** (const QString &search)
- void **setBottomBar** ([SearchViewBottomBar](#) \*const bar)
- void **setup** ()
- QString **write** () const

### Public Member Functions inherited from [Digikam::AbstractSearchGroupContainer](#)

- **AbstractSearchGroupContainer** (QWidget \*const parent=nullptr)  
*Abstract base class for classes that contain SearchGroups To contain common code of [SearchView](#) and [SearchGroup](#), as SearchGroups can have subgroups.*

### Protected Slots

- void **animationFrame** (int)
- void **setTheme** ()
- void **slotAddGroupButton** ()
- void **slotResetButton** ()
- void **startAnimation** ()
- void **timeLineFinished** ()

### Protected Slots inherited from [Digikam::AbstractSearchGroupContainer](#)

- void **removeSendingSearchGroup** ()

### Protected Member Functions

- void [addGroupToLayout](#) ([SearchGroup](#) \*group) override  
*Re-implement: Adds a newly created group to the layout structures.*
- QPixmap **cachedBannerPixmap** (int w, int h) const
- [SearchGroup](#) \* [createSearchGroup](#) () override  
*Re-implement: create and setup a search group.*
- void **paintEvent** (QPaintEvent \*e) override
- void **showEvent** (QShowEvent \*event) override

### Protected Member Functions inherited from [Digikam::AbstractSearchGroupContainer](#)

- void **finishReadingGroups** ()  
*Call when the XML part is finished.*
- void **readGroup** ([SearchXmlCachingReader](#) &reader)  
*Call when a group element is the current element.*
- void **startReadingGroups** ([SearchXmlCachingReader](#) &reader)  
*Call before reading the XML part that could contain group elements.*
- QList< QRect > **startupAnimationAreaOfGroups** () const  
*Collects the data from the same method of all contained groups (position relative to this widget)*
- void **writeGroups** ([SearchXmlWriter](#) &writer) const  
*Write contained groups to writer.*

## Additional Inherited Members

### Public Slots inherited from [Digikam::AbstractSearchGroupContainer](#)

- [SearchGroup](#) \* **addSearchGroup** ()
- void **removeSearchGroup** ([SearchGroup](#) \*group)

### Protected Attributes inherited from [Digikam::AbstractSearchGroupContainer](#)

- int **m\_groupIndex** = 0
- QList< [SearchGroup](#) \* > **m\_groups**

## 9.1166.1 Member Function Documentation

### 9.1166.1.1 addGroupToLayout()

```
void Digikam::SearchView::addGroupToLayout (
    SearchGroup * group ) [override], [protected], [virtual]
```

Implements [Digikam::AbstractSearchGroupContainer](#).

### 9.1166.1.2 bottomBarPixmap()

```
QPixmap Digikam::SearchView::bottomBarPixmap (
    int w,
    int h ) [override], [virtual]
```

Implements [Digikam::SearchViewThemedPartsCache](#).

### 9.1166.1.3 createSearchGroup()

```
SearchGroup * Digikam::SearchView::createSearchGroup ( ) [override], [protected], [virtual]
```

Implements [Digikam::AbstractSearchGroupContainer](#).

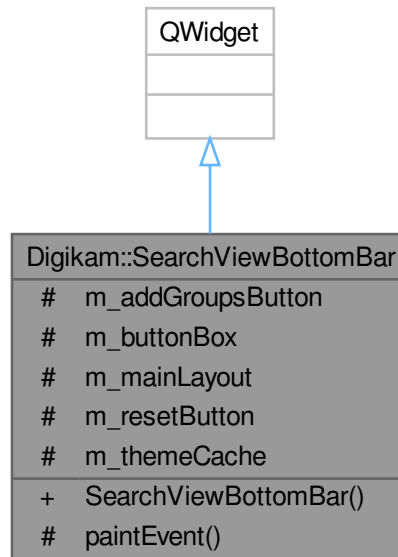
### 9.1166.1.4 groupLabelPixmap()

```
QPixmap Digikam::SearchView::groupLabelPixmap (
    int w,
    int h ) [override], [virtual]
```

Implements [Digikam::SearchViewThemedPartsCache](#).

## 9.1167 Digikam::SearchViewBottomBar Class Reference

Inheritance diagram for Digikam::SearchViewBottomBar:



### Signals

- void **addGroupPressed** ()
- void **cancelPressed** ()
- void **okPressed** ()
- void **resetPressed** ()
- void **tryoutPressed** ()

### Public Member Functions

- **SearchViewBottomBar** ([SearchViewThemedPartsCache](#) \*const cache, QWidget \*const parent=nullptr)

### Protected Member Functions

- void **paintEvent** (QPaintEvent \*) override

### Protected Attributes

- QPushButton \* **m\_addGroupsButton** = nullptr
- QDialogButtonBox \* **m\_buttonBox** = nullptr
- QHBoxLayout \* **m\_mainLayout** = nullptr
- QPushButton \* **m\_resetButton** = nullptr
- [SearchViewThemedPartsCache](#) \* **m\_themeCache** = nullptr

## 9.1168 Digikam::SearchViewThemedPartsCache Class Reference

Inheritance diagram for Digikam::SearchViewThemedPartsCache:

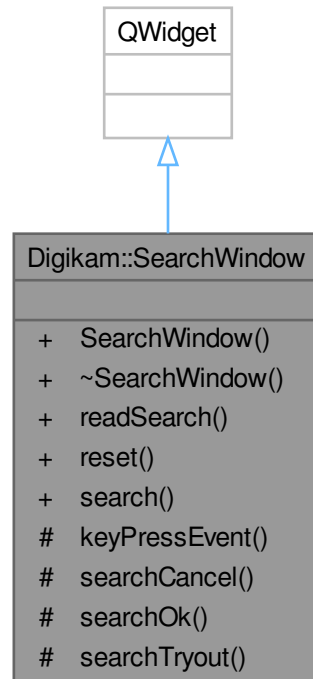


### Public Member Functions

- virtual QPixmap **bottomBarPixmap** (int w, int h)=0
- virtual QPixmap **groupLabelPixmap** (int w, int h)=0

## 9.1169 Digikam::SearchWindow Class Reference

Inheritance diagram for Digikam::SearchWindow:



### Signals

- void `searchEdited` (int id, const QString &query)  
*Signals that the user has finished editing the search.*

### Public Member Functions

- **SearchWindow** ()  
*Create a new [SearchWindow](#) with an empty advanced search.*
- void `readSearch` (int id, const QString &query)  
*Read the given search into the search widgets.*
- void `reset` ()  
*Reset the search widget to an empty search.*
- QString **search** () const  
*Returns the currently produced search string.*

### Protected Slots

- void **searchCancel** ()
- void **searchOk** ()
- void **searchTryout** ()

## Protected Member Functions

- void **keyPressEvent** (QKeyEvent \*) override

## 9.1169.1 Member Function Documentation

### 9.1169.1.1 readSearch()

```
void Digikam::SearchWindow::readSearch (
    int id,
    const QString & query )
```

The id will be emitted with the searchEdited signal.

### 9.1169.1.2 reset()

```
void Digikam::SearchWindow::reset ( )
```

Current id is -1.

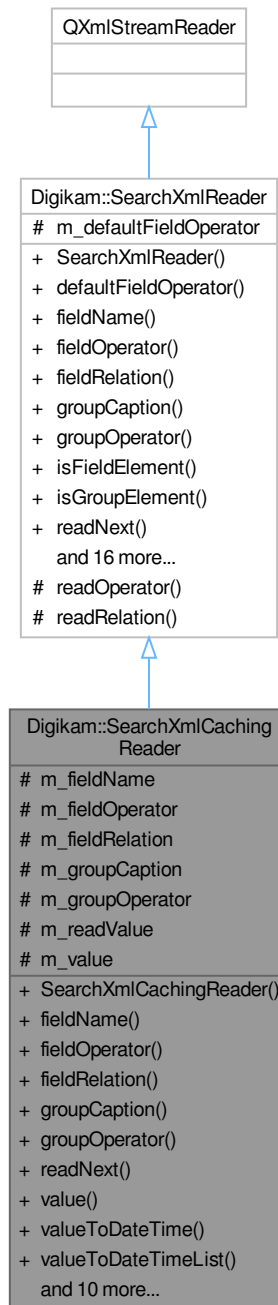
### 9.1169.1.3 searchEdited

```
void Digikam::SearchWindow::searchEdited (
    int id,
    const QString & query ) [signal]
```

The given query is the same as [search\(\)](#).

## 9.1170 Digikam::SearchXmlCachingReader Class Reference

Inheritance diagram for Digikam::SearchXmlCachingReader:



### Public Member Functions

- **SearchXmlCachingReader** (const QString &xml)



This class has the same semantics as [SearchXmlReader](#), but performs some caching and is thus much more relaxed than [SearchXmlReader](#) about the calling order of methods: With this class, you can access properties of a group until the next group is read, access properties and the value of a field until the next field is read, with all calls possible multiple times.

- QString **fieldName** () const
- SearchXml::Operator **fieldOperator** () const
- SearchXml::Relation **fieldRelation** () const
- QString **groupCaption** () const
- SearchXml::Operator **groupOperator** () const
- SearchXml::Element **readNext** ()
- QString **value** ()
- QDateTime **valueToDateTime** ()
- QList< QDateTime > **valueToDateTimeList** ()
- double **valueToDouble** ()
- QList< double > **valueToDoubleList** ()
- QList< double > **valueToDoubleOrDoubleList** ()
- int **valueToInt** ()
- QList< int > **valueToIntList** ()
- QList< int > **valueToIntOrIntList** ()
- qlonglong **valueToLongLong** ()
- QList< qlonglong > **valueToLongLongList** ()
- QStringList **valueToStringList** ()
- QList< QString > **valueToStringOrStringList** ()

## Public Member Functions inherited from [Digikam::SearchXmlReader](#)

- [SearchXmlReader](#) (const QString &xml)
- SearchXml::Operator [defaultFieldOperator](#) () const  
*Returns the default field operator.*
- QString **fieldName** () const
- SearchXml::Operator [fieldOperator](#) () const  
*Returns the field attributes.*
- SearchXml::Relation **fieldRelation** () const
- QString [groupCaption](#) () const  
*Returns the (optional) group caption.*
- SearchXml::Operator [groupOperator](#) () const  
*Returns the group operator.*
- bool **isFieldElement** () const  
*Returns if the current element is a field element (start or end element).*
- bool **isGroupElement** () const  
*Returns if the current element is a group element (start or end element).*
- SearchXml::Element [readNext](#) ()  
*Continue parsing the document.*
- void **readToEndOfElement** ()  
*General helper method: Reads XML until the end element of the current start element is reached.*
- void **readToFirstField** ()  
*General helper method: Reads XML until the first field of the next or first found group is reached.*
- bool [readToStartOfElement](#) (const QString &name)  
*General helper method: Reads XML a start element with the given name is found.*
- QString **value** ()  
*Returns the field values.*
- QDateTime **valueToDateTime** ()

- `QList< QDateTime > valueToDateTimeList ()`
- `double valueToDouble ()`
- `QList< double > valueToDoubleList ()`
- `QList< double > valueToDoubleOrDoubleList ()`
- `int valueToInt ()`
- `QList< int > valueToIntList ()`
- `QList< int > valueToIntOrIntList ()`
- `qulonglong valueToLongLong ()`
- `QList< qulonglong > valueToLongLongList ()`
- `QStringList valueToStringList ()`
- `QList< QString > valueToStringOrStringList ()`

### Protected Attributes

- `QString m_fieldName`
- `SearchXml::Operator m_fieldOperator = SearchXml::And`
- `SearchXml::Relation m_fieldRelation = SearchXml::Equal`
- `QString m_groupCaption`
- `SearchXml::Operator m_groupOperator = SearchXml::And`
- `bool m_readValue = false`
- `QVariant m_value`

### Protected Attributes inherited from [Digikam::SearchXmlReader](#)

- `SearchXml::Operator m_defaultFieldOperator`

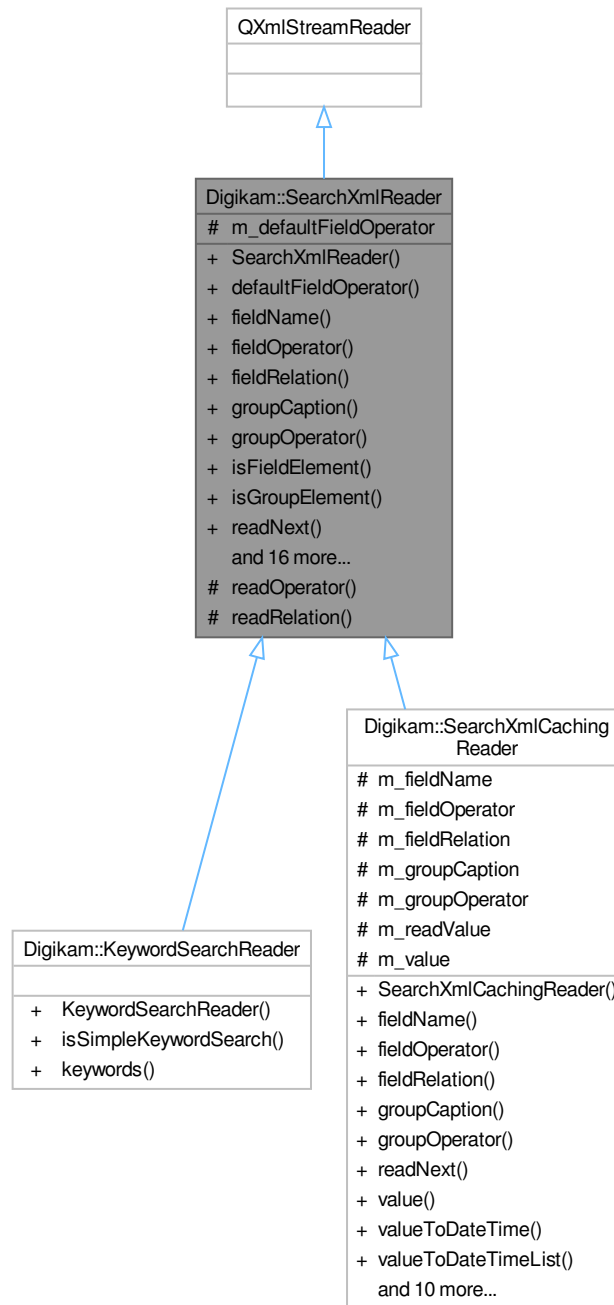
### Additional Inherited Members

### Protected Member Functions inherited from [Digikam::SearchXmlReader](#)

- `SearchXml::Operator readOperator (const QString &, SearchXml::Operator) const`
- `SearchXml::Relation readRelation (const QString &, SearchXml::Relation) const`

## 9.1171 Digikam::SearchXmlReader Class Reference

Inheritance diagram for Digikam::SearchXmlReader:



### Public Member Functions

- **SearchXmlReader** (const QString &xml)
- SearchXml::Operator [defaultFieldOperator](#) () const

- Returns the default field operator.*
- QString **fieldName** () const
- SearchXml::Operator **fieldOperator** () const
  - Returns the field attributes.*
- SearchXml::Relation **fieldRelation** () const
- QString **groupCaption** () const
  - Returns the (optional) group caption.*
- SearchXml::Operator **groupOperator** () const
  - Returns the group operator.*
- bool **isFieldElement** () const
  - Returns if the current element is a field element (start or end element).*
- bool **isGroupElement** () const
  - Returns if the current element is a group element (start or end element).*
- SearchXml::Element **readNext** ()
  - Continue parsing the document.*
- void **readToEndOfElement** ()
  - General helper method: Reads XML until the end element of the current start element is reached.*
- void **readToFirstField** ()
  - General helper method: Reads XML until the first field of the next or first found group is reached.*
- bool **readToStartOfElement** (const QString &name)
  - General helper method: Reads XML a start element with the given name is found.*
- QString **value** ()
  - Returns the field values.*
- QDateTime **valueToDateTime** ()
- QList< QDateTime > **valueToDateTimeList** ()
- double **valueToDouble** ()
- QList< double > **valueToDoubleList** ()
- QList< double > **valueToDoubleOrDoubleList** ()
- int **valueToInt** ()
- QList< int > **valueToIntList** ()
- QList< int > **valueToIntOrIntList** ()
- qlonglong **valueToLongLong** ()
- QList< qlonglong > **valueToLongLongList** ()
- QStringList **valueToStringList** ()
- QList< QString > **valueToStringOrStringList** ()

### Protected Member Functions

- SearchXml::Operator **readOperator** (const QString &, SearchXml::Operator) const
- SearchXml::Relation **readRelation** (const QString &, SearchXml::Relation) const

### Protected Attributes

- SearchXml::Operator **m\_defaultFieldOperator**

## 9.1171.1 Member Function Documentation

### 9.1171.1.1 defaultFieldOperator()

```
SearchXml::Operator Digikam::SearchXmlReader::defaultFieldOperator ( ) const
```

This operator can be overridden by a specific [fieldOperator\(\)](#).

### 9.1171.1.2 fieldOperator()

```
SearchXml::Operator Digikam::SearchXmlReader::fieldOperator ( ) const
```

Only valid if the current element is a field. fieldOperator returns the default operator if the field has not specified any.

### 9.1171.1.3 groupCaption()

```
QString Digikam::SearchXmlReader::groupCaption ( ) const
```

Only valid if the current element is a group.

### 9.1171.1.4 groupOperator()

```
SearchXml::Operator Digikam::SearchXmlReader::groupOperator ( ) const
```

Only valid if the current element is a group.

### 9.1171.1.5 readNext()

```
SearchXml::Element Digikam::SearchXmlReader::readNext ( )
```

Returns the type of the current element.

### 9.1171.1.6 readToStartOfElement()

```
bool Digikam::SearchXmlReader::readToStartOfElement (
    const QString & name )
```

The method goes to the next start element, and from there down the hierarchy, but not further up in the hierarchy. Returns false if the element is not found.

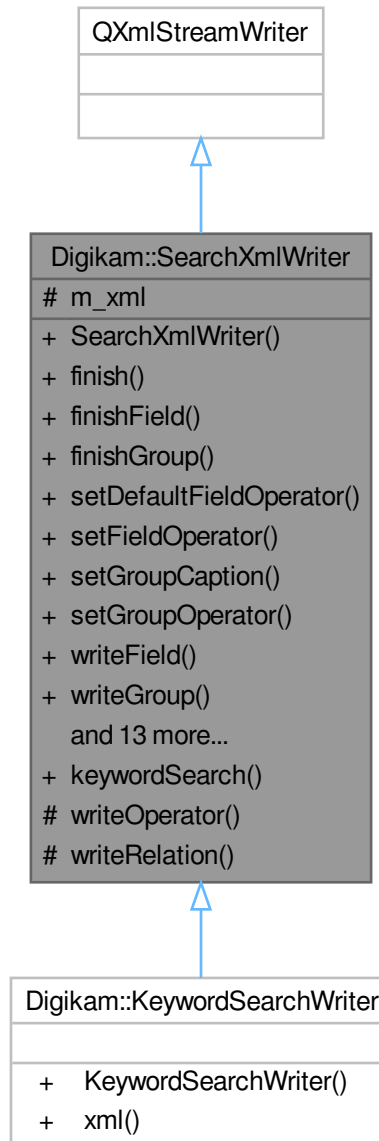
### 9.1171.1.7 value()

```
QString Digikam::SearchXmlReader::value ( )
```

Only valid if the current element is a field. This reads to the end element of the field, and converts the found text/elements to the desired output.

## 9.1172 Digikam::SearchXmlWriter Class Reference

Inheritance diagram for Digikam::SearchXmlWriter:



### Public Member Functions

- **SearchXmlWriter ()**

*Note that [SearchXmlWriter](#) and [SearchXmlGroupWriter](#) rely on you calling the methods following the restrictions set by the documentation; Otherwise you will not produce the desired output.*

- void **finish ()**

*Finish the XML.*

- void **finishField** ()  
*Finish writing the current field.*
- void **finishGroup** ()  
*Finish the current group.*
- void **setDefaultFieldOperator** (SearchXml::Operator op)  
*Sets the default operator for fields in this group "(field1 AND field2 AND ... fieldn)".*
- void **setFieldOperator** (SearchXml::Operator op)  
*Adds an optional operator overriding the default field operator of the group.*
- void **setGroupCaption** (const QString &caption)  
*Sets an optional caption.*
- void **setGroupOperator** (SearchXml::Operator op)  
*Sets the operator applied to the group as a whole "OR (field1 ... fieldn)".*
- void **writeField** (const QString &name, SearchXml::Relation relation)  
*Adds a new field with the given name (entity) and relation, "Rating less than ...".*
- void **writeGroup** ()  
*Adds a group.*
- void **writeValue** (const QDateTime &dateTime)
- void **writeValue** (const QList< double > &valueList, int precision=8)
- void **writeValue** (const QList< float > &valueList, int precision=6)
- void **writeValue** (const QList< int > &valueList)
- void **writeValue** (const QList< QDateTime > &valueList)
- void **writeValue** (const QList< qlonglong > &valueList)
- void **writeValue** (const QString &value)  
*Adds the value, "4" in the case of "Rating less than 4".*
- void **writeValue** (const QStringList &valueList)
- void **writeValue** (double value, int precision=8)
- void **writeValue** (float value, int precision=6)
- void **writeValue** (int value)
- void **writeValue** (qlonglong value)
- QString **xml** () const  
*Get the created XML.*

### Static Public Member Functions

- static QString **keywordSearch** (const QString &keyword)  
*Returns ready-made XML for a query of type "keyword" with the specified text as keyword.*

### Protected Member Functions

- void **writeOperator** (const QString &, SearchXml::Operator)
- void **writeRelation** (const QString &, SearchXml::Relation)

### Protected Attributes

- QString **m\_xml**

## 9.1172.1 Member Function Documentation

### 9.1172.1.1 finish()

```
void Digikam::SearchXmlWriter::finish ( )
```

No further group can be added after calling this. You need to call this before you can get the resulting XML from [xml\(\)](#).

### 9.1172.1.2 finishField()

```
void Digikam::SearchXmlWriter::finishField ( )
```

You shall call this method before adding another field, or closing the group.

### 9.1172.1.3 finishGroup()

```
void Digikam::SearchXmlWriter::finishGroup ( )
```

You cannot add anymore fields after calling this. Note that you will want to call this before writing another group if you want the group on the same level. You can as well add nested groups and call this to close the group afterwards.

### 9.1172.1.4 setDefaultFieldOperator()

```
void Digikam::SearchXmlWriter::setDefaultFieldOperator (
    SearchXml::Operator op )
```

The default operator can in each field be overridden. Default value is AND.

### 9.1172.1.5 setGroupOperator()

```
void Digikam::SearchXmlWriter::setGroupOperator (
    SearchXml::Operator op )
```

Default value is OR.

### 9.1172.1.6 writeField()

```
void Digikam::SearchXmlWriter::writeField (
    const QString & name,
    SearchXml::Relation relation )
```

Ensure that you closed the previous field with [finishField\(\)](#). For a reference of valid field names, look into [ItemQueryBuilder](#). The general rule is that names are like the database fields, but all lower-case.



**9.1172.1.7 writeGroup()**

```
void Digikam::SearchXmlWriter::writeGroup ( )
```

Use the returned group writer to add fields.

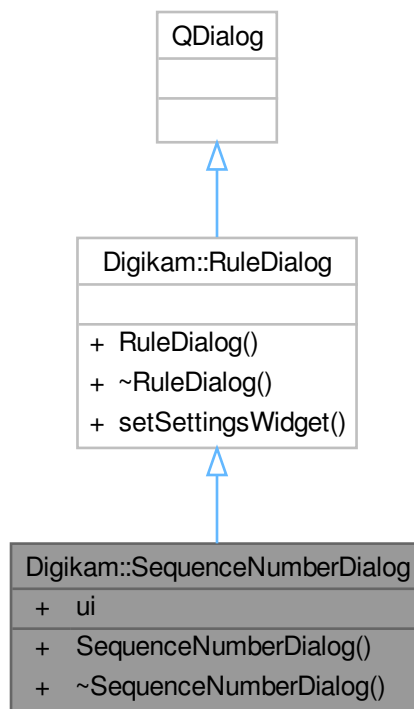
**9.1172.1.8 xml()**

```
QString Digikam::SearchXmlWriter::xml ( ) const
```

The value is only valid if [finish\(\)](#) has been called.

**9.1173 Digikam::SequenceNumberDialog Class Reference**

Inheritance diagram for Digikam::SequenceNumberDialog:

**Public Member Functions**

- `SequenceNumberDialog` ([Rule](#) \*const parent)

**Public Member Functions inherited from [Digikam::RuleDialog](#)**

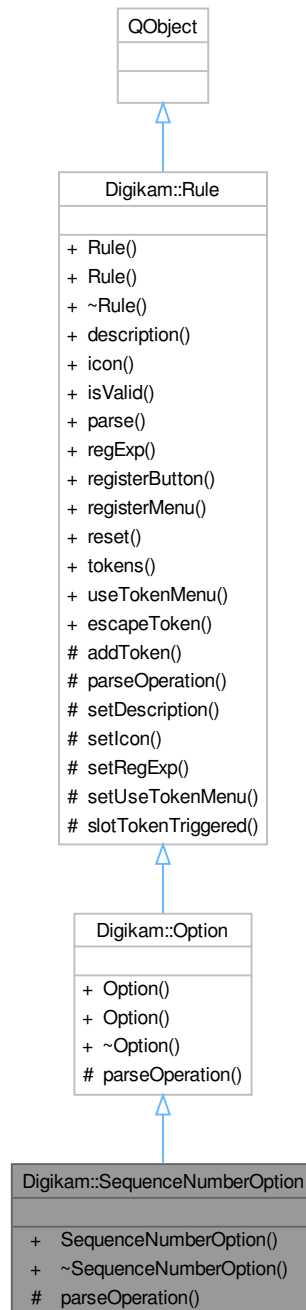
- **RuleDialog** ([Rule](#) \*const parent)
- void **setSettingsWidget** (QWidget \*const settingsWidget)

**Public Attributes**

- Ui::SequenceNumberOptionDialogWidget \*const **ui** = nullptr

## 9.1174 Digikam::SequenceNumberOption Class Reference

Inheritance diagram for Digikam::SequenceNumberOption:



### Protected Member Functions

- QString [parseOperation](#) ([ParseSettings](#) &settings, const QRegularExpressionMatch &match) override  
*TODO: describe me.*

## Protected Member Functions inherited from [Digikam::Rule](#)

- bool [addToken](#) (const QString &id, const QString &description, const QString &actionName=QString())  
*add a token to the parser, every parser should at least assign one token object*
- void [setDescription](#) (const QString &desc)
- void [setIcon](#) (const QString &pixmap)
- void [setRegExp](#) (const QRegularExpression &regExp)
- void [setUseTokenMenu](#) (bool value)  
*If multiple tokens have been assigned to a rule, a menu will be created.*

## Additional Inherited Members

## Public Types inherited from [Digikam::Rule](#)

- enum [IconType](#) { [Action](#) = 0 , [Dialog](#) }

## Signals inherited from [Digikam::Rule](#)

- void [signalTokenTriggered](#) (const QString &)

## Public Member Functions inherited from [Digikam::Option](#)

- [Option](#) (const QString &name, const QString &description)
- [Option](#) (const QString &name, const QString &description, const QString &icon)

## Public Member Functions inherited from [Digikam::Rule](#)

- [Rule](#) (const QString &name)
- [Rule](#) (const QString &name, const QString &icon)
- QString [description](#) () const
- QPixmap [icon](#) (Rule::IconType type=Rule::Action) const
- bool [isValid](#) () const  
*Checks the validity of the parse object.*
- [ParseResults](#) [parse](#) ([ParseSettings](#) &settings)
- QRegularExpression & [regExp](#) () const  
*TODO: This is probably not needed anymore.*
- QPushButton \* [registerButton](#) (QWidget \*parent)  
*Register a button in the parent object.*
- QAction \* [registerMenu](#) (QMenu \*parent)  
*Register a menu action in the parent object.*
- virtual void [reset](#) ()  
*Resets the parser to its initial state.*
- TokenList & [tokens](#) () const
- bool [useTokenMenu](#) () const  
*Returns true if a token menu is used.*

## Static Public Member Functions inherited from [Digikam::Rule](#)

- static QString [escapeToken](#) (const QString &token)  
*Escape the token characters to make them work in regular expressions.*

## Protected Slots inherited from [Digikam::Rule](#)

- virtual void [slotTokenTriggered](#) (const QString &)

## 9.1174.1 Member Function Documentation

### 9.1174.1.1 [parseOperation\(\)](#)

```
QString Digikam::SequenceNumberOption::parseOperation (
    ParseSettings & settings,
    const QRegularExpressionMatch & match ) [override], [protected], [virtual]
```

#### Parameters

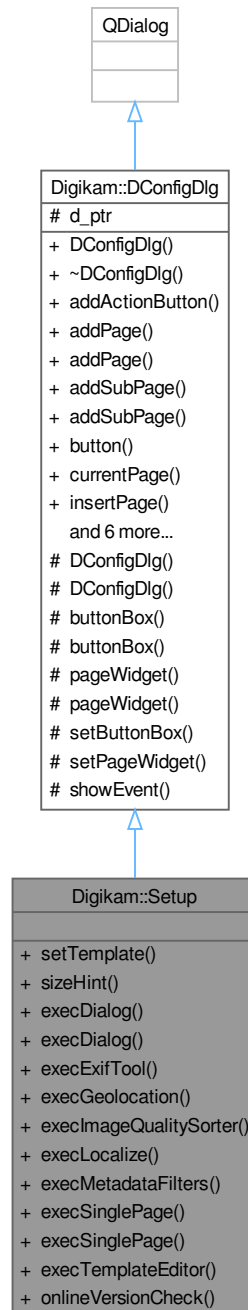
<i>settings</i>	contains settings
<i>match</i>	result of the regular expression match done in <a href="#">Option::parse()</a>

#### Returns

Implements [Digikam::Option](#).

## 9.1175 Digikam::Setup Class Reference

Inheritance diagram for Digikam::Setup:



### Public Types

- enum **Page** {  
**LastPageUsed** = -1 , **DatabasePage** = 0 , **CollectionsPage** , **AlbumViewPage** ,

**ToolTipPage** , **MetadataPage** , **TemplatePage** , **EditorPage** ,  
**ICCPage** , **LightTablePage** , **GeolocationPage** , **ImageQualityPage** ,  
**CameraPage** , **PluginsPage** , **MiscellaneousPage** , **SetupPageEnumLast** }

## Public Types inherited from [Digikam::DConfigDlg](#)

- enum [FaceType](#) {  
**Auto** = DConfigDlgView::Auto , **Plain** = DConfigDlgView::Plain , **List** = DConfigDlgView::List , **Tree** =  
DConfigDlgView::Tree ,  
**Tabbed** = DConfigDlgView::Tabbed }

## Public Member Functions

- void **setTemplate** (const [Template](#) &t)
- QSize **sizeHint** () const override

## Public Member Functions inherited from [Digikam::DConfigDlg](#)

- **DConfigDlg** (QWidget \*const parent=nullptr, Qt::WindowFlags flags=Qt::WindowFlags())  
*Creates a new page dialog.*
- **~DConfigDlg** () override  
*Destroys the page dialog.*
- void **addActionButton** (QAbstractButton \*const [button](#))  
*Set an action button.*
- void **addPage** ([DConfigDlgWdgItem](#) \*const item)  
*Adds a new top level page to the dialog.*
- [DConfigDlgWdgItem](#) \* **addPage** (QWidget \*const widget, const QString &name)  
*Adds a new top level page to the dialog.*
- void **addSubPage** ([DConfigDlgWdgItem](#) \*const parent, [DConfigDlgWdgItem](#) \*const item)  
*Inserts a new sub page in the dialog.*
- [DConfigDlgWdgItem](#) \* **addSubPage** ([DConfigDlgWdgItem](#) \*const parent, QWidget \*const widget, const  
QString &name)  
*Inserts a new sub page in the dialog.*
- QPushButton \* **button** (QDialogButtonBox::StandardButton which) const  
*Returns the QPushButton corresponding to the standard button which, or 0 if the standard button doesn't exist in this  
dialog.*
- [DConfigDlgWdgItem](#) \* **currentPage** () const  
*Returns the.*
- void **insertPage** ([DConfigDlgWdgItem](#) \*const before, [DConfigDlgWdgItem](#) \*const item)  
*Inserts a new page in the dialog.*
- [DConfigDlgWdgItem](#) \* **insertPage** ([DConfigDlgWdgItem](#) \*const before, QWidget \*const widget, const  
QString &name)  
*Inserts a new page in the dialog.*
- void **removePage** ([DConfigDlgWdgItem](#) \*const item)  
*Removes the page associated with the given.*
- void **setConfigGroup** (const QString &group)  
*Sets the config group name for restore or save dialog window size.*
- void **setCurrentPage** ([DConfigDlgWdgItem](#) \*const item)  
*Sets the page which is associated with the given.*
- void **setFaceType** ([FaceType](#) faceType)  
*Sets the face type of the dialog.*
- void **setStandardButtons** (QDialogButtonBox::StandardButtons buttons)  
*Sets the collection of standard buttons displayed by this dialog.*

## Static Public Member Functions

- static bool [execDialog](#) (Page page=LastPageUsed)  
*Show a setup dialog.*
- static bool **execDialog** (QWidget \*const parent, Page page=LastPageUsed)
- static bool **execExifTool** (QWidget \*const parent)
- static bool **execGeolocation** (QWidget \*const parent, int tab)
- static bool **execImageQualitySorter** (QWidget \*const parent)
- static bool **execLocalize** (QWidget \*const parent)
- static bool **execMetadataFilters** (QWidget \*const parent, int tab)
- static bool [execSinglePage](#) (Page page)  
*Show a setup dialog.*
- static bool **execSinglePage** (QWidget \*const parent, Page page)
- static bool **execTemplateEditor** (QWidget \*const parent, const [Template](#) &t)
- static void **onlineVersionCheck** ()

## Additional Inherited Members

### Signals inherited from [Digikam::DConfigDlg](#)

- void [currentPageChanged](#) ([DConfigDlgWdgItem](#) \*current, [DConfigDlgWdgItem](#) \*before)  
*This signal is emitted whenever the current page has changed.*
- void [pageRemoved](#) ([DConfigDlgWdgItem](#) \*page)  
*This signal is emitted whenever a page has been removed.*

### Protected Member Functions inherited from [Digikam::DConfigDlg](#)

- **DConfigDlg** ([DConfigDlgPrivate](#) &dd, [DConfigDlgWdg](#) \*const widget, QWidget \*const parent, Qt::Window↔Flags flags=Qt::WindowFlags())
- [DConfigDlg](#) ([DConfigDlgWdg](#) \*const widget, QWidget \*const parent, Qt::WindowFlags flags=Qt::Window↔Flags())  
*This constructor can be used by subclasses to provide a custom page widget.*
- QDialogButtonBox \* **buttonBox** ()  
*Returns the button box of the dialog or 0 if no button box is set.*
- const QDialogButtonBox \* **buttonBox** () const  
*Returns the button box of the dialog or 0 if no button box is set.*
- [DConfigDlgWdg](#) \* **pageWidget** ()  
*Returns the page widget of the dialog or 0 if no page widget is set.*
- const [DConfigDlgWdg](#) \* **pageWidget** () const  
*Returns the page widget of the dialog or 0 if no page widget is set.*
- void [setButtonBox](#) (QDialogButtonBox \*const box)  
*Set the button box of the dialog.*
- void [setPageWidget](#) ([DConfigDlgWdg](#) \*const widget)  
*Set the page widget of the dialog.*
- void **showEvent** (QShowEvent \*) override

### Protected Attributes inherited from [Digikam::DConfigDlg](#)

- [DConfigDlgPrivate](#) \*const **d\_ptr** = nullptr



## 9.1175.1 Member Function Documentation

### 9.1175.1.1 execDialog()

```
bool Digikam::Setup::execDialog (
    Page page = LastPageUsed ) [static]
```

The specified page will be selected. True is returned if the dialog was closed with Ok.

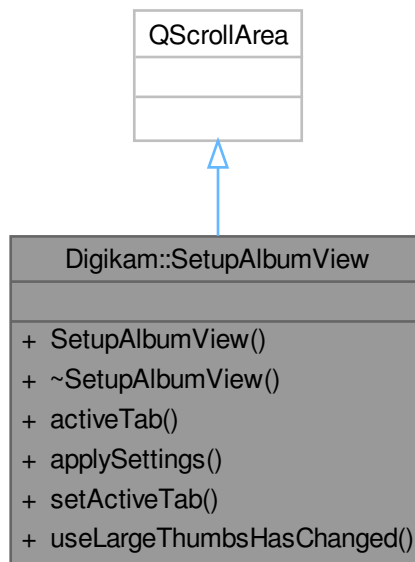
### 9.1175.1.2 execSinglePage()

```
bool Digikam::Setup::execSinglePage (
    Page page ) [static]
```

Only the specified page will be available.

## 9.1176 Digikam::SetupAlbumView Class Reference

Inheritance diagram for Digikam::SetupAlbumView:



### Public Types

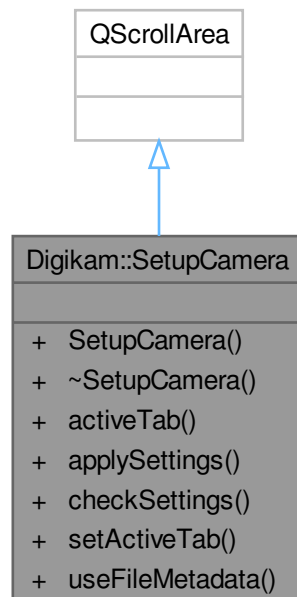
- enum **AlbumTab** {  
**IconView** = 0 , **FolderView** , **Preview** , **FullScreen** ,  
**MimeType** , **Category** }

## Public Member Functions

- **SetupAlbumView** (QWidget \*const parent=nullptr)
- AlbumTab **activeTab** () const
- void **applySettings** ()
- void **setActiveTab** (AlbumTab tab)
- bool **useLargeThumbsHasChanged** () const

## 9.1177 Digikam::SetupCamera Class Reference

Inheritance diagram for Digikam::SetupCamera:



## Public Types

- enum **CameraTab** { **Devices** = 0 , **Behavior** , **ImportFilters** , **ImportWindow** }
- enum **ConflictRule** { **OVERWRITE** = 0 , **DIFFNAME** , **SKIPFILE** }

## Signals

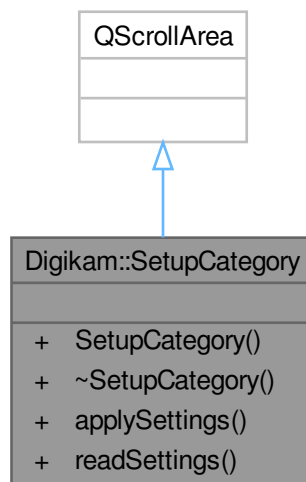
- void **signalUseFileMetadataChanged** (bool)

### Public Member Functions

- **SetupCamera** (QWidget \*const parent=nullptr)
- CameraTab **activeTab** () const
- void **applySettings** ()
- bool **checkSettings** ()
- void **setActiveTab** (CameraTab tab)
- bool **useFileMetadata** ()

## 9.1178 Digikam::SetupCategory Class Reference

Inheritance diagram for Digikam::SetupCategory:

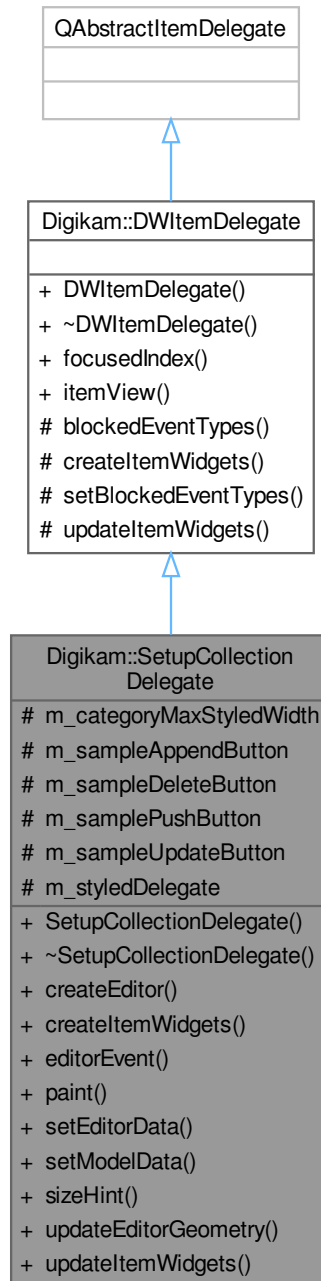


### Public Member Functions

- **SetupCategory** (QWidget \*const parent=nullptr)
- void **applySettings** ()
- void **readSettings** ()

## 9.1179 Digikam::SetupCollectionDelegate Class Reference

Inheritance diagram for Digikam::SetupCollectionDelegate:



### Signals

- void **appendPressed** (int mappedId) const
- void **categoryButtonPressed** (int mappedId) const
- void **deletePressed** (int mappedId) const
- void **updatePressed** (int mappedId) const

## Public Member Functions

- **SetupCollectionDelegate** (QAbstractItemView \*const view, QObject \*const parent=nullptr)
- QWidget \* **createEditor** (QWidget \*parent, const QStyleOptionViewItem &option, const QModelIndex &index) const override
- QList< QWidget \* > **createItemWidgets** (const QModelIndex &index) const override  
*Creates the list of widgets needed for an item.*
- bool **editorEvent** (QEvent \*event, QAbstractItemModel \*model, const QStyleOptionViewItem &option, const QModelIndex &index) override
- void **paint** (QPainter \*painter, const QStyleOptionViewItem &option, const QModelIndex &index) const override
- void **setEditorData** (QWidget \*editor, const QModelIndex &index) const override
- void **setModelData** (QWidget \*editor, QAbstractItemModel \*model, const QModelIndex &index) const override
- QSize **sizeHint** (const QStyleOptionViewItem &option, const QModelIndex &index) const override
- void **updateEditorGeometry** (QWidget \*editor, const QStyleOptionViewItem &option, const QModelIndex &index) const override
- void **updateItemWidgets** (const QList< QWidget \* > &widgets, const QStyleOptionViewItem &option, const QPersistentModelIndex &index) const override  
*Updates a list of widgets for its use inside of the delegate (painting or event handling).*

## Public Member Functions inherited from Digikam::DWItemDelegate

- **DWItemDelegate** (QAbstractItemView \*const itemView, QObject \*const parent=nullptr)  
*Creates a new [ItemDelegate](#) to be used with a given itemview.*
- QPersistentModelIndex **focusedIndex** () const  
*Retrieves the currently focused index.*
- QAbstractItemView \* **itemView** () const  
*Retrieves the item view this delegate is monitoring.*

## Protected Attributes

- int **m\_categoryMaxStyledWidth** = 0
- QToolButton \* **m\_sampleAppendButton** = nullptr
- QToolButton \* **m\_sampleDeleteButton** = nullptr
- QPushButton \* **m\_samplePushButton** = nullptr
- QToolButton \* **m\_sampleUpdateButton** = nullptr
- StyledItemDelegate \* **m\_styledDelegate** = nullptr

## Additional Inherited Members

## Protected Member Functions inherited from Digikam::DWItemDelegate

- QList< QEvent::Type > **blockedEventTypes** (QWidget \*const widget) const  
*Retrieves the list of blocked event types for the given widget.*
- void **setBlockedEventTypes** (QWidget \*const widget, const QList< QEvent::Type > &types) const  
*Sets the list of event types that a widget will block.*

## 9.1179.1 Member Function Documentation

### 9.1179.1.1 createItemWidgets()

```
QList< QWidget * > Digikam::SetupCollectionDelegate::createItemWidgets (
    const QModelIndex & index ) const [override], [virtual]
```

#### Note

No initialization of the widgets is supposed to happen here. The widgets will be initialized based on needs for a given item.

If you want to connect some widget signals to any slot, you should do it here.

- index the index to create widgets for.

#### Note

If you want to know the index for which you are creating widgets, it is available as a QModelIndex Q\_↔ PROPERTY called "goya:creatingWidgetsForIndex". Ensure to add Q\_DECLARE\_METATYPE(QModelIndex) before your method definition to tell QVariant about QModelIndex.

#### Returns

the list of newly created widgets which will be used to interact with an item.

#### See also

[updateItemWidgets\(\)](#)

Implements [Digikam::DWItemDelegate](#).

### 9.1179.1.2 updateItemWidgets()

```
void Digikam::SetupCollectionDelegate::updateItemWidgets (
    const QList< QWidget * > & widgets,
    const QStyleOptionViewItem & option,
    const QPersistentModelIndex & index ) const [override], [virtual]
```

#### Note

All the positioning and sizing should be done in item coordinates.

#### Warning

Do not make widget connections in here, since this method will be called very regularly.

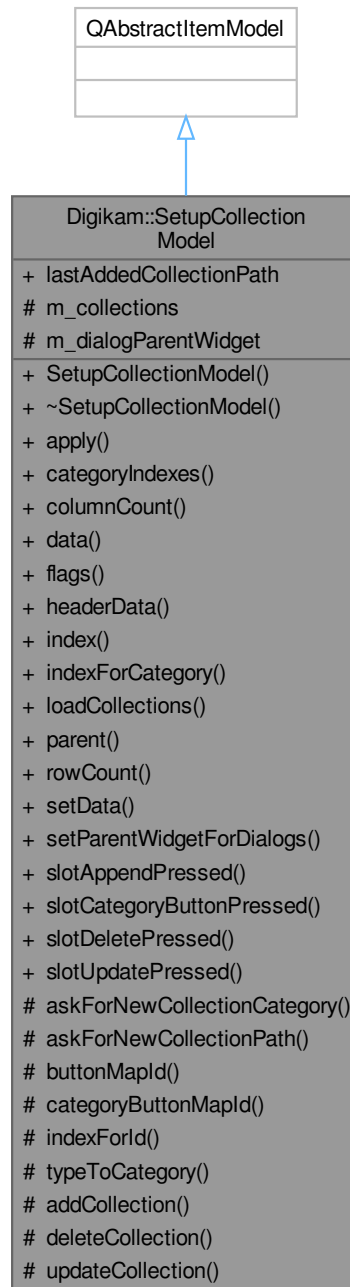
## Parameters

<i>widgets</i>	the widgets to update
<i>option</i>	the current set of style options for the view.
<i>index</i>	the model index of the item currently manipulated.

Implements [Digikam::DWItemDelegate](#).

## 9.1180 Digikam::SetupCollectionModel Class Reference

Inheritance diagram for Digikam::SetupCollectionModel:



### Classes

- class [Item](#)



## Public Types

- enum **Category** { **CategoryLocal** = 0 , **CategoryRemovable** = 1 , **CategoryRemote** = 2 , **NumberOfCategories** }
- enum **Columns** { **ColumnStatus** = 0 , **ColumnName** = 1 , **ColumnPath** = 2 , **ColumnAppendButton** = 3 , **ColumnUpdateButton** = 4 , **ColumnDeleteButton** = 5 , **NumberOfColumns** }
- enum **SetupCollectionDataRole** { **IsCategoryRole** = Qt::UserRole , **CategoryButtonDisplayRole** = Qt::UserRole + 1 , **CategoryButtonMapId** = Qt::UserRole + 2 , **IsAppendRole** = Qt::UserRole + 3 , **AppendDecorationRole** = Qt::UserRole + 4 , **AppendMapId** = Qt::UserRole + 5 , **IsUpdateRole** = Qt::UserRole + 6 , **UpdateDecorationRole** = Qt::UserRole + 7 , **UpdateMapId** = Qt::UserRole + 8 , **IsDeleteRole** = Qt::UserRole + 9 , **DeleteDecorationRole** = Qt::UserRole + 10 , **DeleteMapId** = Qt::UserRole + 11 }

*SetupCollectionModel* is a model specialized for use in *SetupCollectionTreeView*.

## Public Slots

- void **slotAppendPressed** (int mappedId)  
*Forward button clicked signals to this slot.*
- void **slotCategoryButtonPressed** (int mappedId)  
*Forward category button clicked signals to this slot.*
- void **slotDeletePressed** (int mappedId)
- void **slotUpdatePressed** (int mappedId)

## Signals

- void **collectionsLoaded** ()  
*Emitted when all collections were loaded and the model reset in loadCollections.*

## Public Member Functions

- **SetupCollectionModel** (QObject \*const parent=nullptr)  
*Internal data structure:*
- void **apply** ()  
*Apply the changed settings to [CollectionManager](#).*
- QList< QModelIndex > **categoryIndexes** () const
- int **columnCount** (const QModelIndex &parent=QModelIndex()) const override
- QVariant **data** (const QModelIndex &index, int role=Qt::DisplayRole) const override  
*QAbstractItemModel implementation.*
- Qt::ItemFlags **flags** (const QModelIndex &index) const override
- QVariant **headerData** (int section, Qt::Orientation orientation, int role=Qt::DisplayRole) const override
- QModelIndex **index** (int row, int column, const QModelIndex &parent=QModelIndex()) const override
- QModelIndex **indexForCategory** (Category category) const
- void **loadCollections** ()  
*Read collections from [CollectionManager](#).*
- QModelIndex **parent** (const QModelIndex &index) const override
- int **rowCount** (const QModelIndex &parent=QModelIndex()) const override
- bool **setData** (const QModelIndex &index, const QVariant &value, int role=Qt::EditRole) override
- void **setParentWidgetForDialogs** (QWidget \*const widget)  
*Set a widget used as parent for dialogs and message boxes.*

## Public Attributes

- QString **lastAddedCollectionPath**

## Protected Slots

- void **addCollection** (int category)
- void **deleteCollection** (int internalId)
- void **updateCollection** (int internalId)

## Protected Member Functions

- bool **askForNewCollectionCategory** (int \*const category)
- bool **askForNewCollectionPath** (bool adding, int category, QString \*const newPath, QString \*const newLabel)
- int **buttonMapId** (const QModelIndex &index) const
- int **categoryButtonMapId** (const QModelIndex &index) const
- QModelIndex **indexForId** (int id, int column) const

## Static Protected Member Functions

- static Category **typeToCategory** ([CollectionLocation::Type](#) type)

## Protected Attributes

- QList< [Item](#) > **m\_collections**
- QWidget \* **m\_dialogParentWidget** = nullptr

## 9.1180.1 Member Enumeration Documentation

### 9.1180.1.1 SetupCollectionDataRole

enum [Digikam::SetupCollectionModel::SetupCollectionDataRole](#)

It provides a reads the current collections from [CollectionManager](#), displays them in three categories, and supports adding and removing collections

#### Enumerator

IsCategoryRole	Returns true if the model index is the index of a category.
CategoryButtonDisplayRole	The text for the category button.
IsAppendRole	Returns true if the model index is the index of a button.
AppendDecorationRole	The pixmap of the button.
IsUpdateRole	Returns true if the model index is the index of a button.
UpdateDecorationRole	The pixmap of the button.
IsDeleteRole	Returns true if the model index is the index of a button.
DeleteDecorationRole	The pixmap of the button.

## 9.1180.2 Constructor & Destructor Documentation

### 9.1180.2.1 SetupCollectionModel()

```
Digikam::SetupCollectionModel::SetupCollectionModel (
    QObject *const parent = nullptr ) [explicit]
```

The category entries get a model index with INTERNALID and are identified by their row(). The item entries get the index in m\_collections as INTERNALID. No item is ever removed from m\_collections, deleted entries are only marked as such.

Items have a location, a parentId, and a name and label field. parentId always contains the category, needed to implement parent(). The location is the location if it exists, or is null if the item was added. Name and label are null if unchanged, then the values from location are used. They are valid if edited (label) or the location was added (both valid, location null).

## 9.1180.3 Member Function Documentation

### 9.1180.3.1 slotAppendPressed

```
void Digikam::SetupCollectionModel::slotAppendPressed (
    int mappedId ) [slot]
```

mappedId is retrieved with the ButtonMapId role for the model index of the button

### 9.1180.3.2 slotCategoryButtonPressed

```
void Digikam::SetupCollectionModel::slotCategoryButtonPressed (
    int mappedId ) [slot]
```

mappedId is retrieved with the CategoryButtonMapId role for the model index of the button

## 9.1181 Digikam::SetupCollectionModel::Item Class Reference

### Public Member Functions

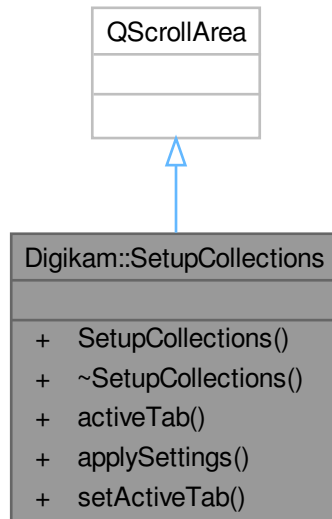
- **Item** (const [CollectionLocation](#) &location)
- **Item** (const QString &path, const QString &label, SetupCollectionModel::Category category)

### Public Attributes

- bool **appended** = false
- QStringList **childs**
- bool **deleted** = false
- QString **label**
- [CollectionLocation](#) **location**
- int **orgIndex** = 0
- int **parentId** = 0
- QString **path**
- bool **updated** = false

## 9.1182 Digikam::SetupCollections Class Reference

Inheritance diagram for Digikam::SetupCollections:



### Public Types

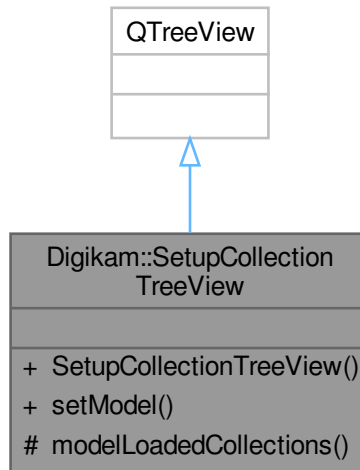
- enum `CollectionsTab` { `Collections = 0` , `IgnoreDirs` }

### Public Member Functions

- `SetupCollections` (`QWidget *const parent=nullptr`)
- `CollectionsTab activeTab` () const
- void `applySettings` ()
- void `setActiveTab` (`CollectionsTab tab`)

## 9.1183 Digikam::SetupCollectionTreeView Class Reference

Inheritance diagram for Digikam::SetupCollectionTreeView:



### Public Member Functions

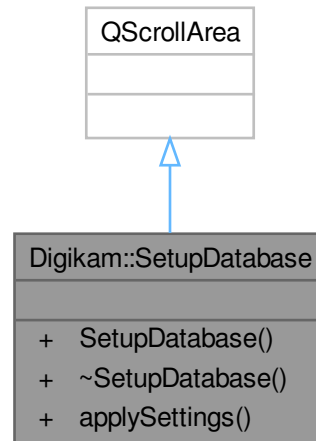
- **SetupCollectionTreeView** (`QWidget *const parent=nullptr`)
- void **setModel** (`SetupCollectionModel *model`)

### Protected Slots

- void **modelLoadedCollections** ()

## 9.1184 Digikam::SetupDatabase Class Reference

Inheritance diagram for Digikam::SetupDatabase:

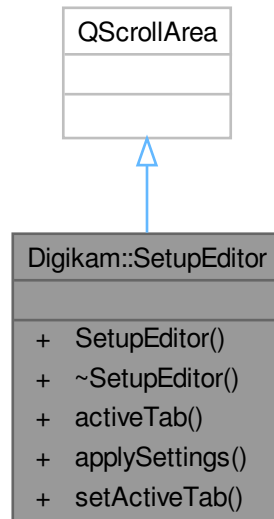


### Public Member Functions

- **SetupDatabase** (QWidget \*const parent=nullptr)
- void **applySettings** ()

## 9.1185 Digikam::SetupEditor Class Reference

Inheritance diagram for Digikam::SetupEditor:



### Public Types

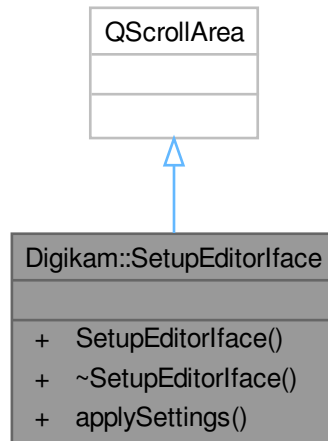
- enum **EditorTab** {  
    **EditorWindow** = 0 , **Versioning** , **SaveSettings** , **RAWBehavior** ,  
    **RAWDefaultSettings** }

### Public Member Functions

- **SetupEditor** (QWidget \*const parent=nullptr)
- EditorTab **activeTab** () const
- void **applySettings** ()
- void **setActiveTab** (EditorTab tab)

## 9.1186 Digikam::SetupEditorIface Class Reference

Inheritance diagram for Digikam::SetupEditorIface:



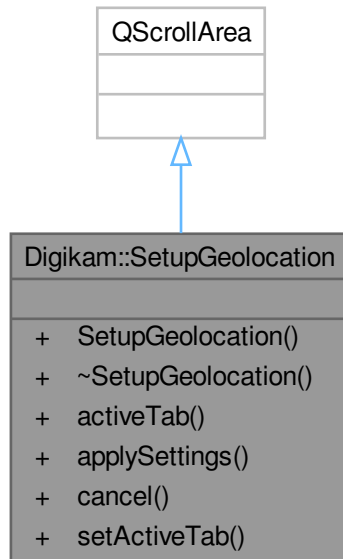
### Public Member Functions

- **SetupEditorIface** (`QWidget *const parent=nullptr`)
- void **applySettings** ()



## 9.1187 Digikam::SetupGeolocation Class Reference

Inheritance diagram for Digikam::SetupGeolocation:



### Public Types

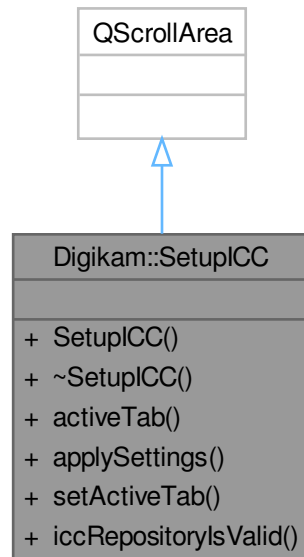
- enum **GeolocationTab** { **MarbleView** = 0 , **MarblePlugins** , **GoogleMaps** }

### Public Member Functions

- **SetupGeolocation** (QWidget \*const parent=nullptr)
- GeolocationTab **activeTab** () const
- void **applySettings** ()
- void **cancel** ()
- void **setActiveTab** (GeolocationTab tab)

## 9.1188 Digikam::SetupICC Class Reference

Inheritance diagram for Digikam::SetupICC:



### Public Types

- enum `ICCTab` { `Behavior = 0`, `Profiles`, `Advanced` }

### Public Member Functions

- [SetupICC](#) (`QDialogButtonBox *const dlgBtnBox`, `QWidget *const parent=nullptr`)
- `ICCTab activeTab () const`
- void `applySettings ()`
- void `setActiveTab (ICCTab tab)`

### Static Public Member Functions

- static bool `iccRepositoryIsValid ()`

## 9.1188.1 Constructor & Destructor Documentation

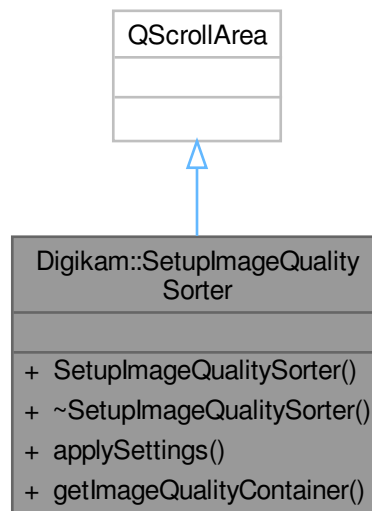
### 9.1188.1.1 SetupICC()

```

Digikam::SetupICC::SetupICC (
    QDialogButtonBox *const dlgBtnBox,
    QWidget *const parent = nullptr ) [explicit]
  
```

## 9.1189 Digikam::SetupImageQualitySorter Class Reference

Inheritance diagram for Digikam::SetupImageQualitySorter:

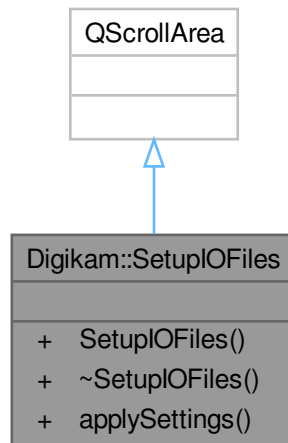


### Public Member Functions

- **SetupImageQualitySorter** (QWidget \*const parent=nullptr)
- void **applySettings** ()
- [ImageQualityContainer](#) **getImageQualityContainer** () const

## 9.1190 Digikam::SetupIOFiles Class Reference

Inheritance diagram for Digikam::SetupIOFiles:

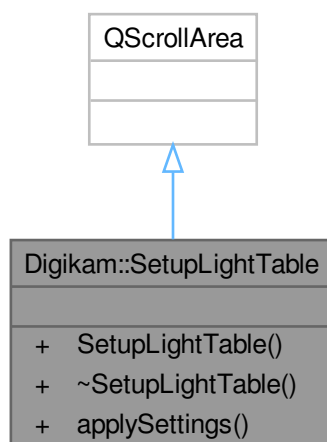


### Public Member Functions

- **SetupIOFiles** (QWidget \*const parent=nullptr)
- void **applySettings** ()

## 9.1191 Digikam::SetupLightTable Class Reference

Inheritance diagram for Digikam::SetupLightTable:

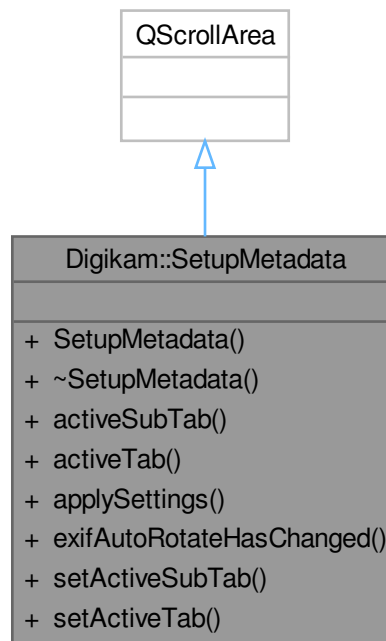


**Public Member Functions**

- **SetupLightTable** (QWidget \*const parent=nullptr)
- void **applySettings** ()

**9.1192 Digikam::SetupMetadata Class Reference**

Inheritance diagram for Digikam::SetupMetadata:

**Public Types**

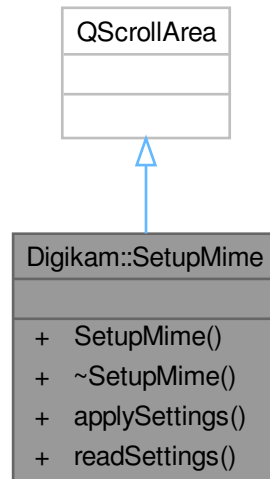
- enum **MetadataSubTab** {  
**ExifViewer** = 0 , **MakernotesViewer** , **IPTCViewer** , **XMPViewer** ,  
**ExifToolViewer** }
- enum **MetadataTab** {  
**Behavior** = 0 , **Sidecars** , **Rotation** , **Display** ,  
**ExifTool** , **Baloo** , **AdvancedConfig** }

**Public Member Functions**

- **SetupMetadata** (QWidget \*const parent=nullptr)
- MetadataSubTab **activeSubTab** () const
- MetadataTab **activeTab** () const
- void **applySettings** ()
- bool **exifAutoRotateHasChanged** () const
- void **setActiveSubTab** (MetadataSubTab tab)
- void **setActiveTab** (MetadataTab tab)

## 9.1193 Digikam::SetupMime Class Reference

Inheritance diagram for Digikam::SetupMime:

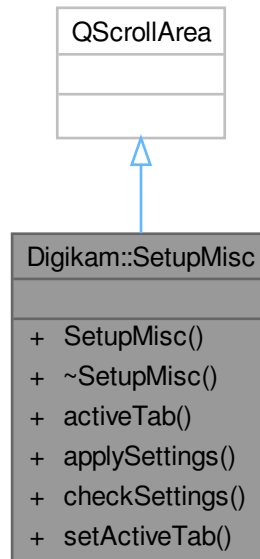


### Public Member Functions

- **SetupMime** (QWidget \*const parent=nullptr)
- void **applySettings** ()
- void **readSettings** ()

## 9.1194 Digikam::SetupMisc Class Reference

Inheritance diagram for Digikam::SetupMisc:



### Public Types

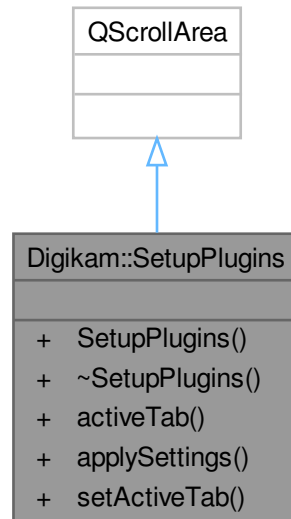
- enum `MiscTab` {  
    `Behaviour = 0` , `Appearance` , `Grouping` , `SpellCheck` ,  
    `Localize` , `System` }

### Public Member Functions

- `SetupMisc` (`QWidget *const parent=nullptr`)
- `MiscTab activeTab () const`
- void `applySettings ()`
- bool `checkSettings ()`
- void `setActiveTab (MiscTab tab)`

## 9.1195 Digikam::SetupPlugins Class Reference

Inheritance diagram for Digikam::SetupPlugins:



### Public Types

- enum `PluginTab` { `Generic = 0` , `Editor` , `Bqm` , `Loaders` }

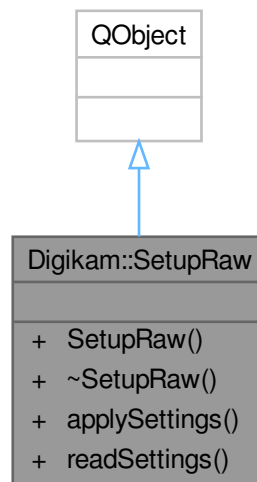
### Public Member Functions

- **SetupPlugins** (`QWidget *const parent=nullptr`)
- `PluginTab` **activeTab** () const
- void **applySettings** ()
- void **setActiveTab** (`PluginTab tab`)



## 9.1196 Digikam::SetupRaw Class Reference

Inheritance diagram for Digikam::SetupRaw:

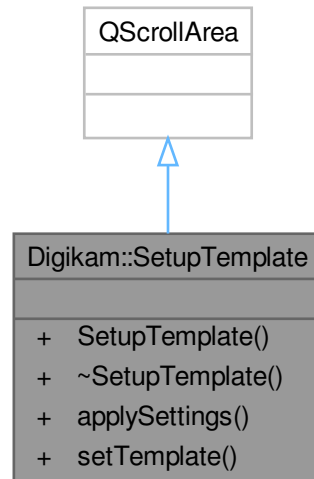


### Public Member Functions

- **SetupRaw** (QTabWidget \*const tab)
- void **applySettings** ()
- void **readSettings** ()

## 9.1197 Digikam::SetupTemplate Class Reference

Inheritance diagram for Digikam::SetupTemplate:

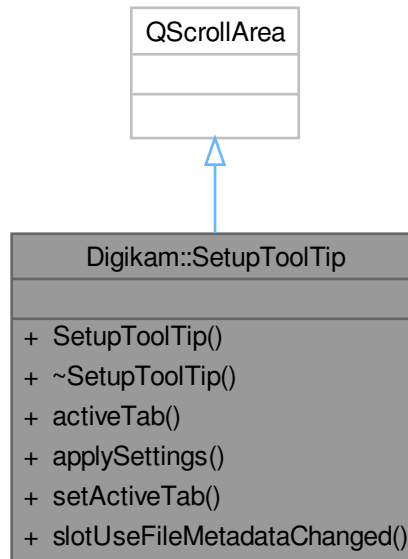


### Public Member Functions

- **SetupTemplate** (`QWidget *const parent=nullptr`)
- void **applySettings** ()
- void **setTemplate** (`const Template &t`)

## 9.1198 Digikam::SetupToolTip Class Reference

Inheritance diagram for Digikam::SetupToolTip:



### Public Types

- enum `ToolTipTab` { `IconItems = 0` , `AlbumItems` , `ImportItems` }

### Public Slots

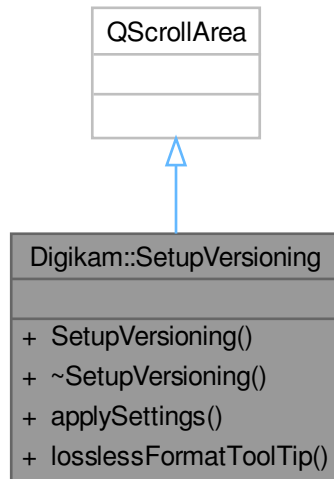
- void `slotUseFileMetadataChanged` (bool)

### Public Member Functions

- **SetupToolTip** (QWidget \*const parent=nullptr)
- ToolTipTab **activeTab** () const
- void **applySettings** ()
- void **setActiveTab** (ToolTipTab tab)

## 9.1199 Digikam::SetupVersioning Class Reference

Inheritance diagram for Digikam::SetupVersioning:



### Public Member Functions

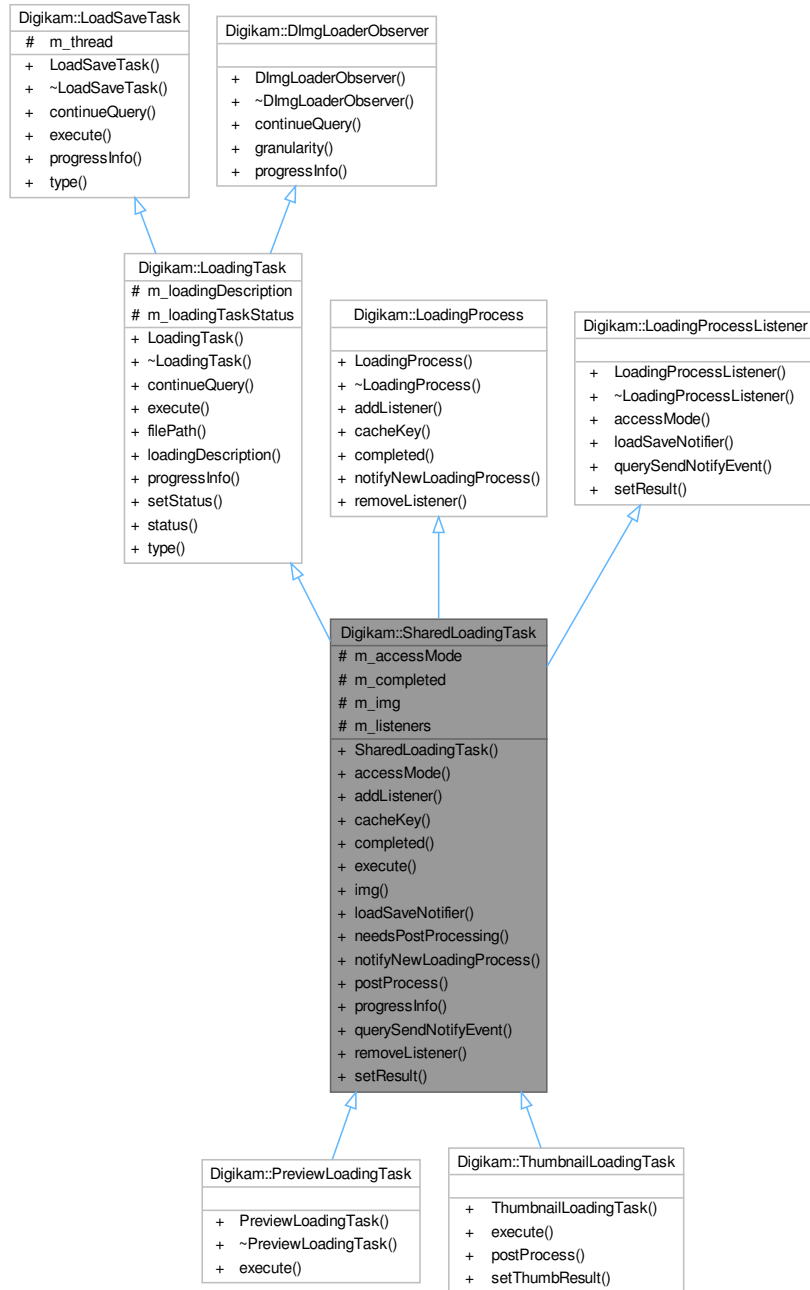
- **SetupVersioning** (QWidget \*const parent=nullptr)
- void **applySettings** ()

### Static Public Member Functions

- static void **losslessFormatToolTip** (QString &formatHelp, bool hasJXLSupport, bool hasWEBPSupport, bool hasAVIFSupport)

## 9.1200 Digikam::SharedLoadingTask Class Reference

Inheritance diagram for Digikam::SharedLoadingTask:



### Public Member Functions

- **SharedLoadingTask** (`LoadSaveThread *const thread`, `const LoadingDescription &description`, `LoadSaveThread::AccessMode mode=LoadSaveThread::AccessModeReadWrite`, `LoadingTaskStatus loadingTaskStatus=LoadingTask↔StatusLoading`)
- `LoadSaveThread::AccessMode accessMode ()` const override

- void [addListener](#) ([LoadingProcessListener](#) \*const listener) override
- QString [cacheKey](#) () const override
- bool [completed](#) () const override
- void [execute](#) () override
- [DImg](#) [img](#) () const
- [LoadSaveNotifier](#) \* [loadSaveNotifier](#) () const override
- bool [needsPostProcessing](#) () const
- void [notifyNewLoadingProcess](#) ([LoadingProcess](#) \*const process, const [LoadingDescription](#) &description) override
- virtual void [postProcess](#) ()
- void [progressInfo](#) (float progress) override
- bool [querySendNotifyEvent](#) () const override
- void [removeListener](#) ([LoadingProcessListener](#) \*const listener) override
- void [setResult](#) (const [LoadingDescription](#) &loadingDescription, const [DImg](#) &img) override

### Public Member Functions inherited from [Digikam::LoadingTask](#)

- [LoadingTask](#) ([LoadSaveThread](#) \*const thread, const [LoadingDescription](#) &description, LoadingTaskStatus loadingTaskStatus=LoadingTaskStatusLoading)
- bool [continueQuery](#) () override
- QString [filePath](#) () const
- const [LoadingDescription](#) & [loadingDescription](#) () const
- void [setStatus](#) (LoadingTaskStatus status)
- LoadingTaskStatus [status](#) () const
- TaskType [type](#) () override

### Public Member Functions inherited from [Digikam::LoadSaveTask](#)

- [LoadSaveTask](#) ([LoadSaveThread](#) \*const thread)

### Public Member Functions inherited from [Digikam::DImgLoaderObserver](#)

- virtual float [granularity](#) ()  
*Return a relative value which determines the granularity, the frequency with which the [DImgLoaderObserver](#) is checked and progress is posted.*

### Protected Attributes

- [LoadSaveThread::AccessMode](#) [m\\_accessMode](#) = [LoadSaveThread::AccessModeReadWrite](#)
- volatile bool [m\\_completed](#) = false
- [DImg](#) [m\\_img](#)
- QList< [LoadingProcessListener](#) \* > [m\\_listeners](#)

### Protected Attributes inherited from [Digikam::LoadingTask](#)

- [LoadingDescription](#) [m\\_loadingDescription](#)
- volatile LoadingTaskStatus [m\\_loadingTaskStatus](#) = LoadingTaskStatusLoading

## Protected Attributes inherited from [Digikam::LoadSaveTask](#)

- [LoadSaveThread](#) \* m\_thread = nullptr

## Additional Inherited Members

## Public Types inherited from [Digikam::LoadingTask](#)

- enum [LoadingTaskStatus](#) { [LoadingTaskStatusLoading](#) , [LoadingTaskStatusPreloading](#) , [LoadingTaskStatusStopping](#) }

## Public Types inherited from [Digikam::LoadSaveTask](#)

- enum [TaskType](#) { [TaskTypeLoading](#) , [TaskTypeSaving](#) }

## 9.1200.1 Member Function Documentation

### 9.1200.1.1 [accessMode\(\)](#)

```
LoadSaveThread::AccessMode Digikam::SharedLoadingTask::accessMode ( ) const [override], [virtual]
```

Implements [Digikam::LoadingProcessListener](#).

### 9.1200.1.2 [addListener\(\)](#)

```
void Digikam::SharedLoadingTask::addListener ( LoadingProcessListener *const listener ) [override], [virtual]
```

Implements [Digikam::LoadingProcess](#).

### 9.1200.1.3 [cacheKey\(\)](#)

```
QString Digikam::SharedLoadingTask::cacheKey ( ) const [override], [virtual]
```

Implements [Digikam::LoadingProcess](#).

### 9.1200.1.4 [completed\(\)](#)

```
bool Digikam::SharedLoadingTask::completed ( ) const [override], [virtual]
```

Implements [Digikam::LoadingProcess](#).

### 9.1200.1.5 [execute\(\)](#)

```
void Digikam::SharedLoadingTask::execute ( ) [override], [virtual]
```

Reimplemented from [Digikam::LoadingTask](#).

### 9.1200.1.6 loadSaveNotifier()

```
LoadSaveNotifier * Digikam::SharedLoadingTask::loadSaveNotifier ( ) const [override], [virtual]
```

Implements [Digikam::LoadingProcessListener](#).

### 9.1200.1.7 notifyNewLoadingProcess()

```
void Digikam::SharedLoadingTask::notifyNewLoadingProcess (
    LoadingProcess *const process,
    const LoadingDescription & description ) [override], [virtual]
```

Implements [Digikam::LoadingProcess](#).

### 9.1200.1.8 progressInfo()

```
void Digikam::SharedLoadingTask::progressInfo (
    float progress ) [override], [virtual]
```

Reimplemented from [Digikam::LoadingTask](#).

### 9.1200.1.9 querySendNotifyEvent()

```
bool Digikam::SharedLoadingTask::querySendNotifyEvent ( ) const [override], [virtual]
```

Implements [Digikam::LoadingProcessListener](#).

### 9.1200.1.10 removeListener()

```
void Digikam::SharedLoadingTask::removeListener (
    LoadingProcessListener *const listener ) [override], [virtual]
```

Implements [Digikam::LoadingProcess](#).

### 9.1200.1.11 setResult()

```
void Digikam::SharedLoadingTask::setResult (
    const LoadingDescription & loadingDescription,
    const DImg & img ) [override], [virtual]
```

Implements [Digikam::LoadingProcessListener](#).



## 9.1201 Digikam::SharedLoadSaveThread Class Reference

Inheritance diagram for Digikam::SharedLoadSaveThread:



### Public Member Functions

- `SharedLoadSaveThread` (`QObject *const parent=nullptr`)
- void `load` (const [LoadingDescription](#) &description, [AccessMode](#) mode, [LoadingPolicy](#) policy=[LoadingPolicyAppend](#))

## Public Member Functions inherited from [Digikam::ManagedLoadSaveThread](#)

- **ManagedLoadSaveThread** (QObject \*const parent=nullptr)
  - Termination is controlled by setting the TerminationPolicy Default is TerminationPolicyTerminateLoading.*
- void **load** (const [LoadingDescription](#) &description)
  - Append a task to load the given file to the task list.*
- void **load** (const [LoadingDescription](#) &description, [LoadingPolicy](#) policy)
- [LoadingPolicy](#) **loadingPolicy** () const
- void **save** (const [DImg](#) &image, const QString &filePath, const QString &format)
  - Append a task to save the image to the task list.*
- void **setLoadingPolicy** ([LoadingPolicy](#) policy)
  - Set the loading policy.*
- void **setTerminationPolicy** ([TerminationPolicy](#) terminationPolicy)
- void **stopAllTasks** ()
- void **stopLoading** (const [LoadingDescription](#) &desc, [LoadingTaskFilter](#) filter=[LoadingTaskFilterAll](#))
  - Same than previous method, but Stop and remove tasks filtered by LoadingDescription.*
- void **stopLoading** (const QString &filePath=QString(), [LoadingTaskFilter](#) filter=[LoadingTaskFilterAll](#))
  - Stop and remove tasks filtered by filePath and policy.*
- void **stopSaving** (const QString &filePath=QString())
  - Stop and remove saving tasks filtered by filePath.*
- [TerminationPolicy](#) **terminationPolicy** () const

## Public Member Functions inherited from [Digikam::LoadSaveThread](#)

- **LoadSaveThread** (QObject \*const parent=nullptr)
- **~LoadSaveThread** () override
  - Destructor: The thread will execute all pending tasks and wait for this upon destruction.*
- void **imageLoaded** (const [LoadingDescription](#) &loadingDescription, const [DImg](#) &img) override
- void **imageSaved** (const QString &filePath, bool success) override
- void **imageStartedLoading** (const [LoadingDescription](#) &loadingDescription) override
- void **imageStartedSaving** (const QString &filePath) override
- void **load** (const [LoadingDescription](#) &description)
  - Append a task to load the given file to the task list.*
- void **loadingProgress** (const [LoadingDescription](#) &loadingDescription, float progress) override
- void **moreCompleteLoadingAvailable** (const [LoadingDescription](#) &oldLoadingDescription, const [LoadingDescription](#) &newLoadingDescription) override
- virtual bool **querySendNotifyEvent** () const
- void **save** (const [DImg](#) &image, const QString &filePath, const QString &format)
  - Append a task to save the image to the task list.*
- void **savingProgress** (const QString &filePath, float progress) override
- void **setNotificationPolicy** ([NotificationPolicy](#) notificationPolicy)
- virtual void **taskHasFinished** ()
- void **thumbnailLoaded** (const [LoadingDescription](#) &loadingDescription, const [QImage](#) &img) override

## Public Member Functions inherited from Digikam::DynamicThread

- [DynamicThread](#) (QObject \*const parent=nullptr)
 

*This class extends QRunnable, so you have to reimplement virtual void [run\(\)](#).*
- [~DynamicThread](#) () override
 

*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool [isFinished](#) () const
- bool [isRunning](#) () const
- QThread::Priority [priority](#) () const
- void [setEmitSignals](#) (bool emitThem)
- void [setPriority](#) (QThread::Priority priority)
 

*Sets the priority for this dynamic thread.*
- State [state](#) () const

## Additional Inherited Members

## Public Types inherited from Digikam::ManagedLoadSaveThread

- enum [LoadingMode](#) { [LoadingModeNormal](#) , [LoadingModeShared](#) }
 

*used by [SharedLoadSaveThread](#) only*
- enum [LoadingPolicy](#) { [LoadingPolicyFirstRemovePrevious](#) , [LoadingPolicyPrepend](#) , [LoadingPolicySimplePrepend](#) , [LoadingPolicyAppend](#) , [LoadingPolicySimpleAppend](#) , [LoadingPolicyPreload](#) }
- enum [LoadingTaskFilter](#) { [LoadingTaskFilterAll](#) , [LoadingTaskFilterPreloading](#) }
- enum [TerminationPolicy](#) { [TerminationPolicyTerminateLoading](#) , [TerminationPolicyTerminatePreloading](#) , [TerminationPolicyWait](#) , [TerminationPolicyTerminateAll](#) }

## Public Types inherited from Digikam::LoadSaveThread

- enum [AccessMode](#) { [AccessModeRead](#) , [AccessModeReadWrite](#) }
 

*used by [SharedLoadSaveThread](#) only*
- enum [NotificationPolicy](#) { [NotificationPolicyDirect](#) , [NotificationPolicyTimeLimited](#) }

## Public Types inherited from Digikam::DynamicThread

- enum [State](#) { [Inactive](#) , [Scheduled](#) , [Running](#) , [Deactivating](#) }

## Public Slots inherited from Digikam::DynamicThread

- void [start](#) ()
- void [stop](#) ()
 

*Stop computation, sets the running flag to false.*
- void [wait](#) ()
 

*Waits until the thread finishes.*

## Signals inherited from [Digikam::LoadSaveThread](#)

- void [signalImageLoaded](#) (const [LoadingDescription](#) &loadingDescription, const [DImg](#) &img)  
*This signal is emitted when the loading process has finished.*
- void **signalImageSaved** (const [QString](#) &filePath, bool success)
- void [signalImageStartedLoading](#) (const [LoadingDescription](#) &loadingDescription)  
*All signals are delivered to the thread from where the [LoadSaveThread](#) object has been created.*
- void **signalImageStartedSaving** (const [QString](#) &filePath)
- void [signalLoadingProgress](#) (const [LoadingDescription](#) &loadingDescription, float progress)  
*This signal is emitted whenever new progress info is available and the notification policy allows emitting the signal.*
- void [signalMoreCompleteLoadingAvailable](#) (const [LoadingDescription](#) &oldLoadingDescription, const [LoadingDescription](#) &newLoadingDescription)  
*This signal is emitted if.*
- void **signalSavingProgress** (const [QString](#) &filePath, float progress)
- void **signalThumbnailLoaded** (const [LoadingDescription](#) &loadingDescription, const [QImage](#) &img)

## Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if `emitSignals` is enabled.*

## Static Public Member Functions inherited from [Digikam::LoadSaveThread](#)

- static int **exifOrientation** (const [QString](#) &filePath, const [DMetadata](#) &metadata, bool isRaw, bool fromRaw↔  
EmbeddedPreview)  
*Retrieves the Exif orientation, either from the info provider if available, or from the metadata.*
- static [LoadSaveFileInfoProvider](#) \* **infoProvider** ()
- static void **setInfoProvider** ([LoadSaveFileInfoProvider](#) \*const infoProvider)

## Protected Member Functions inherited from [Digikam::ManagedLoadSaveThread](#)

- void **load** (const [LoadingDescription](#) &description, [LoadingMode](#) loadingMode, [AccessMode](#) mode=[AccessModeReadWrite](#))
- void **load** (const [LoadingDescription](#) &description, [LoadingMode](#) loadingMode, [LoadingPolicy](#) policy,  
[AccessMode](#) mode=[AccessModeReadWrite](#))
- void **loadPreview** (const [LoadingDescription](#) &description, [LoadingPolicy](#) policy)
- void **loadThumbnail** (const [LoadingDescription](#) &description)
- void **preloadThumbnail** (const [LoadingDescription](#) &description)
- void **preloadThumbnailGroup** (const [QList](#)< [LoadingDescription](#) > &descriptions)
- void **prependThumbnailGroup** (const [QList](#)< [LoadingDescription](#) > &descriptions)
- void **shutDown** ()

## Protected Member Functions inherited from [Digikam::LoadSaveThread](#)

- void **notificationReceived** ()
- void [run](#) () override  
*Implement this pure virtual function in your subclass.*

## Protected Member Functions inherited from Digikam::DynamicThread

- bool **runningFlag** () const volatile  
*In you [run\(\)](#) method, you shall regularly check for [runningFlag\(\)](#) and cleanup and return if false.*
- void **shutDown** ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call [stop\(\)](#) and [wait\(\)](#), knowing that nothing will call [start\(\)](#) anymore after this 3) Be sure the thread will never be running at destruction.*
- void **start** (QMutexLocker< QMutex > &locker)  
*Doing the same as [start\(\)](#), [stop\(\)](#) and [wait](#) above, provide it with a locked QMutexLocker on mutex().*
- void **stop** (const QMutexLocker< QMutex > &locker)
- QMutex \* **threadMutex** () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void **wait** (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from Digikam::ManagedLoadSaveThread

- [LoadingPolicy](#) **m\_loadingPolicy** = [LoadingPolicyAppend](#)
- [TerminationPolicy](#) **m\_terminationPolicy** = [TerminationPolicyTerminateLoading](#)

## Protected Attributes inherited from Digikam::LoadSaveThread

- [LoadSaveTask](#) \* **m\_currentTask** = nullptr
- QMutex **m\_mutex**
- [NotificationPolicy](#) **m\_notificationPolicy** = [NotificationPolicyTimeLimited](#)
- QList< [LoadSaveTask](#) \* > **m\_todo**

## 9.1202 Digikam::SharedQueue< T > Class Template Reference

### Public Member Functions

- void **clear** ()
- bool **empty** ()
- T & **front** ()
- int **maxDepth** () const
- T & **pop\_front** ()
- void **push\_back** (T &&item)
- void **push\_back** (T &item)
- void **setMaxDepth** (int depth)
- int **size** ()

## 9.1203 Digikam::SharpContainer Class Reference

### Public Types

- enum **SharpingMethods** { **SimpleSharp** = 0 , **UnsharpMask** , **Refocus** }

## Public Attributes

- int **method** = SimpleSharp  
*Store SharpingMethods value.*
- double **rfCorrelation** = 0.5
- double **rfGauss** = 0.0
- int **rfMatrix** = 5
- double **rfNoise** = 0.03
- double **rfRadius** = 1.0  
*Refocus.*
- int **ssRadius** = 0  
*Simple sharp.*
- double **umAmount** = 1.0
- bool **umLumaOnly** = false
- double **umRadius** = 1.0  
*Unsharp mask.*
- double **umThreshold** = 0.05

## 9.1204 Digikam::SharpenFilter Class Reference

Inheritance diagram for Digikam::SharpenFilter:



### Public Member Functions

- **SharpenFilter** ([DImg](#) \*const orgImage, [QObject](#) \*const parent=nullptr, double radius=0.0, double sigma=1.0)
- **SharpenFilter** ([DImgThreadedFilter](#) \*const parentFilter, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, double radius=0.0, double sigma=1.0)

*Constructor for slave mode: execute immediately in current thread with specified master filter.*

- **SharpenFilter** (QObject \*const parent=nullptr)
- **FilterAction filterAction** () override
 

*Returns the action description corresponding to currently set options.*
- QString **filterIdentifier** () const override
 

*Return the identifier for this filter in the image history.*
- void **readParameters** (const **FilterAction** &action) override

## Public Member Functions inherited from **Digikam::DImgThreadedFilter**

- **DImgThreadedFilter** (DImg \*const orgImage, QObject \*const parent, const QString &name=QString())
 

*Constructs a filter with all arguments (ready to use).*
- **DImgThreadedFilter** (QObject \*const parent=nullptr, const QString &name=QString())
 

*Constructs a filter without argument.*
- virtual void **cancelFilter** ()
 

*Cancel the threaded computation.*
- const QString & **filterName** ()
- int **filterVersion** () const
- **DImg getTargetImage** ()
- QList< int > **multithreadedSteps** (int stop, int start=0) const
 

*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool **parametersSuccessfullyRead** () const
 

*Optional: error handling for readParameters.*
- virtual QString **readParametersError** (const **FilterAction** &actionThatFailed) const
- void **setFilterName** (const QString &name)
- void **setFilterVersion** (int version)
 

*Replaying a filter action: Set the filter version.*
- void **setOriginalImage** (const **DImg** &orgImage)
- void **setupAndStartDirectly** (const **DImg** &orgImage, **DImgThreadedFilter** \*const master, int progress←Begin=0, int progressEnd=100)
 

*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void **setupFilter** (const **DImg** &orgImage)
 

*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void **startFilter** ()
 

*Start the threaded computation.*
- virtual void **startFilterDirectly** ()
 

*Start computation of this filter, directly in this thread.*
- virtual QList< int > **supportedVersions** () const

## Public Member Functions inherited from **Digikam::DynamicThread**

- **DynamicThread** (QObject \*const parent=nullptr)
 

*This class extends QRunnable, so you have to reimplement virtual void [run\(\)](#).*
- ~**DynamicThread** () override
 

*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool **isFinished** () const
- bool **isRunning** () const
- QThread::Priority **priority** () const
- void **setEmitSignals** (bool emitThem)
- void **setPriority** (QThread::Priority priority)
 

*Sets the priority for this dynamic thread.*
- State **state** () const



### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from [Digikam::DImgThreadedFilter](#)

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.1204.1 Member Function Documentation

### 9.1204.1.1 filterAction()

`FilterAction` Digikam::SharpenFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.1204.1.2 filterIdentifier()

`QString` Digikam::SharpenFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

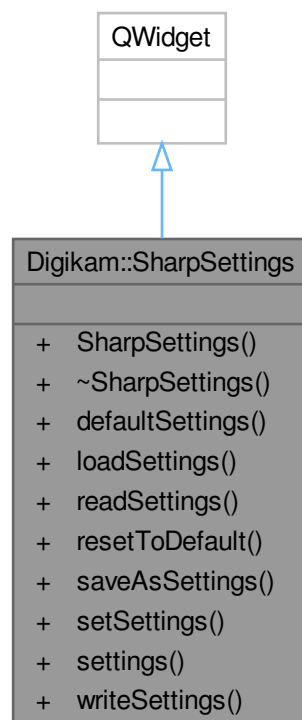
### 9.1204.1.3 readParameters()

```
void Digikam::SharpenFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

## 9.1205 Digikam::SharpSettings Class Reference

Inheritance diagram for Digikam::SharpSettings:



## Signals

- void **signalSettingsChanged** ()

## Public Member Functions

- **SharpSettings** (QWidget \*const parent)
- [SharpContainer](#) **defaultSettings** () const
- void **loadSettings** ()
- void **readSettings** (const KConfigGroup &group)
- void **resetToDefault** ()
- void **saveAsSettings** ()
- void **setSettings** (const [SharpContainer](#) &settings)
- [SharpContainer](#) **settings** () const
- void **writeSettings** (KConfigGroup &group)

## 9.1206 Digikam::ShearFilter Class Reference

Inheritance diagram for Digikam::ShearFilter:



### Public Member Functions

- **ShearFilter** (*Dlmg* \*const orgImage, QObject \*const parent=nullptr, float hAngle=0.0, float vAngle=0.0, bool antialiasing=true, const QColor &background-color=Qt::black, int orgW=0, int orgH=0)

- **ShearFilter** (QObject \*const parent=nullptr)
- **FilterAction filterAction** () override  
*Returns the action description corresponding to currently set options.*
- QString **filterIdentifier** () const override  
*Return the identifier for this filter in the image history.*
- QSize **getNewSize** () const
- void **readParameters** (const FilterAction &action) override

## Public Member Functions inherited from Digikam::DImgThreadedFilter

- **DImgThreadedFilter** (DImg \*const orgImage, QObject \*const parent, const QString &name=QString())  
*Constructs a filter with all arguments (ready to use).*
- **DImgThreadedFilter** (QObject \*const parent=nullptr, const QString &name=QString())  
*Constructs a filter without argument.*
- virtual void **cancelFilter** ()  
*Cancel the threaded computation.*
- const QString & **filterName** ()
- int **filterVersion** () const
- **DImg getTargetImage** ()
- QList< int > **multithreadedSteps** (int stop, int start=0) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool **parametersSuccessfullyRead** () const  
*Optional: error handling for readParameters.*
- virtual QString **readParametersError** (const FilterAction &actionThatFailed) const
- void **setFilterName** (const QString &name)
- void **setFilterVersion** (int version)  
*Replaying a filter action: Set the filter version.*
- void **setOriginalImage** (const DImg &orgImage)
- void **setupAndStartDirectly** (const DImg &orgImage, DImgThreadedFilter \*const master, int progress←Begin=0, int progressEnd=100)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void **setupFilter** (const DImg &orgImage)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void **startFilter** ()  
*Start the threaded computation.*
- virtual void **startFilterDirectly** ()  
*Start computation of this filter, directly in this thread.*
- virtual QList< int > **supportedVersions** () const

## Public Member Functions inherited from Digikam::DynamicThread

- **DynamicThread** (QObject \*const parent=nullptr)  
*This class extends QRunnable, so you have to reimplement virtual void run().*
- **~DynamicThread** () override  
*The destructor calls stop() and wait(), but if you, in your destructor, delete any data that is accessed by your run() method, you must call stop() and wait() before yourself.*
- bool **isFinished** () const
- bool **isRunning** () const
- QThread::Priority **priority** () const
- void **setEmitSignals** (bool emitThem)
- void **setPriority** (QThread::Priority priority)  
*Sets the priority for this dynamic thread.*
- State **state** () const

### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from [Digikam::DImgThreadedFilter](#)

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false



## 9.1206.1 Member Function Documentation

### 9.1206.1.1 filterAction()

`FilterAction` Digikam::ShearFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.1206.1.2 filterIdentifier()

`QString` Digikam::ShearFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.1206.1.3 readParameters()

```
void Digikam::ShearFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

## 9.1207 Digikam::ShowHideVersionsOverlay Class Reference

Inheritance diagram for Digikam::ShowHideVersionsOverlay:



### Public Member Functions

- **ShowHideVersionsOverlay** (QObject \*const parent)
- void **setActive** (bool active) override  
*Will call [createButton\(\)](#).*
- void **setSettings** (const [VersionManagerSettings](#) &settings)

## Public Member Functions inherited from Digikam::HoverButtonDelegateOverlay

- **HoverButtonDelegateOverlay** (QObject \*const parent)
- **ItemViewHoverButton** \* **button** () const

## Public Member Functions inherited from Digikam::AbstractWidgetDelegateOverlay

- **AbstractWidgetDelegateOverlay** (QObject \*const parent)  
*This class provides functionality for using a widget in an overlay.*

## Public Member Functions inherited from Digikam::ItemDelegateOverlay

- **ItemDelegateOverlay** (QObject \*const parent=nullptr)
- virtual bool **acceptsDelegate** (QAbstractItemDelegate \*) const
- QAbstractItemDelegate \* **delegate** () const
- virtual void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)  
*Only these two methods are implemented as virtual methods.*
- virtual void **paint** (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index)
- void **setDelegate** (QAbstractItemDelegate \*delegate)
- void **setView** (QAbstractItemView \*view)
- QAbstractItemView \* **view** () const

## Protected Slots

- void **slotClicked** (bool checked)

## Protected Slots inherited from Digikam::HoverButtonDelegateOverlay

- void **slotEntered** (const QModelIndex &index) override
- void **slotReset** () override

## Protected Slots inherited from Digikam::AbstractWidgetDelegateOverlay

- virtual void **slotEntered** (const QModelIndex &index)  
*Default implementation shows the widget iff the index is valid and checkIndex returns true.*
- virtual void **slotLayoutChanged** ()
- virtual void **slotReset** ()  
*Default implementations of these three slots call `hide()`*
- virtual void **slotRowsRemoved** (const QModelIndex &parent, int start, int end)
- virtual void **slotViewportEntered** ()

## Protected Slots inherited from Digikam::ItemDelegateOverlay

### Protected Member Functions

- bool **checkIndex** (const QModelIndex &index) const override  
*Return true here if you want to show the overlay for the given index.*
- **ItemViewHoverButton** \* **createButton** () override  
*Create your widget here.*
- void **updateButton** (const QModelIndex &index) override  
*Called when a new index is entered.*

## Protected Member Functions inherited from [Digikam::HoverButtonDelegateOverlay](#)

- `QWidget * createWidget ()` override  
*Create your widget here.*
- `void visualChange ()` override  
*Called when any change from the delegate occurs - when the overlay is installed, when size hints, styles or fonts change.*

## Protected Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- `bool checkIndexOnEnter (const QModelIndex &index) const`  
*Utility method called from slotEntered.*
- `bool eventFilter (QObject *obj, QEvent *event)` override
- `virtual void hide ()`  
*Called when the widget shall be hidden (mouse cursor left index, viewport, uninstalled etc.).*
- `virtual QString notifyMultipleMessage (const QModelIndex &, int number)`
- `QWidget * parentWidget () const`  
*Returns the widget to be used as parent for your widget created in [createWidget\(\)](#)*
- `virtual void viewportLeaveEvent (QObject *obj, QEvent *event)`  
*Called when a `QEvent::Leave` of the viewport is received.*
- `virtual void widgetEnterEvent ()`  
*Called when a `QEvent::Enter` resp.*
- `void widgetEnterNotifyMultiple (const QModelIndex &index)`  
*A sample implementation for above methods.*
- `virtual void widgetLeaveEvent ()`
- `void widgetLeaveNotifyMultiple ()`

## Protected Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- `QList< QModelIndex > affectedIndexes (const QModelIndex &index) const`
- `bool affectsMultiple (const QModelIndex &index) const`  
*For the context that an overlay can affect multiple items: Assuming the currently overlaid index is given.*
- `int numberOfAffectedIndexes (const QModelIndex &index) const`
- `bool viewHasMultiSelection () const`  
*Utility method.*

## Protected Attributes

- `VersionItemFilterSettings m_filter`

## Protected Attributes inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- `bool m_mouseButtonPressedOnWidget = false`
- `QWidget * m_widget = nullptr`

## Protected Attributes inherited from [Digikam::ItemDelegateOverlay](#)

- `QAbstractItemDelegate * m_delegate = nullptr`
- `QAbstractItemView * m_view = nullptr`

## Additional Inherited Members

## Signals inherited from [Digikam::ItemDelegateOverlay](#)

- void **hideNotification** ()
- void **requestNotification** (const QModelIndex &index, const QString &message)
- void **update** (const QModelIndex &index)

## 9.1207.1 Member Function Documentation

### 9.1207.1.1 checkIndex()

```
bool Digikam::ShowHideVersionsOverlay::checkIndex (
    const QModelIndex & index ) const [override], [protected], [virtual]
```

The default implementation returns true.

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.1207.1.2 createButton()

```
ItemViewHoverButton * Digikam::ShowHideVersionsOverlay::createButton ( ) [override], [protected],
[virtual]
```

Pass view() as parent.

Implements [Digikam::HoverButtonDelegateOverlay](#).

### 9.1207.1.3 setActive()

```
void Digikam::ShowHideVersionsOverlay::setActive (
    bool active ) [override], [virtual]
```

Reimplemented from [Digikam::HoverButtonDelegateOverlay](#).

### 9.1207.1.4 updateButton()

```
void Digikam::ShowHideVersionsOverlay::updateButton (
    const QModelIndex & index ) [override], [protected], [virtual]
```

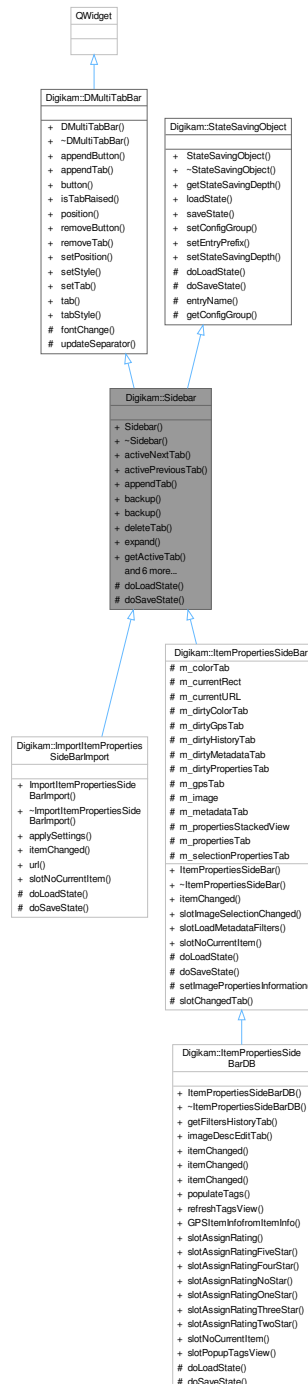
Reposition your button here, adjust and store state.

Implements [Digikam::HoverButtonDelegateOverlay](#).

## 9.1208 Digikam::Sidebar Class Reference

This class handles a sidebar view.

Inheritance diagram for Digikam::Sidebar:



### Signals

- void **signalChangedTab** (QWidget \*w)

*Is emitted, when another tab is activated.*

- void **signalViewChanged** ()

*Is emitted, when tab is shrink or expanded.*

## Public Member Functions

- **Sidebar** (QWidget \*const parent, **SidebarSplitter** \*const sp, Qt::Edge side=Qt::LeftEdge, bool minimized← Default=false)
 

*Creates a new sidebar.*
- void **activeNextTab** ()
 

*Activates a next tab from current one.*
- void **activePreviousTab** ()
 

*Activates a previous tab from current one.*
- void **appendTab** (QWidget \*const w, const QIcon &pic, const QString &title)
 

*Appends a new tab to the sidebar.*
- void **backup** ()
 

*Hide sidebar and backup minimized state.*
- void **backup** (const QList< QWidget \* > &thirdWidgetsToBackup, QList< int > \*const sizes)
 

*Hide sidebar and backup minimized state.*
- void **deleteTab** (QWidget \*const w)
 

*Deletes a tab from the tabbar.*
- void **expand** ()
 

*Redisplays the whole sidebar.*
- QWidget \* **getActiveTab** () const
 

*Returns the currently activated tab, or 0 if no tab is active.*
- bool **isExpanded** () const
 

*Return the visible status of current sidebar tab.*
- void **restore** ()
 

*Show sidebar and restore minimized state.*
- void **restore** (const QList< QWidget \* > &thirdWidgetsToRestore, const QList< int > &sizes)
 

*Show sidebar and restore minimized state.*
- void **setActiveTab** (QWidget \*const w)
 

*Activates a tab.*
- void **shrink** ()
 

*Hides the sidebar (display only the activation buttons)*
- **SidebarSplitter** \* **splitter** () const

## Public Member Functions inherited from Digikam::DMultiTabBar

- **DMultiTabBar** (Qt::Edge pos, QWidget \*const parent=nullptr)
- void **appendButton** (const QIcon &pic, int id=-1, QMenu \*const popup=nullptr, const QString &not\_used\_← yet=QString())
 

*append a new button to the button area.*
- void **appendTab** (const QIcon &pic, int id=-1, const QString &text=QString())
 

*append a new tab to the tab area.*
- **DMultiTabBarButton** \* **button** (int id) const
 

*get a pointer to a button within the button area identified by its ID*
- bool **isTabRaised** (int id) const
 

*return the state of a tab, identified by its ID*
- Qt::Edge **position** () const

- get the tabbar position.*
- void **removeButton** (int id)
  - remove a button with the given ID*
- void **removeTab** (int id)
  - remove a tab with a given ID*
- void **setPosition** (Qt::Edge pos)
  - set the real position of the widget.*
- void **setStyle** (TextStyle style)
  - set the display style of the tabs*
- void **setTab** (int id, bool state)
  - set a tab to "raised"*
- **DMultiTabBarTab \* tab** (int id) const
  - get a pointer to a tab within the tab area, identified by its ID*
- **TextStyle tabStyle** () const
  - get the display style of the tabs*

## Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (QObject \*const host)
  - Constructor.*
- virtual **~StateSavingObject** ()
  - Destructor.*
- [StateSavingDepth](#) **getStateSavingDepth** () const
  - Returns the depth used for state saving or loading.*
- void **loadState** ()
  - Invokes loading the class' state.*
- void **saveState** ()
  - Invokes saving the class' state.*
- virtual void **setConfigGroup** (const KConfigGroup &group)
  - Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void **setEntryPrefix** (const QString &prefix)
  - Define a prefix that will be used for every entry in the config group.*
- void **setStateSavingDepth** (const [StateSavingDepth](#) depth)
  - Sets the depth used for state saving or loading.*

## Protected Member Functions

- void **doLoadState** () override
  - Load the last view state from disk - called by [StateSavingObject::loadState\(\)](#)*
- void **doSaveState** () override
  - Save the view state to disk - called by [StateSavingObject::saveState\(\)](#)*

## Protected Member Functions inherited from [Digikam::DMultiTabBar](#)

- virtual void **fontChange** (const QFont &)
- void **updateSeparator** ()



## Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- QString [entryName](#) (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup [getConfigGroup](#) () const  
*Returns the config group that must be used for state saving and loading.*

## Friends

- class [SidebarSplitter](#)

## Additional Inherited Members

## Public Types inherited from [Digikam::DMultiTabBar](#)

- enum [TextStyle](#) { [ActiveIconText](#) = 0 , [AllIconsText](#) = 2 }
- The list of available styles for [DMultiTabBar](#).*

## Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }
- This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

### 9.1208.1 Detailed Description

Since this class derives from [StateSavingObject](#), you can call [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) for loading/saving of settings. However, if you use multiple sidebar instances in your program, you have to remember to either call [QObject::setObjectName\(\)](#), [StateSavingObject::setEntryPrefix\(\)](#) or [StateSavingObject::setConfigGroup\(\)](#) first.

### 9.1208.2 Constructor & Destructor Documentation

#### 9.1208.2.1 Sidebar()

```
Digikam::Sidebar::Sidebar (
    QWidget *const parent,
    SidebarSplitter *const sp,
    Qt::Edge side = Qt::LeftEdge,
    bool minimizedDefault = false ) [explicit]
```

#### Parameters

<i>parent</i>	sidebar's parent
<i>sp</i>	sets the splitter, which should handle the width. The splitter normally is part of the main view. Internally, the width of the widget stack can be changed by a QSplitter.
<i>side</i>	where the sidebar should be displayed. At the left or right border. Use Qt::LeftEdge or Qt::RightEdge.
<i>minimizedDefault</i>	hide the sidebar when the program is started the first time.

## 9.1208.3 Member Function Documentation

### 9.1208.3.1 activeNextTab()

```
void Digikam::Sidebar::activeNextTab ( )
```

If current one is last, first one is activated.

### 9.1208.3.2 activePreviousTab()

```
void Digikam::Sidebar::activePreviousTab ( )
```

If current one is first, last one is activated.

### 9.1208.3.3 appendTab()

```
void Digikam::Sidebar::appendTab (
    QWidget *const w,
    const QIcon & pic,
    const QString & title )
```

#### Parameters

<i>w</i>	widget which is activated by this tab
<i>pic</i>	icon which is shown in this tab
<i>title</i>	text which is shown it this tab

### 9.1208.3.4 backup()

```
void Digikam::Sidebar::backup (
    const QList< QWidget * > & thirdWidgetsToBackup,
    QList< int > *const sizes )
```

If there are other widgets in this splitter, stores their sizes in the provided list.

### 9.1208.3.5 doLoadState()

```
void Digikam::Sidebar::doLoadState ( ) [override], [protected], [virtual]
```

Implements [Digikam::StateSavingObject](#).

### 9.1208.3.6 doSaveState()

```
void Digikam::Sidebar::doSaveState ( ) [override], [protected], [virtual]
```

Implements [Digikam::StateSavingObject](#).

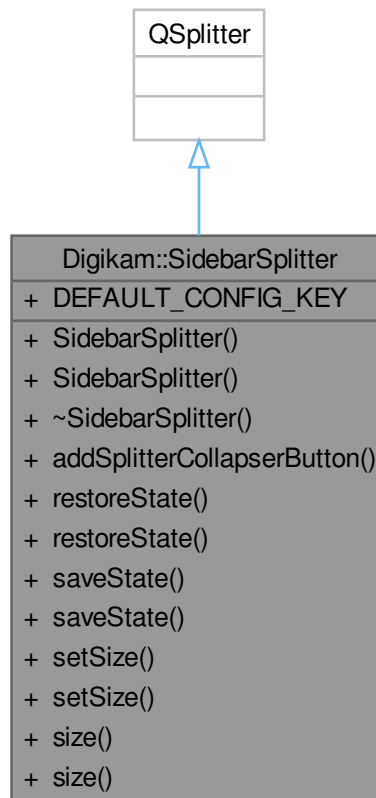
## 9.1208.3.7 restore()

```
void Digikam::Sidebar::restore (
    const QList< QWidget * > & thirdWidgetsToRestore,
    const QList< int > & sizes )
```

Restores other widgets' sizes in splitter.

## 9.1209 Digikam::SidebarSplitter Class Reference

Inheritance diagram for Digikam::SidebarSplitter:



## Public Member Functions

- **SidebarSplitter** (Qt::Orientation orientation, QWidget \*const parent=nullptr)
- **SidebarSplitter** (QWidget \*const parent=nullptr)

*This is a QSplitter with better support for storing its state in config files, especially if Sidebars are contained in the splitter.*

- void **addSplitterCollapserButton** (QWidget \*const widget)
- void **restoreState** (KConfigGroup &group)

- Restores the splitter state from group, handling minimized sidebars correctly.*

  - void [restoreState](#) (KConfigGroup &group, const QString &key)
- Restores the splitter state from group, handling minimized sidebars correctly.*

  - void [saveState](#) (KConfigGroup &group)
- Saves the splitter state to group, handling minimized sidebars correctly.*

  - void [saveState](#) (KConfigGroup &group, const QString &key)
- Saves the splitter state to group, handling minimized sidebars correctly.*

  - void **setSize** (QWidget \*const widget, int [size](#))
  - void [setSize](#) ([Sidebar](#) \*const bar, int [size](#))
- Sets the splitter size for the given sidebar or splitter child widget to size.*

  - int **size** (QWidget \*const widget) const
  - int **size** ([Sidebar](#) \*const bar) const
- Returns the value of sizes() that corresponds to the given [Sidebar](#) or splitter child widget.*

### Static Public Attributes

- static const QString **DEFAULT\_CONFIG\_KEY** = QLatin1String("SplitterState")

### Friends

- class **Sidebar**

## 9.1209.1 Member Function Documentation

### 9.1209.1.1 restoreState() [1/2]

```
void Digikam::SidebarSplitter::restoreState (
    KConfigGroup & group )
```

DEFAULT\_CONFIG\_KEY is used for restoring the state.

### 9.1209.1.2 restoreState() [2/2]

```
void Digikam::SidebarSplitter::restoreState (
    KConfigGroup & group,
    const QString & key )
```

This version uses a specified key in the config group.

### 9.1209.1.3 saveState() [1/2]

```
void Digikam::SidebarSplitter::saveState (
    KConfigGroup & group )
```

DEFAULT\_CONFIG\_KEY is used for storing the state.



## Public Member Functions

- [SidebarWidget](#) (QWidget \*const parent)  
*Constructor.*
- [~SidebarWidget](#) () override=default  
*Destructor.*
- virtual void [applySettings](#) ()=0  
*This method is invoked when the application settings should be (re-) applied to this widget.*
- virtual void [changeAlbumFromHistory](#) (const QList< Album \* > &album)=0  
*This is called on this widget when the history requires to move back to the specified album.*
- virtual const QString [getCaption](#) ()=0  
*Must be implemented to return the title of this sidebar's tab.*
- virtual const QIcon [getIcon](#) ()=0  
*Must be implemented and return the icon that shall be visible for this sidebar widget.*
- virtual void [setActive](#) (bool active)=0  
*This method is called if the visible sidebar widget is changed.*

## Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (QObject \*const host)  
*Constructor.*
- virtual [~StateSavingObject](#) ()  
*Destructor.*
- [StateSavingDepth](#) [getStateSavingDepth](#) () const  
*Returns the depth used for state saving or loading.*
- void [loadState](#) ()  
*Invokes loading the class' state.*
- void [saveState](#) ()  
*Invokes saving the class' state.*
- virtual void [setConfigGroup](#) (const KConfigGroup &group)  
*Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void [setEntryPrefix](#) (const QString &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void [setStateSavingDepth](#) (const [StateSavingDepth](#) depth)  
*Sets the depth used for state saving or loading.*

## Additional Inherited Members

## Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }  
*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

## Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- virtual void [doLoadState](#) ()=0  
*Implement this hook method for state loading.*
- virtual void [doSaveState](#) ()=0  
*Implement this hook method for state saving.*
- QString [entryName](#) (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup [getConfigGroup](#) () const  
*Returns the config group that must be used for state saving and loading.*

## 9.1210.1 Constructor & Destructor Documentation

### 9.1210.1.1 SidebarWidget()

```
Digikam::SidebarWidget::SidebarWidget (
    QWidget *const parent ) [explicit]
```

#### Parameters

<i>parent</i>	the parent of this widget, may be null
---------------	--

## 9.1210.2 Member Function Documentation

### 9.1210.2.1 applySettings()

```
virtual void Digikam::SidebarWidget::applySettings ( ) [pure virtual]
```

Implemented in [Digikam::AlbumFolderViewSideBarWidget](#), [Digikam::DateFolderViewSideBarWidget](#), [Digikam::FuzzySearchSideBarV](#), [Digikam::GPSSearchSideBarWidget](#), [Digikam::LabelsSideBarWidget](#), [Digikam::PeopleSideBarWidget](#), [Digikam::SearchSideBarWidg](#), [Digikam::TagViewSideBarWidget](#), and [Digikam::TimelineSideBarWidget](#).

### 9.1210.2.2 changeAlbumFromHistory()

```
virtual void Digikam::SidebarWidget::changeAlbumFromHistory (
    const QList< Album * > & album ) [pure virtual]
```

Implemented in [Digikam::AlbumFolderViewSideBarWidget](#), [Digikam::DateFolderViewSideBarWidget](#), [Digikam::FuzzySearchSideBarV](#), [Digikam::GPSSearchSideBarWidget](#), [Digikam::LabelsSideBarWidget](#), [Digikam::PeopleSideBarWidget](#), [Digikam::SearchSideBarWidg](#), [Digikam::TagViewSideBarWidget](#), and [Digikam::TimelineSideBarWidget](#).

### 9.1210.2.3 getCaption()

```
virtual const QString Digikam::SidebarWidget::getCaption ( ) [pure virtual]
```

#### Returns

localized title string

Implemented in [Digikam::AlbumFolderViewSideBarWidget](#), [Digikam::DateFolderViewSideBarWidget](#), [Digikam::FuzzySearchSideBarV](#), [Digikam::GPSSearchSideBarWidget](#), [Digikam::LabelsSideBarWidget](#), [Digikam::PeopleSideBarWidget](#), [Digikam::SearchSideBarWidg](#), [Digikam::TagViewSideBarWidget](#), and [Digikam::TimelineSideBarWidget](#).

### 9.1210.2.4 getIcon()

```
virtual const QIcon Digikam::SidebarWidget::getIcon ( ) [pure virtual]
```

#### Returns

pixmap icon

Implemented in [Digikam::AlbumFolderViewSideBarWidget](#), [Digikam::DateFolderViewSideBarWidget](#), [Digikam::FuzzySearchSideBarV](#), [Digikam::GPSSearchSideBarWidget](#), [Digikam::LabelsSideBarWidget](#), [Digikam::PeopleSideBarWidget](#), [Digikam::SearchSideBarWidg](#), [Digikam::TagViewSideBarWidget](#), and [Digikam::TimelineSideBarWidget](#).

### 9.1210.2.5 setActive()

```
virtual void Digikam::SidebarWidget::setActive (
    bool active ) [pure virtual]
```

#### Parameters

<i>active</i>	if true, this widget is the new active widget, if false another widget is active
---------------	--

Implemented in [Digikam::AlbumFolderViewSideBarWidget](#), [Digikam::DateFolderViewSideBarWidget](#), [Digikam::FuzzySearchSideBarV](#), [Digikam::GPSSearchSideBarWidget](#), [Digikam::LabelsSideBarWidget](#), [Digikam::PeopleSideBarWidget](#), [Digikam::SearchSideBarWidg](#), [Digikam::TagViewSideBarWidget](#), and [Digikam::TimelineSideBarWidget](#).

## 9.1211 Digikam::SidecarFinder Class Reference

### Public Member Functions

- **SidecarFinder** (const QList< QUrl > &files)

### Public Attributes

- QList< bool > **localFileModes**
- QList< QUrl > **localFiles**
- QList< QString > **localFileSuffixes**

## 9.1212 Digikam::SimilarityDb Class Reference

### Public Member Functions

- void **clearImageSimilarity** (FuzzyAlgorithm algorithm=FuzzyAlgorithm::Haar)  
*This method removes all image similarity entries for the algorithm.*
- void **copySimilarityAttributes** (qlonglong srcId, qlonglong destId)  
*Copies all similarity-specific information, from image srcId to destId.*
- QList< qlonglong > **getDirtyOrMissingFingerprints** (const QList< [ItemInfo](#) > &imageInfos, FuzzyAlgorithm algorithm=FuzzyAlgorithm::Haar)  
*Returns a list of all item ids (images, videos,...) where either no fingerprint for the given algorithm exists or is outdated because the file is identified as changed since the generation of the fingerprint.*
- QStringList **getDirtyOrMissingFingerprintURLs** (const QList< [ItemInfo](#) > &imageInfos, FuzzyAlgorithm algorithm=FuzzyAlgorithm::Haar)  
*Returns a list of the URLs of all items (images, videos,...) where either no fingerprint for the given algorithm exists or is outdated because the file is identified as changed since the generation of the fingerprint.*
- double **getImageSimilarity** (qlonglong imageID1, qlonglong imageID2, FuzzyAlgorithm algorithm=FuzzyAlgorithm::Haar)  
*Returns the similarity value for two images.*
- QList< FuzzyAlgorithm > **getImageSimilarityAlgorithms** (qlonglong imageID1, qlonglong imageID2)  
*Returns the algorithms for which a similarity value exists for the given image ids.*
- QString **getLegacySetting** (const QString &keyword)  
*Returns the legacy settings with the keyword name.*



- QString [getSetting](#) (const QString &keyword)  
*Returns the setting with the keyword name.*
- bool [hasDirtyOrMissingFingerprint](#) (const [ItemInfo](#) &imageInfo, FuzzyAlgorithm algorithm=FuzzyAlgorithm::Haar) const  
*Checks if the given image has a dirty fingerprint or even none for the given algorithm.*
- bool [hasFingerprint](#) (qulonglong imageId, FuzzyAlgorithm algorithm) const  
*This method checks if the given image has a fingerprint for the given algorithm.*
- bool [hasFingerprints](#) ()  
*This method checks if there are any fingerprints for any algorithm present.*
- bool [hasFingerprints](#) (FuzzyAlgorithm algorithm) const  
*This method checks if there are any fingerprints for the given algorithm.*
- bool [integrityCheck](#) ()  
*This method checks the integrity of the similarity database.*
- QSet< qulonglong > [registeredImageIds](#) () const  
*This method returns all image ids that are present in the similarity db tables.*
- void [removeImageFingerprint](#) (qulonglong imageId, FuzzyAlgorithm algorithm=FuzzyAlgorithm::Haar)  
*This method removes the fingerprint entry for the given imageId and algorithm.*
- void [removeImageSimilarity](#) (qulonglong imageID, FuzzyAlgorithm algorithm=FuzzyAlgorithm::Haar)  
*This method removes the image similarity entries for the imageID and algorithm.*
- void [removeImageSimilarity](#) (qulonglong imageID1, qulonglong imageID2, FuzzyAlgorithm algorithm=FuzzyAlgorithm::Haar)  
*This method removes the image similarity entry for the imageIDs and algorithm.*
- void [setImageSimilarity](#) (qulonglong imageID1, qulonglong imageID2, double value, FuzzyAlgorithm algorithm=FuzzyAlgorithm::Haar)
- bool [setSetting](#) (const QString &keyword, const QString &value)  
*Set the database setting entry given by keyword to the given value.*
- void [vacuum](#) ()  
*This method shrinks the database.*

## Friends

- class [SimilarityDbAccess](#)

## 9.1212.1 Member Function Documentation

### 9.1212.1.1 clearImageSimilarity()

```
void Digikam::SimilarityDb::clearImageSimilarity (
    FuzzyAlgorithm algorithm = FuzzyAlgorithm::Haar )
```

#### Parameters

<i>algorithm</i>	The algorithm.
------------------	----------------

### 9.1212.1.2 getDirtyOrMissingFingerprints()

```
QList< qulonglong > Digikam::SimilarityDb::getDirtyOrMissingFingerprints (
```

```
const QList< ItemInfo > & imageInfos,
FuzzyAlgorithm algorithm = FuzzyAlgorithm::Haar )
```

**Parameters**

<i>imageInfos</i>	The image info objects representing the items.
<i>algorithm</i>	The algorithm.

**Returns**

The ids of the items whose fingerprints are dirty or missing.

**9.1212.1.3 getDirtyOrMissingFingerprintURLs()**

```
QStringList Digikam::SimilarityDb::getDirtyOrMissingFingerprintURLs (
    const QList< ItemInfo > & imageInfos,
    FuzzyAlgorithm algorithm = FuzzyAlgorithm::Haar )
```

**Parameters**

<i>imageInfos</i>	The image info objects representing the items.
<i>algorithm</i>	The algorithm.

**Returns**

The URLs of the items whose fingerprints are dirty or missing.

**9.1212.1.4 getImageSimilarity()**

```
double Digikam::SimilarityDb::getImageSimilarity (
    qlonglong imageID1,
    qlonglong imageID2,
    FuzzyAlgorithm algorithm = FuzzyAlgorithm::Haar )
```

A value of -1 means nonexistence. A value of -2 means that there is a value that cannot be converted into a double

**9.1212.1.5 getImageSimilarityAlgorithms()**

```
QList< FuzzyAlgorithm > Digikam::SimilarityDb::getImageSimilarityAlgorithms (
    qlonglong imageID1,
    qlonglong imageID2 )
```

**Parameters**

<i>imageID1</i>	The first image id.
<i>imageID2</i>	The second image id.

**Returns**

a list of all algorithms for which a similarity value exists.

**9.1212.1.6 getLegacySetting()**

```
QString Digikam::SimilarityDb::getLegacySetting (
    const QString & keyword )
```

**Parameters**

<i>keyword</i>	The setting entry name.
----------------	-------------------------

**Returns**

The setting value.

**9.1212.1.7 getSetting()**

```
QString Digikam::SimilarityDb::getSetting (
    const QString & keyword )
```

**Parameters**

<i>keyword</i>	The setting entry name.
----------------	-------------------------

**Returns**

The setting value.

**9.1212.1.8 hasDirtyOrMissingFingerprint()**

```
bool Digikam::SimilarityDb::hasDirtyOrMissingFingerprint (
    const ItemInfo & imageInfo,
    FuzzyAlgorithm algorithm = FuzzyAlgorithm::Haar ) const
```

**Parameters**

<i>imageInfo</i>	The image info object representing the item.
<i>algorithm</i>	The algorithm used for the fingerprint.

**Returns**

True, if the image either has no or a dirty fingerprint.

### 9.1212.1.9 hasFingerprint()

```
bool Digikam::SimilarityDb::hasFingerprint (
    qlonglong imageId,
    FuzzyAlgorithm algorithm ) const
```

#### Parameters

<i>imageId</i>	The Id of the image to check.
<i>algorithm</i>	The algorithm.

#### Returns

True, if there is a fingerprint.

### 9.1212.1.10 hasFingerprints() [1/2]

```
bool Digikam::SimilarityDb::hasFingerprints ( )
```

#### Returns

True, if fingerprints exist.

### 9.1212.1.11 hasFingerprints() [2/2]

```
bool Digikam::SimilarityDb::hasFingerprints (
    FuzzyAlgorithm algorithm ) const
```

#### Parameters

<i>algorithm</i>	The algorithm.
------------------	----------------

#### Returns

true, if there are fingerprints and false, otherwise.

### 9.1212.1.12 integrityCheck()

```
bool Digikam::SimilarityDb::integrityCheck ( )
```

#### Returns

true, if the integrity check was passed and false, else.

**9.1212.1.13 registeredImageIds()**

```
QSet< qulonglong > Digikam::SimilarityDb::registeredImageIds ( ) const
```

**Returns**

a set of all present image ids.

**9.1212.1.14 removeImageFingerprint()**

```
void Digikam::SimilarityDb::removeImageFingerprint (
    qulonglong imageID,
    FuzzyAlgorithm algorithm = FuzzyAlgorithm::Haar )
```

Also, this automatically removes the entries in the ImageSimilarities table for the given algorithm and image id.

**Parameters**

<i>imageID</i>	The image id.
<i>algorithm</i>	The algorithm.

**9.1212.1.15 removeImageSimilarity() [1/2]**

```
void Digikam::SimilarityDb::removeImageSimilarity (
    qulonglong imageID,
    FuzzyAlgorithm algorithm = FuzzyAlgorithm::Haar )
```

**Parameters**

<i>imageID</i>	The image id.
<i>algorithm</i>	The algorithm.

**9.1212.1.16 removeImageSimilarity() [2/2]**

```
void Digikam::SimilarityDb::removeImageSimilarity (
    qulonglong imageID1,
    qulonglong imageID2,
    FuzzyAlgorithm algorithm = FuzzyAlgorithm::Haar )
```

**Parameters**

<i>imageID1</i>	The first image id.
<i>imageID2</i>	The second image id.
<i>algorithm</i>	The algorithm.

### 9.1212.1.17 setSetting()

```
bool Digikam::SimilarityDb::setSetting (
    const QString & keyword,
    const QString & value )
```

#### Parameters

<i>keyword</i>	The keyword, i.e. setting name.
<i>value</i>	The value.

#### Returns

True, if the value was set and false, else..

## 9.1213 Digikam::SimilarityDbAccess Class Reference

### Public Member Functions

- [SimilarityDbAccess](#) ()  
*This class is written in analogy to [CoreDbAccess](#) (some features stripped off).*
- [SimilarityDbBackend](#) \* **backend** () const
- [SimilarityDb](#) \* **db** () const
- QString **lastError** () const
- void [setLastError](#) (const QString &error)  
*Set the "last error" message.*

### Static Public Member Functions

- static bool [checkReadyForUse](#) ([InitializationObserver](#) \*const observer)  
*This static method checks if the similarity db is ready for use.*
- static void **cleanUpDatabase** ()  
*This static method removes the connection to the similarity database.*
- static void [initDbEngineErrorHandler](#) ([DbEngineErrorHandler](#) \*const errorhandler)  
*This static method initialises the error handler for the similarity db.*
- static bool [isInitialized](#) ()  
*This static method returns if the similarity db is initialised.*
- static [DbEngineParameters](#) [parameters](#) ()  
*This static method returns the current db parameters.*
- static void [setParameters](#) (const [DbEngineParameters](#) &[parameters](#))  
*This static method sets the database parameters that are needed to initialise the db connection.*

## 9.1213.1 Constructor & Destructor Documentation

### 9.1213.1.1 SimilarityDbAccess()

```
Digikam::SimilarityDbAccess::SimilarityDbAccess ( )
```

For documentation, see [coredbaccess.h](#)

## 9.1213.2 Member Function Documentation

### 9.1213.2.1 checkReadyForUse()

```
bool Digikam::SimilarityDbAccess::checkReadyForUse (
    InitializationObserver *const observer ) [static]
```

#### Parameters

<i>observer</i>	the observer.
-----------------	---------------

#### Returns

true, if the database is ready for use.

### 9.1213.2.2 initDbEngineErrorHandler()

```
void Digikam::SimilarityDbAccess::initDbEngineErrorHandler (
    DbEngineErrorHandler *const errorhandler ) [static]
```

#### Parameters

<i>errorhandler</i>	The error handler.
---------------------	--------------------

### 9.1213.2.3 isInitialized()

```
bool Digikam::SimilarityDbAccess::isInitialized ( ) [static]
```

#### Returns

true, if the similarityDb is initialised.

### 9.1213.2.4 parameters()

```
DbEngineParameters Digikam::SimilarityDbAccess::parameters ( ) [static]
```

#### Returns

the current db parameters.

### 9.1213.2.5 setLastError()

```
void Digikam::SimilarityDbAccess::setLastError (
    const QString & error )
```

This method is not for public use.

### 9.1213.2.6 setParameters()

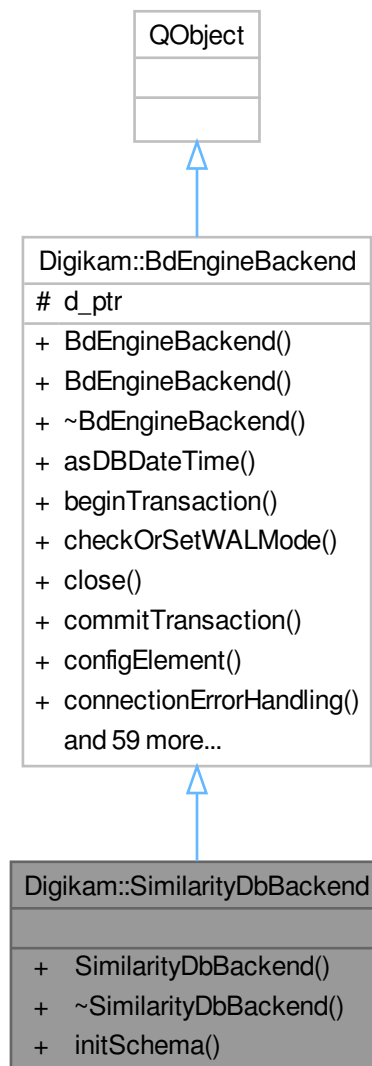
```
void Digikam::SimilarityDbAccess::setParameters (
    const DbEngineParameters & parameters ) [static]
```

## Parameters

<i>parameters</i>	The db parameters.
-------------------	--------------------

## 9.1214 Digikam::SimilarityDbBackend Class Reference

Inheritance diagram for Digikam::SimilarityDbBackend:



### Public Member Functions

- **SimilarityDbBackend** ([DbEngineLocking](#) \*const locking, const QString &backendName=QLatin1String("similarityDatabase-"))
- bool **initSchema** ([SimilarityDbSchemaUpdater](#) \*const updater)  
*Initialize the database schema to the current version, carry out upgrades if necessary.*



## Public Member Functions inherited from Digikam::BdEngineBackend

- [BdEngineBackend](#) (const QString &backendName, [DbEngineLocking](#) \*const locking)  
*Creates a database backend.*
- **BdEngineBackend** (const QString &backendName, [DbEngineLocking](#) \*const locking, [BdEngineBackend](#)←Private &dd)
- QDateTime [asDBDateTime](#) (const QDateTime &dateTime) const  
*Depending on the database backend return a local or UTC date format.*
- [BdEngineBackend::QueryState](#) **beginTransaction** ()  
*Begin a database transaction.*
- bool [checkOrSetWALMode](#) ()  
*Check or set WAL mode for SQLite database if enabled in settings.*
- void **close** ()  
*Close the database connection.*
- [BdEngineBackend::QueryState](#) **commitTransaction** ()  
*Commit the current database transaction.*
- [DbEngineConfigSettings](#) **configElement** () const  
*Return config read from XML, corresponding to this backend's database type.*
- bool [connectionErrorHandling](#) (int retries)  
*Called when an attempted connection to the database failed.*
- [DbEngineSqlQuery](#) **copyQuery** (const [DbEngineSqlQuery](#) &old)  
*Creates a faithful copy of the passed query, with the current db connection.*
- DbType **databaseType** () const  
*Return the database type.*
- bool **exec** ([DbEngineSqlQuery](#) &query)  
*Calls exec/execBatch on the query, and handles debug output if something went wrong.*
- bool **execBatch** ([DbEngineSqlQuery](#) &query)
- [QueryState](#) **execDBAction** (const [DbEngineAction](#) &action, const QMap< QString, QVariant > &bindingMap, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)  
*Performs the database action on the current database.*
- [QueryState](#) **execDBAction** (const [DbEngineAction](#) &action, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)  
*Performs the database action on the current database.*
- [QueryState](#) **execDBAction** (const QString &action, const QMap< QString, QVariant > &bindingMap, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execDBAction** (const QString &action, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- QSqlQuery [execDBActionQuery](#) (const [DbEngineAction](#) &action, const QMap< QString, QVariant > &bindingMap)  
*Performs the database action on the current database.*
- QSqlQuery **execDBActionQuery** (const QString &action, const QMap< QString, QVariant > &bindingMap)
- [QueryState](#) **execDirectSql** (const QString &query)  
*Calls exec on the query, and handles debug output if something went wrong.*
- [QueryState](#) **execDirectSqlWithResult** (const QString &query, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)  
*Calls exec on the query, and handles debug output if something went wrong.*
- [DbEngineSqlQuery](#) **execQuery** (const QString &sql)  
*Executes the statement and returns the query object.*
- [DbEngineSqlQuery](#) **execQuery** (const QString &sql, const QList< QVariant > &boundValues)
- [DbEngineSqlQuery \*\*execQuery\*\* \(const QString &sql, const QMap< QString, QVariant > &bindingMap\)  
\*Method which accept a hashmap with key, values which are used for named binding.\*](#)
- [DbEngineSqlQuery](#) **execQuery** (const QString &sql, const QVariant &boundValue1)

- [DbEngineSqlQuery](#) **execQuery** (const QString &sql, const QVariant &bindValue1, const QVariant &bindValue2)
- [DbEngineSqlQuery](#) **execQuery** (const QString &sql, const QVariant &bindValue1, const QVariant &bindValue2, const QVariant &bindValue3)
- [DbEngineSqlQuery](#) **execQuery** (const QString &sql, const QVariant &bindValue1, const QVariant &bindValue2, const QVariant &bindValue3, const QVariant &bindValue4)
- void **execQuery** ([DbEngineSqlQuery](#) &preparedQuery, const QList< QVariant > &boundValues)
- void **execQuery** ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &bindValue1)
  - Binds the values and executes the prepared query.*
- void **execQuery** ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &bindValue1, const QVariant &bindValue2)
- void **execQuery** ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &bindValue1, const QVariant &bindValue2, const QVariant &bindValue3)
- void **execQuery** ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &bindValue1, const QVariant &bindValue2, const QVariant &bindValue3, const QVariant &bindValue4)
- [QueryState](#) **execSql** (const QString &sql, const QList< QVariant > &boundValues, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** (const QString &sql, const QMap< QString, QVariant > &bindingMap, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
  - Method which accepts a map for named binding.*
- [QueryState](#) **execSql** (const QString &sql, const QVariant &bindValue1, const QVariant &bindValue2, const QVariant &bindValue3, const QVariant &bindValue4, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** (const QString &sql, const QVariant &bindValue1, const QVariant &bindValue2, const QVariant &bindValue3, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** (const QString &sql, const QVariant &bindValue1, const QVariant &bindValue2, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** (const QString &sql, const QVariant &bindValue1, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** (const QString &sql, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
  - Executes the SQL statement, and write the returned data into the values list.*
- [QueryState](#) **execSql** ([DbEngineSqlQuery](#) &preparedQuery, const QList< QVariant > &boundValues, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &bindValue1, const QVariant &bindValue2, const QVariant &bindValue3, const QVariant &bindValue4, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &bindValue1, const QVariant &bindValue2, const QVariant &bindValue3, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &bindValue1, const QVariant &bindValue2, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &bindValue1, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** ([DbEngineSqlQuery](#) &preparedQuery, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execUpsertDBAction** (const [DbEngineAction](#) &action, const QVariant &id, const QStringList &fieldNames, const QList< QVariant > &values)
  - Performs a special DBAction that is usually needed to "INSERT or UPDATE" entries in a table.*
- [QueryState](#) **execUpsertDBAction** (const QString &action, const QVariant &id, const QStringList &fieldNames, const QList< QVariant > &values)
- [DbEngineAction](#) **getDBAction** (const QString &actionName) const
  - Returns a database action with name, specified in actionName, for the current database.*
- [DbEngineSqlQuery](#) **getQuery** ()

- Creates an empty query object waiting for the statement.*

  - [QueryState handleQueryResult](#) ([DbEngineSqlQuery](#) &query, [QList< QVariant >](#) \*const values, [QVariant](#) \*const lastInsertId)
- Checks if there was a connection error.*

  - bool **isCompatible** (const [DbEngineParameters](#) &parameters)
- Checks if the parameters can be used for this database backend.*

  - bool **isInTransaction** () const
- Returns if the database is in a different thread in a transaction.*

  - bool **isOpen** () const
  - bool **isReady** () const
  - [QString lastError](#) ()
- Returns a description of the last error that occurred on this database.*

  - [QSqlError lastSQLError](#) ()
- Returns the last error that occurred on this database.*

  - int **maximumBoundValues** () const
- Returns the maximum number of bound parameters allowed per query.*

  - bool **open** (const [DbEngineParameters](#) &parameters)
- Open the database connection.*

  - [DbEngineSqlQuery prepareQuery](#) (const [QString](#) &sql)
- Creates a query object prepared with the statement, waiting for bound values.*

  - bool **queryErrorHandling** ([DbEngineSqlQuery](#) &query, int retries)
- Called with a failed query.*

  - [QList< QVariant >](#) **readToList** ([DbEngineSqlQuery](#) &query)
- Reads data of returned result set into a list which is returned.*

  - void **rollbackTransaction** ()
- Rollback the current database transaction.*

  - void **setDbEngineErrorHandler** ([DbEngineErrorHandler](#) \*const handler)
- Add a [DbEngineErrorHandler](#).*

  - void **setForeignKeyChecks** (bool check)
- Enables or disables FOREIGN\_KEY\_CHECKS for the database.*

  - [Status status](#) () const
- Returns the current status of the database backend.*

  - [QStringList tables](#) ()
- Returns a list with the names of tables in the database.*

  - bool **transactionErrorHandling** (const [QSqlError](#) &[lastError](#), int retries)

### Additional Inherited Members

### Public Types inherited from [Digikam::BdEngineBackend](#)

- enum **DbType** { [SQLite](#) , [MySQL](#) }
- enum **QueryOperationStatus** { [ExecuteNormal](#) , [Wait](#) , [AbortQueries](#) }
- enum **QueryStateEnum** { [NoErrors](#) , [SQLError](#) , [ConnectionError](#) }
- enum **Status** { [Unavailable](#) , [Open](#) , [OpenSchemaChecked](#) }

### Protected Attributes inherited from [Digikam::BdEngineBackend](#)

- [BdEngineBackendPrivate](#) \*const **d\_ptr** = nullptr

## 9.1214.1 Member Function Documentation

### 9.1214.1.1 initSchema()

```
bool Digikam::SimilarityDbBackend::initSchema (
    SimilarityDbSchemaUpdater *const updater )
```

Shall only be called from the thread that called [open\(\)](#).

## 9.1215 Digikam::SimilarityDbSchemaUpdater Class Reference

### Public Member Functions

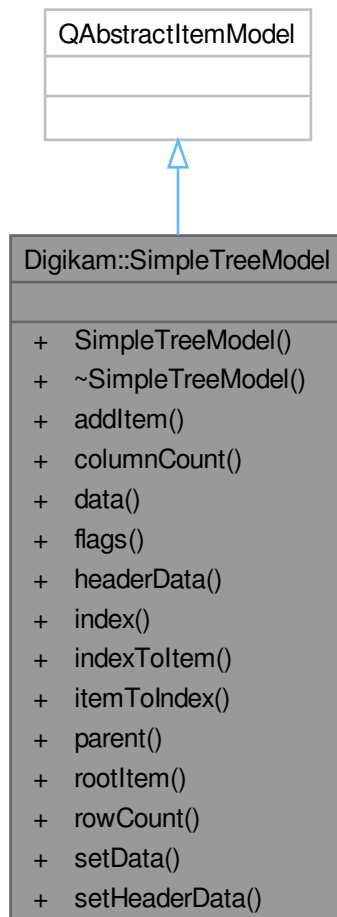
- [SimilarityDbSchemaUpdater](#) ([SimilarityDbAccess](#) \*const dbAccess)
- void [setObserver](#) ([InitializationObserver](#) \*const observer)
- bool [update](#) ()

### Static Public Member Functions

- static int [schemaVersion](#) ()

## 9.1216 Digikam::SimpleTreeModel Class Reference

Inheritance diagram for Digikam::SimpleTreeModel:



### Classes

- class [Item](#)

### Public Member Functions

- **SimpleTreeModel** (const int [columnCount](#), QObject \*const parent=nullptr)
- **Item \* addItem** (Item \*const parentItem=nullptr, const int rowNumber=-1)
- int **columnCount** (const QModelIndex &parent=QModelIndex()) const override  
*QAbstractItemModel:*
- QVariant **data** (const QModelIndex &index, int role=Qt::DisplayRole) const override
- Qt::ItemFlags **flags** (const QModelIndex &index) const override
- QVariant **headerData** (int section, Qt::Orientation orientation, int role) const override

- QModelIndex **index** (int row, int column, const QModelIndex &parent=QModelIndex()) const override
- Item \* **indexToItem** (const QModelIndex &itemIndex) const
- QModelIndex **itemToIndex** (const Item \*const item) const
- QModelIndex **parent** (const QModelIndex &index) const override
- Item \* **rootItem** () const
- int **rowCount** (const QModelIndex &parent=QModelIndex()) const override
- bool **setData** (const QModelIndex &index, const QVariant &value, int role) override
- bool **setHeaderData** (int section, Qt::Orientation orientation, const QVariant &value, int role) override

## 9.1217 Digikam::SimpleTreeModel::Item Class Reference

### Public Attributes

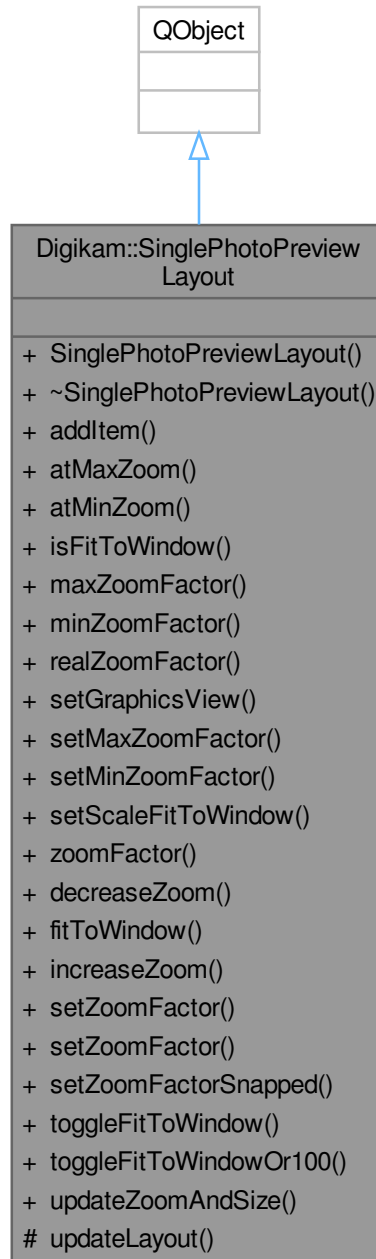
- QString **data**

### Friends

- class **SimpleTreeModel**

## 9.1218 Digikam::SinglePhotoPreviewLayout Class Reference

Inheritance diagram for Digikam::SinglePhotoPreviewLayout:



### Public Types

- enum **SetZoomFlag** { **JustSetFactor** = 0 , **CenterView** = 1 << 0 , **SnapZoomFactor** = 1 << 1 }
- typedef QFlags< SetZoomFlag > **SetZoomFlags**

## Public Slots

- void **decreaseZoom** (const QPoint &viewportAnchor=QPoint())
- void **fitToWindow** ()
- void **increaseZoom** (const QPoint &viewportAnchor=QPoint())
- void **setZoomFactor** (double z, const QPoint &viewportAnchor=QPoint(), SetZoomFlags flags=JustSet←Factor)
- void **setZoomFactor** (double z, SetZoomFlags flags)
- void **setZoomFactorSnapped** (double z)
- void **toggleFitToWindow** ()  
*Toggle between fitToWindow and previous zoom factor.*
- void **toggleFitToWindowOr100** ()  
*Toggle between fitToWindow and zoom factor 1.*
- void **updateZoomAndSize** ()  
*Update settings when size of image or view changed.*

## Signals

- void **fitToWindowToggled** (bool fitToWindow)
- void **zoomFactorChanged** (double)

## Public Member Functions

- **SinglePhotoPreviewLayout** (QObject \*const parent)
- void **addItem** (GraphicsDImgItem \*const item)  
*Set the item to layout.*
- bool **atMaxZoom** () const
- bool **atMinZoom** () const
- bool **isFitToWindow** () const
- double **maxZoomFactor** () const  
*The zoom range for incrementing and decrementing.*
- double **minZoomFactor** () const
- double **realZoomFactor** () const
- void **setGraphicsView** (GraphicsDImgView \*const view)  
*Set the graphics view, and associated scene, to operate on.*
- void **setMaxZoomFactor** (double z)
- void **setMinZoomFactor** (double z)
- void **setScaleFitToWindow** (bool value)  
*Set to true to scale small images to fit to window.*
- double **zoomFactor** () const

## Protected Member Functions

- void **updateLayout** ()

## 9.1218.1 Member Function Documentation

### 9.1218.1.1 addItem()

```
void Digikam::SinglePhotoPreviewLayout::addItem (
    GraphicsDImgItem *const item )
```

For a SinglePhoto layout, typically, you can add only one item.



## 9.1219 Digikam::SketchWidget Class Reference

Inheritance diagram for Digikam::SketchWidget:



### Public Slots

- void **setPenColor** (const QColor &newColor)
- void **setPenWidth** (int newWidth)
- void **slotClear** ()
- void **slotRedo** ()
- void **slotUndo** ()

## Signals

- void **signalPenColorChanged** (const QColor &)
- void **signalPenSizeChanged** (int)
- void **signalSketchChanged** (const QImage &)
- void **signalUndoRedoStateChanged** (bool hasUndo, bool hasRedo)

## Public Member Functions

- **SketchWidget** (QWidget \*const parent=nullptr)
- bool **isClear** () const
- QColor **penColor** () const
- int **penWidth** () const
- void **setSketchImage** (const QImage &image)
- bool **setSketchImageFromXML** (const QString &xml)
- bool **setSketchImageFromXML** (QXmlStreamReader &reader)

*This method set sketch image using XML data based on drawing line history.*

- QImage **sketchImage** () const
- QString **sketchImageToXML** ()
- void **sketchImageToXML** (QXmlStreamWriter &writer)

*This method return the drawing line history as XML, to be stored in database as [SAIbum](#) data.*

## Protected Member Functions

- void **keyPressEvent** (QKeyEvent \*) override
- void **keyReleaseEvent** (QKeyEvent \*) override
- void **mouseMoveEvent** (QMouseEvent \*) override
- void **mousePressEvent** (QMouseEvent \*) override
- void **mouseReleaseEvent** (QMouseEvent \*) override
- void **paintEvent** (QPaintEvent \*) override
- void **wheelEvent** (QWheelEvent \*) override

## 9.1219.1 Member Function Documentation

### 9.1219.1.1 setSketchImageFromXML()

```
bool Digikam::SketchWidget::setSketchImageFromXML (
    QXmlStreamReader & reader )
```

Return true if data are imported successfully.

## 9.1220 Digikam::SlideVideo Class Reference

Inheritance diagram for Digikam::SlideVideo:



### Public Slots

- void **slotPositionChanged** (int position)
- void **slotVolumeChanged** (int volume)

### Signals

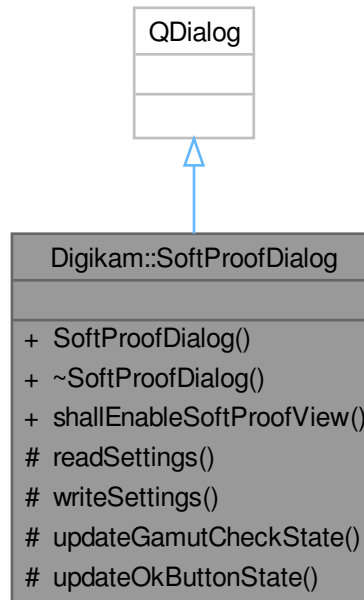
- void **signalVideoDuration** (qint64)
- void **signalVideoFinished** ()
- void **signalVideoLoaded** (bool)
- void **signalVideoPosition** (qint64)
- void **signalVideoVolume** (int)

### Public Member Functions

- **SlideVideo** (QWidget \*const parent)
- void **backward** ()
- void **forward** ()
- void **pause** (bool)
- void **setCurrentUrl** (const QUrl &url)
- void **setInfoInterface** ([DInfoInterface](#) \*const iface)
- void **stop** ()

## 9.1221 Digikam::SoftProofDialog Class Reference

Inheritance diagram for Digikam::SoftProofDialog:



### Public Member Functions

- **SoftProofDialog** (QWidget \*const parent)
- bool **shallEnableSoftProofView** () const

### Protected Slots

- void **updateGamutCheckState** ()
- void **updateOkButtonState** ()

### Protected Member Functions

- void **readSettings** ()
- void **writeSettings** ()

## 9.1222 Digikam::SolidHardwareDlg Class Reference

Inheritance diagram for Digikam::SolidHardwareDlg:



### Public Member Functions

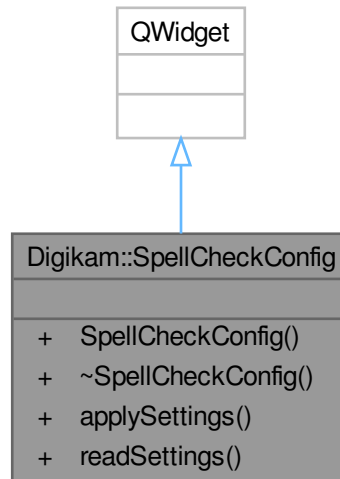
- **SolidHardwareDlg** (QWidget \*const parent)

### Public Member Functions inherited from [Digikam::InfoDlg](#)

- **InfoDlg** (QWidget \*const parent)
- QDialogButtonBox \* **buttonBox** () const
- QTreeWidget \* **listView** () const
- QWidget \* **mainWidget** () const
- virtual void **setInfoMap** (const QMap< QString, QString > &list)
- QTabWidget \* **tabView** () const

## 9.1223 Digikam::SpellCheckConfig Class Reference

Inheritance diagram for Digikam::SpellCheckConfig:



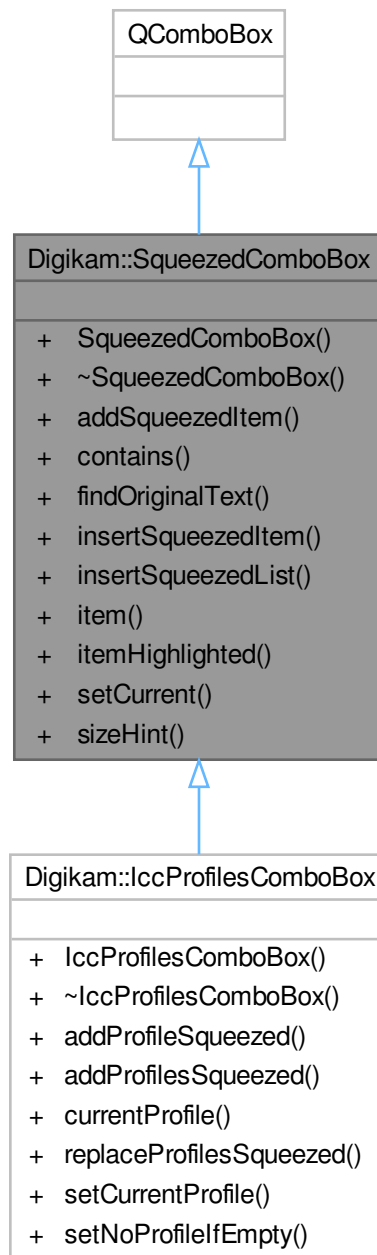
### Public Member Functions

- **SpellCheckConfig** (`QWidget *const parent=nullptr`)
- void **applySettings** ()
- void **readSettings** ()

## 9.1224 Digikam::SqueezedComboBox Class Reference

This widget is a `QComboBox`, but then a little bit different.

Inheritance diagram for Digikam::SqueezedComboBox:



## Signals

- void **signalItemActivated** (const QString &)

## Public Member Functions

- [SqueezedComboBox](#) (QWidget \*const parent=nullptr, const char \*name=nullptr)

- Constructor.*
- `~SqueezedComboBox ()` override
  - destructor*
  - void `addSqueezedItem` (const QString &newItem, const QVariant &userData=QVariant())  
*Append an item.*
  - bool `contains` (const QString &text) const  
*Returns true if the combobox contains the original (not-squeezed) version of text.*
  - int `findOriginalText` (const QString &text, Qt::CaseSensitivity cs=Qt::CaseSensitive) const  
*Returns the index of the combobox if found the original (not-squeezed) version of text.*
  - void `insertSqueezedItem` (const QString &newItem, int index, const QVariant &userData=QVariant())  
*This inserts a item to the list.*
  - void `insertSqueezedList` (const QStringList &newItems, int index)  
*This inserts items to the list.*
  - QString `item` (int index) const  
*This method returns the full text (not squeezed) for the index.*
  - QString `itemHighlighted` () const  
*This method returns the full text (not squeezed) of the currently highlighted item.*
  - void `setCurrent` (const QString &itemText)  
*Set the current item to the one matching the given text.*
  - QSize `sizeHint` () const override  
*Sets the `sizeHint()` of this widget.*

### 9.1224.1 Detailed Description

It only shows the right part of the items depending on de size of the widget. When it is not possible to show the complete item, it will be shortened and "..." will be prepended.

### 9.1224.2 Constructor & Destructor Documentation

#### 9.1224.2.1 SqueezedComboBox()

```
Digikam::SqueezedComboBox::SqueezedComboBox (
    QWidget *const parent = nullptr,
    const char * name = nullptr ) [explicit]
```

#### Parameters

<i>parent</i>	the parent widget
<i>name</i>	the name to give to the widget

### 9.1224.3 Member Function Documentation

#### 9.1224.3.1 addSqueezedItem()

```
void Digikam::SqueezedComboBox::addSqueezedItem (
    const QString & newItem,
    const QVariant & userData = QVariant() )
```



## Parameters

<i>newItem</i>	the original (long version) of the item which needs to be added to the combobox
<i>userData</i>	custom meta-data assigned to new item.

**9.1224.3.2 contains()**

```
bool Digikam::SqueezedComboBox::contains (
    const QString & text ) const
```

## Parameters

<i>text</i>	the original (not-squeezed) text to check for
-------------	---

**9.1224.3.3 findOriginalText()**

```
int Digikam::SqueezedComboBox::findOriginalText (
    const QString & text,
    Qt::CaseSensitivity cs = Qt::CaseSensitive ) const
```

## Parameters

<i>text</i>	the original (not-squeezed) text to find for
<i>cs</i>	case sensitive or case insensitive search

**9.1224.3.4 insertSqueezedItem()**

```
void Digikam::SqueezedComboBox::insertSqueezedItem (
    const QString & newItem,
    int index,
    const QVariant & userData = QVariant() )
```

See `QComboBox::insertItem()` for details. Please do not use `QComboBox::insertItem()` to this widget, as that will fail.

## Parameters

<i>newItem</i>	the original (long version) of the item which needs to be added to the combobox
<i>index</i>	the position in the widget.
<i>userData</i>	custom meta-data assigned to new item.

**9.1224.3.5 insertSqueezedList()**

```
void Digikam::SqueezedComboBox::insertSqueezedList (
    const QStringList & newItems,
    int index )
```

See `QComboBox::insertItems()` for details. Please do not use `QComboBox::insertItems()` to this widget, as that will fail.

#### Parameters

<i>newItems</i>	the originals (long version) of the items which needs to be added to the combobox
<i>index</i>	the position in the widget.

#### 9.1224.3.6 `item()`

```
QString Digikam::SqueezedComboBox::item (
    int index ) const
```

#### Parameters

<i>index</i>	the position in the widget.
--------------	-----------------------------

#### Returns

full text of the item

#### 9.1224.3.7 `itemHighlighted()`

```
QString Digikam::SqueezedComboBox::itemHighlighted ( ) const
```

#### Returns

full text of the highlighted item

#### 9.1224.3.8 `setCurrent()`

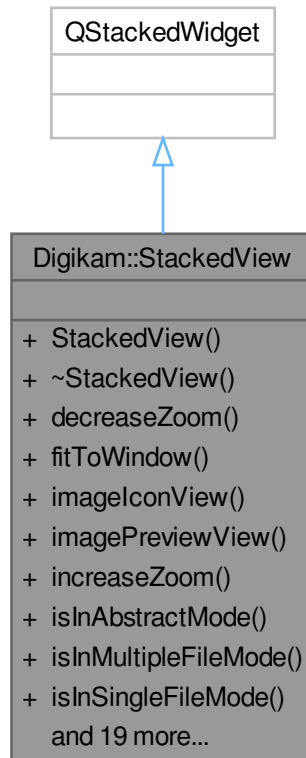
```
void Digikam::SqueezedComboBox::setCurrent (
    const QString & itemText )
```

#### Parameters

<i>itemText</i>	the original (long version) of the item text
-----------------	--

## 9.1225 Digikam::StackedView Class Reference

Inheritance diagram for Digikam::StackedView:



### Public Types

- enum `StackedViewMode` {  
**StackedViewModeFirst** = 0 , **IconViewMode** = 0 , **PreviewImageMode** = 1 , **WelcomePageMode** = 2 ,  
**TableViewMode** = 3 , **TrashViewMode** = 4 , **MapWidgetMode** = 5 , **MediaPlayerMode** = 6 ,  
**StackedViewModeLast** = 6 }

### Signals

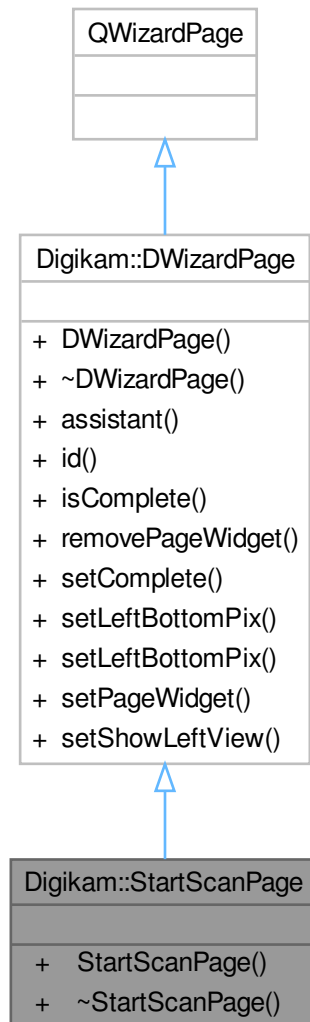
- void **signalAddToExistingQueue** (int)
- void **signalDeleteItem** ()
- void **signalEscapePreview** ()
- void **signalGotoAlbumAndItem** (const [ItemInfo](#) &)
- void **signalGotoDateAndItem** (const [ItemInfo](#) &)
- void **signalGotoTagAndItem** (int)
- void **signalNextItem** ()
- void **signalPopupTagsView** ()
- void **signalPrevItem** ()
- void **signalViewModeChanged** ()
- void **signalZoomFactorChanged** (double)

## Public Member Functions

- **StackedView** (QWidget \*const parent=nullptr)
  - void **decreaseZoom** ()
  - void **fitToWindow** ()
  - [DigikamItemView](#) \* **imageIconView** () const
  - [ItemPreviewView](#) \* **imagePreviewView** () const
  - void **increaseZoom** ()
  - bool **isInAbstractMode** () const
  - bool **isInMultipleFileMode** () const
  - bool **isInSingleFileMode** () const
- Single-file mode is image preview or media player, multi-file is icon view or map, abstract modes do not handle files (welcome page)*
- [MapWidgetView](#) \* **mapWidgetView** () const
  - bool **maxZoom** ()
  - bool **minZoom** ()
  - void **previewLoaded** ()
  - void **setDockArea** (QMainWindow \*)
  - void **setPreviewItem** (const [ItemInfo](#) &info=[ItemInfo](#)(), const [ItemInfo](#) &previous=[ItemInfo](#)(), const [ItemInfo](#) &next=[ItemInfo](#)())
  - void **setViewMode** (const StackedViewMode mode, bool focus=false)
  - void **setZoomFactor** (double z)
  - void **setZoomFactorSnapped** (double z)
  - [TableView](#) \* **tableView** () const
  - [ItemThumbnailBar](#) \* **thumbBar** () const
  - [ThumbBarDock](#) \* **thumbBarDock** () const
  - void **toggleFitToWindowOr100** ()
  - [TrashView](#) \* **trashView** () const
  - StackedViewMode **viewMode** () const
  - double **zoomFactor** ()
  - double **zoomMax** ()
  - double **zoomMin** ()
  - void **zoomTo100Percents** ()

## 9.1226 Digikam::StartScanPage Class Reference

Inheritance diagram for Digikam::StartScanPage:



### Public Member Functions

- `StartScanPage` (`QWizard *const dlg`)

### Public Member Functions inherited from [Digikam::DWizardPage](#)

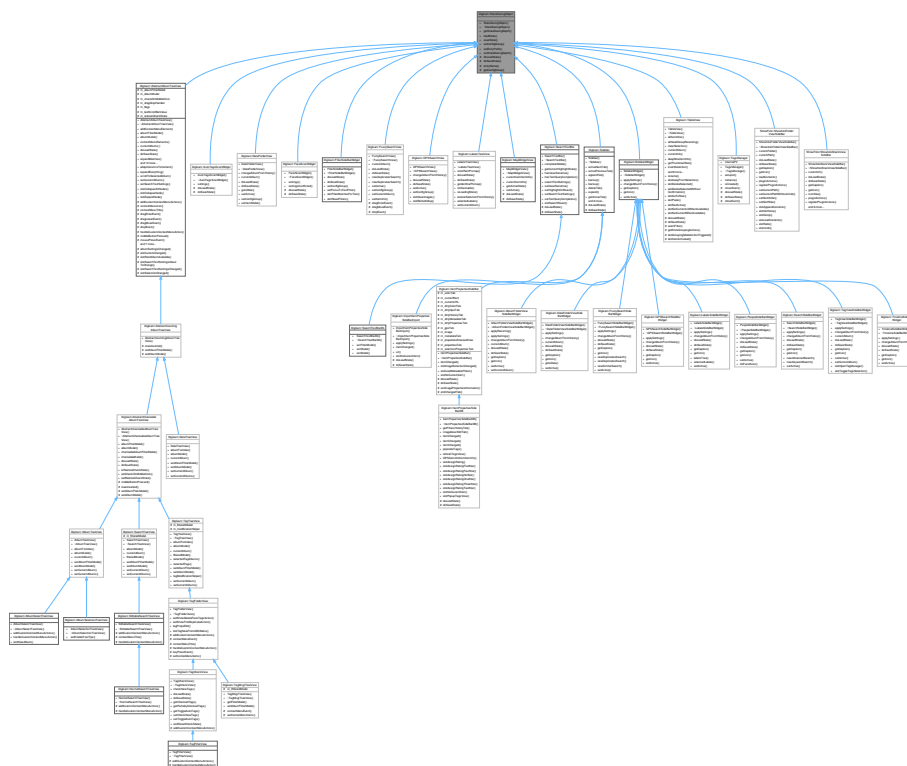
- `DWizardPage` (`QWizard *const dlg, const QString &title`)
- `QWizard * assistant () const`
- `int id () const`
- `bool isComplete () const` override

- void **removePageWidget** (QWidget \*const w)
- void **setComplete** (bool b)
- void **setLeftBottomPix** (const QIcon &icon)
- void **setLeftBottomPix** (const QPixmap &pix)
- void **setPageWidget** (QWidget \*const w)
- void **setShowLeftView** (bool v)

## 9.1227 Digikam::StateSavingObject Class Reference

An interface-like class with utility methods and a general public interface to support state saving and restoring for objects via KConfig.

Inheritance diagram for Digikam::StateSavingObject:



### Public Types

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }
- This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

### Public Member Functions

- [StateSavingObject](#) (QObject \*const host)
- Constructor.*
- virtual [~StateSavingObject](#) ()
- Destructor.*
- [StateSavingDepth](#) [getStateSavingDepth](#) () const

- Returns the depth used for state saving or loading.*
- void **loadState** ()
  - Invokes loading the class' state.*
- void **saveState** ()
  - Invokes saving the class' state.*
- virtual void **setConfigGroup** (const KConfigGroup &group)
  - Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void **setEntryPrefix** (const QString &prefix)
  - Define a prefix that will be used for every entry in the config group.*
- void **setStateSavingDepth** (const StateSavingDepth depth)
  - Sets the depth used for state saving or loading.*

### Protected Member Functions

- virtual void **doLoadState** ()=0
  - Implement this hook method for state loading.*
- virtual void **doSaveState** ()=0
  - Implement this hook method for state saving.*
- QString **entryName** (const QString &base) const
  - Always use this method to create config group entry names.*
- KConfigGroup **getConfigGroup** () const
  - Returns the config group that must be used for state saving and loading.*

## 9.1227.1 Detailed Description

Use this class as a Mixin.

The public interface for loading and saving state is implemented designed as template methods. To store or restore the state of a class, inherit from this class via multiple inheritance and implement [doLoadState\(\)](#) and [doSaveState\(\)](#). In these methods always use the protected method [getConfigGroup\(\)](#) to access a config group. Also always use the [entryName\(\)](#) method for generating keys in the config (for prefixes, see below).

Ensure that this class is inherited after a QObject-based class and pass "this" as constructor argument.

By default a config group based on Qt's object name of the class is used. This behaviour can be changed by setting a dedicated config group via [setConfigGroup\(\)](#). This is useful for to externally control the config group and shouldn't be used inside the implementing class.

Additionally to setting the config group, also a prefix for each config group entry can be defined via [setEntryPrefix\(\)](#). This may be useful if multiple instances of the same class shall be stored in the same config group or can generally be a good idea to make the config more readable and recognizable. By default this prefix is empty.

This class also supports recursive saving / loading invocations based on the QT object hierarchy. As default, calls to [loadState\(\)](#) or [saveState\(\)](#) only invoke the [doLoadState\(\)](#) or [doStateSave\(\)](#) method of the called instance. This behaviour can be changed with [setStateSavingDepth\(\)](#) to automatically call children of the instance. Various modes are supported as documented in [StateSavingDepth](#).

Author

jwienke

## 9.1227.2 Member Enumeration Documentation

### 9.1227.2.1 StateSavingDepth

```
enum Digikam::StateSavingObject::StateSavingDepth
```

## Enumerator

INSTANCE	Only the instance the saving / restoring was invoked on is saved / restored.
DIRECT_CHILDREN	The instance itself and all direct children of this instance implementing <a href="#">StateSavingObject</a> are saved / restored.
RECURSIVE	The instance and all children in the complete hierarchy are saved / restored.

## 9.1227.3 Constructor & Destructor Documentation

### 9.1227.3.1 StateSavingObject()

```
Digikam::StateSavingObject::StateSavingObject (
    QObject *const host ) [explicit]
```

Must be called after any QObject-based constructor.

## Parameters

<i>host</i>	self-reference to access the object name, simply pass "this" as argument
-------------	--

## 9.1227.4 Member Function Documentation

### 9.1227.4.1 doLoadState()

```
virtual void Digikam::StateSavingObject::doLoadState ( ) [protected], [pure virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implemented in [Digikam::DateFolderView](#), [Digikam::FilterSideBarWidget](#), [Digikam::AlbumFolderViewSideBarWidget](#), [Digikam::DateFolderViewSideBarWidget](#), [Digikam::FuzzySearchSideBarWidget](#), [Digikam::GPSSearchSideBarWidget](#), [Digikam::LabelsSideBarWidget](#), [Digikam::PeopleSideBarWidget](#), [Digikam::SearchSideBarWidget](#), [Digikam::TagViewSideBarWidget](#), [Digikam::TimelineSideBarWidget](#), [Digikam::MapWidgetView](#), [Digikam::TableView](#), [Digikam::AbstractAlbumTreeView](#), [Digikam::AbstractCheckableAlbumTreeView](#), [Digikam::LabelsTreeView](#), [Digikam::ImportItemPropertiesSideBarImport](#), [Digikam::ItemPropertiesSideBar](#), [Digikam::ItemPropertiesSideBarDB](#), [Digikam::AutoTagsScanWidget](#), [Digikam::TagsManager](#), [Digikam::TagCheckView](#), [Digikam::Sidebar](#), [Digikam::SearchTextBar](#), [ShowFoto::ShowfotoFolderViewSideBar](#), [ShowFoto::ShowfotoStackViewSideBar](#), [Digikam::FaceScanWidget](#), [Digikam::FuzzySearchView](#), and [Digikam::GPSSearchView](#).

### 9.1227.4.2 doSaveState()

```
virtual void Digikam::StateSavingObject::doSaveState ( ) [protected], [pure virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implemented in [Digikam::DateFolderView](#), [Digikam::FilterSideBarWidget](#), [Digikam::AlbumFolderViewSideBarWidget](#), [Digikam::DateFolderViewSideBarWidget](#), [Digikam::FuzzySearchSideBarWidget](#), [Digikam::GPSSearchSideBarWidget](#), [Digikam::LabelsSideBarWidget](#), [Digikam::PeopleSideBarWidget](#), [Digikam::SearchSideBarWidget](#), [Digikam::TagViewSideBarWidget](#), [Digikam::TimelineSideBarWidget](#), [Digikam::MapWidgetView](#), [Digikam::TableView](#), [Digikam::AbstractAlbumTreeView](#), [Digikam::AbstractCheckableAlbumTreeView](#), [Digikam::LabelsTreeView](#), [Digikam::ImportItemPropertiesSideBarImport](#), [Digikam::ItemPropertiesSideBar](#), [Digikam::ItemPropertiesSideBarDB](#), [Digikam::AutoTagsScanWidget](#), [Digikam::TagsManager](#), [Digikam::TagCheckView](#), [Digikam::Sidebar](#), [Digikam::SearchTextBar](#), [ShowFoto::ShowfotoFolderViewSideBar](#), [ShowFoto::ShowfotoStackViewSideBar](#), [Digikam::FaceScanWidget](#), [Digikam::FuzzySearchView](#), and [Digikam::GPSSearchView](#).



### 9.1227.4.3 entryName()

```
QString Digikam::StateSavingObject::entryName (
    const QString & base ) const [protected]
```

This allows to manipulate the entry keys externally by eg. setting a prefix.

#### Parameters

<i>base</i>	original name planned for the config group entry
-------------	--

#### Returns

entry name after manipulating it with externally set parameters

### 9.1227.4.4 getConfigGroup()

```
KConfigGroup Digikam::StateSavingObject::getConfigGroup ( ) const [protected]
```

#### Returns

config group for state saving and loading

### 9.1227.4.5 getStateSavingDepth()

```
StateSavingObject::StateSavingDepth Digikam::StateSavingObject::getStateSavingDepth ( ) const
```

Default is [StateSavingDepth::INSTANCE](#).

#### Returns

state saving / restoring depth

### 9.1227.4.6 setConfigGroup()

```
void Digikam::StateSavingObject::setConfigGroup (
    const KConfigGroup & group ) [virtual]
```

If this method is not called, a group based on the object name is used.

You can re-implement this method to pass the group set here to child objects. Don't forget to call this method in your implementation.

#### Parameters

<i>group</i>	config group to use for state saving and restoring
--------------	--

Reimplemented in [Digikam::DateFolderView](#), [Digikam::FilterSideBarWidget](#), [Digikam::FuzzySearchView](#), and [Digikam::GPSSearchView](#).

#### 9.1227.4.7 setEntryPrefix()

```
void Digikam::StateSavingObject::setEntryPrefix (
    const QString & prefix ) [virtual]
```

The default prefix is empty.

You can re-implement this method to pass the prefix set here to child objects. Don't forget to call this method in your implementation.

##### Parameters

<i>prefix</i>	the prefix to use for the config entries
---------------	--

#### 9.1227.4.8 setStateSavingDepth()

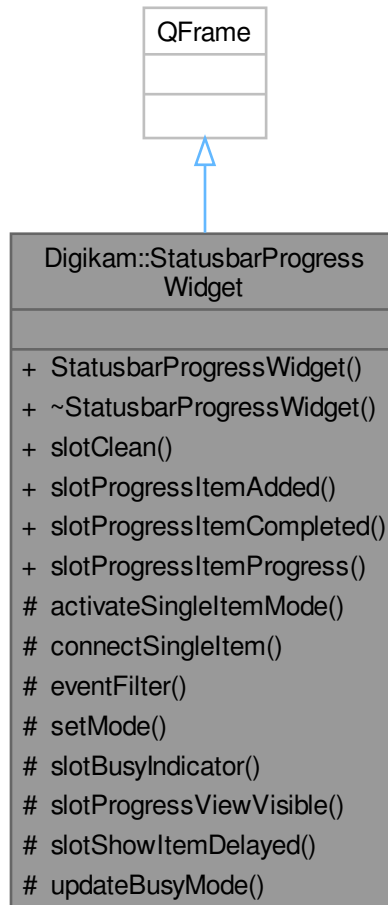
```
void Digikam::StateSavingObject::setStateSavingDepth (
    const StateSavingDepth depth )
```

##### Parameters

<i>depth</i>	new depth to use
--------------	------------------

## 9.1228 Digikam::StatusBarProgressWidget Class Reference

Inheritance diagram for Digikam::StatusBarProgressWidget:



### Public Slots

- void `slotClean` ()
- void `slotProgressItemAdded` ([ProgressItem](#) \*i)
- void `slotProgressItemCompleted` ([ProgressItem](#) \*i)
- void `slotProgressItemProgress` ([ProgressItem](#) \*i, unsigned int value)

### Public Member Functions

- `StatusBarProgressWidget` ([ProgressView](#) \*const progressView, `QWidget` \*const parent, bool button=true)

**Protected Slots**

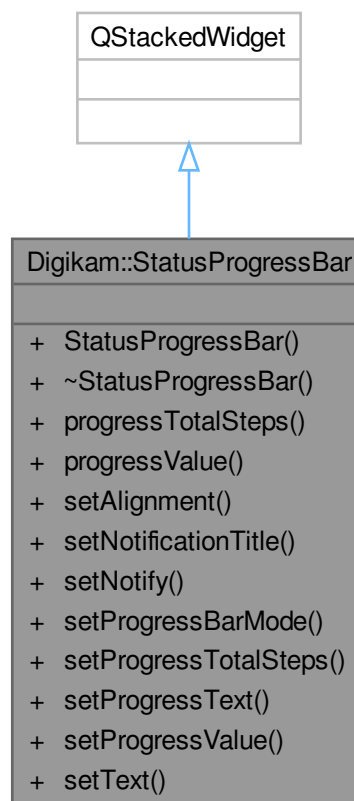
- void **slotBusyIndicator** ()
- void **slotProgressViewVisible** (bool)
- void **slotShowItemDelayed** ()
- void **updateBusyMode** ()

**Protected Member Functions**

- void **activateSingleItemMode** ()
- void **connectSingleItem** ()
- bool **eventFilter** (QObject \*, QEvent \*) override
- void **setMode** ()

**9.1229 Digikam::StatusProgressBar Class Reference**

Inheritance diagram for Digikam::StatusProgressBar:

**Public Types**

- enum **StatusProgressBarMode** { `TextMode = 0` , `ProgressBarMode` , `CancelProgressBarMode` }

### Public Slots

- void **setProgressText** (const QString &text)
- void **setProgressValue** (int v)
- void **setText** (const QString &text)

### Signals

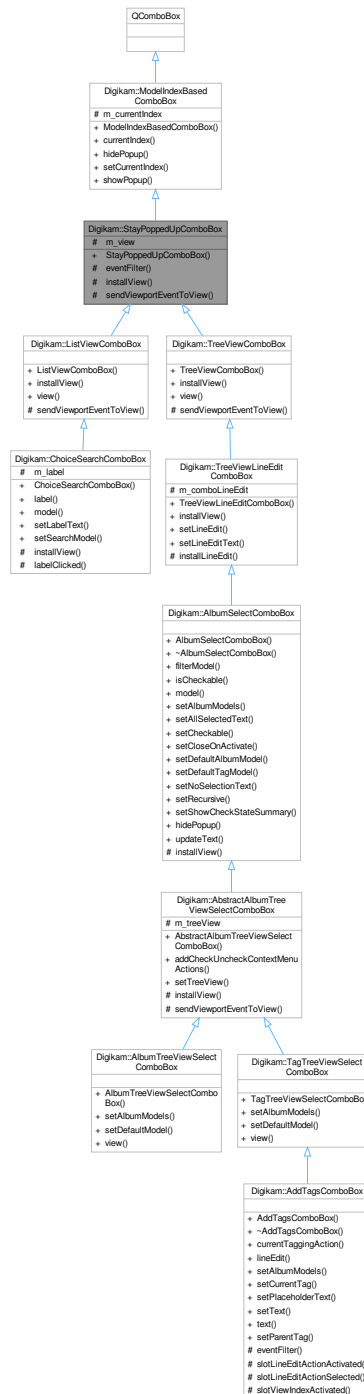
- void **signalCancelButtonPressed** ()

### Public Member Functions

- **StatusProgressBar** (QWidget \*const parent=nullptr)
- int **progressTotalSteps** () const
- int **progressValue** () const
- void **setAlignment** (Qt::Alignment a)
- void **setNotificationTitle** (const QString &title, const QIcon &icon)
- void **setNotify** (bool b)
- void **setProgressBarMode** (int mode, const QString &text=QString())
- void **setProgressTotalSteps** (int v)

## 9.1230 Digikam::StayPoppedUpComboBox Class Reference

Inheritance diagram for Digikam::StayPoppedUpComboBox:



### Public Member Functions

- [StayPoppedUpComboBox](#) (QWidget \*const parent=nullptr)

*This class provides an abstract QComboBox with a custom view (which is created by implementing subclasses) instead of the usual QListView.*

## Public Member Functions inherited from Digikam::ModelIndexBasedComboBox

- [ModelIndexBasedComboBox](#) (QWidget \*const parent=nullptr)  
*QComboBox has a current index based on a single integer.*
- QModelIndex **currentIndex** () const
- void **hidePopup** () override
- void **setCurrentIndex** (const QModelIndex &index)
- void **showPopup** () override

## Protected Member Functions

- bool **eventFilter** (QObject \*watched, QEvent \*event) override
- void **installView** (QAbstractItemView \*view)  
*Replace the standard combo box list view with the given view.*
- virtual void **sendViewportEventToView** (QEvent \*e)=0  
*Implement in subclass: Send the given event to the viewportEvent() method of m\_view.*

## Protected Attributes

- QAbstractItemView \* **m\_view** = nullptr

## Protected Attributes inherited from Digikam::ModelIndexBasedComboBox

- QPersistentModelIndex **m\_currentIndex**

## 9.1230.1 Constructor & Destructor Documentation

### 9.1230.1.1 StayPoppedUpComboBox()

```
Digikam::StayPoppedUpComboBox::StayPoppedUpComboBox (
    QWidget *const parent = nullptr ) [explicit]
```

The Pop-up of the combo box will stay open after selecting an item; it will be closed by clicking outside, but not inside the widget. You need three steps: Construct the object, call setModel() with an appropriate QAbstractItemModel, then call [installView\(\)](#) to replace the standard combo box view with a view.

## 9.1230.2 Member Function Documentation

### 9.1230.2.1 installView()

```
void Digikam::StayPoppedUpComboBox::installView (
    QAbstractItemView * view ) [protected]
```

The view will be set as the view of the combo box (including re-parenting) and be stored in the m\_view variable.

### 9.1230.2.2 `sendViewportEventToView()`

```
virtual void Digikam::StayPoppedUpComboBox::sendViewportEventToView (  
    QEvent * e ) [protected], [pure virtual]
```

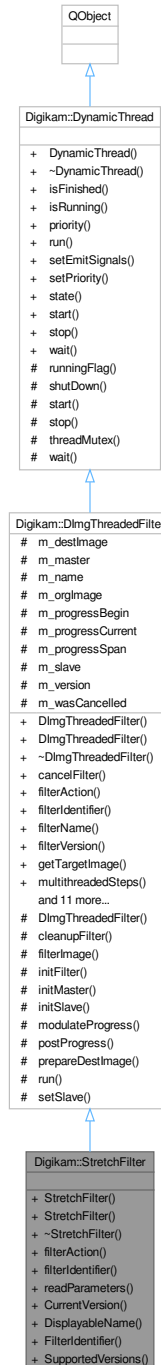
This method is protected for a usual `QAbstractItemView`. You can override, pass a view, and call parent implementation. The existing view will be used. You must then also reimplement `sendViewportEventToView`.

Implemented in [Digikam::AbstractAlbumTreeViewSelectComboBox](#), [Digikam::TreeViewComboBox](#), and [Digikam::ListViewComboBox](#).



## 9.1231 Digikam::StretchFilter Class Reference

Inheritance diagram for Digikam::StretchFilter:



### Public Member Functions

- **StretchFilter** ([Dlmg](#) \*const orgImage, const [Dlmg](#) \*const reflImage, [QObject](#) \*const parent=nullptr)
- **StretchFilter** ([QObject](#) \*const parent=nullptr)

- [FilterAction filterAction \(\)](#) override  
*Returns the action description corresponding to currently set options.*
- [QString filterIdentifier \(\)](#) const override  
*Return the identifier for this filter in the image history.*
- void [readParameters \(const FilterAction &action\)](#) override

## Public Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter \(DImg \\*const orgImage, QObject \\*const parent, const QString &name=QString\(\)\)](#)  
*Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter \(QObject \\*const parent=nullptr, const QString &name=QString\(\)\)](#)  
*Constructs a filter without argument.*
- virtual void [cancelFilter \(\)](#)  
*Cancel the threaded computation.*
- const [QString &filterName \(\)](#)
- int [filterVersion \(\)](#) const
- [DImg getTargetImage \(\)](#)
- [QList< int > multithreadedSteps \(int stop, int start=0\)](#) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool [parametersSuccessfullyRead \(\)](#) const  
*Optional: error handling for readParameters.*
- virtual [QString readParametersError \(const FilterAction &actionThatFailed\)](#) const
- void [setFilterName \(const QString &name\)](#)
- void [setFilterVersion \(int version\)](#)  
*Replaying a filter action: Set the filter version.*
- void [setOriginalImage \(const DImg &orgImage\)](#)
- void [setupAndStartDirectly \(const DImg &orgImage, DImgThreadedFilter \\*const master, int progress←Begin=0, int progressEnd=100\)](#)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void [setupFilter \(const DImg &orgImage\)](#)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void [startFilter \(\)](#)  
*Start the threaded computation.*
- virtual void [startFilterDirectly \(\)](#)  
*Start computation of this filter, directly in this thread.*
- virtual [QList< int > supportedVersions \(\)](#) const

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread \(QObject \\*const parent=nullptr\)](#)  
*This class extends QRunnable, so you have to reimplement virtual void [run\(\)](#).*
- [~DynamicThread \(\)](#) override  
*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool [isFinished \(\)](#) const
- bool [isRunning \(\)](#) const
- [QThread::Priority priority \(\)](#) const
- void [setEmitSignals \(bool emitThem\)](#)
- void [setPriority \(QThread::Priority priority\)](#)  
*Sets the priority for this dynamic thread.*
- State [state \(\)](#) const

### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from [Digikam::DImgThreadedFilter](#)

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.1231.1 Member Function Documentation

### 9.1231.1.1 filterAction()

`FilterAction` Digikam::StretchFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

### 9.1231.1.2 filterIdentifier()

`QString` Digikam::StretchFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

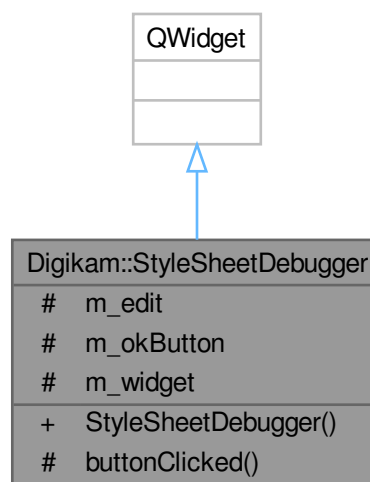
### 9.1231.1.3 readParameters()

```
void Digikam::StretchFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

## 9.1232 Digikam::StyleSheetDebugger Class Reference

Inheritance diagram for Digikam::StyleSheetDebugger:



## Public Member Functions

- [StyleSheetDebugger](#) (QWidget \*const object)

*This widget is for development purpose only: It allows the developer to change the style sheet on a widget dynamically.*

## Protected Slots

- void **buttonClicked** ()

## Protected Attributes

- QTextEdit \* **m\_edit** = nullptr
- QPushButton \* **m\_okButton** = nullptr
- QWidget \* **m\_widget** = nullptr

## 9.1232.1 Constructor & Destructor Documentation

### 9.1232.1.1 StyleSheetDebugger()

```
Digikam::StyleSheetDebugger::StyleSheetDebugger (
    QWidget *const object ) [explicit]
```

If you want to develop or debug the stylesheet on your widget, add temporary code: `new StyleSheetDebugger(myWidget);` That's all. Change the style sheet by editing it and pressing Ok.

## 9.1233 Digikam::SubjectData Class Reference

### Public Member Functions

- **SubjectData** (const QString &n, const QString &m, const QString &d)

### Public Attributes

- QString **detail**  
*English and Detail Name of subject.*
- QString **matter**  
*English and Matter Name of subject.*
- QString **name**  
*English and Name of subject.*

## 9.1234 Digikam::SubjectEdit Class Reference

Inheritance diagram for Digikam::SubjectEdit:



### Public Member Functions

- **SubjectEdit** (QWidget \*const parent)

## Public Member Functions inherited from [Digikam::SubjectWidget](#)

- **SubjectWidget** (QWidget \*const parent, bool sizesLimited=false)
- void **setSubjectsList** (const QStringList &list)
- QStringList **subjectsList** () const

## Additional Inherited Members

## Signals inherited from [Digikam::SubjectWidget](#)

- void **signalModified** ()

## Protected Slots inherited from [Digikam::SubjectWidget](#)

- virtual void **slotAddSubject** ()
- virtual void **slotDelSubject** ()
- virtual void **slotEditOptionChanged** (int)
- virtual void **slotRefChanged** ()
- virtual void **slotRepSubject** ()
- virtual void **slotSubjectSelectionChanged** ()
- virtual void **slotSubjectsToggled** (bool)

## Protected Member Functions inherited from [Digikam::SubjectWidget](#)

- virtual QString **buildSubject** () const
- virtual bool **loadSubjectCodesFromXML** (const QUrl &url)

## Protected Attributes inherited from [Digikam::SubjectWidget](#)

- [DTextEdit](#) \* **m\_detailEdit** = nullptr
- QString **m\_iprDefault**
- [QLineEdit](#) \* **m\_iprEdit** = nullptr
- [DTextEdit](#) \* **m\_matterEdit** = nullptr
- [DTextEdit](#) \* **m\_nameEdit** = nullptr
- [QLabel](#) \* **m\_note** = nullptr
- [QLineEdit](#) \* **m\_refEdit** = nullptr
- [QCheckBox](#) \* **m\_subjectsCheck** = nullptr



## 9.1235 Digikam::SubjectWidget Class Reference

Inheritance diagram for Digikam::SubjectWidget:



### Signals

- void **signalModified** ()

**Public Member Functions**

- **SubjectWidget** (QWidget \*const parent, bool sizesLimited=false)
- void **setSubjectsList** (const QStringList &list)
- QStringList **subjectsList** () const

**Protected Slots**

- virtual void **slotAddSubject** ()
- virtual void **slotDelSubject** ()
- virtual void **slotEditOptionChanged** (int)
- virtual void **slotRefChanged** ()
- virtual void **slotRepSubject** ()
- virtual void **slotSubjectSelectionChanged** ()
- virtual void **slotSubjectsToggled** (bool)

**Protected Member Functions**

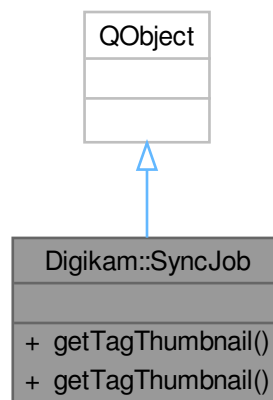
- virtual QString **buildSubject** () const
- virtual bool **loadSubjectCodesFromXML** (const QUrl &url)

**Protected Attributes**

- [DTextEdit](#) \* **m\_detailEdit** = nullptr
- QString **m\_iprDefault**
- [QLineEdit](#) \* **m\_iprEdit** = nullptr
- [DTextEdit](#) \* **m\_matterEdit** = nullptr
- [DTextEdit](#) \* **m\_nameEdit** = nullptr
- [QLabel](#) \* **m\_note** = nullptr
- [QLineEdit](#) \* **m\_refEdit** = nullptr
- [QCheckBox](#) \* **m\_subjectsCheck** = nullptr

**9.1236 Digikam::SyncJob Class Reference**

Inheritance diagram for Digikam::SyncJob:



### Static Public Member Functions

- static QPixmap **getTagThumbnail** (const QString &name, int size)
- static QPixmap **getTagThumbnail** (TAlbum \*const album)

*Load the image or icon for the tag thumbnail.*

## 9.1237 Digikam::SystemSettings Class Reference

### Public Types

- enum **ProxyType** { **HttpProxy** = 0 , **Socks5Proxy** }

*This enum is used to specify the proxy that is used.*

### Public Member Functions

- **SystemSettings** (const QString &name)
- void **saveSettings** ()

### Public Attributes

- bool **enableAesthetic** = false
- bool **enableAutoTags** = false
- bool **enableFaceEngine** = false
- bool **enableHWTCnv** = false
- bool **enableHWVideo** = false
- bool **enableLogging** = false
- bool **enableOpenCL** = false
- bool **proxyAuth** = false
- QString **proxyPass**
- int **proxyPort** = 8080
- int **proxyType** = **HttpProxy**
- QString **proxyUrl**
- QString **proxyUser**
- bool **softwareOpenGL** = false
- QString **videoBackend** = QLatin1String("ffmpeg")

### 9.1237.1 Member Enumeration Documentation

#### 9.1237.1.1 ProxyType

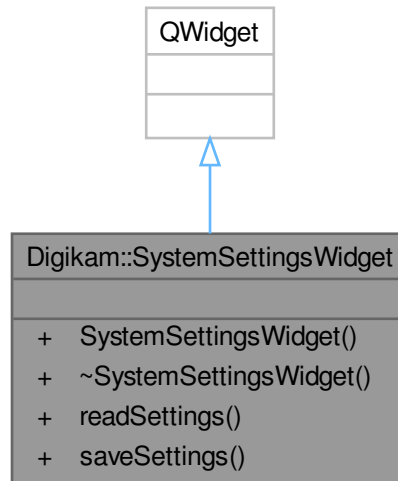
enum [Digikam::SystemSettings::ProxyType](#)

#### Enumerator

HttpProxy	Uses an Http proxy.
Socks5Proxy	Uses a Socks5 proxy.

## 9.1238 Digikam::SystemSettingsWidget Class Reference

Inheritance diagram for Digikam::SystemSettingsWidget:

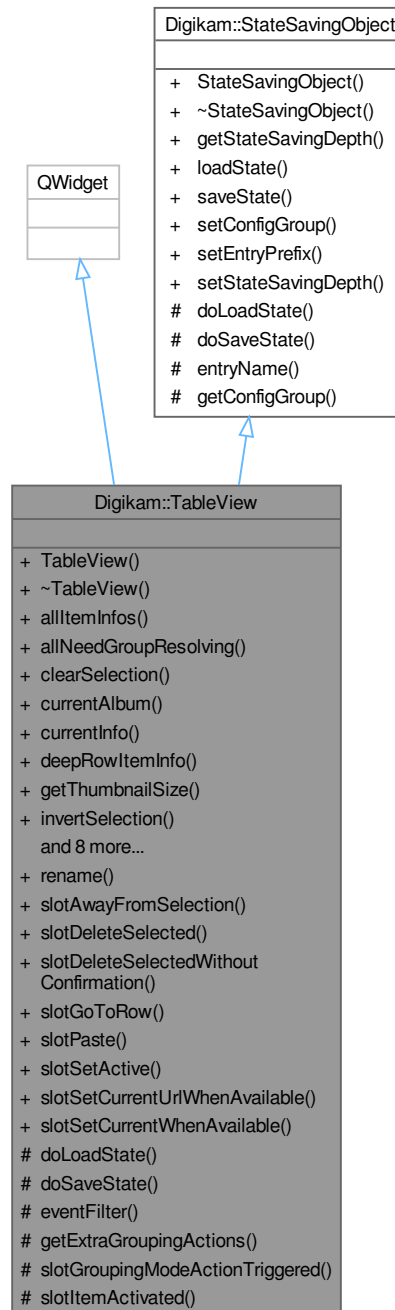


### Public Member Functions

- **SystemSettingsWidget** (`QWidget *const parent`)
- void **readSettings** ()
- void **saveSettings** ()

## 9.1239 Digikam::TableView Class Reference

Inheritance diagram for Digikam::TableView:



### Public Slots

- void **rename** ()
- void [slotAwayFromSelection](#) ()

*Unselects the current selection and changes the current item.*

- void [slotDeleteSelected](#) (const [ItemViewUtilities::DeleteMode](#) deleteMode=[ItemViewUtilities::DeleteUseTrash](#))
- void [slotDeleteSelectedWithoutConfirmation](#) (const [ItemViewUtilities::DeleteMode](#) deleteMode=[ItemViewUtilities::DeleteUseTrash](#))
- void [slotGoToRow](#) (const int rowNumber, const bool relativeMove)
- void [slotPaste](#) ()
- void [slotSetActive](#) (const bool isActive)
- void [slotSetCurrentUrlWhenAvailable](#) (const [QUrl](#) &url)
- void [slotSetCurrentWhenAvailable](#) (const [qlonglong](#) id)

## Signals

- void [signalInsertSelectedToExistingQueue](#) (int queue)
- void [signalItemsChanged](#) ()
- void [signalPopupTagsView](#) ()
- void [signalPreviewRequested](#) (const [ItemInfo](#) &info)
- void [signalShowContextMenu](#) ([QContextMenuEvent](#) \*event, const [QList](#)< [QAction](#) \* > &actions)
- void [signalShowContextMenuOnInfo](#) ([QContextMenuEvent](#) \*event, const [ItemInfo](#) &info, const [QList](#)< [QAction](#) \* > &actions, [ItemFilterModel](#) \*filterModel=nullptr)
- void [signalZoomInStep](#) ()
- void [signalZoomOutStep](#) ()

## Public Member Functions

- [TableView](#) ([QItemSelectionModel](#) \*const selectionModel, [DCategorizedSortFilterProxyModel](#) \*const imageFilterModel, [QWidget](#) \*const parent)
- [ItemInfoList](#) [allItemInfos](#) (bool grouping=false) const
- bool [allNeedGroupResolving](#) (const [OperationType](#) type) const
- void [clearSelection](#) ()
- [Album](#) \* [currentAlbum](#) () const
- [ItemInfo](#) [currentInfo](#) () const
- [ItemInfo](#) [deepRowItemInfo](#) (const int rowNumber, const bool relative) const
- [ThumbnailSize](#) [getThumbnailSize](#) () const
- void [invertSelection](#) ()
- [ItemInfo](#) [nextInfo](#) () const
- int [numberOfSelectedItems](#) () const
- [ItemInfo](#) [previousInfo](#) () const
- void [selectAll](#) ()
- [ItemInfoList](#) [selectedItemInfos](#) (bool grouping=false) const
- [ItemInfoList](#) [selectedItemInfosCurrentFirst](#) (bool grouping=false) const
- bool [selectedNeedGroupResolving](#) (const [OperationType](#) type) const
- void [setThumbnailSize](#) (const [ThumbnailSize](#) &size)

## Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (QObject \*const host)  
*Constructor.*
- virtual `~StateSavingObject ()`  
*Destructor.*
- [StateSavingDepth](#) `getStateSavingDepth ()` const  
*Returns the depth used for state saving or loading.*
- void `loadState ()`  
*Invokes loading the class' state.*
- void `saveState ()`  
*Invokes saving the class' state.*
- virtual void `setConfigGroup` (const KConfigGroup &group)  
*Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void `setEntryPrefix` (const QString &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void `setStateSavingDepth` (const [StateSavingDepth](#) depth)  
*Sets the depth used for state saving or loading.*

## Protected Slots

- void `slotGroupingModeActionTriggered ()`
- void `slotItemActivated` (const QModelIndex &tableViewIndex)

## Protected Member Functions

- void `doLoadState ()` override  
*Implement this hook method for state loading.*
- void `doSaveState ()` override  
*Implement this hook method for state saving.*
- bool `eventFilter` (QObject \*watched, QEvent \*event) override
- QList< QAction \* > `getExtraGroupingActions ()`

## Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- QString `entryName` (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup `getConfigGroup ()` const  
*Returns the config group that must be used for state saving and loading.*

## Additional Inherited Members

## Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { `INSTANCE` , `DIRECT_CHILDREN` , `RECURSIVE` }  
*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

## 9.1239.1 Member Function Documentation

### 9.1239.1.1 doLoadState()

```
void Digikam::TableView::doLoadState ( ) [override], [protected], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

### 9.1239.1.2 doSaveState()

```
void Digikam::TableView::doSaveState ( ) [override], [protected], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

### 9.1239.1.3 invertSelection()

```
void Digikam::TableView::invertSelection ( )
```

### 9.1239.1.4 selectAll()

```
void Digikam::TableView::selectAll ( )
```

### 9.1239.1.5 slotAwayFromSelection

```
void Digikam::TableView::slotAwayFromSelection ( ) [slot]
```

### 9.1239.1.6 slotDeleteSelected

```
void Digikam::TableView::slotDeleteSelected (
    const ItemViewUtilities::DeleteMode deleteMode = ItemViewUtilities::DeleteUseTrash
) [slot]
```

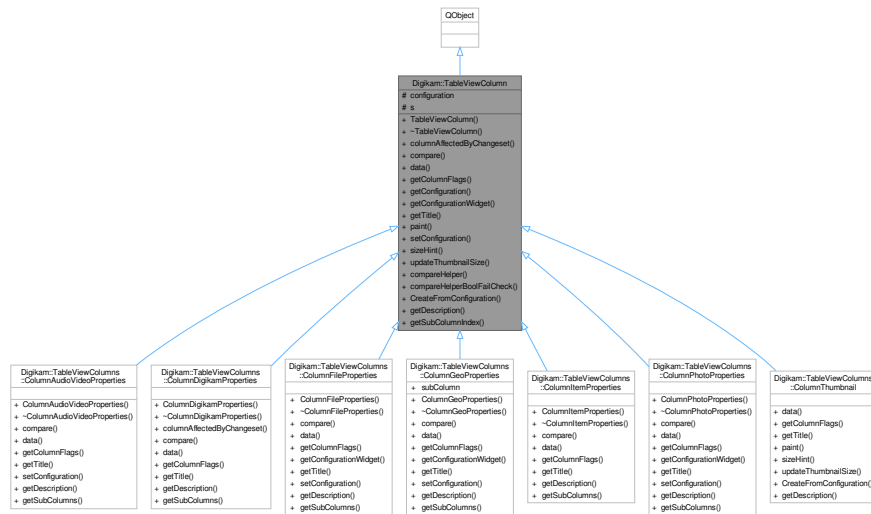
### 9.1239.1.7 slotSetCurrentWhenAvailable

```
void Digikam::TableView::slotSetCurrentWhenAvailable (
    const qlonglong id ) [slot]
```



## 9.1240 Digikam::TableViewColumn Class Reference

Inheritance diagram for Digikam::TableViewColumn:



### Public Types

- enum **ColumnCompareResult** { **CmpEqual** = 0 , **CmpABiggerB** = 1 , **CmpALessB** = 2 }
- enum **ColumnFlag** { **ColumnNoFlags** = 0 , **ColumnCustomPainting** = 1 , **ColumnCustomSorting** = 2 , **ColumnHasConfigurationWidget** = 4 }
- typedef QFlags< ColumnFlag > **ColumnFlags**

### Signals

- void **signalAllDataChanged** ()
- void **signalDataChanged** (const qlonglong imageId)

### Public Member Functions

- **TableViewColumn** (**TableViewShared** \*const tableViewShared, const **TableViewColumnConfiguration** &p← Configuration, **QObject** \*const parent=nullptr)
- virtual bool **columnAffectedByChangeset** (const **ImageChangeset** &imageChangeset) const
- virtual **ColumnCompareResult** **compare** (**TableViewModel::Item** \*const itemA, **TableViewModel::Item** \*const itemB) const  
*This function should never be called, because subclasses have to do the comparison on their own.*
- virtual **QVariant** **data** (**TableViewModel::Item** \*const item, const int role) const
- virtual **ColumnFlags** **getColumnFlags** () const
- virtual **TableViewColumnConfiguration** **getConfiguration** () const
- virtual **TableViewColumnConfigurationWidget** \* **getConfigurationWidget** (**QWidget** \*const parentWidget) const
- virtual **QString** **getTitle** () const =0
- virtual bool **paint** (**QPainter** \*const painter, const **QStyleOptionViewItem** &option, **TableViewModel::Item** \*const item) const
- virtual void **setConfiguration** (const **TableViewColumnConfiguration** &newConfiguration)
- virtual **QSize** **sizeHint** (const **QStyleOptionViewItem** &option, **TableViewModel::Item** \*const item) const
- virtual void **updateThumbnailSize** ()

## Static Public Member Functions

- `template<class MyType >`  
`static ColumnCompareResult compareHelper (const MyType &A, const MyType &B)`
- `static bool compareHelperBoolFailCheck (const bool okA, const bool okB, ColumnCompareResult *const result)`
- `template<typename columnClass >`  
`static bool CreateFromConfiguration (TableViewShared *const tableViewShared, const TableViewColumnConfiguration &pConfiguration, TableViewColumn **const pNewColumn, QObject *const parent)`
- `static TableViewColumnDescription getDescription ()`
- `template<typename columnClass >`  
`static bool getSubColumnIndex (const QString &subColumnId, typename columnClass::SubColumn *const subColumn)`

## Protected Attributes

- [TableViewColumnConfiguration](#) **configuration**
- [TableViewShared](#) \*const **s** = nullptr

## 9.1240.1 Member Function Documentation

### 9.1240.1.1 columnAffectedByChangeset()

```
bool Digikam::TableViewColumn::columnAffectedByChangeset (
    const ImageChangeset & imageChangeset ) const [virtual]
```

Reimplemented in [Digikam::TableViewColumns::ColumnDigikamProperties](#).

### 9.1240.1.2 compare()

```
TableViewColumn::ColumnCompareResult Digikam::TableViewColumn::compare (
    TableViewModel::Item *const itemA,
    TableViewModel::Item *const itemB ) const [virtual]
```

But it can not be pure, since then every subclass which does not do custom comparison would have to implement an empty stub.

Reimplemented in [Digikam::TableViewColumns::ColumnAudioVideoProperties](#), [Digikam::TableViewColumns::ColumnDigikamProperties](#), [Digikam::TableViewColumns::ColumnFileProperties](#), [Digikam::TableViewColumns::ColumnGeoProperties](#), [Digikam::TableViewColumns::ColumnImageProperties](#), and [Digikam::TableViewColumns::ColumnPhotoProperties](#).

### 9.1240.1.3 data()

```
QVariant Digikam::TableViewColumn::data (
    TableViewModel::Item *const item,
    const int role ) const [virtual]
```

Reimplemented in [Digikam::TableViewColumns::ColumnDigikamProperties](#), [Digikam::TableViewColumns::ColumnFileProperties](#), [Digikam::TableViewColumns::ColumnGeoProperties](#), [Digikam::TableViewColumns::ColumnImageProperties](#), and [Digikam::TableViewColumns::ColumnPhotoProperties](#).

#### 9.1240.1.4 getColumnFlags()

```
TableViewColumn::ColumnFlags Digikam::TableViewColumn::getColumnFlags ( ) const [virtual]
```

Reimplemented in [Digikam::TableViewColumns::ColumnAudioVideoProperties](#).

#### 9.1240.1.5 paint()

```
bool Digikam::TableViewColumn::paint (
    QPainter *const painter,
    const QStyleOptionViewItem & option,
    TableViewModel::Item *const item ) const [virtual]
```

Reimplemented in [Digikam::TableViewColumns::ColumnThumbnail](#).

#### 9.1240.1.6 sizeHint()

```
QSize Digikam::TableViewColumn::sizeHint (
    const QStyleOptionViewItem & option,
    TableViewModel::Item *const item ) const [virtual]
```

Reimplemented in [Digikam::TableViewColumns::ColumnThumbnail](#).

#### 9.1240.1.7 updateThumbnailSize()

```
void Digikam::TableViewColumn::updateThumbnailSize ( ) [virtual]
```

Reimplemented in [Digikam::TableViewColumns::ColumnThumbnail](#).

## 9.1241 Digikam::TableViewColumnConfiguration Class Reference

### Public Member Functions

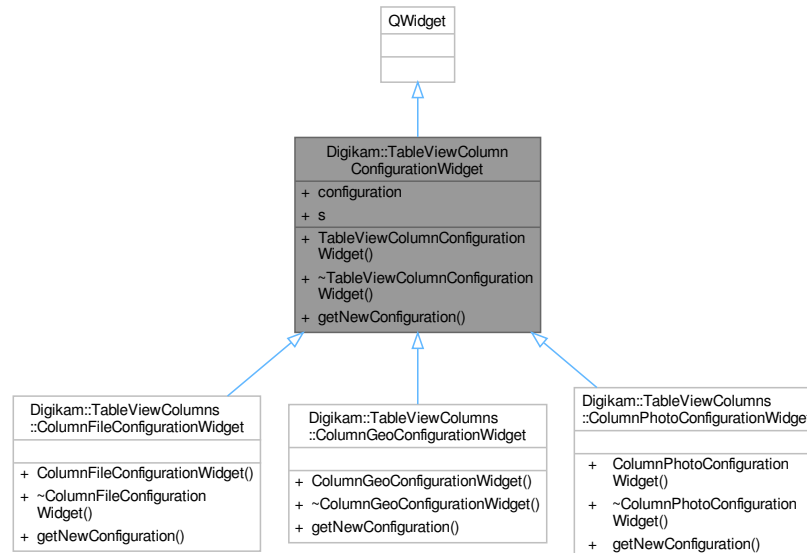
- **TableViewColumnConfiguration** (const QString &id=QString())
- QString **getSetting** (const QString &key, const QString &defaultValue=QString()) const
- void **loadSettings** (const KConfigGroup &configGroup)
- void **saveSettings** (KConfigGroup &configGroup) const

### Public Attributes

- QString **columnId**
- QHash< QString, QString > **columnSettings**

## 9.1242 Digikam::TableViewColumnConfigurationWidget Class Reference

Inheritance diagram for Digikam::TableViewColumnConfigurationWidget:



### Public Member Functions

- **TableViewColumnConfigurationWidget** ([TableViewShared](#) \*const sharedObject, const [TableViewColumnConfiguration](#) &currentConfiguration, QWidget \*const parent=nullptr)
- virtual [TableViewColumnConfiguration](#) **getNewConfiguration** ()=0

### Public Attributes

- [TableViewColumnConfiguration](#) **configuration**
- [TableViewShared](#) \*const **s** = nullptr

## 9.1243 Digikam::TableViewColumnDescription Class Reference

### Public Types

- typedef QList< [TableViewColumnDescription](#) > **List**

### Public Member Functions

- **TableViewColumnDescription** (const QString &id, const QString &title, const QString &setting↔ Key=QString(), const QString &settingValue=QString())
- void **addSetting** (const QString &key, const QString &value)
- void **addSubColumn** (const [TableViewColumnDescription](#) &subColumnDescription)
- [TableViewColumnDescription](#) **setIcon** (const QString &iconName)
- [TableViewColumnConfiguration](#) **toConfiguration** () const

**Static Public Member Functions**

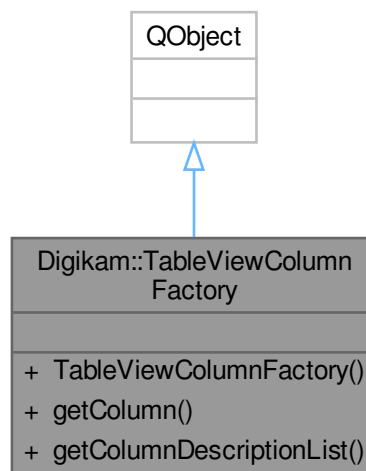
- static bool **FindInListById** (const TableViewColumnDescription::List &listToSearch, const QString &targetId, [TableViewColumnDescription](#) \*const resultDescription)

**Public Attributes**

- QString **columnIcon**
- QString **columnId**
- QHash< QString, QString > **columnSettings**
- QString **columnName**
- QList< [TableViewColumnDescription](#) > **subColumns**

## 9.1244 Digikam::TableViewColumnFactory Class Reference

Inheritance diagram for Digikam::TableViewColumnFactory:

**Public Member Functions**

- **TableViewColumnFactory** ([TableViewShared](#) \*const tableViewShared, QWidget \*const parent)
- [TableViewColumn](#) \* **getColumn** (const [TableViewColumnConfiguration](#) &columnConfiguration)

**Static Public Member Functions**

- static QList< [TableViewColumnDescription](#) > **getColumnDescriptionList** ()

## 9.1245 Digikam::TableViewColumnProfile Class Reference

### Public Member Functions

- void [loadSettings](#) (const KConfigGroup &configGroup)
- void **saveSettings** (KConfigGroup &configGroup)

### Public Attributes

- QList< [TableViewColumnConfiguration](#) > **columnConfigurationList**
- QByteArray **headerState**
- QString **name**

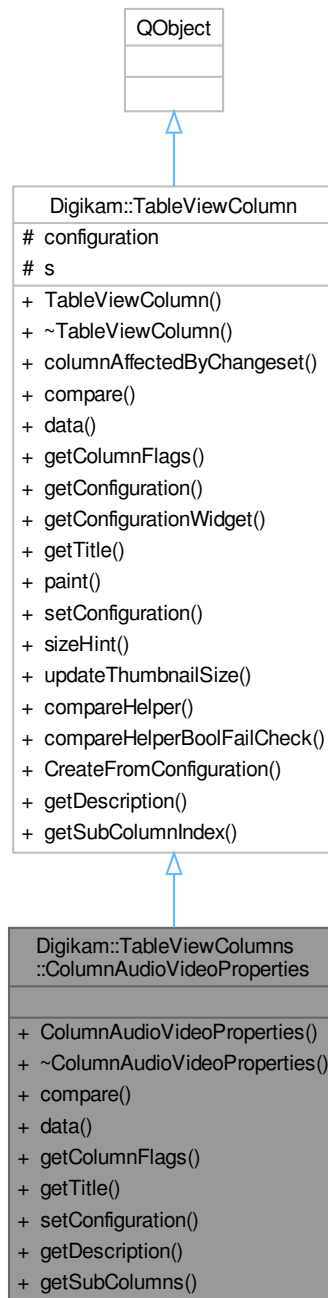
### 9.1245.1 Member Function Documentation

#### 9.1245.1.1 loadSettings()

```
void Digikam::TableViewColumnProfile::loadSettings (
    const KConfigGroup & configGroup )
```

## 9.1246 Digikam::TableViewColumns::ColumnAudioVideoProperties Class Reference

Inheritance diagram for Digikam::TableViewColumns::ColumnAudioVideoProperties:



### Public Types

- enum `SubColumn` {  
`SubColumnAudioBitRate = 0`, `SubColumnAudioChannelType = 1`, `SubColumnAudioCodec = 2`, `Sub`↔

```
ColumnDuration = 3 ,
SubColumnFrameRate = 4 , SubColumnVideoCodec = 5 }
```

## Public Types inherited from [Digikam::TableViewColumn](#)

- enum **ColumnCompareResult** { **CmpEqual** = 0 , **CmpABiggerB** = 1 , **CmpALessB** = 2 }
- enum **ColumnFlag** { **ColumnNoFlags** = 0 , **ColumnCustomPainting** = 1 , **ColumnCustomSorting** = 2 , **ColumnHasConfigurationWidget** = 4 }
- typedef QFlags< ColumnFlag > **ColumnFlags**

## Public Member Functions

- **ColumnAudioVideoProperties** ([TableViewShared](#) \*const tableViewShared, const [TableViewColumnConfiguration](#) &pConfiguration, const SubColumn pSubColumn, QObject \*const parent=nullptr)
- ColumnCompareResult **compare** ([TableViewModel::Item](#) \*const itemA, [TableViewModel::Item](#) \*const itemB) const override
 

*This function should never be called, because subclasses have to do the comparison on their own.*
- QVariant **data** ([TableViewModel::Item](#) \*const item, const int role) const override
- ColumnFlags **getColumnFlags** () const override
- QString **getTitle** () const override
- void **setConfiguration** (const [TableViewColumnConfiguration](#) &newConfiguration) override

## Public Member Functions inherited from [Digikam::TableViewColumn](#)

- **TableViewColumn** ([TableViewShared](#) \*const tableViewShared, const [TableViewColumnConfiguration](#) &pConfiguration, QObject \*const parent=nullptr)
- virtual bool **columnAffectedByChangeset** (const [ImageChangeset](#) &imageChangeset) const
- virtual [TableViewColumnConfiguration](#) **getConfiguration** () const
- virtual [TableViewColumnConfigurationWidget](#) \* **getConfigurationWidget** (QWidget \*const parentWidget) const
- virtual bool **paint** (QPainter \*const painter, const QStyleOptionViewItem &option, [TableViewModel::Item](#) \*const item) const
- virtual QSize **sizeHint** (const QStyleOptionViewItem &option, [TableViewModel::Item](#) \*const item) const
- virtual void **updateThumbnailSize** ()

## Static Public Member Functions

- static [TableViewColumnDescription](#) **getDescription** ()
- static QStringList **getSubColumns** ()

## Static Public Member Functions inherited from [Digikam::TableViewColumn](#)

- template<class MyType >
 

```
static ColumnCompareResult compareHelper (const MyType &A, const MyType &B)
```
- static bool **compareHelperBoolFailCheck** (const bool okA, const bool okB, ColumnCompareResult \*const result)
- template<typename columnClass >
 

```
static bool CreateFromConfiguration (TableViewShared *const tableViewShared, const TableViewColumnConfiguration &pConfiguration, TableViewColumn **const pNewColumn, QObject *const parent)
```
- static [TableViewColumnDescription](#) **getDescription** ()
- template<typename columnClass >
 

```
static bool getSubColumnIndex (const QString &subColumnId, typename columnClass::SubColumn *const subColumn)
```



## Additional Inherited Members

## Signals inherited from [Digikam::TableViewColumn](#)

- void **signalAllDataChanged** ()
- void **signalDataChanged** (const qlonglong imageId)

## Protected Attributes inherited from [Digikam::TableViewColumn](#)

- [TableViewColumnConfiguration](#) **configuration**
- [TableViewShared](#) \*const **s** = nullptr

## 9.1246.1 Member Function Documentation

### 9.1246.1.1 compare()

```
TableViewColumn::ColumnCompareResult Digikam::TableViewColumns::ColumnAudioVideoProperties↔
::compare (
    TableViewModel::Item *const itemA,
    TableViewModel::Item *const itemB ) const [override], [virtual]
```

But it can not be pure, since then every subclass which does not do custom comparison would have to implement an empty stub.

Reimplemented from [Digikam::TableViewColumn](#).

### 9.1246.1.2 data()

```
QVariant Digikam::TableViewColumns::ColumnAudioVideoProperties::data (
    TableViewModel::Item *const item,
    const int role ) const [override], [virtual]
```

Reimplemented from [Digikam::TableViewColumn](#).

### 9.1246.1.3 getColumnFlags()

```
TableViewColumn::ColumnFlags Digikam::TableViewColumns::ColumnAudioVideoProperties::get↔
ColumnFlags ( ) const [override], [virtual]
```

Reimplemented from [Digikam::TableViewColumn](#).

### 9.1246.1.4 getTitle()

```
QString Digikam::TableViewColumns::ColumnAudioVideoProperties::getTitle ( ) const [override],
[virtual]
```

Implements [Digikam::TableViewColumn](#).

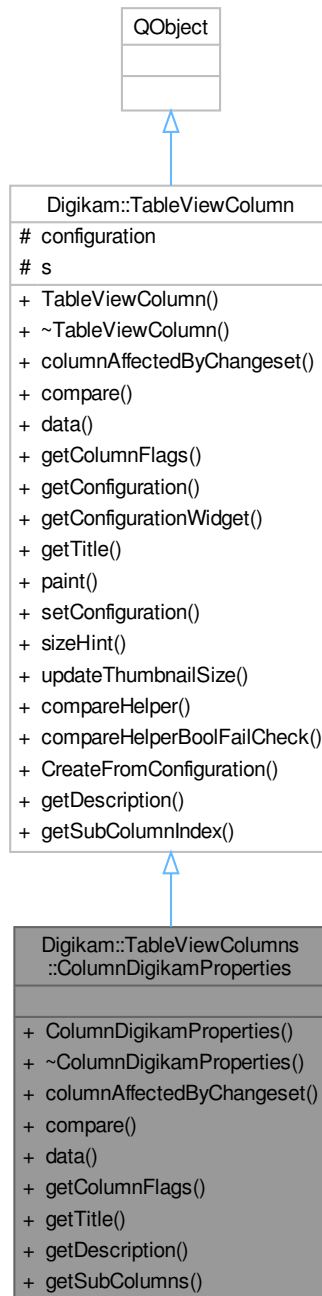
**9.1246.1.5 setConfiguration()**

```
void Digikam::TableViewColumns::ColumnAudioVideoProperties::setConfiguration (
    const TableViewColumnConfiguration & newConfiguration ) [override], [virtual]
```

Reimplemented from [Digikam::TableViewColumn](#).

## 9.1247 Digikam::TableViewColumns::ColumnDigikamProperties Class Reference

Inheritance diagram for Digikam::TableViewColumns::ColumnDigikamProperties:



### Public Types

- enum **SubColumn** {  
**SubColumnRating** = 0 , **SubColumnPickLabel** = 1 , **SubColumnColorLabel** = 2 , **SubColumnTitle** = 3 ,  
**SubColumnCaption** = 4 , **SubColumnTags** = 5 }

## Public Types inherited from [Digikam::TableViewColumn](#)

- enum **ColumnCompareResult** { **CmpEqual** = 0 , **CmpABiggerB** = 1 , **CmpALessB** = 2 }
- enum **ColumnFlag** { **ColumnNoFlags** = 0 , **ColumnCustomPainting** = 1 , **ColumnCustomSorting** = 2 , **ColumnHasConfigurationWidget** = 4 }
- typedef QFlags< ColumnFlag > **ColumnFlags**

## Public Member Functions

- **ColumnDigikamProperties** ([TableViewShared](#) \*const tableViewShared, const [TableViewColumnConfiguration](#) &pConfiguration, const SubColumn pSubColumn, QObject \*const parent=nullptr)
- bool [columnAffectedByChangeset](#) (const [ImageChangeset](#) &imageChangeset) const override
- ColumnCompareResult [compare](#) ([TableViewModel::Item](#) \*const itemA, [TableViewModel::Item](#) \*const itemB) const override  
*This function should never be called, because subclasses have to do the comparison on their own.*
- QVariant [data](#) ([TableViewModel::Item](#) \*const item, const int role) const override
- ColumnFlags [getColumnFlags](#) () const override
- QString [getTitle](#) () const override

## Public Member Functions inherited from [Digikam::TableViewColumn](#)

- **TableViewColumn** ([TableViewShared](#) \*const tableViewShared, const [TableViewColumnConfiguration](#) &pConfiguration, QObject \*const parent=nullptr)
- virtual [TableViewColumnConfiguration](#) [getConfiguration](#) () const
- virtual [TableViewColumnConfigurationWidget](#) \* [getConfigurationWidget](#) (QWidget \*const parentWidget) const
- virtual bool [paint](#) (QPainter \*const painter, const QStyleOptionViewItem &option, [TableViewModel::Item](#) \*const item) const
- virtual void [setConfiguration](#) (const [TableViewColumnConfiguration](#) &newConfiguration)
- virtual QSize [sizeHint](#) (const QStyleOptionViewItem &option, [TableViewModel::Item](#) \*const item) const
- virtual void [updateThumbnailSize](#) ()

## Static Public Member Functions

- static [TableViewColumnDescription](#) [getDescription](#) ()
- static QStringList [getSubColumns](#) ()

## Static Public Member Functions inherited from [Digikam::TableViewColumn](#)

- template<class MyType >  
static ColumnCompareResult [compareHelper](#) (const MyType &A, const MyType &B)
- static bool [compareHelperBoolFailCheck](#) (const bool okA, const bool okB, ColumnCompareResult \*const result)
- template<typename columnClass >  
static bool [CreateFromConfiguration](#) ([TableViewShared](#) \*const tableViewShared, const [TableViewColumnConfiguration](#) &pConfiguration, [TableViewColumn](#) \*\*const pNewColumn, QObject \*const parent)
- static [TableViewColumnDescription](#) [getDescription](#) ()
- template<typename columnClass >  
static bool [getSubColumnIndex](#) (const QString &subColumnId, typename columnClass::SubColumn \*const subColumn)

## Additional Inherited Members

## Signals inherited from [Digikam::TableViewColumn](#)

- void **signalAllDataChanged** ()
- void **signalDataChanged** (const qlonglong imageId)

## Protected Attributes inherited from [Digikam::TableViewColumn](#)

- [TableViewColumnConfiguration](#) **configuration**
- [TableViewShared](#) \*const **s** = nullptr

## 9.1247.1 Member Function Documentation

### 9.1247.1.1 columnAffectedByChangeset()

```
bool Digikam::TableViewColumns::ColumnDigikamProperties::columnAffectedByChangeset (
    const ImageChangeset & imageChangeset ) const [override], [virtual]
```

Reimplemented from [Digikam::TableViewColumn](#).

### 9.1247.1.2 compare()

```
TableViewColumn::ColumnCompareResult Digikam::TableViewColumns::ColumnDigikamProperties←
::compare (
    TableViewModel::Item *const itemA,
    TableViewModel::Item *const itemB ) const [override], [virtual]
```

But it can not be pure, since then every subclass which does not do custom comparison would have to implement an empty stub.

Reimplemented from [Digikam::TableViewColumn](#).

### 9.1247.1.3 data()

```
QVariant Digikam::TableViewColumns::ColumnDigikamProperties::data (
    TableViewModel::Item *const item,
    const int role ) const [override], [virtual]
```

Reimplemented from [Digikam::TableViewColumn](#).

### 9.1247.1.4 getColumnFlags()

```
TableViewColumn::ColumnFlags Digikam::TableViewColumns::ColumnDigikamProperties::getColumn←
Flags ( ) const [override], [virtual]
```

Reimplemented from [Digikam::TableViewColumn](#).

### 9.1247.1.5 getDescription()

```
TableViewColumnDescription Digikam::TableViewColumns::ColumnDigikamProperties::getDescription
( ) [static]
```

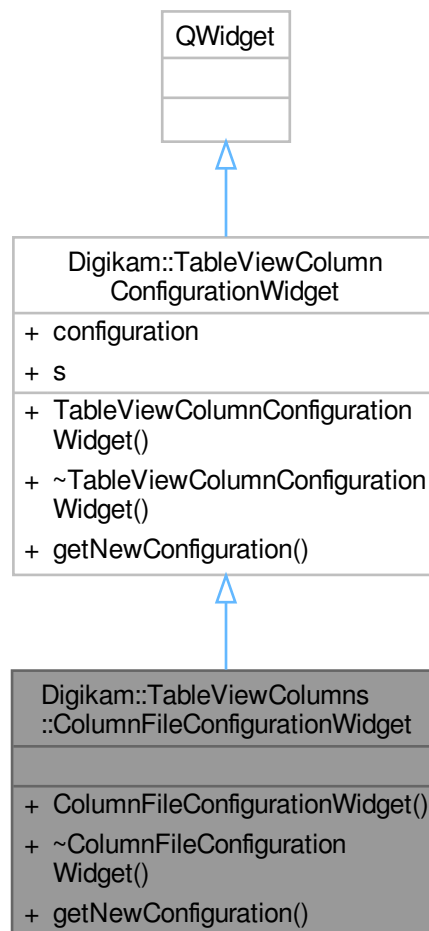
### 9.1247.1.6 getTitle()

```
QString Digikam::TableViewColumns::ColumnDigikamProperties::getTitle ( ) const [override],
[virtual]
```

Implements [Digikam::TableViewColumn](#).

## 9.1248 Digikam::TableViewColumns::ColumnFileConfigurationWidget Class Reference

Inheritance diagram for Digikam::TableViewColumns::ColumnFileConfigurationWidget:



## Public Member Functions

- **ColumnFileConfigurationWidget** ([TableViewShared](#) \*const sharedObject, const [TableViewColumnConfiguration](#) &columnConfiguration, [QWidget](#) \*const parentWidget)
- [TableViewColumnConfiguration](#) `getNewConfiguration` () override

## Public Member Functions inherited from [Digikam::TableViewColumnConfigurationWidget](#)

- **TableViewColumnConfigurationWidget** ([TableViewShared](#) \*const sharedObject, const [TableViewColumnConfiguration](#) &currentConfiguration, [QWidget](#) \*const parent=nullptr)

## Additional Inherited Members

## Public Attributes inherited from [Digikam::TableViewColumnConfigurationWidget](#)

- [TableViewColumnConfiguration](#) `configuration`
- [TableViewShared](#) \*const `s` = nullptr

## 9.1248.1 Member Function Documentation

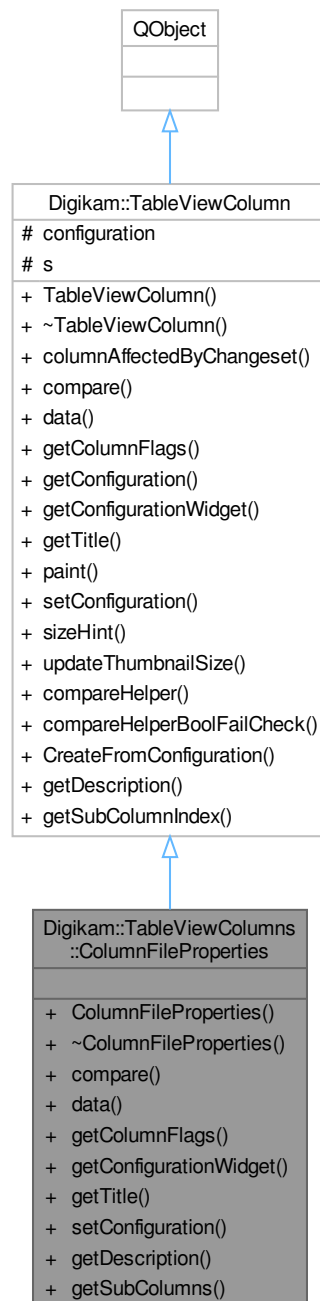
### 9.1248.1.1 `getNewConfiguration()`

[TableViewColumnConfiguration](#) `Digikam::TableViewColumns::ColumnFileConfigurationWidget::getNewConfiguration` ( ) [override], [virtual]

Implements [Digikam::TableViewColumnConfigurationWidget](#).

## 9.1249 Digikam::TableViewColumns::ColumnFileProperties Class Reference

Inheritance diagram for Digikam::TableViewColumns::ColumnFileProperties:



### Public Types

- enum `SubColumn` { `SubColumnName` = 0 , `SubColumnFilePath` = 1 , `SubColumnSize` = 2 , `SubColumnLastModified` = 3 }



## Public Types inherited from Digikam::TableViewColumn

- enum **ColumnCompareResult** { **CmpEqual** = 0 , **CmpABiggerB** = 1 , **CmpALessB** = 2 }
- enum **ColumnFlag** { **ColumnNoFlags** = 0 , **ColumnCustomPainting** = 1 , **ColumnCustomSorting** = 2 , **ColumnHasConfigurationWidget** = 4 }
- typedef QFlags< ColumnFlag > **ColumnFlags**

## Public Member Functions

- **ColumnFileProperties** ([TableViewShared](#) \*const tableViewShared, const [TableViewColumnConfiguration](#) &pConfiguration, const SubColumn pSubColumn, QObject \*const parent=nullptr)
- ColumnCompareResult **compare** ([TableViewModel::Item](#) \*const itemA, [TableViewModel::Item](#) \*const itemB) const override
 

*This function should never be called, because subclasses have to do the comparison on their own.*
- QVariant **data** ([TableViewModel::Item](#) \*const item, const int role) const override
- ColumnFlags **getColumnFlags** () const override
- [TableViewColumnConfigurationWidget](#) \* **getConfigurationWidget** (QWidget \*const parentWidget) const override
- QString **getTitle** () const override
- void **setConfiguration** (const [TableViewColumnConfiguration](#) &newConfiguration) override

## Public Member Functions inherited from Digikam::TableViewColumn

- **TableViewColumn** ([TableViewShared](#) \*const tableViewShared, const [TableViewColumnConfiguration](#) &pConfiguration, QObject \*const parent=nullptr)
- virtual bool **columnAffectedByChangeset** (const [ImageChangeset](#) &imageChangeset) const
- virtual [TableViewColumnConfiguration](#) **getConfiguration** () const
- virtual bool **paint** (QPainter \*const painter, const QStyleOptionViewItem &option, [TableViewModel::Item](#) \*const item) const
- virtual QSize **sizeHint** (const QStyleOptionViewItem &option, [TableViewModel::Item](#) \*const item) const
- virtual void **updateThumbnailSize** ()

## Static Public Member Functions

- static [TableViewColumnDescription](#) **getDescription** ()
- static QStringList **getSubColumns** ()

## Static Public Member Functions inherited from Digikam::TableViewColumn

- template<class MyType >
 

static ColumnCompareResult **compareHelper** (const MyType &A, const MyType &B)
- static bool **compareHelperBoolFailCheck** (const bool okA, const bool okB, ColumnCompareResult \*const result)
- template<typename columnClass >
 

static bool **CreateFromConfiguration** ([TableViewShared](#) \*const tableViewShared, const [TableViewColumnConfiguration](#) &pConfiguration, [TableViewColumn](#) \*\*const pNewColumn, QObject \*const parent)
- static [TableViewColumnDescription](#) **getDescription** ()
- template<typename columnClass >
 

static bool **getSubColumnIndex** (const QString &subColumnId, typename columnClass::SubColumn \*const subColumn)

## Additional Inherited Members

### Signals inherited from [Digikam::TableViewColumn](#)

- void **signalAllDataChanged** ()
- void **signalDataChanged** (const qlonglong imageld)

### Protected Attributes inherited from [Digikam::TableViewColumn](#)

- [TableViewColumnConfiguration](#) **configuration**
- [TableViewShared](#) \*const **s** = nullptr

## 9.1249.1 Member Function Documentation

### 9.1249.1.1 compare()

```
TableViewColumn::ColumnCompareResult Digikam::TableViewColumns::ColumnFileProperties::compare
(
    TableViewModel::Item *const itemA,
    TableViewModel::Item *const itemB ) const [override], [virtual]
```

But it can not be pure, since then every subclass which does not do custom comparison would have to implement an empty stub.

Reimplemented from [Digikam::TableViewColumn](#).

### 9.1249.1.2 data()

```
QVariant Digikam::TableViewColumns::ColumnFileProperties::data (
    TableViewModel::Item *const item,
    const int role ) const [override], [virtual]
```

Reimplemented from [Digikam::TableViewColumn](#).

### 9.1249.1.3 getColumnFlags()

```
TableViewColumn::ColumnFlags Digikam::TableViewColumns::ColumnFileProperties::getColumnFlags (
) const [override], [virtual]
```

Reimplemented from [Digikam::TableViewColumn](#).

### 9.1249.1.4 getConfigurationWidget()

```
TableViewColumnConfigurationWidget * Digikam::TableViewColumns::ColumnFileProperties::get↔
ConfigurationWidget (
    QWidget *const parentWidget ) const [override], [virtual]
```

Reimplemented from [Digikam::TableViewColumn](#).

**9.1249.1.5 getTitle()**

```
QString Digikam::TableViewColumns::ColumnFileProperties::getTitle ( ) const [override], [virtual]
```

Implements [Digikam::TableViewColumn](#).

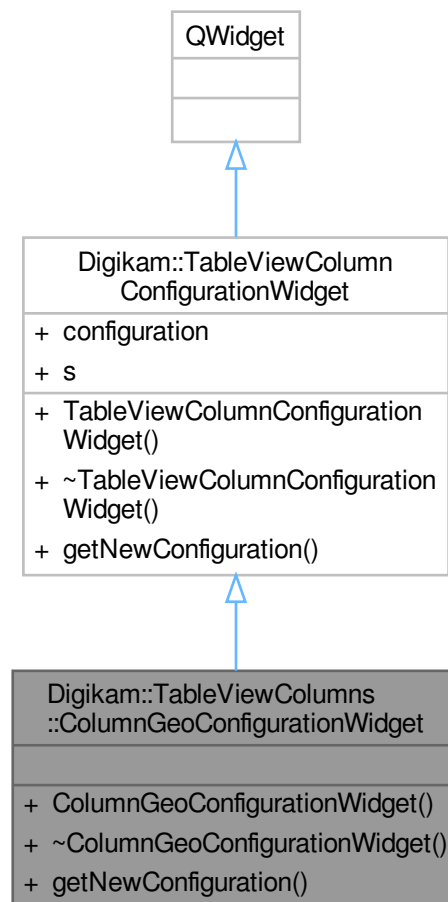
**9.1249.1.6 setConfiguration()**

```
void Digikam::TableViewColumns::ColumnFileProperties::setConfiguration (
    const TableViewColumnConfiguration & newConfiguration ) [override], [virtual]
```

Reimplemented from [Digikam::TableViewColumn](#).

## 9.1250 Digikam::TableViewColumns::ColumnGeoConfigurationWidget Class Reference

Inheritance diagram for Digikam::TableViewColumns::ColumnGeoConfigurationWidget:



## Public Member Functions

- **ColumnGeoConfigurationWidget** ([TableViewShared](#) \*const sharedObject, const [TableViewColumnConfiguration](#) &columnConfiguration, [QWidget](#) \*const parentWidget)
- [TableViewColumnConfiguration](#) `getNewConfiguration` () override

## Public Member Functions inherited from [Digikam::TableViewColumnConfigurationWidget](#)

- **TableViewColumnConfigurationWidget** ([TableViewShared](#) \*const sharedObject, const [TableViewColumnConfiguration](#) &currentConfiguration, [QWidget](#) \*const parent=nullptr)

## Additional Inherited Members

## Public Attributes inherited from [Digikam::TableViewColumnConfigurationWidget](#)

- [TableViewColumnConfiguration](#) **configuration**
- [TableViewShared](#) \*const **s** = nullptr

## 9.1250.1 Member Function Documentation

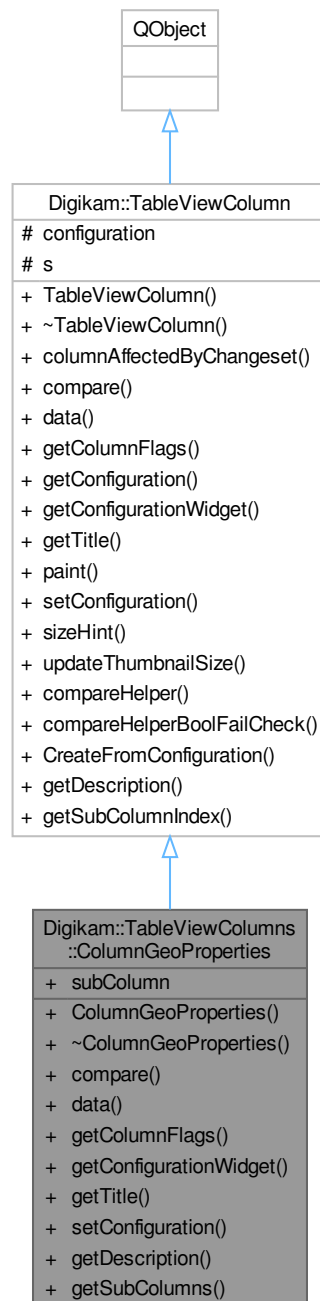
### 9.1250.1.1 `getNewConfiguration()`

[TableViewColumnConfiguration](#) `Digikam::TableViewColumns::ColumnGeoConfigurationWidget::getNewConfiguration` ( ) [override], [virtual]

Implements [Digikam::TableViewColumnConfigurationWidget](#).

## 9.1251 Digikam::TableViewColumns::ColumnGeoProperties Class Reference

Inheritance diagram for Digikam::TableViewColumns::ColumnGeoProperties:



### Public Types

- enum **SubColumn** { **SubColumnHasCoordinates** = 0 , **SubColumnCoordinates** = 1 , **SubColumnAltitude** = 2 }

## Public Types inherited from [Digikam::TableViewColumn](#)

- enum **ColumnCompareResult** { **CmpEqual** = 0 , **CmpABiggerB** = 1 , **CmpALessB** = 2 }
- enum **ColumnFlag** { **ColumnNoFlags** = 0 , **ColumnCustomPainting** = 1 , **ColumnCustomSorting** = 2 , **ColumnHasConfigurationWidget** = 4 }
- typedef QFlags< ColumnFlag > **ColumnFlags**

## Public Member Functions

- **ColumnGeoProperties** ([TableViewShared](#) \*const tableViewShared, const [TableViewColumnConfiguration](#) &pConfiguration, const SubColumn pSubColumn, QObject \*const parent=nullptr)
- ColumnCompareResult **compare** ([TableViewModel::Item](#) \*const itemA, [TableViewModel::Item](#) \*const itemB) const override  
*This function should never be called, because subclasses have to do the comparison on their own.*
- QVariant **data** ([TableViewModel::Item](#) \*const item, const int role) const override
- ColumnFlags **getColumnFlags** () const override
- [TableViewColumnConfigurationWidget](#) \* **getConfigurationWidget** (QWidget \*const parentWidget) const override
- QString **getTitle** () const override
- void **setConfiguration** (const [TableViewColumnConfiguration](#) &newConfiguration) override

## Public Member Functions inherited from [Digikam::TableViewColumn](#)

- **TableViewColumn** ([TableViewShared](#) \*const tableViewShared, const [TableViewColumnConfiguration](#) &pConfiguration, QObject \*const parent=nullptr)
- virtual bool **columnAffectedByChangeset** (const [ImageChangeset](#) &imageChangeset) const
- virtual [TableViewColumnConfiguration](#) **getConfiguration** () const
- virtual bool **paint** (QPainter \*const painter, const QStyleOptionViewItem &option, [TableViewModel::Item](#) \*const item) const
- virtual QSize **sizeHint** (const QStyleOptionViewItem &option, [TableViewModel::Item](#) \*const item) const
- virtual void **updateThumbnailSize** ()

## Static Public Member Functions

- static [TableViewColumnDescription](#) **getDescription** ()
- static QStringList **getSubColumns** ()

## Static Public Member Functions inherited from [Digikam::TableViewColumn](#)

- template<class MyType >  
static ColumnCompareResult **compareHelper** (const MyType &A, const MyType &B)
- static bool **compareHelperBoolFailCheck** (const bool okA, const bool okB, ColumnCompareResult \*const result)
- template<typename columnClass >  
static bool **CreateFromConfiguration** ([TableViewShared](#) \*const tableViewShared, const [TableViewColumnConfiguration](#) &pConfiguration, [TableViewColumn](#) \*\*const pNewColumn, QObject \*const parent)
- static [TableViewColumnDescription](#) **getDescription** ()
- template<typename columnClass >  
static bool **getSubColumnIndex** (const QString &subColumnId, typename columnClass::SubColumn \*const subColumn)

## Public Attributes

- enum Digikam::TableViewColumns::ColumnGeoProperties::SubColumn **subColumn**

## Additional Inherited Members

## Signals inherited from [Digikam::TableViewColumn](#)

- void **signalAllDataChanged** ()
- void **signalDataChanged** (const qlonglong imageId)

## Protected Attributes inherited from [Digikam::TableViewColumn](#)

- [TableViewColumnConfiguration](#) **configuration**
- [TableViewShared](#) \*const **s** = nullptr

## 9.1251.1 Member Function Documentation

### 9.1251.1.1 compare()

```
TableViewColumn::ColumnCompareResult Digikam::TableViewColumns::ColumnGeoProperties::compare (
    TableViewModel::Item *const itemA,
    TableViewModel::Item *const itemB ) const [override], [virtual]
```

But it can not be pure, since then every subclass which does not do custom comparison would have to implement an empty stub.

Reimplemented from [Digikam::TableViewColumn](#).

### 9.1251.1.2 data()

```
QVariant Digikam::TableViewColumns::ColumnGeoProperties::data (
    TableViewModel::Item *const item,
    const int role ) const [override], [virtual]
```

Reimplemented from [Digikam::TableViewColumn](#).

### 9.1251.1.3 getColumnFlags()

```
TableViewColumn::ColumnFlags Digikam::TableViewColumns::ColumnGeoProperties::getColumnFlags (
) const [override], [virtual]
```

Reimplemented from [Digikam::TableViewColumn](#).

#### 9.1251.1.4 `getConfigurationWidget()`

```
TableViewColumnConfigurationWidget * Digikam::TableViewColumns::ColumnGeoProperties::get←  
ConfigurationWidget (   
    QWidget *const parentWidget ) const [override], [virtual]
```

Reimplemented from [Digikam::TableViewColumn](#).

#### 9.1251.1.5 `getTitle()`

```
QString Digikam::TableViewColumns::ColumnGeoProperties::getTitle ( ) const [override], [virtual]
```

Implements [Digikam::TableViewColumn](#).

#### 9.1251.1.6 `setConfiguration()`

```
void Digikam::TableViewColumns::ColumnGeoProperties::setConfiguration (   
    const TableViewColumnConfiguration & newConfiguration ) [override], [virtual]
```

Reimplemented from [Digikam::TableViewColumn](#).



## 9.1252 Digikam::TableViewColumns::ColumnItemProperties Class Reference

Inheritance diagram for Digikam::TableViewColumns::ColumnItemProperties:



### Public Types

- enum `SubColumn` {  
`SubColumnWidth = 0` , `SubColumnHeight = 1` , `SubColumnDimensions = 2` , `SubColumnPixelCount =`

```

3 ,
SubColumnBitDepth = 4 , SubColumnColorMode = 5 , SubColumnType = 6 , SubColumnCreationDate↔
DateTime = 7 ,
SubColumnDigitizationDateTime = 8 , SubColumnAspectRatio = 9 , SubColumnSimilarity = 10 }

```

## Public Types inherited from [Digikam::TableViewColumn](#)

- enum **ColumnCompareResult** { **CmpEqual** = 0 , **CmpABiggerB** = 1 , **CmpALessB** = 2 }
- enum **ColumnFlag** { **ColumnNoFlags** = 0 , **ColumnCustomPainting** = 1 , **ColumnCustomSorting** = 2 , **ColumnHasConfigurationWidget** = 4 }
- typedef QFlags< ColumnFlag > **ColumnFlags**

## Public Member Functions

- **ColumnItemProperties** ([TableViewShared](#) \*const tableViewShared, const [TableViewColumnConfiguration](#) &pConfiguration, const SubColumn pSubColumn, QObject \*const parent=nullptr)
- ColumnCompareResult **compare** ([TableViewModel::Item](#) \*const itemA, [TableViewModel::Item](#) \*const itemB) const override  
*This function should never be called, because subclasses have to do the comparison on their own.*
- QVariant **data** ([TableViewModel::Item](#) \*const item, const int role) const override
- ColumnFlags **getColumnFlags** () const override
- QString **getTitle** () const override

## Public Member Functions inherited from [Digikam::TableViewColumn](#)

- **TableViewColumn** ([TableViewShared](#) \*const tableViewShared, const [TableViewColumnConfiguration](#) &pConfiguration, QObject \*const parent=nullptr)
- virtual bool **columnAffectedByChangeset** (const [ImageChangeset](#) &imageChangeset) const
- virtual [TableViewColumnConfiguration](#) **getConfiguration** () const
- virtual [TableViewColumnConfigurationWidget](#) \* **getConfigurationWidget** (QWidget \*const parentWidget) const
- virtual bool **paint** (QPainter \*const painter, const QStyleOptionViewItem &option, [TableViewModel::Item](#) \*const item) const
- virtual void **setConfiguration** (const [TableViewColumnConfiguration](#) &newConfiguration)
- virtual QSize **sizeHint** (const QStyleOptionViewItem &option, [TableViewModel::Item](#) \*const item) const
- virtual void **updateThumbnailSize** ()

## Static Public Member Functions

- static [TableViewColumnDescription](#) **getDescription** ()
- static QStringList **getSubColumns** ()

## Static Public Member Functions inherited from [Digikam::TableViewColumn](#)

- template<class MyType >  
static ColumnCompareResult **compareHelper** (const MyType &A, const MyType &B)
- static bool **compareHelperBoolFailCheck** (const bool okA, const bool okB, ColumnCompareResult \*const result)
- template<typename columnClass >  
static bool **CreateFromConfiguration** ([TableViewShared](#) \*const tableViewShared, const [TableViewColumnConfiguration](#) &pConfiguration, [TableViewColumn](#) \*\*const pNewColumn, QObject \*const parent)
- static [TableViewColumnDescription](#) **getDescription** ()
- template<typename columnClass >  
static bool **getSubColumnIndex** (const QString &subColumnId, typename columnClass::SubColumn \*const subColumn)

## Additional Inherited Members

## Signals inherited from [Digikam::TableViewColumn](#)

- void **signalAllDataChanged** ()
- void **signalDataChanged** (const qlonglong imageId)

## Protected Attributes inherited from [Digikam::TableViewColumn](#)

- [TableViewColumnConfiguration](#) **configuration**
- [TableViewShared](#) \*const **s** = nullptr

## 9.1252.1 Member Function Documentation

### 9.1252.1.1 compare()

```
TableViewColumn::ColumnCompareResult Digikam::TableViewColumns::ColumnItemProperties::compare
(
    TableViewModel::Item *const itemA,
    TableViewModel::Item *const itemB ) const [override], [virtual]
```

But it can not be pure, since then every subclass which does not do custom comparison would have to implement an empty stub.

Reimplemented from [Digikam::TableViewColumn](#).

### 9.1252.1.2 data()

```
QVariant Digikam::TableViewColumns::ColumnItemProperties::data (
    TableViewModel::Item *const item,
    const int role ) const [override], [virtual]
```

Reimplemented from [Digikam::TableViewColumn](#).

### 9.1252.1.3 getColumnFlags()

```
TableViewColumn::ColumnFlags Digikam::TableViewColumns::ColumnItemProperties::getColumnFlags (
) const [override], [virtual]
```

Reimplemented from [Digikam::TableViewColumn](#).

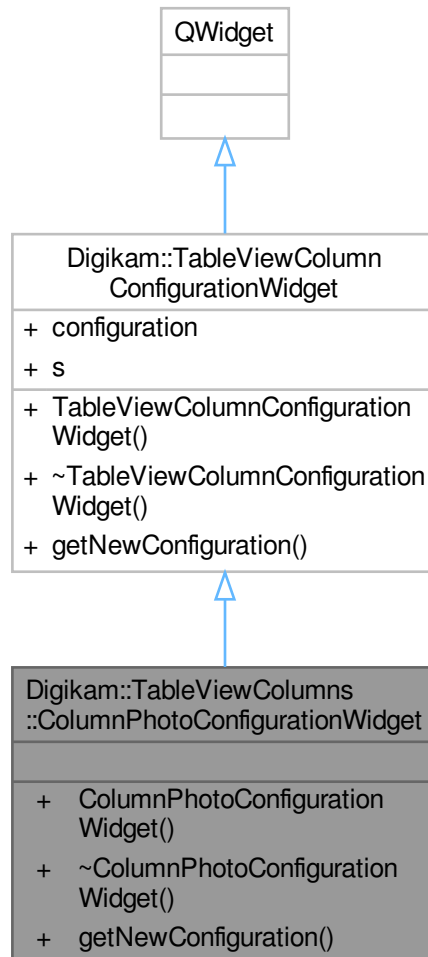
### 9.1252.1.4 getTitle()

```
QString Digikam::TableViewColumns::ColumnItemProperties::getTitle ( ) const [override], [virtual]
```

Implements [Digikam::TableViewColumn](#).

## 9.1253 Digikam::TableViewColumns::ColumnPhotoConfigurationWidget Class Reference

Inheritance diagram for Digikam::TableViewColumns::ColumnPhotoConfigurationWidget:



### Public Member Functions

- **ColumnPhotoConfigurationWidget** ([TableViewShared](#) \*const sharedObject, const [TableViewColumnConfiguration](#) &columnConfiguration, [QWidget](#) \*const parentWidget)
- [TableViewColumnConfiguration](#) `getNewConfiguration` () override

### Public Member Functions inherited from [Digikam::TableViewColumnConfigurationWidget](#)

- **TableViewColumnConfigurationWidget** ([TableViewShared](#) \*const sharedObject, const [TableViewColumnConfiguration](#) &currentConfiguration, [QWidget](#) \*const parent=nullptr)

## Additional Inherited Members

### Public Attributes inherited from [Digikam::TableViewColumnConfigurationWidget](#)

- [TableViewColumnConfiguration](#) **configuration**
- [TableViewShared](#) \*const **s** = nullptr

## 9.1253.1 Member Function Documentation

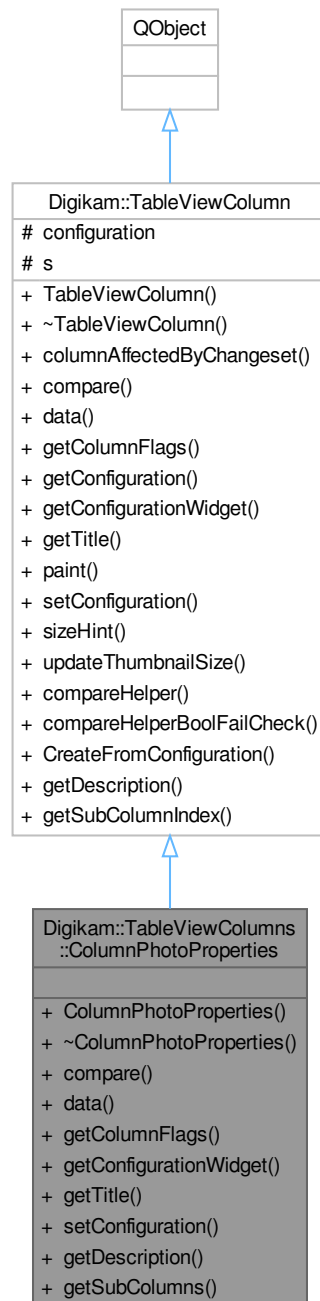
### 9.1253.1.1 [getNewConfiguration\(\)](#)

[TableViewColumnConfiguration](#) Digikam::TableViewColumns::ColumnPhotoConfigurationWidget::get↔  
NewConfiguration ( ) [override], [virtual]

Implements [Digikam::TableViewColumnConfigurationWidget](#).

## 9.1254 Digikam::TableViewColumns::ColumnPhotoProperties Class Reference

Inheritance diagram for Digikam::TableViewColumns::ColumnPhotoProperties:



### Public Types

- enum `SubColumn` {  
`SubColumnCameraMaker = 0` , `SubColumnCameraModel = 1` , `SubColumnLens = 2` , `SubColumnAperture = 3` ,

```
SubColumnFocal = 4 , SubColumnExposure = 5 , SubColumnSensitivity = 6 , SubColumnMode↔
Program = 7 ,
SubColumnFlash = 8 , SubColumnWhiteBalance = 9 }
```

## Public Types inherited from Digikam::TableViewColumn

- enum **ColumnCompareResult** { **CmpEqual** = 0 , **CmpABiggerB** = 1 , **CmpALessB** = 2 }
- enum **ColumnFlag** { **ColumnNoFlags** = 0 , **ColumnCustomPainting** = 1 , **ColumnCustomSorting** = 2 , **ColumnHasConfigurationWidget** = 4 }
- typedef QFlags< ColumnFlag > **ColumnFlags**

## Public Member Functions

- **ColumnPhotoProperties** ([TableViewShared](#) \*const tableViewShared, const [TableViewColumnConfiguration](#) &pConfiguration, const SubColumn pSubColumn, QObject \*const parent=nullptr)
- ColumnCompareResult **compare** ([TableViewModel::Item](#) \*const itemA, [TableViewModel::Item](#) \*const itemB) const override
 

*This function should never be called, because subclasses have to do the comparison on their own.*
- QVariant **data** ([TableViewModel::Item](#) \*const item, const int role) const override
- ColumnFlags **getColumnFlags** () const override
- [TableViewColumnConfigurationWidget](#) \* **getConfigurationWidget** (QWidget \*const parentWidget) const override
- QString **getTitle** () const override
- void **setConfiguration** (const [TableViewColumnConfiguration](#) &newConfiguration) override

## Public Member Functions inherited from Digikam::TableViewColumn

- **TableViewColumn** ([TableViewShared](#) \*const tableViewShared, const [TableViewColumnConfiguration](#) &p↔Configuration, QObject \*const parent=nullptr)
- virtual bool **columnAffectedByChangeset** (const [ImageChangeset](#) &imageChangeset) const
- virtual [TableViewColumnConfiguration](#) **getConfiguration** () const
- virtual bool **paint** (QPainter \*const painter, const QStyleOptionViewItem &option, [TableViewModel::Item](#) \*const item) const
- virtual QSize **sizeHint** (const QStyleOptionViewItem &option, [TableViewModel::Item](#) \*const item) const
- virtual void **updateThumbnailSize** ()

## Static Public Member Functions

- static [TableViewColumnDescription](#) **getDescription** ()
- static QStringList **getSubColumns** ()

## Static Public Member Functions inherited from Digikam::TableViewColumn

- template<class MyType >
 

```
static ColumnCompareResult compareHelper (const MyType &A, const MyType &B)
```
- static bool **compareHelperBoolFailCheck** (const bool okA, const bool okB, ColumnCompareResult \*const result)
- template<typename columnClass >
 

```
static bool CreateFromConfiguration (TableViewShared *const tableViewShared, const TableViewColumnConfiguration &pConfiguration, TableViewColumn **const pNewColumn, QObject *const parent)
```
- static [TableViewColumnDescription](#) **getDescription** ()
- template<typename columnClass >
 

```
static bool getSubColumnIndex (const QString &subColumnId, typename columnClass::SubColumn *const subColumn)
```

## Additional Inherited Members

### Signals inherited from [Digikam::TableViewColumn](#)

- void **signalAllDataChanged** ()
- void **signalDataChanged** (const qlonglong imageld)

### Protected Attributes inherited from [Digikam::TableViewColumn](#)

- [TableViewColumnConfiguration](#) **configuration**
- [TableViewShared](#) \*const **s** = nullptr

## 9.1254.1 Member Function Documentation

### 9.1254.1.1 compare()

```
TableViewColumn::ColumnCompareResult Digikam::TableViewColumns::ColumnPhotoProperties::compare
(
    TableViewModel::Item *const itemA,
    TableViewModel::Item *const itemB ) const [override], [virtual]
```

But it can not be pure, since then every subclass which does not do custom comparison would have to implement an empty stub.

Reimplemented from [Digikam::TableViewColumn](#).

### 9.1254.1.2 data()

```
QVariant Digikam::TableViewColumns::ColumnPhotoProperties::data (
    TableViewModel::Item *const item,
    const int role ) const [override], [virtual]
```

Reimplemented from [Digikam::TableViewColumn](#).

### 9.1254.1.3 getColumnFlags()

```
TableViewColumn::ColumnFlags Digikam::TableViewColumns::ColumnPhotoProperties::getColumnFlags
( ) const [override], [virtual]
```

Reimplemented from [Digikam::TableViewColumn](#).

### 9.1254.1.4 getConfigurationWidget()

```
TableViewColumnConfigurationWidget * Digikam::TableViewColumns::ColumnPhotoProperties::get↔
ConfigurationWidget (
    QWidget *const parentWidget ) const [override], [virtual]
```

Reimplemented from [Digikam::TableViewColumn](#).



### 9.1254.1.5 getTitle()

```
QString Digikam::TableViewColumns::ColumnPhotoProperties::getTitle ( ) const [override],  
[virtual]
```

Implements [Digikam::TableViewColumn](#).

### 9.1254.1.6 setConfiguration()

```
void Digikam::TableViewColumns::ColumnPhotoProperties::setConfiguration (   
    const TableViewColumnConfiguration & newConfiguration ) [override], [virtual]
```

Reimplemented from [Digikam::TableViewColumn](#).

## 9.1255 Digikam::TableViewColumns::ColumnThumbnail Class Reference

Inheritance diagram for Digikam::TableViewColumns::ColumnThumbnail:



### Public Member Functions

- QVariant `data` (`TableViewModel::Item *const item`, `const int role`) `const` override
- ColumnFlags `getColumnFlags` () `const` override

- QString [getTitle](#) () const override
- bool [paint](#) (QPainter \*const painter, const QStyleOptionViewItem &option, [TableViewModel::Item](#) \*const item) const override
- QSize [sizeHint](#) (const QStyleOptionViewItem &option, [TableViewModel::Item](#) \*const item) const override
- void [updateThumbnailSize](#) () override

## Public Member Functions inherited from [Digikam::TableViewColumn](#)

- [TableViewColumn](#) ([TableViewShared](#) \*const tableViewShared, const [TableViewColumnConfiguration](#) &pConfiguration, QObject \*const parent=nullptr)
- virtual bool [columnAffectedByChangeset](#) (const [ImageChangeset](#) &imageChangeset) const
- virtual ColumnCompareResult [compare](#) ([TableViewModel::Item](#) \*const itemA, [TableViewModel::Item](#) \*const itemB) const
 

*This function should never be called, because subclasses have to do the comparison on their own.*
- virtual [TableViewColumnConfiguration](#) [getConfiguration](#) () const
- virtual [TableViewColumnConfigurationWidget](#) \* [getConfigurationWidget](#) (QWidget \*const parentWidget) const
- virtual void [setConfiguration](#) (const [TableViewColumnConfiguration](#) &newConfiguration)

## Static Public Member Functions

- static bool [CreateFromConfiguration](#) ([TableViewShared](#) \*const tableViewShared, const [TableViewColumnConfiguration](#) &pConfiguration, [TableViewColumn](#) \*\*const pNewColumn, QWidget \*const parent)
- static [TableViewColumnDescription](#) [getDescription](#) ()

## Static Public Member Functions inherited from [Digikam::TableViewColumn](#)

- template<class MyType >
  - static ColumnCompareResult [compareHelper](#) (const MyType &A, const MyType &B)
- static bool [compareHelperBoolFailCheck](#) (const bool okA, const bool okB, ColumnCompareResult \*const result)
- template<typename columnClass >
  - static bool [CreateFromConfiguration](#) ([TableViewShared](#) \*const tableViewShared, const [TableViewColumnConfiguration](#) &pConfiguration, [TableViewColumn](#) \*\*const pNewColumn, QObject \*const parent)
  - static [TableViewColumnDescription](#) [getDescription](#) ()
- template<typename columnClass >
  - static bool [getSubColumnIndex](#) (const QString &subColumnId, typename columnClass::SubColumn \*const subColumn)

## Additional Inherited Members

## Public Types inherited from [Digikam::TableViewColumn](#)

- enum [ColumnCompareResult](#) { [CmpEqual](#) = 0 , [CmpABiggerB](#) = 1 , [CmpALessB](#) = 2 }
- enum [ColumnFlag](#) { [ColumnNoFlags](#) = 0 , [ColumnCustomPainting](#) = 1 , [ColumnCustomSorting](#) = 2 , [ColumnHasConfigurationWidget](#) = 4 }
- typedef QFlags< [ColumnFlag](#) > [ColumnFlags](#)

## Signals inherited from [Digikam::TableViewColumn](#)

- void **signalAllDataChanged** ()
- void **signalDataChanged** (const qlonglong imageId)

## Protected Attributes inherited from [Digikam::TableViewColumn](#)

- [TableViewColumnConfiguration](#) **configuration**
- [TableViewShared](#) \*const **s** = nullptr

## 9.1255.1 Member Function Documentation

### 9.1255.1.1 data()

```
QVariant Digikam::TableViewColumns::ColumnThumbnail::data (
    TableViewModel::Item *const item,
    const int role ) const [override], [virtual]
```

Reimplemented from [Digikam::TableViewColumn](#).

### 9.1255.1.2 getColumnFlags()

```
TableViewColumn::ColumnFlags Digikam::TableViewColumns::ColumnThumbnail::getColumnFlags ( )
const [override], [virtual]
```

Reimplemented from [Digikam::TableViewColumn](#).

### 9.1255.1.3 getTitle()

```
QString Digikam::TableViewColumns::ColumnThumbnail::getTitle ( ) const [override], [virtual]
```

Implements [Digikam::TableViewColumn](#).

### 9.1255.1.4 paint()

```
bool Digikam::TableViewColumns::ColumnThumbnail::paint (
    QPainter *const painter,
    const QStyleOptionViewItem & option,
    TableViewModel::Item *const item ) const [override], [virtual]
```

Reimplemented from [Digikam::TableViewColumn](#).

### 9.1255.1.5 sizeHint()

```
QSize Digikam::TableViewColumns::ColumnThumbnail::sizeHint (
    const QStyleOptionViewItem & option,
    TableViewModel::Item *const item ) const [override], [virtual]
```

Reimplemented from [Digikam::TableViewColumn](#).

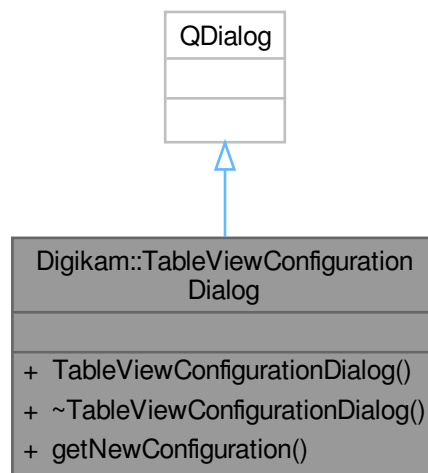
### 9.1255.1.6 updateThumbnailSize()

```
void Digikam::TableViewColumns::ColumnThumbnail::updateThumbnailSize ( ) [override], [virtual]
```

Reimplemented from [Digikam::TableViewColumn](#).

## 9.1256 Digikam::TableViewConfigurationDialog Class Reference

Inheritance diagram for Digikam::TableViewConfigurationDialog:

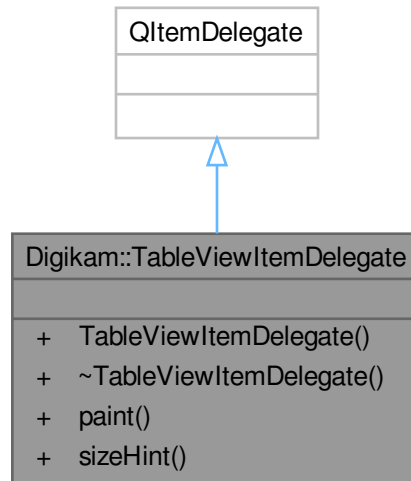


### Public Member Functions

- **TableViewConfigurationDialog** ([TableViewShared](#) \*const sharedObject, const int columnIndex, QWidget \*const parentWidget)
- [TableViewColumnConfiguration](#) **getNewConfiguration** () const

## 9.1257 Digikam::TableViewItemDelegate Class Reference

Inheritance diagram for Digikam::TableViewItemDelegate:



### Public Member Functions

- **TableViewItemDelegate** ([TableViewShared](#) \*const tableViewShared, QObject \*const parent=nullptr)
- void **paint** (QPainter \*painter, const QStyleOptionViewItem &option, const QModelIndex &tableViewModelIndex) const override
- QSize **sizeHint** (const QStyleOptionViewItem &option, const QModelIndex &tableViewModelIndex) const override

### 9.1257.1 Member Function Documentation

#### 9.1257.1.1 sizeHint()

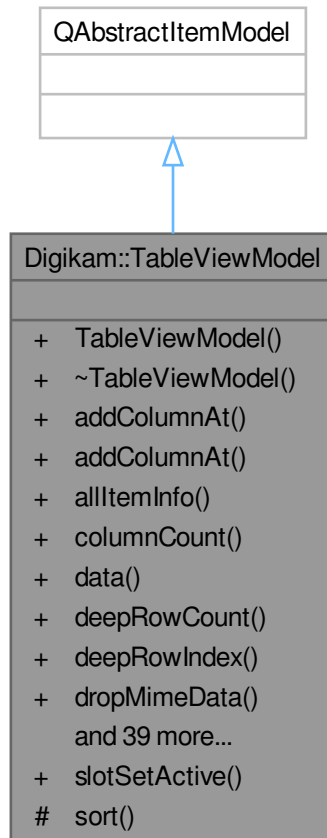
```

QSize Digikam::TableViewItemDelegate::sizeHint (
    const QStyleOptionViewItem & option,
    const QModelIndex & tableViewModelIndex ) const [override]
  
```

we have to take the maximum of all columns for the height

## 9.1258 Digikam::TableViewModel Class Reference

Inheritance diagram for Digikam::TableViewModel:



### Classes

- class [Item](#)

### Public Types

- typedef [DatabaseFields::Hash](#)< QVariant > **DatabaseFieldsHashRaw**
- enum **GroupingMode** { **GroupingHideGrouped** = 0 , **GroupingIgnoreGrouping** = 1 , **GroupingShow**↔  
**SubItems** = 2 }

### Public Slots

- void **slotSetActive** (const bool isActive)

## Signals

- void **signalGroupingModeChanged** ()

## Public Member Functions

- **TableViewModel** ([TableViewShared](#) \*const sharedObject, QObject \*const [parent](#)=nullptr)
- void **addColumnAt** (const [TableViewColumnConfiguration](#) &cpp, const int targetColumn=-1)
- void **addColumnAt** (const [TableViewColumnDescription](#) &description, const int targetColumn=-1)
- QList< [ItemInfo](#) > **allItemInfo** () const
- int **columnCount** (const QModelIndex &i) const override
- QVariant **data** (const QModelIndex &i, int role) const override
- int **deepRowCount** () const
- QModelIndex **deepRowIndex** (const int rowNumber) const
- bool **dropMimeData** (const QMimeData \*data, Qt::DropAction action, int row, int column, const QModelIndex &parent) override
- int **findChildSortedPosition** ([Item](#) \*const parentItem, [Item](#) \*const childItem)
- int **firstDeepRowNotInList** (const QList< QModelIndex > &needleList)
- Qt::ItemFlags **flags** (const QModelIndex &index) const override
- QModelIndex **fromItemFilterModelIndex** (const QModelIndex &imageFilterModelIndex)
- QModelIndex **fromItemModelIndex** (const QModelIndex &imageModelIndex)
- [TableViewColumn](#) \* **getColumnObject** (const int columnIndex)
- QList< [TableViewColumn](#) \* > **getColumnObjects** ()
- [TableViewColumnProfile](#) **getColumnProfile** () const
- GroupingMode **groupingMode** () const
- bool **hasChildren** (const QModelIndex &parent=QModelIndex()) const override
- QVariant **headerData** (int section, Qt::Orientation orientation, int role) const override
- qlonglong **imageId** (const QModelIndex &anIndex) const
- QList< qlonglong > **imageIds** (const QModelIndexList &indexList) const
- [ItemInfo](#) **imageInfo** (const QModelIndex &index) const
- QList< [ItemInfo](#) > **imageInfos** (const QModelIndexList &indexList) const
- QModelIndex **index** (int row, int column, const QModelIndex &parent=QModelIndex()) const override
- QModelIndex **indexFromImageId** (const qlonglong imageId, const int columnIndex) const
- int **indexToDeepRowNumber** (const QModelIndex &index) const
- [ItemInfo](#) **infoFromItem** ([Item](#) \*const item) const
- [ItemInfoList](#) **infosFromItems** (const QList< [Item](#) \* > &items) const
- QVariant **itemDatabaseFieldRaw** ([Item](#) \*const item, const [DatabaseFields::Set](#) &requestedField)
- [DatabaseFieldsHashRaw](#) **itemDatabaseFieldsRaw** ([Item](#) \*const item, const [DatabaseFields::Set](#) &requestedSet)
- [Item](#) \* **itemFromImageId** (const qlonglong imageId) const
- [Item](#) \* **itemFromIndex** (const QModelIndex &i) const
- QModelIndex **itemIndex** ([Item](#) \*const item) const
- bool **lessThan** ([Item](#) \*const itemA, [Item](#) \*const itemB)
- void **loadColumnProfile** (const [TableViewColumnProfile](#) &columnProfile)
- QMimeData \* **mimeData** (const QModelIndexList &indexes) const override
- QStringList **mimeTypes** () const override
- QModelIndex **parent** (const QModelIndex &childIndex) const override
- void **removeColumnAt** (const int columnIndex)
- int **rowCount** (const QModelIndex &parent) const override
- void **scheduleResort** ()
- void **setGroupingMode** (const GroupingMode newGroupingMode)
- QList< [Item](#) \* > **sortItems** (const QList< [Item](#) \* > &itemList)
- Qt::DropActions **supportedDropActions** () const override
- *drag-and-drop related functions*
- QModelIndex **toColId** (const QModelIndex &anIndex) const
- QModelIndex **toItemFilterModelIndex** (const QModelIndex &i) const
- QModelIndex **toItemModelIndex** (const QModelIndex &i) const



## Protected Member Functions

- void [sort](#) (int column, Qt::SortOrder order=Qt::AscendingOrder) override

## 9.1258.1 Member Function Documentation

### 9.1258.1.1 addColumnAt()

```
void Digikam::TableViewModel::addColumnAt (
    const TableViewColumnDescription & description,
    const int targetColumn = -1 )
```

### 9.1258.1.2 flags()

```
Qt::ItemFlags Digikam::TableViewModel::flags (
    const QModelIndex & index ) const [override]
```

### 9.1258.1.3 indexFromImageId()

```
QModelIndex Digikam::TableViewModel::indexFromImageId (
    const qlonglong imageId,
    const int columnIndex ) const
```

### 9.1258.1.4 infoFromItem()

```
ItemInfo Digikam::TableViewModel::infoFromItem (
    TableViewModel::Item *const item ) const
```

### 9.1258.1.5 loadColumnProfile()

```
void Digikam::TableViewModel::loadColumnProfile (
    const TableViewColumnProfile & columnProfile )
```

### 9.1258.1.6 parent()

```
QModelIndex Digikam::TableViewModel::parent (
    const QModelIndex & childIndex ) const [override]
```

### 9.1258.1.7 sort()

```
void Digikam::TableViewModel::sort (
    int column,
    Qt::SortOrder order = Qt::AscendingOrder ) [override], [protected]
```

## 9.1259 Digikam::TableViewModel::Item Class Reference

### Public Member Functions

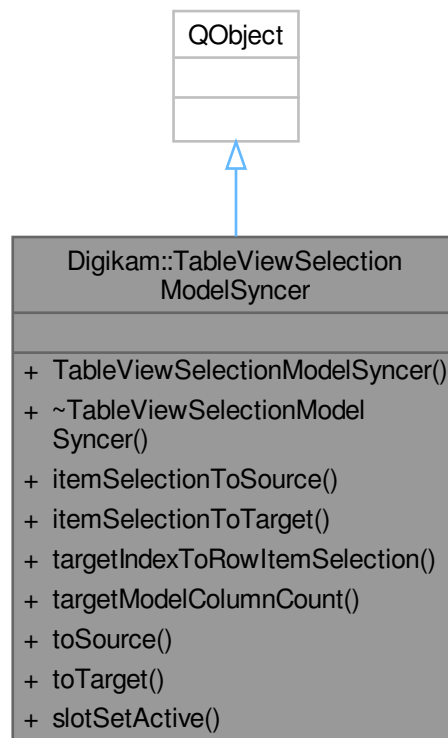
- void **addChild** ([Item](#) \*const newChild)
- [Item](#) \* **findChildWithImageId** (const qlonglong searchImageId)
- void **insertChild** (const int pos, [Item](#) \*const newChild)
- void **takeChild** ([Item](#) \*const oldChild)

### Public Attributes

- QList< [Item](#) \* > **children**
- qlonglong **imageId** = 0
- [Item](#) \* **parent** = nullptr

## 9.1260 Digikam::TableViewSelectionModeSyncer Class Reference

Inheritance diagram for Digikam::TableViewSelectionModeSyncer:



### Public Slots

- void **slotSetActive** (const bool isActive)

**Public Member Functions**

- [TableViewSelectionModeSyncer](#) ([TableViewShared](#) \*const sharedObject, QObject \*const parent=nullptr)
- QItemSelection **itemSelectionToSource** (const QItemSelection &selection) const
- QItemSelection **itemSelectionToTarget** (const QItemSelection &selection) const
- QItemSelection **targetIndexToRowItemSelection** (const QModelIndex &targetIndex) const
- int **targetModelColumnCount** () const
- QModelIndex **toSource** (const QModelIndex &targetIndex) const
- QModelIndex **toTarget** (const QModelIndex &sourceIndex) const

**9.1260.1 Constructor & Destructor Documentation****9.1260.1.1 TableViewSelectionModeSyncer()**

```
Digikam::TableViewSelectionModeSyncer::TableViewSelectionModeSyncer (
    TableViewShared *const sharedObject,
    QObject *const parent = nullptr ) [explicit]
```

**9.1261 Digikam::TableViewShared Class Reference****Public Attributes**

- [TableViewColumnFactory](#) \* **columnFactory** = nullptr
- [ItemFilterModel](#) \* **imageFilterModel** = nullptr
- QItemSelectionModel \* **imageFilterSelectionModel** = nullptr
- [ItemModel](#) \* **imageModel** = nullptr
- bool **isActive** = false
- [TableViewItemDelegate](#) \* **itemDelegate** = nullptr
- [TableView](#) \* **tableView** = nullptr
- [TableViewModel](#) \* **tableViewModel** = nullptr
- QItemSelectionModel \* **tableViewSelectionModel** = nullptr
- [TableViewSelectionModeSyncer](#) \* **tableViewSelectionModeSyncer** = nullptr
- [ThumbnailLoadThread](#) \* **thumbnailLoadThread** = nullptr
- [TableViewTreeView](#) \* **treeView** = nullptr

## 9.1262 Digikam::TableViewTreeView Class Reference

Inheritance diagram for Digikam::TableViewTreeView:



### Signals

- void **signalZoomInStep** ()
- void **signalZoomOutStep** ()

### Public Member Functions

- **TableViewTreeView** ([TableViewShared](#) \*const tableViewShared, [QWidget](#) \*const parent=nullptr)
- [Album](#) \* **albumAt** (const [QPoint](#) &pos) const

### Public Member Functions inherited from [Digikam::DragDropViewImplementation](#)

- virtual void **copy** ()
- virtual void **cut** ()
- virtual void **paste** ()

## Public Member Functions inherited from Digikam::GroupingViewImplementation

- [ItemInfoList](#) **getHiddenGroupedInfos** (const [ItemInfoList](#) &infos) const
- bool **needGroupResolving** ([OperationType](#) type, const [ItemInfoList](#) &infos) const
- [ItemInfoList](#) **resolveGrouping** (const [ItemInfoList](#) &infos) const

## Protected Member Functions

- [AbstractItemDragDropHandler](#) \* **dragDropHandler** () const override  
*You need to implement these three methods Returns the drag drop handler.*
- bool **eventFilter** (QObject \*watched, QEvent \*event) override
- bool **hasHiddenGroupedImages** (const [ItemInfo](#) &info) const override  
*must be implemented by parent view*
- QModelIndex **mapIndexForDragDrop** (const QModelIndex &index) const override  
*Maps the given index of the view's model to an index of the handler's model, which can be a source model of the view's model.*
- QPixmap **pixmapForDrag** (const QList< QModelIndex > &indexes) const override  
*Creates a pixmap for dragging the given indexes.*
- void **wheelEvent** (QWheelEvent \*event) override

## Protected Member Functions inherited from Digikam::DragDropViewImplementation

- virtual QAbstractItemView \* **asView** ()=0  
*This one is implemented by DECLARE\_VIEW\_DRAG\_DROP\_METHODS.*
- bool **decodelsCutSelection** (const QMimeData \*mimeData)
- void **dragEnterEvent** (QDragEnterEvent \*event)  
*Implements the relevant QAbstractItemView methods for drag and drop.*
- void **dragMoveEvent** (QDragMoveEvent \*e)
- void **dropEvent** (QDropEvent \*e)
- void **encodelsCutSelection** (QMimeData \*mime, bool isCutSelection)
- void **startDrag** (Qt::DropActions supportedActions)

### 9.1262.1 Detailed Description

### 9.1262.2 Member Function Documentation

#### 9.1262.2.1 dragDropHandler()

```
AbstractItemDragDropHandler * Digikam::TableViewTreeView::dragDropHandler ( ) const [override],
[protected], [virtual]
```

Implements [Digikam::DragDropViewImplementation](#).

#### 9.1262.2.2 hasHiddenGroupedImages()

```
bool Digikam::TableViewTreeView::hasHiddenGroupedImages (
    const ItemInfo & ) const [override], [protected], [virtual]
```

Reimplemented from [Digikam::GroupingViewImplementation](#).

### 9.1262.2.3 mapIndexForDragDrop()

```
QModelIndex Digikam::TableViewTreeView::mapIndexForDragDrop (
    const QModelIndex & index ) const [override], [protected], [virtual]
```

Implements [Digikam::DragDropViewImplementation](#).

### 9.1262.2.4 pixmapForDrag()

```
QPixmap Digikam::TableViewTreeView::pixmapForDrag (
    const QList< QModelIndex > & indexes ) const [override], [protected], [virtual]
```

Implements [Digikam::DragDropViewImplementation](#).

## 9.1263 Digikam::TagChangeset Class Reference

### Public Types

- enum [Operation](#) {
  - Unknown** , **Added** , **Moved** , **Deleted** ,
  - Renamed** , **Reparented** , **IconChanged** , [PropertiesChanged](#) }

### Public Member Functions

- **TagChangeset** (int tagId, [Operation](#) operation)
- [Operation](#) **operation** () const
- [TagChangeset](#) & **operator**<< (const QDBusArgument &argument)
- const [TagChangeset](#) & **operator**>> (QDBusArgument &argument) const
- int **tagId** () const

### 9.1263.1 Member Enumeration Documentation

#### 9.1263.1.1 Operation

```
enum Digikam::TagChangeset::Operation
```

#### Enumerator

PropertiesChanged	ImageTagProperties Table.
-------------------	---------------------------



## Public Types inherited from [Digikam::AbstractAlbumTreeView](#)

- enum [Flag](#) {  
[CreateDefaultModel](#) , [CreateDefaultFilterModel](#) , [CreateDefaultDelegate](#) , [ShowCountAccordingToSettings](#) ,  
[AlwaysShowInclusiveCounts](#) , **DefaultFlags** = [CreateDefaultFilterModel](#) | [CreateDefaultDelegate](#) | [ShowCountAccordingToSettings](#) }
- typedef QFlags< [Flag](#) > **Flags**

## Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }  
*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

## Public Slots

- void **slotResetCheckState** ()  
*Resets the whole tag filter.*

## Public Slots inherited from [Digikam::TagFolderView](#)

- void **slotTagNewFromABCMenu** (const QString &personName)

## Public Slots inherited from [Digikam::TagTreeView](#)

- void **setCurrentAlbum** (int tagId, bool selectInAlbumManager=true)
- void **setCurrentAlbums** (const QList< [Album](#) \* > &tags, bool selectInAlbumManager=true)

## Public Slots inherited from [Digikam::AbstractAlbumTreeView](#)

- void **adaptColumnsToContent** ()  
*Adapt the column sizes to the contents of the tree view.*
- void **expandEverything** (const QModelIndex &index)  
*Expands the complete tree under the given index.*
- void **scrollToSelectedAlbum** ()  
*Scrolls to the first selected album if there is one.*
- void **setCurrentAlbums** (const QList< [Album](#) \* > &albums, bool selectInAlbumManager=true)  
*Selects the given album.*
- void **setSearchTextSettings** (const [SearchTextSettings](#) &settings)
- void **slotCollapseAllNodes** ()  
*slotCollapseAllNodes - collapse all nodes without root node*
- void **slotCollapseNode** ()  
*slotCollapseNode - collapse recursively selected nodes*
- void **slotExpandNode** ()  
*slotExpandNode - expands recursively selected nodes*



## Signals

- void [checkedTagsChanged](#) (const QList< [TAlbum](#) \* > &includedTags, const QList< [TAlbum](#) \* > &excludedTags)  
*Emitted if the checked tags have changed.*

## Signals inherited from [Digikam::TagFolderView](#)

- void [signalFindDuplicates](#) (const QList< [TAlbum](#) \* > &albums)

## Signals inherited from [Digikam::TagTreeView](#)

- void [assignTags](#) (int tagId, const QList< int > &imageIds)

## Signals inherited from [Digikam::AbstractAlbumTreeView](#)

- void [currentAlbumChanged](#) ([Album](#) \*currentAlbum)  
*Emitted when the currently selected album changes.*
- void [selectedAlbumsChanged](#) (const QList< [Album](#) \* > &selectedAlbums)  
*Emitted when the current selection changes.*

## Public Member Functions

- [TagCheckView](#) (QWidget \*const parent, [TagModel](#) \*const tagModel)
- bool [checkNewTags](#) () const
- void [doLoadState](#) () override  
*Implements state loading for the album tree view in a somewhat clumsy procedure because the model may not be fully loaded when this method is called.*
- void [doSaveState](#) () override  
*Implement this hook method for state saving.*
- QList< [TAlbum](#) \* > [getCheckedTags](#) () const
- QList< [TAlbum](#) \* > [getPartiallyCheckedTags](#) () const
- ToggleAutoTags [getToggleAutoTags](#) () const
- void [setCheckNewTags](#) (bool checkNewTags)  
*if.*
- void [setToggleAutoTags](#) (ToggleAutoTags toggle)

## Public Member Functions inherited from [Digikam::TagFolderView](#)

- [TagFolderView](#) (QWidget \*const parent, [TagModel](#) \*const model)  
*Constructor.*
- [~TagFolderView](#) () override  
*Destructor.*
- void [setShowDeleteFaceTagsAction](#) (bool show)  
*Define whether to show the "Delete People Tags" action in context menus or not.*
- void [setShowFindDuplicateAction](#) (bool show)  
*Define whether to show the "find duplicate" action in context menus or not.*
- void [tagPropsEdit](#) ()  
*Open tag for editing.*

## Public Member Functions inherited from [Digikam::TagTreeView](#)

- [TagTreeView](#) (QWidget \*const parent=nullptr, Flags flags=DefaultFlags)
- [TAlbum](#) \* [albumForIndex](#) (const QModelIndex &index) const
- [TagModel](#) \* [albumModel](#) () const
- [TAlbum](#) \* [currentAlbum](#) () const  
*currentAlbum Even if multiple selection is enabled current [Album](#) can be only one, the last clicked item if you need selected items, see [selectedAlbums\(\)](#) It's NOT the same as [AlbumManager::currentAlbums\(\)](#)*
- [TagPropertiesFilterModel](#) \* [filteredModel](#) () const  
*Contains only the tags filtered by properties - prefer to [albumModel\(\)](#)*
- QList< [TAlbum](#) \* > [selectedTagAlbums](#) ()
- QList< [Album](#) \* > [selectedTags](#) ()  
*selectedTags - return a list of all selected items in tag model*
- void [setAlbumFilterModel](#) ([TagPropertiesFilterModel](#) \*const [filteredModel](#), [CheckableAlbumFilterModel](#) \*const [filterModel](#))
- void [setAlbumModel](#) ([TagModel](#) \*const [model](#))
- [TagModificationHelper](#) \* [tagModificationHelper](#) () const

## Public Member Functions inherited from [Digikam::AbstractCheckableAlbumTreeView](#)

- [AbstractCheckableAlbumTreeView](#) (QWidget \*const parent, Flags flags)  
*Models of these view can be checkable, they need not.*
- [CheckableAlbumFilterModel](#) \* [albumFilterModel](#) () const
- [AbstractCheckableAlbumModel](#) \* [albumModel](#) () const  
*Manage check state through the model directly.*
- [CheckableAlbumFilterModel](#) \* [checkableAlbumFilterModel](#) () const
- [AbstractCheckableAlbumModel](#) \* [checkableModel](#) () const
- bool [isRestoreCheckState](#) () const  
*Tells if the check state is restored while loading / saving state.*
- void [setCheckOnMiddleClick](#) (bool doThat)  
*Enable checking on middle mouse button click (default: on).*
- void [setRestoreCheckState](#) (bool restore)  
*Set whether to restore check state or not.*

## Public Member Functions inherited from [Digikam::AbstractCountingAlbumTreeView](#)

- [AbstractCountingAlbumTreeView](#) (QWidget \*const parent, Flags flags)

## Public Member Functions inherited from [Digikam::AbstractAlbumTreeView](#)

- [AbstractAlbumTreeView](#) (QWidget \*const parent, Flags flags)  
*Constructs an album tree view.*
- void [addContextMenuElement](#) ([ContextMenuElement](#) \*const [element](#))
- [AlbumFilterModel](#) \* [albumFilterModel](#) () const
- [AbstractSpecificAlbumModel](#) \* [albumModel](#) () const
- QList< [ContextMenuElement](#) \* > [contextMenuElements](#) () const
- template<class A >  
QList< A \* > [currentAlbums](#) ()
- bool [expandMatches](#) (const QModelIndex &index)  
*Ensures that every current match is visible by expanding all parent entries.*

- QModelIndex [indexVisuallyAt](#) (const QPoint &p)  
*This is a combination of [indexAt\(\)](#) checked with [visualRect\(\)](#).*
- void **removeContextMenuElement** ([ContextMenuElement](#) \*const element)
- QList< [Album](#) \* > **selectedItems** ()  
*[selectedItems\(\)](#) -*
- void [setAlbumManagerCurrentAlbum](#) (const bool setCurrentAlbum)  
*Some treeviews shall control the global current album kept by [AlbumManager](#).*
- void [setContextMenuIcon](#) (const QPixmap &pixmap)  
*Set the context menu title and icon.*
- void **setContextMenuTitle** (const QString &title)
- void [setEnabledContextMenu](#) (const bool enable)  
*Determines the global decision to show a popup menu or not.*
- void **setExpandNewCurrentItem** (const bool doThat)  
*Expand an item when making it the new current item.*
- void **setExpandOnSingleClick** (const bool doThat)  
*Enable expanding of tree items on single click on the item (default: off)*
- void [setSelectAlbumOnClick](#) (const bool selectOnClick)  
*Sets whether to select an album on click via the album manager or not.*
- void [setSelectOnContextMenu](#) (const bool select)  
*Sets whether to select the album under the mouse cursor on a context menu request (so that the album is shown using the album manager) or not.*
- bool **viewportEvent** (QEvent \*event) override  
*For internal use only.*

## Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (QObject \*const host)  
*Constructor.*
- virtual ~**StateSavingObject** ()  
*Destructor.*
- [StateSavingDepth](#) [getStateSavingDepth](#) () const  
*Returns the depth used for state saving or loading.*
- void **loadState** ()  
*Invokes loading the class' state.*
- void **saveState** ()  
*Invokes saving the class' state.*
- virtual void [setConfigGroup](#) (const KConfigGroup &group)  
*Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void [setEntryPrefix](#) (const QString &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void [setStateSavingDepth](#) (const [StateSavingDepth](#) depth)  
*Sets the depth used for state saving or loading.*

## Protected Member Functions

- void [addCustomContextMenuActions](#) ([ContextMenuHelper](#) &cmh, [Album](#) \*album) override  
*Hook method to add custom actions to the generated context menu.*

### Protected Member Functions inherited from [Digikam::TagFolderView](#)

- void [addCustomContextMenuActions](#) ([ContextMenuHelper](#) &cmh, [Album](#) \*album) override  
*Hook method to add custom actions to the generated context menu.*
- void [contextMenuEvent](#) ([QContextMenuEvent](#) \*event) override  
*Reimplement contextMenuEvent from AbstractAlbumTree to support multiple selection.*
- [QString](#) [contextMenuTitle](#) () const override  
*Hook method to implement that returns the title for the context menu.*
- void [handleCustomContextMenuAction](#) ([QAction](#) \*action, const [AlbumPointer](#)< [Album](#) > &album) override  
*Hook method to handle the custom context menu actions that were added with addCustomContextMenuActions.*
- void [keyPressEvent](#) ([QKeyEvent](#) \*event) override
- virtual void [setContextMenuItems](#) ([ContextMenuHelper](#) &cmh, const [QList](#)< [Album](#) \* > &albums)  
*Implementation of AddCustomContextMenuActions(see above) that handle multiple selection.*

### Protected Member Functions inherited from [Digikam::AbstractCheckableAlbumTreeView](#)

- void [middleButtonPressed](#) ([Album](#) \*a) override
- void [rowsInserted](#) (const [QModelIndex](#) &parent, int start, int end) override
- void [setAlbumFilterModel](#) ([CheckableAlbumFilterModel](#) \*const filterModel)
- void [setAlbumModel](#) ([AbstractCheckableAlbumModel](#) \*const model)

### Protected Member Functions inherited from [Digikam::AbstractCountingAlbumTreeView](#)

- void [rowsInserted](#) (const [QModelIndex](#) &parent, int start, int end) override
- void [setAlbumFilterModel](#) ([AlbumFilterModel](#) \*const filterModel)
- void [setAlbumModel](#) ([AbstractCountingAlbumModel](#) \*const model)

### Protected Member Functions inherited from [Digikam::AbstractAlbumTreeView](#)

- virtual [QPixmap](#) [contextMenuIcon](#) () const  
*Hook method that can be implemented to return a special icon used for the context menu.*
- void [dragEnterEvent](#) ([QDragEnterEvent](#) \*e) override
- void [dragLeaveEvent](#) ([QDragLeaveEvent](#) \*e) override
- void [dragMoveEvent](#) ([QDragMoveEvent](#) \*e) override
- void [dropEvent](#) ([QDropEvent](#) \*e) override
- void [mousePressEvent](#) ([QMouseEvent](#) \*e) override  
*Other helper methods.*
- virtual [QPixmap](#) [pixmapForDrag](#) (const [QStyleOptionViewItem](#) &option, [QList](#)< [QModelIndex](#) > indexes)  
*TODO: Move to delegate, when we have one.*
- void [rowsAboutToBeRemoved](#) (const [QModelIndex](#) &parent, int start, int end) override
- void [rowsInserted](#) (const [QModelIndex](#) &index, int start, int end) override
- void [setAlbumFilterModel](#) ([AlbumFilterModel](#) \*const filterModel)
- void [setAlbumModel](#) ([AbstractSpecificAlbumModel](#) \*const model)
- virtual bool [showContextMenuAt](#) ([QContextMenuEvent](#) \*event, [Album](#) \*albumForEvent)  
*Hook method to implement that determines if a context menu shall be displayed for the given event at the position coded in the event.*
- void [startDrag](#) ([Qt::DropActions](#) supportedActions) override

## Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- QString [entryName](#) (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup [getConfigGroup](#) () const  
*Returns the config group that must be used for state saving and loading.*

## Additional Inherited Members

## Protected Slots inherited from [Digikam::AbstractAlbumTreeView](#)

- void [albumSettingsChanged](#) ()
- void [slotCurrentChanged](#) ()
- virtual void [slotRootAlbumAvailable](#) ()  
*override if implemented behavior is not as intended*
- void [slotSearchTextSettingsAboutToChange](#) (bool searched, bool willSearch)
- void [slotSearchTextSettingsChanged](#) (bool wasSearching, bool searching)
- void [slotSelectionChanged](#) ()

## Protected Attributes inherited from [Digikam::TagTreeView](#)

- [TagPropertiesFilterModel](#) \* [m\\_filteredModel](#) = nullptr
- [TagModificationHelper](#) \* [m\\_modificationHelper](#) = nullptr

## Protected Attributes inherited from [Digikam::AbstractAlbumTreeView](#)

- [AlbumFilterModel](#) \* [m\\_albumFilterModel](#) = nullptr
- [AbstractSpecificAlbumModel](#) \* [m\\_albumModel](#) = nullptr
- bool [m\\_checkOnMiddleClick](#) = false
- [AlbumModelDragDropHandler](#) \* [m\\_dragDropHandler](#) = nullptr
- Flags [m\\_flags](#) = DefaultFlags
- int [m\\_lastScrollBarValue](#) = 0
- bool [m\\_restoreCheckState](#) = false

## 9.1264.1 Member Function Documentation

### 9.1264.1.1 [addCustomContextMenuActions\(\)](#)

```
void Digikam::TagCheckView::addCustomContextMenuActions (
    ContextMenuHelper & cmh,
    Album * album ) [override], [protected], [virtual]
```

#### Parameters

<i>cmh</i>	helper object to create the context menu
<i>album</i>	tag on which the context menu will be created. May be null if it is requested on no tag entry

Reimplemented from [Digikam::AbstractAlbumTreeView](#).

Reimplemented in [Digikam::TagFilterView](#).

### 9.1264.1.2 checkedTagsChanged

```
void Digikam::TagCheckView::checkedTagsChanged (
    const QList< TAlbum * > & includedTags,
    const QList< TAlbum * > & excludedTags ) [signal]
```

#### Parameters

<i>includedTags</i>	a list of selected tag ids processed.
<i>excludedTags</i>	a list of tag ids ignored.

### 9.1264.1.3 doLoadState()

```
void Digikam::TagCheckView::doLoadState ( ) [override], [virtual]
```

Therefore the config is first parsed into `d->statesByAlbumId` which holds the state of all tree view entries indexed by album id. Afterwards an initial sync run is done restoring the state of all model entries that are already present at this time. Every processed entry is removed from `d->statesByAlbumId`. If there are still entries left in this map we assume that the model is not fully loaded at the moment. Therefore the `rowsInserted` signal is connected to a slot that restores the state of new rows based on the remaining entries in `d->statesByAlbumId`.

Reimplemented from [Digikam::AbstractCheckableAlbumTreeView](#).

### 9.1264.1.4 doSaveState()

```
void Digikam::TagCheckView::doSaveState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Reimplemented from [Digikam::AbstractCheckableAlbumTreeView](#).

### 9.1264.1.5 setCheckNewTags()

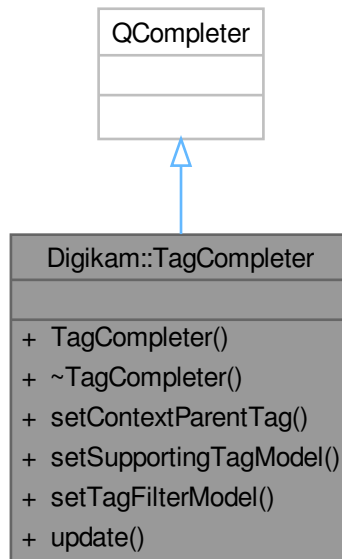
```
void Digikam::TagCheckView::setCheckNewTags (
    bool checkNewTags )
```

#### Parameters

<i>checkNewTags</i>	is switched on, a tag that is created from <i>within</i> this view, typically via the context menu, will automatically be set checked.
---------------------	--

## 9.1265 Digikam::TagCompleter Class Reference

Inheritance diagram for Digikam::TagCompleter:



### Signals

- void **signalActivated** (const [TaggingAction](#) &action)
- void **signalHighlighted** (const [TaggingAction](#) &action)

### Public Member Functions

- **TagCompleter** (QObject \*const parent=nullptr)  
*A completion object operating on a [TagModel](#).*
- void **setContextParentTag** (int parentTagId)  
*Set a parent tag which may by the user be considered as a parent for a new tag during completion.*
- void **setSupportingTagModel** ([TagModel](#) \*const supportingModel)  
*Set a supporting model from which the completer may get data for its display.*
- void **setTagFilterModel** ([AlbumFilterModel](#) \*const supportingModel)
- void **update** (const QString &fragment)  
*Update the completer for the given fragment.*

### 9.1265.1 Member Function Documentation

#### 9.1265.1.1 setSupportingTagModel()

```
void Digikam::TagCompleter::setSupportingTagModel (
    TagModel *const supportingModel )
```

Optional.

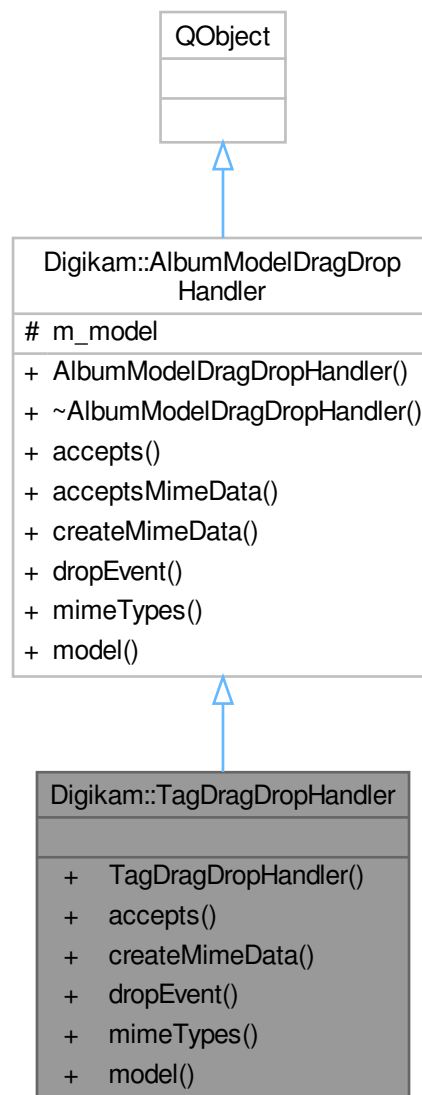
## 9.1266 Digikam::TagData Struct Reference

### Public Attributes

- QString **tagName**
- Type **tagType** = TypeChild
- QString **tipName**

## 9.1267 Digikam::TagDragDropHandler Class Reference

Inheritance diagram for Digikam::TagDragDropHandler:





## Signals

- void **assignTags** (const QList< qlonglong > &imageIDs, const QList< int > &tagIDs)

## Public Member Functions

- **TagDragDropHandler** ([TagModel](#) \*const model)
- Qt::DropAction **accepts** (const QDropEvent \*e, const QModelIndex &dropIndex) override  
*Returns if the given mime data is accepted for drop on dropIndex.*
- QMimeData \* **createMimeData** (const QList< [Album](#) \* > &) override  
*Create a mime data object for starting a drag from the given Albums.*
- bool **dropEvent** (QAbstractItemView \*view, const QDropEvent \*e, const QModelIndex &droppedOn) override  
*Gives the view and the occurring drop event.*
- QStringList **mimeTypes** () const override  
*Returns the supported mime types.*
- [TagModel](#) \* **model** () const

## Public Member Functions inherited from [Digikam::AlbumModelDragDropHandler](#)

- **AlbumModelDragDropHandler** ([AbstractAlbumModel](#) \*model)
- virtual bool **acceptsMimeData** (const QMimeData \*data)  
*Returns if the given mime data can be handled.*
- [AbstractAlbumModel](#) \* **model** () const

## Additional Inherited Members

## Protected Attributes inherited from [Digikam::AlbumModelDragDropHandler](#)

- [AbstractAlbumModel](#) \* **m\_model** = nullptr

## 9.1267.1 Member Function Documentation

### 9.1267.1.1 accepts()

```
Qt::DropAction Digikam::TagDragDropHandler::accepts (
    const QDropEvent * e,
    const QModelIndex & dropIndex ) [override], [virtual]
```

Returns the proposed action, or Qt::IgnoreAction if not accepted.

Reimplemented from [Digikam::AlbumModelDragDropHandler](#).

### 9.1267.1.2 createMimeData()

```
QMimeData * Digikam::TagDragDropHandler::createMimeData (
    const QList< Album * > & ) [override], [virtual]
```

Reimplemented from [Digikam::AlbumModelDragDropHandler](#).

### 9.1267.1.3 dropEvent()

```
bool Digikam::TagDragDropHandler::dropEvent (
    QAbstractItemView * view,
    const QDropEvent * e,
    const QModelIndex & droppedOn ) [override], [virtual]
```

The index is the index where the drop was dropped on. It may be invalid (dropped on decoration, viewport) Returns true if the event is to be accepted.

Reimplemented from [Digikam::AlbumModelDragDropHandler](#).

### 9.1267.1.4 mimeTypees()

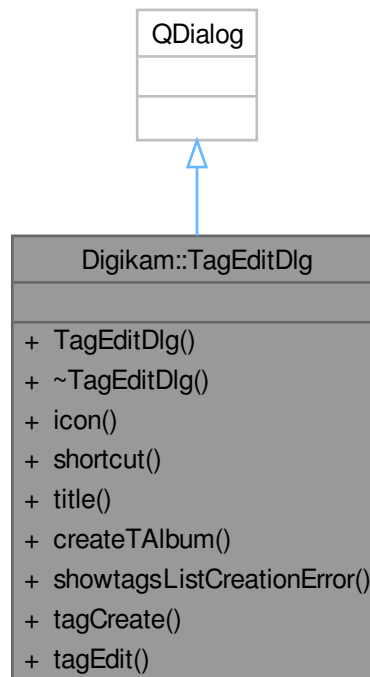
```
QStringList Digikam::TagDragDropHandler::mimeTypees ( ) const [override], [virtual]
```

Called by the default implementation of model's [mimeTypees\(\)](#).

Reimplemented from [Digikam::AlbumModelDragDropHandler](#).

## 9.1268 Digikam::TagEditDlg Class Reference

Inheritance diagram for Digikam::TagEditDlg:



## Public Member Functions

- **TagEditDlg** (QWidget \*const parent, [TAlbum](#) \*const album, bool create=false)
- QString **icon** () const
- QKeySequence **shortcut** () const
- QString **title** () const

## Static Public Member Functions

- static AlbumList **createTAlbum** ([TAlbum](#) \*const mainRootAlbum, const QString &tagStr, const QString &icon, const QKeySequence &ks, QMap< QString, QString > &errMap)  
*Create a list of new Tag album using a list of tags hierarchies separated by ",".*
- static void **showtagsListCreationError** (QWidget \*const parent, const QMap< QString, QString > &errMap)
- static bool **tagCreate** (QWidget \*const parent, [TAlbum](#) \*const album, QString &title, QString &icon, QKeySequence &ks)
- static bool **tagEdit** (QWidget \*const parent, [TAlbum](#) \*const album, QString &title, QString &icon, QKeySequence &ks)

## 9.1268.1 Member Function Documentation

### 9.1268.1.1 createTAlbum()

```
AlbumList Digikam::TagEditDlg::createTAlbum (
    TAlbum *const mainRootAlbum,
    const QString & tagStr,
    const QString & icon,
    const QKeySequence & ks,
    QMap< QString, QString > & errMap ) [static]
```

A hierarchy of tags is a string path of tags name separated by "/". If a hierarchy start by "/" or if mainRootAlbum is null, it will be created from root tag album, else it will be created from mainRootAlbum as parent album. 'errMap' is Map of [TAlbum](#) path and error message if tag creation failed. Return the list of created Albums.

## 9.1269 Digikam::TagFilterView Class Reference

A view to filter the currently displayed album by tags.



## Public Member Functions inherited from Digikam::TagCheckView

- **TagCheckView** (QWidget \*const parent, TagModel \*const tagModel)
- bool **checkNewTags** () const
- void **doLoadState** () override
 

*Implements state loading for the album tree view in a somewhat clumsy procedure because the model may not be fully loaded when this method is called.*
- void **doSaveState** () override
 

*Implement this hook method for state saving.*
- QList< TAlbum \* > **getCheckedTags** () const
- QList< TAlbum \* > **getPartiallyCheckedTags** () const
- ToggleAutoTags **getToggleAutoTags** () const
- void **setCheckNewTags** (bool checkNewTags)
- *If.*
- void **setToggleAutoTags** (ToggleAutoTags toggle)

## Public Member Functions inherited from Digikam::TagFolderView

- **TagFolderView** (QWidget \*const parent, TagModel \*const model)
 

*Constructor.*
- ~**TagFolderView** () override
 

*Destructor.*
- void **setShowDeleteFaceTagsAction** (bool show)
 

*Define whether to show the "Delete People Tags" action in context menus or not.*
- void **setShowFindDuplicateAction** (bool show)
 

*Define whether to show the "find duplicate" action in context menus or not.*
- void **tagPropsEdit** ()
 

*Open tag for editing.*

## Public Member Functions inherited from Digikam::TagTreeView

- **TagTreeView** (QWidget \*const parent=nullptr, Flags flags=DefaultFlags)
- TAlbum \* **albumForIndex** (const QModelIndex &index) const
- TagModel \* **albumModel** () const
- TAlbum \* **currentAlbum** () const
 

*currentAlbum Even if multiple selection is enabled current Album can be only one, the last clicked item if you need selected items, see selectedAlbums() It's NOT the same as AlbumManager::currentAlbums()*
- TagPropertiesFilterModel \* **filteredModel** () const
 

*Contains only the tags filtered by properties - prefer to albumModel()*
- QList< TAlbum \* > **selectedTagAlbums** ()
- QList< Album \* > **selectedTags** ()
 

*selectedTags - return a list of all selected items in tag model*
- void **setAlbumFilterModel** (TagPropertiesFilterModel \*const filteredModel, CheckableAlbumFilterModel \*const filterModel)
- void **setAlbumModel** (TagModel \*const model)
- TagModificationHelper \* **tagModificationHelper** () const

## Public Member Functions inherited from [Digikam::AbstractCheckableAlbumTreeView](#)

- [AbstractCheckableAlbumTreeView](#) (QWidget \*const parent, Flags flags)
  - Models of these view can be checkable, they need not.*
- [CheckableAlbumFilterModel](#) \* [albumFilterModel](#) () const
- [AbstractCheckableAlbumModel](#) \* [albumModel](#) () const
  - Manage check state through the model directly.*
- [CheckableAlbumFilterModel](#) \* [checkableAlbumFilterModel](#) () const
- [AbstractCheckableAlbumModel](#) \* [checkableModel](#) () const
- bool [isRestoreCheckState](#) () const
  - Tells if the check state is restored while loading / saving state.*
- void [setCheckOnMiddleClick](#) (bool doThat)
  - Enable checking on middle mouse button click (default: on).*
- void [setRestoreCheckState](#) (bool restore)
  - Set whether to restore check state or not.*

## Public Member Functions inherited from [Digikam::AbstractCountingAlbumTreeView](#)

- [AbstractCountingAlbumTreeView](#) (QWidget \*const parent, Flags flags)

## Public Member Functions inherited from [Digikam::AbstractAlbumTreeView](#)

- [AbstractAlbumTreeView](#) (QWidget \*const parent, Flags flags)
  - Constructs an album tree view.*
- void [addContextMenuElement](#) ([ContextMenuElement](#) \*const element)
- [AlbumFilterModel](#) \* [albumFilterModel](#) () const
- [AbstractSpecificAlbumModel](#) \* [albumModel](#) () const
- QList< [ContextMenuElement](#) \* > [contextMenuElements](#) () const
- template<class A >
  - QList< A \* > [currentAlbums](#) ()
- bool [expandMatches](#) (const QModelIndex &index)
  - Ensures that every current match is visible by expanding all parent entries.*
- QModelIndex [indexVisuallyAt](#) (const QPoint &p)
  - This is a combination of [indexAt\(\)](#) checked with [visualRect\(\)](#).*
- void [removeContextMenuElement](#) ([ContextMenuElement](#) \*const element)
- QList< [Album](#) \* > [selectedItems](#) ()
  - [selectedItems\(\)](#) -*
- void [setAlbumManagerCurrentAlbum](#) (const bool setCurrentAlbum)
  - Some treeviews shall control the global current album kept by [AlbumManager](#).*
- void [setContextMenuIcon](#) (const QPixmap &pixmap)
  - Set the context menu title and icon.*
- void [setContextMenuTitle](#) (const QString &title)
- void [setEnabledContextMenu](#) (const bool enable)
  - Determines the global decision to show a popup menu or not.*
- void [setExpandNewCurrentItem](#) (const bool doThat)
  - Expand an item when making it the new current item.*
- void [setExpandOnSingleClick](#) (const bool doThat)
  - Enable expanding of tree items on single click on the item (default: off)*
- void [setSelectAlbumOnClick](#) (const bool selectOnClick)
  - Sets whether to select an album on click via the album manager or not.*
- void [setSelectOnContextMenu](#) (const bool select)
  - Sets whether to select the album under the mouse cursor on a context menu request (so that the album is shown using the album manager) or not.*
- bool [viewportEvent](#) (QEvent \*event) override
  - For internal use only.*

## Public Member Functions inherited from Digikam::StateSavingObject

- [StateSavingObject](#) (QObject \*const host)
  - Constructor.*
- virtual `~StateSavingObject ()`
  - Destructor.*
- [StateSavingDepth](#) `getStateSavingDepth ()` const
  - Returns the depth used for state saving or loading.*
- void `loadState ()`
  - Invokes loading the class' state.*
- void `saveState ()`
  - Invokes saving the class' state.*
- virtual void `setConfigGroup` (const KConfigGroup &group)
  - Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void `setEntryPrefix` (const QString &prefix)
  - Define a prefix that will be used for every entry in the config group.*
- void `setStateSavingDepth` (const [StateSavingDepth](#) depth)
  - Sets the depth used for state saving or loading.*

## Protected Member Functions

- void `addCustomContextMenuActions` ([ContextMenuHelper](#) &cmh, [Album](#) \*album) override
  - Hook method to add custom actions to the generated context menu.*
- void `handleCustomContextMenuAction` (QAction \*action, const [AlbumPointer](#)< [Album](#) > &album) override
  - Hook method to handle the custom context menu actions that were added with addCustomContextMenuActions.*

## Protected Member Functions inherited from Digikam::TagFolderView

- void `addCustomContextMenuActions` ([ContextMenuHelper](#) &cmh, [Album](#) \*album) override
  - Hook method to add custom actions to the generated context menu.*
- void `contextMenuEvent` (QContextMenuEvent \*event) override
  - Reimplement contextMenuEvent from AbstractAlbumTree to support multiple selection.*
- QString `contextMenuTitle ()` const override
  - Hook method to implement that returns the title for the context menu.*
- void `handleCustomContextMenuAction` (QAction \*action, const [AlbumPointer](#)< [Album](#) > &album) override
  - Hook method to handle the custom context menu actions that were added with addCustomContextMenuActions.*
- void `keyPressEvent` (QKeyEvent \*event) override
- virtual void `setContextMenuItems` ([ContextMenuHelper](#) &cmh, const QList< [TAlbum](#) \* > &albums)
  - Implementation of AddCustomContextMenuActions(see above) that handle multiple selection.*

## Protected Member Functions inherited from Digikam::AbstractCheckableAlbumTreeView

- void `middleButtonPressed` ([Album](#) \*a) override
- void `rowsInserted` (const QModelIndex &parent, int start, int end) override
- void `setAlbumFilterModel` ([CheckableAlbumFilterModel](#) \*const filterModel)
- void `setAlbumModel` ([AbstractCheckableAlbumModel](#) \*const model)

## Protected Member Functions inherited from [Digikam::AbstractCountingAlbumTreeView](#)

- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- void **setAlbumFilterModel** ([AlbumFilterModel](#) \*const filterModel)
- void **setAlbumModel** ([AbstractCountingAlbumModel](#) \*const model)

## Protected Member Functions inherited from [Digikam::AbstractAlbumTreeView](#)

- virtual QPixmap **contextMenuIcon** () const  
*Hook method that can be implemented to return a special icon used for the context menu.*
- void **dragEnterEvent** (QDragEnterEvent \*e) override
- void **dragLeaveEvent** (QDragLeaveEvent \*e) override
- void **dragMoveEvent** (QDragMoveEvent \*e) override
- void **dropEvent** (QDropEvent \*e) override
- void **mousePressEvent** (QMouseEvent \*e) override  
*Other helper methods.*
- virtual QPixmap  **pixmapForDrag** (const QStyleOptionViewItem &option, QList< QModelIndex > indexes)  
*TODO: Move to delegate, when we have one.*
- void **rowsAboutToBeRemoved** (const QModelIndex &parent, int start, int end) override
- void **rowsInserted** (const QModelIndex &index, int start, int end) override
- void **setAlbumFilterModel** ([AlbumFilterModel](#) \*const filterModel)
- void **setAlbumModel** ([AbstractSpecificAlbumModel](#) \*const model)
- virtual bool **showContextMenuAt** (QContextMenuEvent \*event, [Album](#) \*albumForEvent)  
*Hook method to implement that determines if a context menu shall be displayed for the given event at the position coded in the event.*
- void **startDrag** (Qt::DropActions supportedActions) override

## Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- QString **entryName** (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup **getConfigGroup** () const  
*Returns the config group that must be used for state saving and loading.*

## Additional Inherited Members

## Public Types inherited from [Digikam::TagCheckView](#)

- enum **ToggleAutoTags** { **NoToggleAuto** = 0 , **Children** , **Parents** , **ChildrenAndParents** }

## Public Types inherited from [Digikam::AbstractAlbumTreeView](#)

- enum **Flag** {  
[CreateDefaultFilterModel](#) , [CreateDefaultFilterModel](#) , [CreateDefaultDelegate](#) , [ShowCountAccordingToSettings](#) ,  
[AlwaysShowInclusiveCounts](#) , **DefaultFlags** = [CreateDefaultFilterModel](#) | [CreateDefaultDelegate](#) | [Show↔](#)  
[CountAccordingToSettings](#) }
- typedef QFlags< [Flag](#) > **Flags**



## Public Types inherited from Digikam::StateSavingObject

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }

This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.

## Public Slots inherited from Digikam::TagCheckView

- void [slotResetCheckState](#) ()

Resets the whole tag filter.

## Public Slots inherited from Digikam::TagFolderView

- void [slotTagNewFromABCMenu](#) (const QString &personName)

## Public Slots inherited from Digikam::TagTreeView

- void [setCurrentAlbum](#) (int tagId, bool selectInAlbumManager=true)
- void [setCurrentAlbums](#) (const QList< Album \* > &tags, bool selectInAlbumManager=true)

## Public Slots inherited from Digikam::AbstractAlbumTreeView

- void [adaptColumnsToContent](#) ()  
*Adapt the column sizes to the contents of the tree view.*
- void [expandEverything](#) (const QModelIndex &index)  
*Expands the complete tree under the given index.*
- void [scrollToSelectedAlbum](#) ()  
*Scrolls to the first selected album if there is one.*
- void [setCurrentAlbums](#) (const QList< Album \* > &albums, bool selectInAlbumManager=true)  
*Selects the given album.*
- void [setSearchTextSettings](#) (const [SearchTextSettings](#) &settings)
- void [slotCollapseAllNodes](#) ()  
*slotCollapseAllNodes - collapse all nodes without root node*
- void [slotCollapseNode](#) ()  
*slotCollapseNode - collapse recursively selected nodes*
- void [slotExpandNode](#) ()  
*slotExpandNode - expands recursively selected nodes*

## Signals inherited from Digikam::TagCheckView

- void [checkedTagsChanged](#) (const QList< TAlbum \* > &includedTags, const QList< TAlbum \* > &excludedTags)  
*Emitted if the checked tags have changed.*

## Signals inherited from Digikam::TagFolderView

- void [signalFindDuplicates](#) (const QList< TAlbum \* > &albums)

## Signals inherited from [Digikam::TagTreeView](#)

- void **assignTags** (int tagId, const QList< int > &imageIds)

## Signals inherited from [Digikam::AbstractAlbumTreeView](#)

- void **currentAlbumChanged** ([Album](#) \*currentAlbum)  
*Emitted when the currently selected album changes.*
- void **selectedAlbumsChanged** (const QList< [Album](#) \* > &selectedAlbums)  
*Emitted when the current selection changes.*

## Protected Slots inherited from [Digikam::AbstractAlbumTreeView](#)

- void **albumSettingsChanged** ()
- void **slotCurrentChanged** ()
- virtual void **slotRootAlbumAvailable** ()  
*override if implemented behavior is not as intended*
- void **slotSearchTextSettingsAboutToChange** (bool searched, bool willSearch)
- void **slotSearchTextSettingsChanged** (bool wasSearching, bool searching)
- void **slotSelectionChanged** ()

## Protected Attributes inherited from [Digikam::TagTreeView](#)

- [TagPropertiesFilterModel](#) \* **m\_filteredModel** = nullptr
- [TagModificationHelper](#) \* **m\_modificationHelper** = nullptr

## Protected Attributes inherited from [Digikam::AbstractAlbumTreeView](#)

- [AlbumFilterModel](#) \* **m\_albumFilterModel** = nullptr
- [AbstractSpecificAlbumModel](#) \* **m\_albumModel** = nullptr
- bool **m\_checkOnMiddleClick** = false
- [AlbumModelDragDropHandler](#) \* **m\_dragDropHandler** = nullptr
- Flags **m\_flags** = DefaultFlags
- int **m\_lastScrollBarValue** = 0
- bool **m\_restoreCheckState** = false

### 9.1269.1 Detailed Description

Author

jwienke

### 9.1269.2 Constructor & Destructor Documentation

#### 9.1269.2.1 TagFilterView()

```
Digikam::TagFilterView::TagFilterView (
    QWidget *const parent,
    TagModel *const tagFilterModel ) [explicit]
```

## Parameters

<i>parent</i>	the parent for qt parent child mechanism
<i>tagFilterModel</i>	tag model to work on

**9.1269.3 Member Function Documentation****9.1269.3.1 addCustomContextMenuActions()**

```
void Digikam::TagFilterView::addCustomContextMenuActions (
    ContextMenuHelper & cmh,
    Album * album ) [override], [protected], [virtual]
```

## Parameters

<i>cmh</i>	helper object to create the context menu
<i>album</i>	tag on which the context menu will be created. May be null if it is requested on no tag entry

Reimplemented from [Digikam::TagCheckView](#).

**9.1269.3.2 handleCustomContextMenuAction()**

```
void Digikam::TagFilterView::handleCustomContextMenuAction (
    QAction * action,
    const AlbumPointer< Album > & album ) [override], [protected], [virtual]
```

## Parameters

<i>action</i>	the action that was chosen by the user, may be null if none of the custom actions were selected
<i>album</i>	the tag on which the context menu was requested. May be null if there was no

Reimplemented from [Digikam::AbstractAlbumTreeView](#).



## Public Slots inherited from [Digikam::TagTreeView](#)

- void **setCurrentAlbum** (int tagId, bool selectInAlbumManager=true)
- void **setCurrentAlbums** (const QList< Album \* > &tags, bool selectInAlbumManager=true)

## Public Slots inherited from [Digikam::AbstractAlbumTreeView](#)

- void **adaptColumnsToContent** ()  
*Adapt the column sizes to the contents of the tree view.*
- void **expandEverything** (const QModelIndex &index)  
*Expands the complete tree under the given index.*
- void **scrollToSelectedAlbum** ()  
*Scrolls to the first selected album if there is one.*
- void **setCurrentAlbums** (const QList< Album \* > &albums, bool selectInAlbumManager=true)  
*Selects the given album.*
- void **setSearchTextSettings** (const SearchTextSettings &settings)
- void **slotCollapseAllNodes** ()  
*slotCollapseAllNodes - collapse all nodes without root node*
- void **slotCollapseNode** ()  
*slotCollapseNode - collapse recursively selected nodes*
- void **slotExpandNode** ()  
*slotExpandNode - expands recursively selected nodes*

## Signals

- void **signalFindDuplicates** (const QList< TAlbum \* > &albums)

## Signals inherited from [Digikam::TagTreeView](#)

- void **assignTags** (int tagId, const QList< int > &imageIds)

## Signals inherited from [Digikam::AbstractAlbumTreeView](#)

- void **currentAlbumChanged** (Album \*currentAlbum)  
*Emitted when the currently selected album changes.*
- void **selectedAlbumsChanged** (const QList< Album \* > &selectedAlbums)  
*Emitted when the current selection changes.*

## Public Member Functions

- [TagFolderView](#) (QWidget \*const parent, [TagModel](#) \*const model)  
*Constructor.*
- [~TagFolderView](#) () override  
*Destructor.*
- void **setShowDeleteFaceTagsAction** (bool show)  
*Define whether to show the "Delete People Tags" action in context menus or not.*
- void **setShowFindDuplicateAction** (bool show)  
*Define whether to show the "find duplicate" action in context menus or not.*
- void **tagPropsEdit** ()  
*Open tag for editing.*

## Public Member Functions inherited from [Digikam::TagTreeView](#)

- [TagTreeView](#) (QWidget \*const parent=nullptr, Flags flags=DefaultFlags)
- [TAlbum](#) \* [albumForIndex](#) (const QModelIndex &index) const
- [TagModel](#) \* [albumModel](#) () const
- [TAlbum](#) \* [currentAlbum](#) () const
 

*currentAlbum Even if multiple selection is enabled current Album can be only one, the last clicked item if you need selected items, see [selectedAlbums\(\)](#) It's NOT the same as [AlbumManager::currentAlbums\(\)](#)*
- [TagPropertiesFilterModel](#) \* [filteredModel](#) () const
 

*Contains only the tags filtered by properties - prefer to albumModel()*
- QList< [TAlbum](#) \* > [selectedTagAlbums](#) ()
- QList< [Album](#) \* > [selectedTags](#) ()
 

*selectedTags - return a list of all selected items in tag model*
- void [setAlbumFilterModel](#) ([TagPropertiesFilterModel](#) \*const [filteredModel](#), [CheckableAlbumFilterModel](#) \*const [filterModel](#))
- void [setAlbumModel](#) ([TagModel](#) \*const [model](#))
- [TagModificationHelper](#) \* [tagModificationHelper](#) () const

## Public Member Functions inherited from [Digikam::AbstractCheckableAlbumTreeView](#)

- [AbstractCheckableAlbumTreeView](#) (QWidget \*const parent, Flags flags)
 

*Models of these view can be checkable, they need not.*
- [CheckableAlbumFilterModel](#) \* [albumFilterModel](#) () const
- [AbstractCheckableAlbumModel](#) \* [albumModel](#) () const
 

*Manage check state through the model directly.*
- [CheckableAlbumFilterModel](#) \* [checkableAlbumFilterModel](#) () const
- [AbstractCheckableAlbumModel](#) \* [checkableModel](#) () const
- void [doLoadState](#) () override
 

*Implements state loading for the album tree view in a somewhat clumsy procedure because the model may not be fully loaded when this method is called.*
- void [doSaveState](#) () override
 

*Implement this hook method for state saving.*
- bool [isRestoreCheckState](#) () const
 

*Tells if the check state is restored while loading / saving state.*
- void [setCheckOnMiddleClick](#) (bool [doThat](#))
 

*Enable checking on middle mouse button click (default: on).*
- void [setRestoreCheckState](#) (bool [restore](#))
 

*Set whether to restore check state or not.*

## Public Member Functions inherited from [Digikam::AbstractCountingAlbumTreeView](#)

- [AbstractCountingAlbumTreeView](#) (QWidget \*const parent, Flags flags)

## Public Member Functions inherited from Digikam::AbstractAlbumTreeView

- [AbstractAlbumTreeView](#) (QWidget \*const parent, Flags flags)
  - Constructs an album tree view.*
- void **addContextMenuElement** ([ContextMenuElement](#) \*const element)
- [AlbumFilterModel](#) \* **albumFilterModel** () const
- [AbstractSpecificAlbumModel](#) \* **albumModel** () const
- QList< [ContextMenuElement](#) \* > **contextMenuElements** () const
- template<class A >
  - QList< A \* > **currentAlbums** ()
- bool **expandMatches** (const QModelIndex &index)
  - Ensures that every current match is visible by expanding all parent entries.*
- QModelIndex **indexVisuallyAt** (const QPoint &p)
  - This is a combination of indexAt() checked with visualRect().*
- void **removeContextMenuElement** ([ContextMenuElement](#) \*const element)
- QList< [Album](#) \* > **selectedItems** ()
  - selectedItems() -*
- void **setAlbumManagerCurrentAlbum** (const bool setCurrentAlbum)
  - Some treeviews shall control the global current album kept by [AlbumManager](#).*
- void **setContextMenuIcon** (const QPixmap &pixmap)
  - Set the context menu title and icon.*
- void **setContextMenuTitle** (const QString &title)
- void **setEnabledContextMenu** (const bool enable)
  - Determines the global decision to show a popup menu or not.*
- void **setExpandNewCurrentItem** (const bool doThat)
  - Expand an item when making it the new current item.*
- void **setExpandOnSingleClick** (const bool doThat)
  - Enable expanding of tree items on single click on the item (default: off)*
- void **setSelectAlbumOnClick** (const bool selectOnClick)
  - Sets whether to select an album on click via the album manager or not.*
- void **setSelectOnContextMenu** (const bool select)
  - Sets whether to select the album under the mouse cursor on a context menu request (so that the album is shown using the album manager) or not.*
- bool **viewportEvent** (QEvent \*event) override
  - For internal use only.*

## Public Member Functions inherited from Digikam::StateSavingObject

- [StateSavingObject](#) (QObject \*const host)
  - Constructor.*
- virtual ~**StateSavingObject** ()
  - Destructor.*
- [StateSavingDepth](#) **getStateSavingDepth** () const
  - Returns the depth used for state saving or loading.*
- void **loadState** ()
  - Invokes loading the class' state.*
- void **saveState** ()
  - Invokes saving the class' state.*
- virtual void **setConfigGroup** (const KConfigGroup &group)
  - Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void **setEntryPrefix** (const QString &prefix)
  - Define a prefix that will be used for every entry in the config group.*
- void **setStateSavingDepth** (const [StateSavingDepth](#) depth)
  - Sets the depth used for state saving or loading.*

### Protected Member Functions

- void [addCustomContextMenuActions](#) ([ContextMenuHelper](#) &cmh, [Album](#) \*album) override  
*Hook method to add custom actions to the generated context menu.*
- void [contextMenuEvent](#) ([QContextMenuEvent](#) \*event) override  
*Reimplement contextMenuEvent from AbstractAlbumTree to support multiple selection.*
- [QString](#) [contextMenuTitle](#) () const override  
*Hook method to implement that returns the title for the context menu.*
- void [handleCustomContextMenuAction](#) ([QAction](#) \*action, const [AlbumPointer](#)< [Album](#) > &album) override  
*Hook method to handle the custom context menu actions that were added with addCustomContextMenuActions.*
- void [keyPressEvent](#) ([QKeyEvent](#) \*event) override
- virtual void [setContextMenuItems](#) ([ContextMenuHelper](#) &cmh, const [QList](#)< [TAlbum](#) \* > &albums)  
*Implementation of AddCustomContextMenuActions(see above) that handle multiple selection.*

### Protected Member Functions inherited from [Digikam::AbstractCheckableAlbumTreeView](#)

- void [middleButtonPressed](#) ([Album](#) \*a) override
- void [rowsInserted](#) (const [QModelIndex](#) &parent, int start, int end) override
- void [setAlbumFilterModel](#) ([CheckableAlbumFilterModel](#) \*const filterModel)
- void [setAlbumModel](#) ([AbstractCheckableAlbumModel](#) \*const model)

### Protected Member Functions inherited from [Digikam::AbstractCountingAlbumTreeView](#)

- void [rowsInserted](#) (const [QModelIndex](#) &parent, int start, int end) override
- void [setAlbumFilterModel](#) ([AlbumFilterModel](#) \*const filterModel)
- void [setAlbumModel](#) ([AbstractCountingAlbumModel](#) \*const model)

### Protected Member Functions inherited from [Digikam::AbstractAlbumTreeView](#)

- virtual [QPixmap](#) [contextMenuIcon](#) () const  
*Hook method that can be implemented to return a special icon used for the context menu.*
- void [dragEnterEvent](#) ([QDragEnterEvent](#) \*e) override
- void [dragLeaveEvent](#) ([QDragLeaveEvent](#) \*e) override
- void [dragMoveEvent](#) ([QDragMoveEvent](#) \*e) override
- void [dropEvent](#) ([QDropEvent](#) \*e) override
- void [mousePressEvent](#) ([QMouseEvent](#) \*e) override  
*Other helper methods.*
- virtual [QPixmap](#) [pixmapForDrag](#) (const [QStyleOptionViewItem](#) &option, [QList](#)< [QModelIndex](#) > indexes)  
*TODO: Move to delegate, when we have one.*
- void [rowsAboutToBeRemoved](#) (const [QModelIndex](#) &parent, int start, int end) override
- void [rowsInserted](#) (const [QModelIndex](#) &index, int start, int end) override
- void [setAlbumFilterModel](#) ([AlbumFilterModel](#) \*const filterModel)
- void [setAlbumModel](#) ([AbstractSpecificAlbumModel](#) \*const model)
- virtual bool [showContextMenuAt](#) ([QContextMenuEvent](#) \*event, [Album](#) \*albumForEvent)  
*Hook method to implement that determines if a context menu shall be displayed for the given event at the position coded in the event.*
- void [startDrag](#) ([Qt::DropActions](#) supportedActions) override



## Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- `QString` `entryName` (const `QString` &base) const  
*Always use this method to create config group entry names.*
- `KConfigGroup` `getConfigGroup` () const  
*Returns the config group that must be used for state saving and loading.*

## Additional Inherited Members

## Public Types inherited from [Digikam::AbstractAlbumTreeView](#)

- enum `Flag` {  
`CreateDefaultModel`, `CreateDefaultFilterModel`, `CreateDefaultDelegate`, `ShowCountAccordingToSettings`,  
`AlwaysShowInclusiveCounts`, `DefaultFlags` = `CreateDefaultFilterModel` | `CreateDefaultDelegate` | `ShowCountAccordingToSettings` }
- typedef `QFlags`< `Flag` > `Flags`

## Public Types inherited from [Digikam::StateSavingObject](#)

- enum `StateSavingDepth` { `INSTANCE`, `DIRECT_CHILDREN`, `RECURSIVE` }  
*This enum defines the "depth" of the `StateSavingObject::loadState()` and `StateSavingObject::saveState()` methods.*

## Protected Slots inherited from [Digikam::AbstractAlbumTreeView](#)

- void `albumSettingsChanged` ()
- void `slotCurrentChanged` ()
- virtual void `slotRootAlbumAvailable` ()  
*override if implemented behavior is not as intended*
- void `slotSearchTextSettingsAboutToChange` (bool searched, bool willSearch)
- void `slotSearchTextSettingsChanged` (bool wasSearching, bool searching)
- void `slotSelectionChanged` ()

## Protected Attributes inherited from [Digikam::TagTreeView](#)

- `TagPropertiesFilterModel` \* `m_filteredModel` = nullptr
- `TagModificationHelper` \* `m_modificationHelper` = nullptr

## Protected Attributes inherited from [Digikam::AbstractAlbumTreeView](#)

- `AlbumFilterModel` \* `m_albumFilterModel` = nullptr
- `AbstractSpecificAlbumModel` \* `m_albumModel` = nullptr
- bool `m_checkOnMiddleClick` = false
- `AlbumModelDragDropHandler` \* `m_dragDropHandler` = nullptr
- `Flags` `m_flags` = `DefaultFlags`
- int `m_lastScrollBarValue` = 0
- bool `m_restoreCheckState` = false

## 9.1270.1 Constructor & Destructor Documentation

### 9.1270.1.1 TagFolderView()

```
Digikam::TagFolderView::TagFolderView (
    QWidget *const parent,
    TagModel *const model )
```

## Parameters

<i>parent</i>	the parent for Qt's parent child mechanism
<i>model</i>	tag model to display

This ensures that the View appears sorted

## 9.1270.2 Member Function Documentation

### 9.1270.2.1 addCustomContextMenuActions()

```
void Digikam::TagFolderView::addCustomContextMenuActions (
    ContextMenuHelper & cmh,
    Album * album ) [override], [protected], [virtual]
```

The default implementation adds actions to reset the tag icon and to find duplicates in a tag album. If you want to use these actions, remember to call this class' implementation of this method and the handleCustomContextMenuAction in your derived class.

## Parameters

<i>cmh</i>	helper object to create the context menu
<i>album</i>	tag on which the context menu will be created. May be null if it is requested on no tag entry

Reimplemented from [Digikam::AbstractAlbumTreeView](#).

### 9.1270.2.2 contextMenuEvent()

```
void Digikam::TagFolderView::contextMenuEvent (
    QContextMenuEvent * event ) [override], [protected]
```

## Parameters

<i>event</i>	context menu event triggered by right click
--------------	---

If no item is selected append root tag

### 9.1270.2.3 contextMenuTitle()

```
QString Digikam::TagFolderView::contextMenuTitle ( ) const [override], [protected], [virtual]
```

## Returns

title for the context menu

Reimplemented from [Digikam::AbstractAlbumTreeView](#).

**9.1270.2.4 handleCustomContextMenuAction()**

```
void Digikam::TagFolderView::handleCustomContextMenuAction (
    QAction * action,
    const AlbumPointer< Album > & album ) [override], [protected], [virtual]
```

**Parameters**

<i>action</i>	the action that was chosen by the user, may be null if none of the custom actions were selected
<i>album</i>	the tag on which the context menu was requested. May be null if there was no

Reimplemented from [Digikam::AbstractAlbumTreeView](#).

**9.1270.2.5 setContextMenuItems()**

```
void Digikam::TagFolderView::setContextMenuItems (
    ContextMenuHelper & cmh,
    const QList< TAlbum * > & albums ) [protected], [virtual]
```

If only one element is selected, only AddCustomContextMenuActions is called

**Parameters**

<i>cmh</i>	- helper object to create context menu
<i>albums</i>	- vector of selected albums to be used on menu actions

Reimplemented in [Digikam::TagMngrTreeView](#).

**9.1270.2.6 setShowDeleteFaceTagsAction()**

```
void Digikam::TagFolderView::setShowDeleteFaceTagsAction (
    bool show )
```

**Parameters**

<i>show</i>	if <code>true</code> the action to delete people tags in the tag album is displayed
-------------	---

**9.1270.2.7 setShowFindDuplicateAction()**

```
void Digikam::TagFolderView::setShowFindDuplicateAction (
    bool show )
```

**Parameters**

<i>show</i>	if <code>true</code> the action to find duplicate images in the tag album is displayed
-------------	--

## 9.1271 Digikam::TaggingAction Class Reference

### Public Types

- enum [Type](#) { **NoAction** , **AssignTag** , **CreateNewTag** }

*Describes two possible actions: Assigning an existing tag, known by tag id, or creation of a new tag, with a given tag name and a parent tag.*

### Public Member Functions

- **TaggingAction** ()=default  
*Create a NoAction.*
- [TaggingAction](#) (const QString &name, int parentTagId)  
*Create a new tag with the given name.*
- **TaggingAction** (int tagId)  
*Assign the existing tag with given id.*
- bool **isValid** () const
- QString **newTagName** () const  
*If shallCreateNewTag(), returns the tag name and the parent tag id, 0 for toplevel tag.*
- bool **operator==** (const [TaggingAction](#) &other) const
- int **parentTagId** () const
- bool **shallAssignTag** () const
- bool **shallCreateNewTag** () const
- int **tagId** () const  
*If shallAssignTag(), returns the tag id.*
- [Type](#) **type** () const

### Protected Attributes

- int **m\_tagId** = -1
- QString **m\_tagName**
- [Type](#) **m\_type** = NoAction

## 9.1271.1 Constructor & Destructor Documentation

### 9.1271.1.1 TaggingAction()

```
Digikam::TaggingAction::TaggingAction (
    const QString & name,
    int parentTagId )
```

The parent shall be the tag with the given id, or 0 for a toplevel tag.

## 9.1272 Digikam::TaggingActionFactory Class Reference

### Classes

- class [ConstraintInterface](#)

## Public Types

- enum [NameMatchMode](#) { [MatchStartingWithFragment](#) , [MatchContainingFragment](#) }

## Public Member Functions

- QList< [TaggingAction](#) > **actions** () const  
*Returns the sorted list of suggested tagging actions, based on the above settings.*
- [ConstraintInterface](#) \* **constraintInterface** () const
- [TaggingAction](#) **defaultTaggingAction** () const  
*Returns one single action, which is decided to be the presumably best action based on the settings.*
- QString **fragment** () const
- int **indexOfDefaultAction** () const  
*Returns the index of the default action in the list returned by generate()*
- int **indexOfLastRecentAction** () const  
*Returns the index of the last recent action in the list returned by actions()*
- [NameMatchMode](#) **nameMatchMode** () const
- int **parentTagId** () const
- void **reset** ()  
*reset all settings to the default (no fragment, no actions)*
- void **setConstraintInterface** ([ConstraintInterface](#) \*const iface)  
*Allows to filter the scope of suggested tags.*
- void **setFragment** (const QString &fragment)  
*Set a fragment of a tag name to generate possible tags, as known from completers.*
- void **setNameMatchMode** ([NameMatchMode](#) mode)  
*Set the matching mode for the tag name.*
- void **setParentTag** (int parentTagId)  
*Set a tag which may by the user be intended to be the parent of a newly created tag.*
- QString **suggestedUIString** (const [TaggingAction](#) &action) const  
*Returns a string to be used in the UI for the given [TaggingAction](#), interpreted in the context of the current settings.*

## Static Public Member Functions

- static [TaggingAction](#) **defaultTaggingAction** (const QString &tagName, int parentTagId=0)

## 9.1272.1 Member Enumeration Documentation

### 9.1272.1.1 NameMatchMode

enum [Digikam::TaggingActionFactory::NameMatchMode](#)

#### Enumerator

MatchStartingWithFragment	Default: use the "startingWith" method.
MatchContainingFragment	use the "contains" method

## 9.1272.2 Member Function Documentation

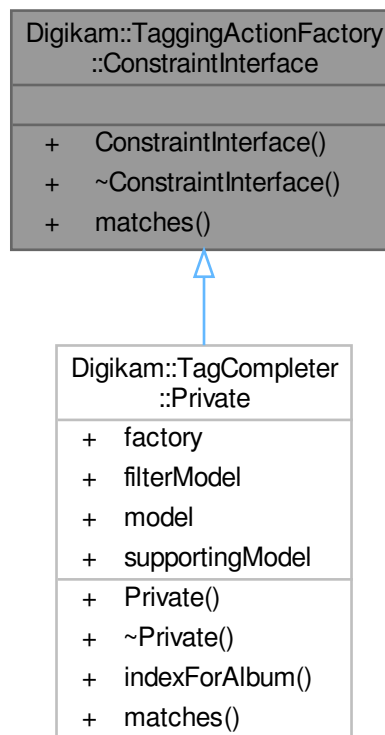
### 9.1272.2.1 setConstraintInterface()

```
void Digikam::TaggingActionFactory::setConstraintInterface (
    ConstraintInterface *const iface )
```

Pass an implementation of [ConstraintInterface](#) (remains in your ownership). [actions\(\)](#) will then only suggest to assign tags for which [matches\(\)](#) is true

## 9.1273 Digikam::TaggingActionFactory::ConstraintInterface Class Reference

Inheritance diagram for Digikam::TaggingActionFactory::ConstraintInterface:



### Public Member Functions

- virtual bool **matches** (int tagId)=0

## 9.1274 Digikam::TagInfo Class Reference

A container class for transporting tag information from the database to [AlbumManager](#).

### Public Types

- typedef QList< [TagInfo](#) > **List**

### Public Member Functions

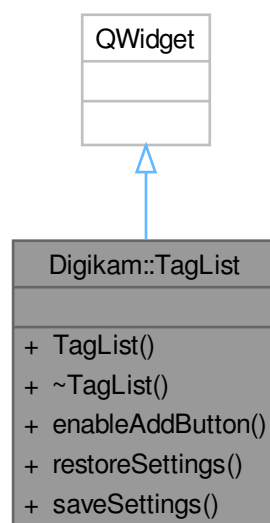
- bool **isNull** () const
- bool **operator**< (const [TagInfo](#) &info) const

### Public Attributes

- QString **icon**
- qlonglong **iconId** = 0
- int **id** = 0
- QString **name**
- int **pid** = 0

## 9.1275 Digikam::TagList Class Reference

Inheritance diagram for Digikam::TagList:



## Public Member Functions

- **TagList** ([TagMngrTreeView](#) \*const treeView, QWidget \*const parent)
- void **enableAddButton** (bool value)  
*enableAddButton* - disable Add Button when selection is empty or only root tag is selected
- void **restoreSettings** ()  
*restoreSettings* - read settings from digikam\_tagsmanagerrc config and populate model with data
- void **saveSettings** ()  
*saveSettings* - save settings to digiKam\_tagsmanagerrc KConfig

## 9.1275.1 Member Function Documentation

### 9.1275.1.1 restoreSettings()

```
void Digikam::TagList::restoreSettings ( )
```

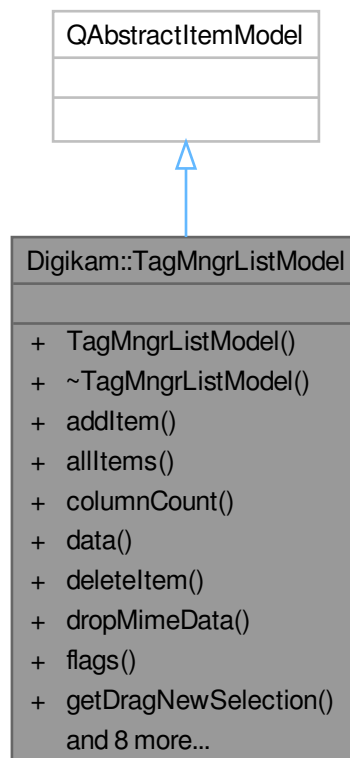
If config is empty add generic All Tags

Use this map to find all List Items that contain specific tag usually to remove deleted tag

"All Tags" item should be selected

## 9.1276 Digikam::TagMngrListModel Class Reference

Inheritance diagram for Digikam::TagMngrListModel:





## Public Member Functions

- **TagMgrListModel** (QObject \*const parent=nullptr)
- **ListItem \* addItem** (QList< QVariant > values)
  - addItem - add new item to list*
- QList< **ListItem \* > **allItems**** () const
  - allItems - return all items from List, usually to be saved in KConfig*
- int **columnCount** (const QModelIndex &parent=QModelIndex()) const override
- QVariant **data** (const QModelIndex &index, int role) const override
  - Standard methods to be implemented when subclassing QAbstractListModel.*
- void **deleteItem** (**ListItem \*const** item)
- bool **dropMimeData** (const QMimeData \*data, Qt::DropAction action, int row, int column, const QModelIndex &parent) override
- Qt::ItemFlags **flags** (const QModelIndex &index) const override
- QList< int > **getDragNewSelection** () const
- QVariant **headerData** (int section, Qt::Orientation orientation, int role=Qt::DisplayRole) const override
- QModelIndex **index** (int row, int column, const QModelIndex &parent=QModelIndex()) const override
- QMimeData \* **mimeData** (const QModelIndexList &indexes) const override
- QStringList **mimeTypes** () const override
- QModelIndex **parent** (const QModelIndex &index) const override
- int **rowCount** (const QModelIndex &parent=QModelIndex()) const override
- bool **setData** (const QModelIndex &index, const QVariant &value, int role) override
- Qt::DropActions **supportedDropActions** () const override
  - Reimplemented methods for handling drag-n-drop, encoding and decoding mime types.*

## 9.1276.1 Member Function Documentation

### 9.1276.1.1 addItem()

```
ListItem * Digikam::TagMgrListModel::addItem (
    QList< QVariant > values )
```

#### Parameters

<i>values</i>	- A list of data for item: Name as QString, QBrush as background and qlonglong as id
---------------	--

#### Returns

- pointer to newly created listitem

containsItem will return a valid pointer if item with the same values is already added to it's children list.

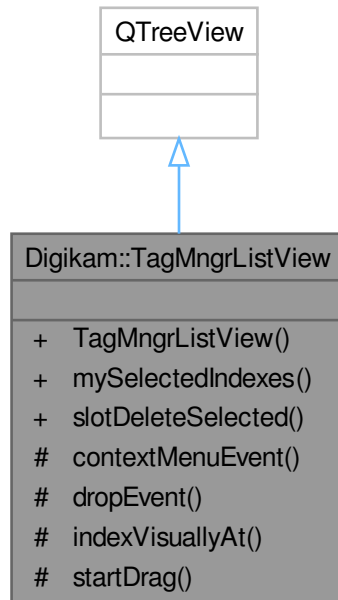
### 9.1276.1.2 dropMimeData()

```
bool Digikam::TagMgrListModel::dropMimeData (
    const QMimeData * data,
    Qt::DropAction action,
    int row,
    int column,
    const QModelIndex & parent ) [override]
```

After drag-n-drop selection is messed up, store the interval were new items are and TagsMgrListView will update selection

## 9.1277 Digikam::TagMngrListView Class Reference

Inheritance diagram for Digikam::TagMngrListView:



### Public Slots

- void **slotDeleteSelected** ()  
*slotDeleteSelected - delete selected item from Quick Access List*

### Public Member Functions

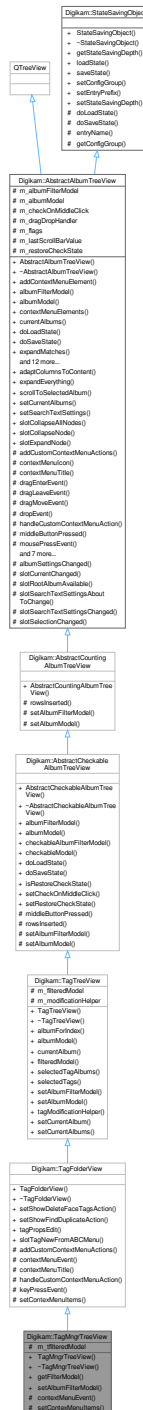
- **TagMngrListView** (QWidget \*const parent=nullptr)
- QModelIndexList **mySelectedIndexes** ()

### Protected Member Functions

- void **contextMenuEvent** (QContextMenuEvent \*event) override  
*contextMenuEvent - reimplemented method from QListView to handle custom context menu*
- void **dropEvent** (QDropEvent \*e) override
- QModelIndex **indexVisuallyAt** (const QPoint &p)
- void **startDrag** (Qt::DropActions supportedActions) override  
*Reimplemented methods to enable custom drag-n-drop in QListView.*

## 9.1278 Digikam::TagMngrTreeView Class Reference

Inheritance diagram for Digikam::TagMngrTreeView:



### Public Member Functions

- **TagMngrTreeView** ([TagsManager](#) \*const parent, [TagModel](#) \*const model)
- [TagsManagerFilterModel](#) \* **getFilterModel** () const

- void **setAlbumFilterModel** ([TagsManagerFilterModel](#) \*const [filteredModel](#), [CheckableAlbumFilterModel](#) \*const [filterModel](#))  
*setAlbumFilterModel reimplement from AbstractAlbumTree*

### Public Member Functions inherited from [Digikam::TagFolderView](#)

- [TagFolderView](#) ([QWidget](#) \*const [parent](#), [TagModel](#) \*const [model](#))  
*Constructor.*
- [~TagFolderView](#) () override  
*Destructor.*
- void **setShowDeleteFaceTagsAction** (bool [show](#))  
*Define whether to show the "Delete People Tags" action in context menus or not.*
- void **setShowFindDuplicateAction** (bool [show](#))  
*Define whether to show the "find duplicate" action in context menus or not.*
- void **tagPropsEdit** ()  
*Open tag for editing.*

### Public Member Functions inherited from [Digikam::TagTreeView](#)

- [TagTreeView](#) ([QWidget](#) \*const [parent](#)=nullptr, [Flags](#) [flags](#)=DefaultFlags)
- [TAlbum](#) \* **albumForIndex** (const [QModelIndex](#) &[index](#)) const
- [TagModel](#) \* **albumModel** () const
- [TAlbum](#) \* **currentAlbum** () const  
*currentAlbum Even if multiple selection is enabled current Album can be only one, the last clicked item if you need selected items, see selectedAlbums() It's NOT the same as AlbumManager::currentAlbums()*
- [TagPropertiesFilterModel](#) \* **filteredModel** () const  
*Contains only the tags filtered by properties - prefer to albumModel()*
- [QList](#)< [TAlbum](#) \* > **selectedTagAlbums** ()
- [QList](#)< [Album](#) \* > **selectedTags** ()  
*selectedTags - return a list of all selected items in tag model*
- void **setAlbumFilterModel** ([TagPropertiesFilterModel](#) \*const [filteredModel](#), [CheckableAlbumFilterModel](#) \*const [filterModel](#))
- void **setAlbumModel** ([TagModel](#) \*const [model](#))
- [TagModificationHelper](#) \* **tagModificationHelper** () const

### Public Member Functions inherited from [Digikam::AbstractCheckableAlbumTreeView](#)

- [AbstractCheckableAlbumTreeView](#) ([QWidget](#) \*const [parent](#), [Flags](#) [flags](#))  
*Models of these view can be checkable, they need not.*
- [CheckableAlbumFilterModel](#) \* **albumFilterModel** () const
- [AbstractCheckableAlbumModel](#) \* **albumModel** () const  
*Manage check state through the model directly.*
- [CheckableAlbumFilterModel](#) \* **checkableAlbumFilterModel** () const
- [AbstractCheckableAlbumModel](#) \* **checkableModel** () const
- void **doLoadState** () override  
*Implements state loading for the album tree view in a somewhat clumsy procedure because the model may not be fully loaded when this method is called.*
- void **doSaveState** () override  
*Implement this hook method for state saving.*
- bool **isRestoreCheckState** () const  
*Tells if the check state is restored while loading / saving state.*
- void **setCheckOnMiddleClick** (bool [doThat](#))  
*Enable checking on middle mouse button click (default: on).*
- void **setRestoreCheckState** (bool [restore](#))  
*Set whether to restore check state or not.*

## Public Member Functions inherited from Digikam::AbstractCountingAlbumTreeView

- **AbstractCountingAlbumTreeView** (QWidget \*const parent, Flags flags)

## Public Member Functions inherited from Digikam::AbstractAlbumTreeView

- **AbstractAlbumTreeView** (QWidget \*const parent, Flags flags)  
*Constructs an album tree view.*
- void **addContextMenuElement** (ContextMenuElement \*const element)
- AlbumFilterModel \* **albumFilterModel** () const
- AbstractSpecificAlbumModel \* **albumModel** () const
- QList< ContextMenuElement \* > **contextMenuElements** () const
- template<class A >  
QList< A \* > **currentAlbums** ()
- bool **expandMatches** (const QModelIndex &index)  
*Ensures that every current match is visible by expanding all parent entries.*
- QModelIndex **indexVisuallyAt** (const QPoint &p)  
*This is a combination of indexAt() checked with visualRect().*
- void **removeContextMenuElement** (ContextMenuElement \*const element)
- QList< Album \* > **selectedItems** ()  
*selectedItems()* -
- void **setAlbumManagerCurrentAlbum** (const bool setCurrentAlbum)  
*Some treeviews shall control the global current album kept by AlbumManager.*
- void **setContextMenuIcon** (const QPixmap &pixmap)  
*Set the context menu title and icon.*
- void **setContextMenuTitle** (const QString &title)
- void **setEnabledContextMenu** (const bool enable)  
*Determines the global decision to show a popup menu or not.*
- void **setExpandNewCurrentItem** (const bool doThat)  
*Expand an item when making it the new current item.*
- void **setExpandOnSingleClick** (const bool doThat)  
*Enable expanding of tree items on single click on the item (default: off)*
- void **setSelectAlbumOnClick** (const bool selectOnClick)  
*Sets whether to select an album on click via the album manager or not.*
- void **setSelectOnContextMenu** (const bool select)  
*Sets whether to select the album under the mouse cursor on a context menu request (so that the album is shown using the album manager) or not.*
- bool **viewportEvent** (QEvent \*event) override  
*For internal use only.*

## Public Member Functions inherited from Digikam::StateSavingObject

- **StateSavingObject** (QObject \*const host)  
*Constructor.*
- virtual ~**StateSavingObject** ()  
*Destructor.*
- **StateSavingDepth** **getStateSavingDepth** () const  
*Returns the depth used for state saving or loading.*
- void **loadState** ()  
*Invokes loading the class' state.*

- void **saveState** ()  
*Invokes saving the class' state.*
- virtual void **setConfigGroup** (const KConfigGroup &group)  
*Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void **setEntryPrefix** (const QString &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void **setStateSavingDepth** (const [StateSavingDepth](#) depth)  
*Sets the depth used for state saving or loading.*

### Protected Member Functions

- void **contextMenuEvent** (QContextMenuEvent \*event) override  
*contextMenuEvent Reimplement contextMenuEvent from AbstractAlbumTree to support multiple selection*
- void **setContextMenuItems** ([ContextMenuHelper](#) &cmh, const QList< [TAlbum](#) \* > &albums) override  
*setContextMenuItems Reimplemented method from TagsFolderView.*

### Protected Member Functions inherited from [Digikam::TagFolderView](#)

- void **addCustomContextMenuActions** ([ContextMenuHelper](#) &cmh, [Album](#) \*album) override  
*Hook method to add custom actions to the generated context menu.*
- void **contextMenuEvent** (QContextMenuEvent \*event) override  
*Reimplement contextMenuEvent from AbstractAlbumTree to support multiple selection.*
- QString **contextMenuTitle** () const override  
*Hook method to implement that returns the title for the context menu.*
- void **handleCustomContextMenuAction** (QAction \*action, const [AlbumPointer](#)< [Album](#) > &album) override  
*Hook method to handle the custom context menu actions that were added with addCustomContextMenuActions.*
- void **keyPressEvent** (QKeyEvent \*event) override

### Protected Member Functions inherited from [Digikam::AbstractCheckableAlbumTreeView](#)

- void **middleButtonPressed** ([Album](#) \*a) override
- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- void **setAlbumFilterModel** ([CheckableAlbumFilterModel](#) \*const filterModel)
- void **setAlbumModel** ([AbstractCheckableAlbumModel](#) \*const model)

### Protected Member Functions inherited from [Digikam::AbstractCountingAlbumTreeView](#)

- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- void **setAlbumFilterModel** ([AlbumFilterModel](#) \*const filterModel)
- void **setAlbumModel** ([AbstractCountingAlbumModel](#) \*const model)

## Protected Member Functions inherited from Digikam::AbstractAlbumTreeView

- virtual QPixmap **contextMenuIcon** () const  
*Hook method that can be implemented to return a special icon used for the context menu.*
- void **dragEnterEvent** (QDragEnterEvent \*e) override
- void **dragLeaveEvent** (QDragLeaveEvent \*e) override
- void **dragMoveEvent** (QDragMoveEvent \*e) override
- void **dropEvent** (QDropEvent \*e) override
- void **mousePressEvent** (QMouseEvent \*e) override  
*Other helper methods.*
- virtual QPixmap  **pixmapForDrag** (const QStyleOptionViewItem &option, QList< QModelIndex > indexes)  
*TODO: Move to delegate, when we have one.*
- void **rowsAboutToBeRemoved** (const QModelIndex &parent, int start, int end) override
- void **rowsInserted** (const QModelIndex &index, int start, int end) override
- void **setAlbumFilterModel** (AlbumFilterModel \*const filterModel)
- void **setAlbumModel** (AbstractSpecificAlbumModel \*const model)
- virtual bool **showContextMenuAt** (QContextMenuEvent \*event, Album \*albumForEvent)  
*Hook method to implement that determines if a context menu shall be displayed for the given event at the position coded in the event.*
- void **startDrag** (Qt::DropActions supportedActions) override

## Protected Member Functions inherited from Digikam::StateSavingObject

- QString **entryName** (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup **getConfigGroup** () const  
*Returns the config group that must be used for state saving and loading.*

## Protected Attributes

- **TagsManagerFilterModel** \* **m\_tfilteredModel** = nullptr

## Protected Attributes inherited from Digikam::TagTreeView

- **TagPropertiesFilterModel** \* **m\_filteredModel** = nullptr
- **TagModificationHelper** \* **m\_modificationHelper** = nullptr

## Protected Attributes inherited from Digikam::AbstractAlbumTreeView

- **AlbumFilterModel** \* **m\_albumFilterModel** = nullptr
- **AbstractSpecificAlbumModel** \* **m\_albumModel** = nullptr
- bool **m\_checkOnMiddleClick** = false
- **AlbumModelDragDropHandler** \* **m\_dragDropHandler** = nullptr
- Flags **m\_flags** = DefaultFlags
- int **m\_lastScrollBarValue** = 0
- bool **m\_restoreCheckState** = false

## Additional Inherited Members

### Public Types inherited from [Digikam::AbstractAlbumTreeView](#)

- enum [Flag](#) { [CreateDefaultModel](#) , [CreateDefaultFilterModel](#) , [CreateDefaultDelegate](#) , [ShowCountAccordingToSettings](#) , [AlwaysShowInclusiveCounts](#) , **DefaultFlags** = [CreateDefaultFilterModel](#) | [CreateDefaultDelegate](#) | [ShowCountAccordingToSettings](#) }
- typedef QFlags< [Flag](#) > **Flags**

### Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }  
*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

### Public Slots inherited from [Digikam::TagFolderView](#)

- void **slotTagNewFromABCMenu** (const QString &personName)

### Public Slots inherited from [Digikam::TagTreeView](#)

- void **setCurrentAlbum** (int tagId, bool selectInAlbumManager=true)
- void **setCurrentAlbums** (const QList< [Album](#) \* > &tags, bool selectInAlbumManager=true)

### Public Slots inherited from [Digikam::AbstractAlbumTreeView](#)

- void **adaptColumnsToContent** ()  
*Adapt the column sizes to the contents of the tree view.*
- void **expandEverything** (const QModelIndex &index)  
*Expands the complete tree under the given index.*
- void **scrollToSelectedAlbum** ()  
*Scrolls to the first selected album if there is one.*
- void **setCurrentAlbums** (const QList< [Album](#) \* > &albums, bool selectInAlbumManager=true)  
*Selects the given album.*
- void **setSearchTextSettings** (const [SearchTextSettings](#) &settings)
- void **slotCollapseAllNodes** ()  
*slotCollapseAllNodes - collapse all nodes without root node*
- void **slotCollapseNode** ()  
*slotCollapseNode - collapse recursively selected nodes*
- void **slotExpandNode** ()  
*slotExpandNode - expands recursively selected nodes*

### Signals inherited from [Digikam::TagFolderView](#)

- void **signalFindDuplicates** (const QList< [TAlbum](#) \* > &albums)



## Signals inherited from [Digikam::TagTreeView](#)

- void **assignTags** (int tagId, const QList< int > &imageIDs)

## Signals inherited from [Digikam::AbstractAlbumTreeView](#)

- void **currentAlbumChanged** ([Album](#) \*currentAlbum)  
*Emitted when the currently selected album changes.*
- void **selectedAlbumsChanged** (const QList< [Album](#) \* > &selectedAlbums)  
*Emitted when the current selection changes.*

## Protected Slots inherited from [Digikam::AbstractAlbumTreeView](#)

- void **albumSettingsChanged** ()
- void **slotCurrentChanged** ()
- virtual void **slotRootAlbumAvailable** ()  
*override if implemented behavior is not as intended*
- void **slotSearchTextSettingsAboutToChange** (bool searched, bool willSearch)
- void **slotSearchTextSettingsChanged** (bool wasSearching, bool searching)
- void **slotSelectionChanged** ()

## 9.1278.1 Member Function Documentation

### 9.1278.1.1 contextMenuEvent()

```
void Digikam::TagMngrTreeView::contextMenuEvent (
    QContextMenuEvent * event ) [override], [protected]
```

#### Parameters

<i>event</i>	context menu event triggered by right click
--------------	---

Append root tag if no nodes are selected

### 9.1278.1.2 setContextMenuItems()

```
void Digikam::TagMngrTreeView::setContextMenuItems (
    ContextMenuHelper & cmh,
    const QList< TAlbum * > & albums ) [override], [protected], [virtual]
```

Will set custom actions for Tags Manager. Some actions are also available in toolbar

#### Parameters

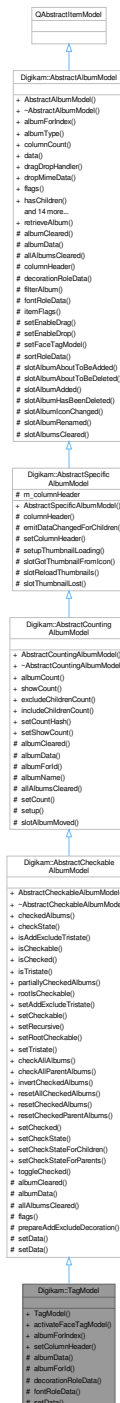
<i>cmh</i>	<a href="#">ContextMenuHelper</a> class to help setting some basic actions
<i>albums</i>	List of currently selected albums

This is a dummy action, delete is disable for root tag

Reimplemented from [Digikam::TagFolderView](#).

## 9.1279 Digikam::TagModel Class Reference

Inheritance diagram for Digikam::TagModel:



**Public Member Functions**

- **TagModel** ([RootAlbumBehavior](#) rootBehavior=[IncludeRootAlbum](#), [QObject](#) \*const parent=nullptr)  
*Create a model containing all tags.*
- void **activateFaceTagModel** ()
- [TAlbum](#) \* **albumForIndex** (const [QModelIndex](#) &index) const
- void **setColumnHeader** (const [QString](#) &header)

**Public Member Functions inherited from [Digikam::AbstractCheckableAlbumModel](#)**

- [AbstractCheckableAlbumModel](#) ([Album::Type](#) albumType, [Album](#) \*const rootAlbum, [RootAlbumBehavior](#) rootBehavior=[IncludeRootAlbum](#), [QObject](#) \*const parent=nullptr)  
*Abstract base class that manages the check state of Albums.*
- [QList](#)< [Album](#) \* > **checkedAlbums** () const  
*Returns a list of album with check state Checked.*
- [Qt::CheckState](#) **checkState** ([Album](#) \*album) const  
*Returns the check state of the album.*
- bool **isAddExcludeTristate** () const
- bool **isCheckable** () const
- bool **isChecked** ([Album](#) \*album) const  
*Returns if the given album has the check state Checked.*
- bool **isTristate** () const
- [QList](#)< [Album](#) \* > **partiallyCheckedAlbums** () const  
*Returns a list of album with partially check state Checked.*
- bool **rootIsCheckable** () const
- void **setAddExcludeTristate** (bool b)  
*Sets a special tristate mode, which offers the three modes "unchecked", "added" and "excluded", where "excluded" corresponds to partially checked internally, but is reflected in the treeview through the decoration only.*
- void **setCheckable** (bool isCheckable)  
*Triggers if the albums in this model are checkable.*
- void **setRecursive** (bool recursive)  
*If an item gets checked, all childs get checked as well, If an item gets unchecked, all childs get unchecked as well.*
- void **setRootCheckable** (bool rootIsCheckable)  
*Triggers if the root album is checkable.*
- void **setTristate** (bool isTristate)  
*Triggers if the albums in this model are tristate.*

**Public Member Functions inherited from [Digikam::AbstractCountingAlbumModel](#)**

- [AbstractCountingAlbumModel](#) ([Album::Type](#) albumType, [Album](#) \*const rootAlbum, [RootAlbumBehavior](#) rootBehavior=[IncludeRootAlbum](#), [QObject](#) \*const parent=nullptr)  
*Supports displaying a count alongside the album name in DisplayRole.*
- virtual int **albumCount** ([Album](#) \*album) const  
*Returns the number of included items for this album.*
- bool **showCount** () const

**Public Member Functions inherited from [Digikam::AbstractSpecificAlbumModel](#)**

- [AbstractSpecificAlbumModel](#) ([Album::Type](#) albumType, [Album](#) \*const rootAlbum, [RootAlbumBehavior](#) rootBehavior=[IncludeRootAlbum](#), [QObject](#) \*const parent=nullptr)  
*Abstract base class, do not instantiate.*

## Public Member Functions inherited from [Digikam::AbstractAlbumModel](#)

- [AbstractAlbumModel](#) ([Album::Type](#) albumType, [Album](#) \*const rootAlbum, [RootAlbumBehavior](#) rootBehavior=[IncludeRootAlbum](#), [QObject](#) \*const parent=nullptr)
  - Create an [AbstractAlbumModel](#) object for albums with the given type.*
- [Album](#) \* **albumForIndex** (const [QModelIndex](#) &index) const
  - Returns the album object associated with the given model index.*
- [Album::Type](#) **albumType** () const
  - Returns the [Album::Type](#) of the contained albums.*
- int **columnCount** (const [QModelIndex](#) &parent=[QModelIndex](#)()) const override
- [QVariant](#) **data** (const [QModelIndex](#) &index, int role=[Qt::DisplayRole](#)) const override
- [AlbumModelDragDropHandler](#) \* **dragDropHandler** () const
  - Returns the drag drop handler, or 0 if none is installed.*
- bool **dropMimeData** (const [QMimeData](#) \*data, [Qt::DropAction](#) action, int row, int column, const [QModelIndex](#) &parent) override
- [Qt::ItemFlags](#) **flags** (const [QModelIndex](#) &index) const override
- bool **hasChildren** (const [QModelIndex](#) &parent=[QModelIndex](#)()) const override
- [QVariant](#) **headerData** (int section, [Qt::Orientation](#) orientation, int role=[Qt::DisplayRole](#)) const override
- [QModelIndex](#) **index** (int row, int column, const [QModelIndex](#) &parent=[QModelIndex](#)()) const override
- [QModelIndex](#) **indexForAlbum** ([Album](#) \*album) const
  - Return the [QModelIndex](#) for the given album, or an invalid index if the album is not contained in this model.*
- bool **isFaceTagModel** () const
  - Returns true if the album model a face tag model.*
- [QMimeData](#) \* **mimeData** (const [QModelIndexList](#) &indexes) const override
- [QStringList](#) **mimeTypes** () const override
- [QModelIndex](#) **parent** (const [QModelIndex](#) &index) const override
- [Album](#) \* **rootAlbum** () const
- [RootAlbumBehavior](#) **rootAlbumBehavior** () const
  - Returns the root album behavior set for this model.*
- [QModelIndex](#) **rootAlbumIndex** () const
  - Return the index corresponding to the root album.*
- int **rowCount** (const [QModelIndex](#) &parent=[QModelIndex](#)()) const override
- void **setDragDropHandler** ([AlbumModelDragDropHandler](#) \*handler)
  - Set a drag drop handler.*
- void **setDropIndex** (const [QModelIndex](#) &index)
  - Set current index from [QDragMoveEvent](#).*
- [Qt::DropActions](#) **supportedDropActions** () const override

## Protected Member Functions

- [QVariant](#) **albumData** ([Album](#) \*a, int role) const override
  - For subclassing convenience: A part of the implementation of data()*
- [Album](#) \* **albumForId** (int id) const override
  - need to implement in subclass*
- [QVariant](#) **decorationRoleData** ([Album](#) \*a) const override
  - For subclassing convenience: A part of the implementation of data()*
- [QVariant](#) **fontRoleData** ([Album](#) \*a) const override
  - For subclassing convenience: A part of the implementation of data()*
- bool **setData** (const [QModelIndex](#) &index, const [QVariant](#) &value, int role=[Qt::EditRole](#)) override

## Protected Member Functions inherited from [Digikam::AbstractCheckableAlbumModel](#)

- void [albumCleared](#) ([Album](#) \*album) override  
*Notification when an entry is removed.*
- void [allAlbumsCleared](#) () override  
*Notification when all entries are removed.*
- Qt::ItemFlags **flags** (const [QModelIndex](#) &index) const override
- void **prepareAddExcludeDecoration** ([Album](#) \*a, [QPixmap](#) &icon) const  
*If in AddExcludeTristate mode, changes the icon as to indicate the state.*
- bool **setData** (const [QModelIndex](#) &index, const [QVariant](#) &value, int role, bool recursive)
- bool [setData](#) (const [QModelIndex](#) &index, const [QVariant](#) &value, int role=[Qt::EditRole](#)) override

## Protected Member Functions inherited from [Digikam::AbstractCountingAlbumModel](#)

- void [albumCleared](#) ([Album](#) \*album) override  
*Notification when an entry is removed.*
- virtual [QString](#) [albumName](#) ([Album](#) \*a) const  
*Can reimplement in subclass.*
- void [allAlbumsCleared](#) () override  
*Notification when all entries are removed.*
- void **setCount** ([Album](#) \*album, int count)  
*If you do not use setCountHash, excludeChildrenCount and includeChildrenCount, you can set a count here.*
- void **setup** ()  
*Call this method in children class constructors to init signal/slots connections.*

## Protected Member Functions inherited from [Digikam::AbstractSpecificAlbumModel](#)

- [QString](#) [columnHeader](#) () const override  
*For subclassing convenience: A part of the implementation of headerData()*
- void **emitDataChangedForChildren** ([Album](#) \*album)
- void **setColumnHeader** (const [QString](#) &header)
- void **setupThumbnailLoading** ()  
*You need to call this from your constructor if you intend to load the thumbnail facilities of this class.*

## Protected Member Functions inherited from [Digikam::AbstractAlbumModel](#)

- virtual bool [filterAlbum](#) ([Album](#) \*album) const  
*Returns true for those and only those albums that shall be contained in this model.*
- virtual Qt::ItemFlags **itemFlags** ([Album](#) \*album) const  
*For subclassing convenience: A part of the implementation of itemFlags()*
- void [setEnableDrag](#) (bool enable)  
*Switch on drag and drop globally for all items.*
- void **setEnableDrop** (bool enable)
- void **setFaceTagModel** (bool enable)
- virtual [QVariant](#) [sortRoleData](#) ([Album](#) \*a) const  
*For subclassing convenience: A part of the implementation of data()*

## Additional Inherited Members

### Public Types inherited from [Digikam::AbstractAlbumModel](#)

- enum [AlbumDataRole](#) {  
[AlbumTitleRole](#) = Qt::UserRole , [AlbumTypeRole](#) = Qt::UserRole + 1 , [AlbumPointerRole](#) = Qt::UserRole + 2  
, [AlbumIdRole](#) = Qt::UserRole + 3 ,  
[AlbumGlobalIdRole](#) = Qt::UserRole + 4 , [AlbumSortRole](#) = Qt::UserRole + 5 }
  - enum [RootAlbumBehavior](#) { [IncludeRootAlbum](#) , [IgnoreRootAlbum](#) }
- [AbstractAlbumModel](#) is the abstract base class for all models that present [Album](#) objects as managed by [AlbumManager](#).*

### Public Slots inherited from [Digikam::AbstractCheckableAlbumModel](#)

- void **checkAllAlbums** (const QModelIndex &parent=QModelIndex())  
*Checks all albums beneath the given parent.*
- void **checkAllParentAlbums** (const QModelIndex &child)  
*Checks all parent albums starting at the child, including it.*
- void **invertCheckedAlbums** (const QModelIndex &parent=QModelIndex())  
*Inverts the checked state of all albums under the given parent.*
- void **resetAllCheckedAlbums** ()  
*Resets the checked state of all albums to Qt::Unchecked.*
- void **resetCheckedAlbums** (const QModelIndex &parent=QModelIndex())  
*Resets the checked state of all albums under the given parent.*
- void **resetCheckedParentAlbums** (const QModelIndex &child)  
*Resets the checked state of all parents of the child including it.*
- void **setChecked** ([Album](#) \*album, bool isChecked)  
*Sets the check state of album to Checked or Unchecked.*
- void **setCheckState** ([Album](#) \*album, Qt::CheckState state)  
*Sets the check state of the album.*
- void **setCheckStateForChildren** ([Album](#) \*album, Qt::CheckState state)  
*Sets the checked state recursively for all children of but not for the given album.*
- void **setCheckStateForParents** ([Album](#) \*album, Qt::CheckState state)  
*Sets the checked state recursively for all parents of but not for the given album.*
- void **toggleChecked** ([Album](#) \*album)  
*Toggles the check state of album between Checked or Unchecked.*

### Public Slots inherited from [Digikam::AbstractCountingAlbumModel](#)

- void **excludeChildrenCount** (const QModelIndex &index)  
*Displays only the count of the album, without adding child albums' counts.*
- void **includeChildrenCount** (const QModelIndex &index)  
*Displays sum of the count of the album and child albums' counts.*
- void **setCountHash** (const QHash< int, int > &idCountHash)  
*Enable displaying the count.*
- void **setShowCount** (bool show)  
*Call to enable or disable showing the count. Default is false.*

### Signals inherited from [Digikam::AbstractCheckableAlbumModel](#)

- void [checkStateChanged](#) ([Album](#) \*album, Qt::CheckState [checkState](#))  
*Emitted when the check state of an album changes.*

### Signals inherited from [Digikam::AbstractCountingAlbumModel](#)

- void [signalUpdateAlbumCount](#) ([Album](#) \*album)

### Signals inherited from [Digikam::AbstractAlbumModel](#)

- void [rootAlbumAvailable](#) ()  
*This is initialized once after creation, if the root album becomes available, if it was not already available at time of construction.*

### Static Public Member Functions inherited from [Digikam::AbstractAlbumModel](#)

- static [Album](#) \* [retrieveAlbum](#) (const QModelIndex &index)  
*Returns the album represented by the index.*

### Protected Slots inherited from [Digikam::AbstractCountingAlbumModel](#)

- void [slotAlbumMoved](#) ([Album](#) \*album)

### Protected Slots inherited from [Digikam::AbstractSpecificAlbumModel](#)

- void [slotGotThumbnailFromIcon](#) ([Album](#) \*album, const QPixmap &thumbnail)
- void [slotReloadThumbnails](#) ()
- void [slotThumbnailLost](#) ([Album](#) \*album)

### Protected Slots inherited from [Digikam::AbstractAlbumModel](#)

- void [slotAlbumAboutToBeAdded](#) ([Album](#) \*album, [Album](#) \*parent, [Album](#) \*prev)
- void [slotAlbumAboutToBeDeleted](#) ([Album](#) \*album)
- void [slotAlbumAdded](#) ([Album](#) \*)
- void [slotAlbumHasBeenDeleted](#) ([Album](#) \*album)
- void [slotAlbumIconChanged](#) ([Album](#) \*album)
- void [slotAlbumRenamed](#) ([Album](#) \*album)
- void [slotAlbumsCleared](#) ()

### Protected Attributes inherited from [Digikam::AbstractSpecificAlbumModel](#)

- QString [m\\_columnHeader](#)

## 9.1279.1 Member Function Documentation

### 9.1279.1.1 albumData()

```
QVariant Digikam::TagModel::albumData (
    Album * a,
    int role ) const [override], [protected], [virtual]
```

#### Note

these can be reimplemented in a subclass

Reimplemented from [Digikam::AbstractCheckableAlbumModel](#).

### 9.1279.1.2 albumForId()

```
Album * Digikam::TagModel::albumForId (
    int id ) const [override], [protected], [virtual]
```

Implements [Digikam::AbstractCountingAlbumModel](#).

### 9.1279.1.3 decorationRoleData()

```
QVariant Digikam::TagModel::decorationRoleData (
    Album * a ) const [override], [protected], [virtual]
```

Reimplemented from [Digikam::AbstractAlbumModel](#).

### 9.1279.1.4 fontRoleData()

```
QVariant Digikam::TagModel::fontRoleData (
    Album * a ) const [override], [protected], [virtual]
```

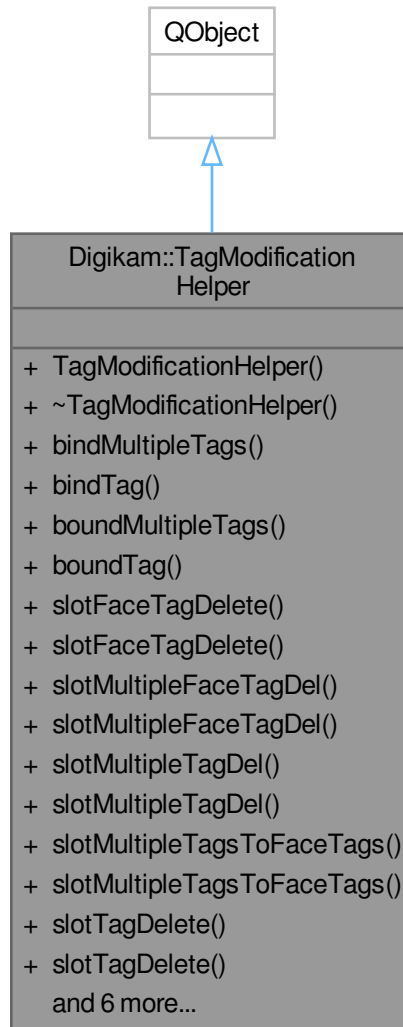
Reimplemented from [Digikam::AbstractAlbumModel](#).



## 9.1280 Digikam::TagModificationHelper Class Reference

Utility class providing methods to modify tag albums ([TAlbum](#)) in a way useful to implement views.

Inheritance diagram for Digikam::TagModificationHelper:



### Public Slots

- void **slotFaceTagDelete** ()  
*must use bindTag and a QAction*
- void **slotFaceTagDelete** ([TAlbum](#) \*tag)  
*Deletes the given face tag and after prompting the user for this.*
- void **slotMultipleFaceTagDel** ()  
*must use bindMultipleTags and a QAction*
- void **slotMultipleFaceTagDel** (const QList< [TAlbum](#) \* > &tags)

- Delete multiple face tags and prompt user only once for all The tags itself are not deleted.*

  - void **slotMultipleTagDel** ()
    - must use bindMultipleTags and a QAction*
  - void **slotMultipleTagDel** (const QList< TAlbum \* > &tags)
    - must use bindTag and a QAction*
  - void **slotMultipleTagsToFaceTags** ()
    - must use bindMultipleTags and a QAction*
  - void **slotMultipleTagsToFaceTags** (const QList< TAlbum \* > &tags)
    - Marks the tags as face tags if they are not already.*
  - void **slotTagDelete** ()
    - must use bindTag and a QAction*
  - void **slotTagDelete** (TAlbum \*tag)
    - Deletes the given tag and after prompting the user for this.*
  - void **slotTagEdit** ()
    - must use bindTag and a QAction*
  - void **slotTagEdit** (TAlbum \*tag)
    - Edits the given tag via a user dialog.*
  - TAlbum \* **slotTagNew** ()
    - Same as above, but this slot can be triggered from a QAction if a parent tag is bound to this action, see below.*
  - TAlbum \* **slotTagNew** (TAlbum \*parent, const QString &title=QString(), const QString &iconName=QString())
    - Creates one ore more new tags under the given parent.*
  - void **slotTagToFaceTag** ()
    - must use bindTag and a QAction*
  - void **slotTagToFaceTag** (TAlbum \*tag)
    - Marks the tag as face tag if it is not already.*

## Signals

- void **aboutToDeleteTag** (TAlbum \*tag)
- void **tagCreated** (TAlbum \*tag)
- void **tagEdited** (TAlbum \*tag)

## Public Member Functions

- **TagModificationHelper** (QObject \*const parent, QWidget \*const dialogParent)
  - Constructor.*
- **~TagModificationHelper** () override
  - Destructor.*
- void **bindMultipleTags** (QAction \*action, const QList< TAlbum \* > &tags)
  - Set QVector's pointer into action's data.*
- void **bindTag** (QAction \*action, TAlbum \*parent) const
  - Sets the tag that the given action operates on.*
- QList< TAlbum \* > **boundMultipleTags** (QObject \*sender)
  - Return QVector pointer bound with bindMultipleTags.*
- TAlbum \* **boundTag** (QObject \*action) const
  - Returns the tag bound with bindTag.*

## 9.1280.1 Detailed Description

This class can do background processing for batch tag operations. So be sure that the signals indicating the progress of these operations are used.

### Author

jwienke

## 9.1280.2 Constructor & Destructor Documentation

### 9.1280.2.1 TagModificationHelper()

```
Digikam::TagModificationHelper::TagModificationHelper (
    QObject *const parent,
    QWidget *const dialogParent ) [explicit]
```

#### Parameters

<i>parent</i>	the parent for qt parent child mechanism
<i>dialogParent</i>	parent widget for dialogs displayed by this object

## 9.1280.3 Member Function Documentation

### 9.1280.3.1 bindMultipleTags()

```
void Digikam::TagModificationHelper::bindMultipleTags (
    QAction * action,
    const QList< TAlbum * > & tags )
```

Make sure that QVector is not a local object and it's not destroyed before boundMultipleTags are called

#### Parameters

<i>action</i>	- action to store pointer
<i>tags</i>	- QVector pointer to be stored

### 9.1280.3.2 bindTag()

```
void Digikam::TagModificationHelper::bindTag (
    QAction * action,
    TAlbum * parent ) const
```

You must call bindTag and then connect the action's triggered to the desired slot, [slotTagNew\(\)](#), [slotTagEdit\(\)](#) or [slotTagDelete\(\)](#). Note: Changes the Action's user data.

### 9.1280.3.3 boundMultipleTags()

```
QList< TAlbum * > Digikam::TagModificationHelper::boundMultipleTags (
    QObject * sender )
```

Use when context menu should delete more than one item: multiple-selection.

### 9.1280.3.4 boundTag()

```
TAlbum * Digikam::TagModificationHelper::boundTag (
    QObject * action ) const
```

The given QObject shall be a QAction, but for convenience the given object will be checked with `qobject_cast` first, so you can pass `QObject::sender()`.

### 9.1280.3.5 slotFaceTagDelete

```
void Digikam::TagModificationHelper::slotFaceTagDelete (
    TAlbum * tag ) [slot]
```

The tag itself is not deleted. Only its property as face tag.

#### Parameters

<i>tag</i>	the face tag to delete
------------	------------------------

### 9.1280.3.6 slotMultipleFaceTagDel

```
void Digikam::TagModificationHelper::slotMultipleFaceTagDel (
    const QList< TAlbum * > & tags ) [slot]
```

Only their properties as face tags.

#### Parameters

<i>tags</i>	face tags to be deleted.
-------------	--------------------------

### 9.1280.3.7 slotMultipleTagDel

```
void Digikam::TagModificationHelper::slotMultipleTagDel (
    const QList< TAlbum * > & tags ) [slot]
```

Delete multiple tags and prompt user only once for all

#### Parameters

<i>tags</i>	the tags to be deleted, without root tag
-------------	--

Tags must be deleted from children to parents, if we don't want to step on invalid index. Use QMap to order them by distance to root tag

QMultimap doesn't provide reverse iterator, use QList.

### 9.1280.3.8 slotMultipleTagsToFaceTags

```
void Digikam::TagModificationHelper::slotMultipleTagsToFaceTags (
    const QList< TAlbum * > & tags ) [slot]
```

#### Parameters

<i>tags</i>	the tags to mark.
-------------	-------------------

### 9.1280.3.9 slotTagDelete

```
void Digikam::TagModificationHelper::slotTagDelete (
    TAlbum * tag ) [slot]
```

#### Parameters

<i>tag</i>	the tag to delete, must not be the root tag album
------------	---

### 9.1280.3.10 slotTagEdit

```
void Digikam::TagModificationHelper::slotTagEdit (
    TAlbum * tag ) [slot]
```

#### Parameters

<i>tag</i>	the tag to change
------------	-------------------

### 9.1280.3.11 slotTagNew [1/2]

```
TAlbum * Digikam::TagModificationHelper::slotTagNew ( ) [slot]
```

Without this mechanism, will add a toplevel tag.

#### Returns

new tag created or 0 if no tag was created

**9.1280.3.12 slotTagNew [2/2]**

```
TAlbum * Digikam::TagModificationHelper::slotTagNew (
    TAlbum * parent,
    const QString & title = QString(),
    const QString & iconName = QString() ) [slot]
```

If only the parent is given, then a dialog is shown to create new tags. Else, if also a title and optionally an icon are given, then these values will be used directly to create the tag.

**Parameters**

<i>parent</i>	the parent tag album under which to create the new tags. May be 0 to use the root album
<i>title</i>	if this isn't an empty string, then this tag name is suggested
<i>iconName</i>	an optional name for the icon to suggest for the new tag

**Returns**

new tag album or 0 if not created

**9.1280.3.13 slotTagToFaceTag**

```
void Digikam::TagModificationHelper::slotTagToFaceTag (
    TAlbum * tag ) [slot]
```

**Parameters**

<i>tag</i>	the tag to mark
------------	-----------------

**9.1281 Digikam::TagProperties Class Reference****Public Member Functions**

- [TagProperties](#) ()
  - This class provides a wrapper over the Database methods to access the properties of a tag.*
- **TagProperties** (const [TagProperties](#) &other)
- **TagProperties** (int tagId)
  - Access the properties of the given tag.*
- void **addProperty** (const QString &key, const QString &value)
  - Adds the given property.*
- bool **hasProperty** (const QString &key) const
  - Returns true if the property is set.*
- bool **hasProperty** (const QString &key, const QString &value) const
  - Returns true if the property is set, with exactly the given value.*
- bool **isNull** () const
- [TagProperties](#) & **operator=** (const [TagProperties](#) &other)
- QMap< QString, QString > **properties** () const

- Returns a map of all key->value pairs.*
- `QStringList propertyKeys () const`  
*Returns all set property keys.*
- `void removeProperties (const QString &key)`  
*Remove all occurrences of the property.*
- `void removeProperty (const QString &key, const QString &value)`  
*Remove the given property/value.*
- `void setProperty (const QString &key, const QString &value)`  
*Set the given property. Replaces all previous occurrences of this property.*
- `int tagId () const`
- `QString value (const QString &key) const`  
*Returns the value of the given property.*

### Static Public Member Functions

- `static TagProperties getOrCreate (const QString &tagPath)`  
*Finds the tag for the given tag path or creates a new tag.*

## 9.1281.1 Constructor & Destructor Documentation

### 9.1281.1.1 TagProperties()

```
Digikam::TagProperties::TagProperties ( )
```

It is meant to be a short-lived object, it does not listen to external database changes.

## 9.1281.2 Member Function Documentation

### 9.1281.2.1 addProperty()

```
void Digikam::TagProperties::addProperty (
    const QString & key,
    const QString & value )
```

Does not change any previous occurrences of this property, allowing multiple properties with the same key.

### 9.1281.2.2 getOrCreate()

```
TagProperties Digikam::TagProperties::getOrCreate (
    const QString & tagPath ) [static]
```

Then returns the tag properties for this tag.

### 9.1281.2.3 value()

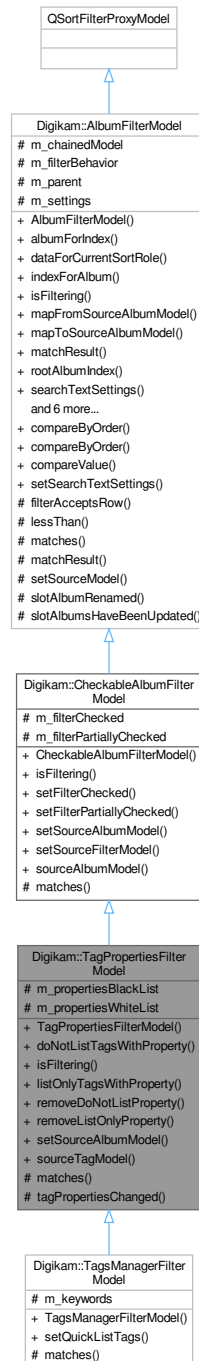
```
QString Digikam::TagProperties::value (
    const QString & key ) const
```

If the property is not set, a null string is returned. But a null string is also returned if the property is set, but without a value. Use `hasProperty` to check that case.

## 9.1282 Digikam::TagPropertiesFilterModel Class Reference

[Filter](#) model for tags that can filter by tag property.

Inheritance diagram for Digikam::TagPropertiesFilterModel:



### Public Member Functions

- **TagPropertiesFilterModel** (QObject \*const parent=nullptr)



- void **doNotListTagsWithProperty** (const QString &property)
- bool **isFiltering** () const override  
*Returns if the currently applied filters will result in any filtering.*
- void **listOnlyTagsWithProperty** (const QString &property)
- void **removeDoNotListProperty** (const QString &property)
- void **removeListOnlyProperty** (const QString &property)
- void **setSourceAlbumModel** (TagModel \*const source)
- TagModel \* **sourceTagModel** () const

## Public Member Functions inherited from Digikam::CheckableAlbumFilterModel

- **CheckableAlbumFilterModel** (QObject \*const parent=nullptr)
- void **setFilterChecked** (bool filter)
- void **setFilterPartiallyChecked** (bool filter)
- void **setSourceAlbumModel** (AbstractCheckableAlbumModel \*const source)
- void **setSourceFilterModel** (CheckableAlbumFilterModel \*const source)
- AbstractCheckableAlbumModel \* **sourceAlbumModel** () const

## Public Member Functions inherited from Digikam::AlbumFilterModel

- **AlbumFilterModel** (QObject \*const parent=nullptr)
- Album \* **albumForIndex** (const QModelIndex &index) const  
*Convenience methods.*
- QVariant **dataForCurrentSortRole** (Album \*album) const
- QModelIndex **indexForAlbum** (Album \*album) const
- QModelIndex **mapFromSourceAlbumModel** (const QModelIndex &index) const
- QModelIndex **mapToSourceAlbumModel** (const QModelIndex &index) const
- MatchResult **matchResult** (const QModelIndex &index) const  
*Returns the MatchResult of an index of this model.*
- QModelIndex **rootAlbumIndex** () const
- SearchTextSettings **searchTextSettings** () const  
*Returns the settings currently used for filtering.*
- void **setFilterBehavior** (FilterBehavior behavior)  
*Sets the filter behavior.*
- void **setSourceAlbumModel** (AbstractAlbumModel \*const source)  
*Sets the source model.*
- void **setSourceFilterModel** (AlbumFilterModel \*const source)  
*Sets a chained filter model.*
- AbstractAlbumModel \* **sourceAlbumModel** () const
- AlbumFilterModel \* **sourceFilterModel** () const
- void **updateFilter** ()  
*Force invalidateFilter() externally.*

## Protected Slots

- void **tagPropertiesChanged** (TAlbum \*)

## Protected Slots inherited from Digikam::AlbumFilterModel

- void **slotAlbumRenamed** (Album \*album)
- void **slotAlbumsHaveBeenUpdated** (int type)

### Protected Member Functions

- bool [matches](#) ([Album](#) \*album) const override  
*This method provides the basic match checking algorithm.*

### Protected Member Functions inherited from [Digikam::AlbumFilterModel](#)

- bool [filterAcceptsRow](#) (int source\_row, const QModelIndex &source\_parent) const override
- bool [lessThan](#) (const QModelIndex &left, const QModelIndex &right) const override
- [MatchResult](#) [matchResult](#) ([Album](#) \*album) const  
*Returns if the filter matches this album (same logic as filterAcceptsRow).*
- void [setSourceModel](#) (QAbstractItemModel \*const model) override  
*Use setSourceAlbumModel.*

### Protected Attributes

- QSet< QString > [m\\_propertiesBlackList](#)
- QSet< QString > [m\\_propertiesWhiteList](#)

### Protected Attributes inherited from [Digikam::CheckableAlbumFilterModel](#)

- bool [m\\_filterChecked](#) = false
- bool [m\\_filterPartiallyChecked](#) = false

### Protected Attributes inherited from [Digikam::AlbumFilterModel](#)

- QPointer< [AlbumFilterModel](#) > [m\\_chainedModel](#) = nullptr
- [FilterBehavior](#) [m\\_filterBehavior](#) = [FullFiltering](#)
- QObject \* [m\\_parent](#) = nullptr
- [SearchTextSettings](#) [m\\_settings](#)

### Additional Inherited Members

### Public Types inherited from [Digikam::AlbumFilterModel](#)

- enum [FilterBehavior](#) { [SimpleFiltering](#) , [FullFiltering](#) , [StrictFiltering](#) }
- enum [MatchResult](#) {  
[NoMatch](#) = 0 , [DirectMatch](#) , [ParentMatch](#) , [ChildMatch](#) ,  
[SpecialMatch](#) }

### Public Slots inherited from [Digikam::AlbumFilterModel](#)

- void [setSearchTextSettings](#) (const [SearchTextSettings](#) &settings)  
*Accepts new settings used for filtering and applies them to the model.*

## Signals inherited from [Digikam::AlbumFilterModel](#)

- void [hasSearchResult](#) (bool hasResult)  
*Indicates whether the newly applied filter results in a search result or not.*
- void [searchTextSettingsAboutToChange](#) (bool searched, bool willSearch)  
*This signal indicates that a new [SearchTextSettings](#) arrived and is about to be applied to the model.*
- void [searchTextSettingsChanged](#) (bool wasSearching, bool searched)  
*Indicates that new search text settings were applied.*
- void **signalFilterChanged** ()  
*Indicates that a new filter was applied to the model.*

## Static Public Member Functions inherited from [Digikam::AlbumFilterModel](#)

- template<typename T >  
static int **compareByOrder** (const T &a, const T &b, Qt::SortOrder sortOrder)
- static int **compareByOrder** (int compareResult, Qt::SortOrder sortOrder)  
*Takes a typical result from a compare method (0 is equal, -1 is less than, 1 is greater than) and applies the given sort order to it.*
- template<typename T >  
static int **compareValue** (const T &a, const T &b)  
*Returns the usual compare result of -1, 0, or 1 for lessThan, equals and greaterThan.*

### 9.1282.1 Member Function Documentation

#### 9.1282.1.1 isFiltering()

```
bool Digikam::TagPropertiesFilterModel::isFiltering ( ) const [override], [virtual]
```

##### Returns

`true` if the current selected filter could result in any filtering without checking if this really happens.

Reimplemented from [Digikam::CheckableAlbumFilterModel](#).

#### 9.1282.1.2 matches()

```
bool Digikam::TagPropertiesFilterModel::matches (
    Album * album ) const [override], [protected], [virtual]
```

Return true if this single album matches the current criteria. This method can be overridden to provide custom filtering.

##### Parameters

<code>album</code>	the album to tell if it matches the filter criteria or not.
--------------------	---

Reimplemented from [Digikam::CheckableAlbumFilterModel](#).

Reimplemented in [Digikam::TagsManagerFilterModel](#).

## 9.1283 Digikam::TagProperty Class Reference

### Public Member Functions

- bool **isNull** () const

### Public Attributes

- QString **property**
- int **tagId** = -1
- QString **value**

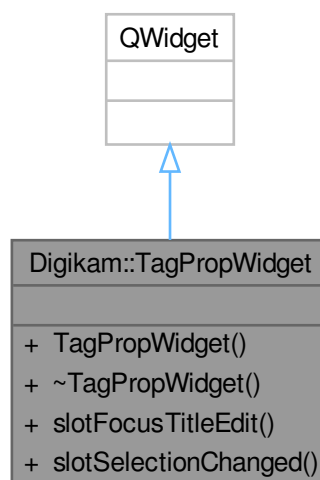
## 9.1284 Digikam::TagPropertyName Class Reference

### Static Public Member Functions

- static QLatin1String **faceEngineName** ()
- static QLatin1String **faceEngineUuid** ()
- static QLatin1String **ignoredPerson** ()
- static QLatin1String **person** ()
- static QLatin1String **tagKeyboardShortcut** ()
- static QLatin1String **unconfirmedPerson** ()
- static QLatin1String **unknownPerson** ()

## 9.1285 Digikam::TagPropWidget Class Reference

Inheritance diagram for Digikam::TagPropWidget:



### Public Types

- enum **ItemsEnable** { **DisabledAll** , **EnabledAll** , **IconOnly** }

### Public Slots

- void **slotFocusTitleEdit** ()
- void **slotSelectionChanged** (const QList< Album \* > &albums)

### Signals

- void **signalTitleEditReady** ()

### Public Member Functions

- **TagPropWidget** (QWidget \*const parent)

## 9.1286 Digikam::TagRegion Class Reference

### Public Types

- enum **Type** { **Invalid** , **Rect** }

### Public Member Functions

- **TagRegion** ()=default  
*Use this small class to convert between the formatted textual representation of a tag region in the database and the corresponding object.*
- **TagRegion** (const QRect &rect)  
*Construct with the region.*
- **TagRegion** (const QString &descriptor)  
*Construct with the textual descriptor.*
- bool **intersects** (const **TagRegion** &other, double fraction=0)  
*Returns true if this and the other region intersect.*
- bool **isValid** () const
- bool **operator!=** (const **TagRegion** &other) const
- bool **operator==** (const **TagRegion** &other) const
- QRect **toRect** () const  
*If type is Rect, returns the contained rectangle.*
- QVariant **toVariant** () const  
*Stores in / loads from a variant.*
- QString **toXml** () const  
*Returns an XML textual representation of this region.*
- Type **type** () const

## Static Public Member Functions

- static QRectF [absoluteToRelative](#) (const QRect &region, const QSize &fullSize)  
*Takes absolute region and full size to return the original relative region.*
- static QSize [adjustToOrientation](#) (QRect &region, int orientation, const QSize &fullSize)  
*Rotate and flip region to [MetaEngine::ImageOrientation](#).*
- static [TagRegion fromVariant](#) (const QVariant &var)
- static QRect [mapFromOriginalSize](#) (const [DImg](#) &reducedSizeImage, const QRect &fullSizeDetail)
- static QRect [mapFromOriginalSize](#) (const QSize &fullImageSize, const QSize &reducedImageSize, const QRect &fullSizeDetail)
- static QRect [mapToOriginalSize](#) (const [DImg](#) &reducedSizeImage, const QRect &reducedSizeDetail)  
*Takes the original and reduced size from the [DImg](#).*
- static QRect [mapToOriginalSize](#) (const QSize &fullImageSize, const QSize &reducedImageSize, const QRect &reducedSizeDetail)  
*Converts detail rectangles taken from a reduced size image to the original size, and vice versa.*
- static QRect [relativeToAbsolute](#) (const QRectF &region, const [DImg](#) &reducedSizeImage)  
*Takes the original and reduced size from the [DImg](#), maps to original size.*
- static QRect [relativeToAbsolute](#) (const QRectF &region, const QSize &fullSize)  
*Takes a relative region and a full size and returns the absolute region.*
- static void [reverseToOrientation](#) (QRect &region, int orientation, const QSize &fullSize)  
*Reverse rotate and flip region to [MetaEngine::ImageOrientation](#).*

## Protected Attributes

- Type `m_type` = Invalid
- QVariant `m_value`

## 9.1286.1 Constructor & Destructor Documentation

### 9.1286.1.1 TagRegion()

```
Digikam::TagRegion::TagRegion ( ) [default]
```

Construct an invalid region.

## 9.1286.2 Member Function Documentation

### 9.1286.2.1 absoluteToRelative()

```
QRectF Digikam::TagRegion::absoluteToRelative (
    const QRect & region,
    const QSize & fullSize ) [static]
```

Used to write back rectangles into image's XMP. see [MetadataHub::write](#).

### 9.1286.2.2 adjustToOrientation()

```
QSize Digikam::TagRegion::adjustToOrientation (
    QRect & region,
    int orientation,
    const QSize & fullSize ) [static]
```

The value region are calculated for the new image orientation.

### 9.1286.2.3 intersects()

```
bool Digikam::TagRegion::intersects (
    const TagRegion & other,
    double fraction = 0 )
```

*fraction* describes the relative overlap area needed to return true: If *fraction* is 0, returns true if the regions intersect at all. If *fraction* is 1, returns true only if *other* is completely contained in this region. If *fraction* is  $x$ ,  $0 < x < 1$ , returns true if the area of this region covered by the other is greater than  $x$ . Invalid areas never intersect.

### 9.1286.2.4 reverseToOrientation()

```
void Digikam::TagRegion::reverseToOrientation (
    QRect & region,
    int orientation,
    const QSize & fullSize ) [static]
```

The value region are calculated for the new image orientation.

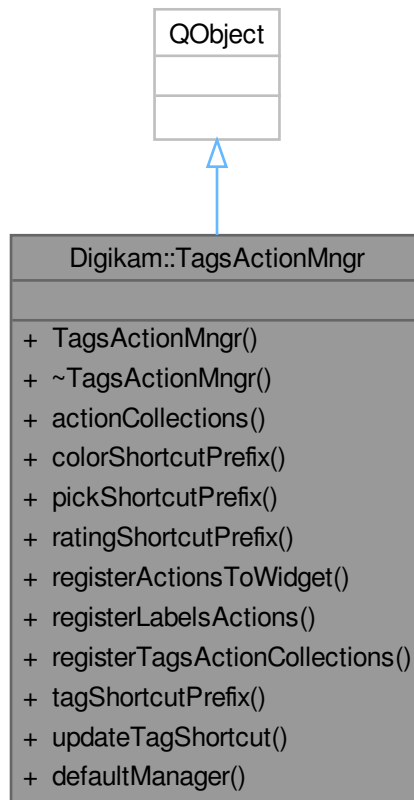
### 9.1286.2.5 toVariant()

```
QVariant Digikam::TagRegion::toVariant ( ) const
```

Will only use native QVariant types.

## 9.1287 Digikam::TagsActionMngr Class Reference

Inheritance diagram for Digikam::TagsActionMngr:



### Signals

- void **signalShortcutPressed** (const QString &shortcut, int val)

### Public Member Functions

- **TagsActionMngr** (QWidget \*const parent)
- QList< KActionCollection \* > **actionCollections** () const  
*Return the list of whole action collections managed.*
- QString **colorShortcutPrefix** () const
- QString **pickShortcutPrefix** () const
- QString **ratingShortcutPrefix** () const
- void **registerActionsToWidget** (QWidget \*const wdg)
- void **registerLabelsActions** (KActionCollection \*const ac)  
*Register all labels actions to collections managed with keyboard shortcuts.*
- void **registerTagsActionCollections** ()  
*Register all tag actions to collections managed with keyboard shortcuts.*
- QString **tagShortcutPrefix** () const
- void **updateTagShortcut** (int tagId, const QKeySequence &ks, bool delAction=true)  
*Updates the shortcut action for a tag.*



## Static Public Member Functions

- static [TagsActionMngr](#) \* **defaultManager** ()

## 9.1287.1 Member Function Documentation

### 9.1287.1.1 registerLabelsActions()

```
void Digikam::TagsActionMngr::registerLabelsActions (
    KActionCollection *const ac )
```

Unlike tags actions, labels shortcuts are stored in XML GUI file of each root windows, to be able to customize it through KDE keyboards shortcuts config panel. This method must be called before to `DXmlGuiWindow::createGui()`, typically when window actions are registered to `ActionCollection` instance.

### 9.1287.1.2 registerTagsActionCollections()

```
void Digikam::TagsActionMngr::registerTagsActionCollections ( )
```

Because Tags shortcuts are stored in database this method must be called after database initialization and after that all root window instances have been created.

### 9.1287.1.3 updateTagShortcut()

```
void Digikam::TagsActionMngr::updateTagShortcut (
    int tagId,
    const QKeySequence & ks,
    bool delAction = true )
```

Call this when a shortcut was added, removed or changed.

## 9.1288 Digikam::TagsCache Class Reference

Inheritance diagram for Digikam::TagsCache:



### Public Types

- enum `HiddenTagsPolicy` { `NoHiddenTags` , `IncludeHiddenTags` }
- enum `LeadingSlashPolicy` { `NoLeadingSlash` , `IncludeLeadingSlash` }

### Signals

- void `tagAboutToBeDeleted` (QString name)
- void `tagAdded` (int tagId)

*These signals are provided for convenience; for finer grained information use [CoreDbWatch](#).*

- void `tagDeleted` (int tagId)

## Public Member Functions

- bool [canBeWrittenToMetadata](#) (int tagId) const  
*Returns if a tag shall be written to the metadata of a file.*
- int [colorLabelForTag](#) (int tagId)  
*Return color label id corresponding of internal tags ID.*
- int [colorLabelFromTags](#) (const QList< int > &tagIds)  
*From the given list of tags, returns the color label corresponding to the first encountered tag which is a color label tag.*
- QVector< int > [colorLabelTags](#) ()  
*Returns all color label tags, where index is the label id and value the tag id.*
- bool [containsPublicTags](#) (const QList< int > &tagIds) const  
*Returns true if the given list of tag ids contains at least one non-internal tag.*
- int [createTag](#) (const QString &tagPathToCreate)  
*Add the tag described by the given tag path, and all missing parent tags, to the database.*
- QList< int > [createTags](#) (const QStringList &tagPaths)
- int [getOrCreateInternalTag](#) (const QString &tagName)  
*For the given tag name (not path!), find the existing tag or creates a new internal tags under the usual tag path used for internal tags.*
- int [getOrCreateTag](#) (const QString &tagPath)  
*A combination of tagForPath and createTag: Finds ids for the given tagPaths.*
- QList< int > [getOrCreateTags](#) (const QStringList &tagPaths)
- int [getOrCreateTagWithProperty](#) (const QString &tagPath, const QString &property, const QString &value=QString())  
*Calls getOrCreateTag for the given path, and ensures that the tag has assigned the given property.*
- bool [hasProperty](#) (int tagId, const QString &property, const QString &value=QString()) const  
*Tests if the tag has the given property: a) just has the property.*
- bool [hasTag](#) (int id) const  
*Returns true if the tag for the given id exists.*
- bool [isInternalTag](#) (int tagId) const  
*Returns if a tag is to be regarded program-internal, that is, a technical implementation detail not visible to the user at any time.*
- int [parentTag](#) (int id) const  
*Returns the parent tag id, or 0 if a toplevel tag or tag does not exist.*
- QList< int > [parentTags](#) (int id) const  
*Returns the parent tag ids of the given tag, starting with the toplevel tag, ending with the direct parent tag.*
- int [pickLabelForTag](#) (int tagId)  
*Return pick label id corresponding of internal tags ID.*
- int [pickLabelFromTags](#) (const QList< int > &tagIds)  
*From the given list of tags, returns the pick label corresponding to the first encountered tag which is a pick label tag.*
- QVector< int > [pickLabelTags](#) ()  
*Returns all pick label tags, where index is the label id and value the tag id.*
- QMap< QString, QString > [properties](#) (int tagId) const  
*Returns a list or a map of the properties of the tag.*
- QString [propertyValue](#) (int tagId, const QString &property) const  
*Returns the value of the property.*
- QStringList [propertyValues](#) (int tagId, const QString &property) const
- QList< int > [publicTags](#) (const QList< int > &tagIds) const  
*From the given list of tag ids, filter out any internal tags and return only public tags.*
- QStringList [shortenedTagPaths](#) (const QList< int > &ids, [LeadingSlashPolicy](#) slashPolicy=[IncludeLeadingSlash](#), [HiddenTagsPolicy](#) hiddenTagsPolicy=[IncludeHiddenTags](#)) const  
*Utility method.*

- QStringList **shortenedTagPaths** (const QList< int > &ids, QList< int > \*sortedIds, [LeadingSlashPolicy](#) slashPolicy=[IncludeLeadingSlash](#), HiddenTagsPolicy hiddenTagsPolicy=[IncludeHiddenTags](#)) const
- int [tagForColorLabel](#) (int label)
  - Return internal tags ID corresponding of color label id.*
- int [tagForName](#) (const QString &tagName, int parentId=0) const
  - Returns the id of the tag with the given name and parent tag.*
- int [tagForPath](#) (const QString &path) const
  - Returns the tag matched exactly by the given path.*
- int [tagForPickLabel](#) (int label)
  - Return internal tags ID corresponding of pick label id.*
- QString [tagName](#) (int id) const
  - Returns the name of the tag with the given id.*
- QStringList **tagNames** (const QList< int > &ids, HiddenTagsPolicy hiddenTagsPolicy=[IncludeHiddenTags](#)) const
- QString [tagPath](#) (int id, [LeadingSlashPolicy](#) slashPolicy=[IncludeLeadingSlash](#)) const
  - Returns the path of the tag with the given id.*
- QStringList **tagPaths** (const QList< int > &ids, [LeadingSlashPolicy](#) slashPolicy=[IncludeLeadingSlash](#), HiddenTagsPolicy hiddenTagsPolicy=[IncludeHiddenTags](#)) const
- QList< int > **tagsContaining** (const QString &fragment, Qt::CaseSensitivity caseSensitivity=Qt::Case↔Insensitive, HiddenTagsPolicy hiddenTagsPolicy=[NoHiddenTags](#))
  - Returns a list of tag ids whose tag name (not path) starts with / contains the given fragment.*
- QList< int > [tagsForName](#) (const QString &tagName, HiddenTagsPolicy hiddenTagsPolicy=[NoHiddenTags](#)) const
  - Finds all tags with the given name.*
- QList< int > **tagsForPaths** (const QStringList &tagPaths) const
- QList< int > **tagsStartingWith** (const QString &begin, Qt::CaseSensitivity caseSensitivity=Qt::Case↔Insensitive, HiddenTagsPolicy hiddenTagsPolicy=[NoHiddenTags](#))
- QList< int > [tagsWithProperty](#) (const QString &property, const QString &value=QString()) const
  - Finds all tags with the given property.*
- QList< int > [tagsWithPropertyCached](#) (const QString &property) const
  - This method is equivalent to calling tagsWithProperty(property), but the immediate result will be cached for subsequent calls.*

### Static Public Member Functions

- static [TagsCache](#) \* **instance** ()
- static QLatin1String **propertyNameDigikamInternalTag** ()
- static QLatin1String **propertyNameExcludedFromWriting** ()
- static QLatin1String **tagPathOfDigikamInternalTags** ([LeadingSlashPolicy](#) slashPolicy=[IncludeLeadingSlash](#))

### Friends

- class **ChangingDB**
- class **CoreDbAccess**
- class **TagsCacheCreator**

## 9.1288.1 Member Enumeration Documentation

### 9.1288.1.1 LeadingSlashPolicy

enum [Digikam::TagsCache::LeadingSlashPolicy](#)

## Enumerator

NoLeadingSlash	Ex: "Places/Cities/Paris".
IncludeLeadingSlash	Ex: "/Places/Cities/Paris".

## 9.1288.2 Member Function Documentation

### 9.1288.2.1 canBeWrittenToMetadata()

```
bool Digikam::TagsCache::canBeWrittenToMetadata (
    int tagId ) const
```

Always returns false if the tag is a program-internal tag.

### 9.1288.2.2 colorLabelForTag()

```
int Digikam::TagsCache::colorLabelForTag (
    int tagId )
```

see ColorLabel values from globals.h. Return -1 if not it's found.

### 9.1288.2.3 colorLabelFromTags()

```
int Digikam::TagsCache::colorLabelFromTags (
    const QList< int > & tagIds )
```

Returns -1 if no tag in the list is a color label tag.

### 9.1288.2.4 createTag()

```
int Digikam::TagsCache::createTag (
    const QString & tagPathToCreate )
```

Returns the tag id. Use this if you know that tag path does not exist. If you are unsure, use getOrCreateTag.

### 9.1288.2.5 getOrCreateTag()

```
int Digikam::TagsCache::getOrCreateTag (
    const QString & tagPath )
```

If a tag does not exist yet and create is true, it will be created. Otherwise the id 0 is returned for this path.

### 9.1288.2.6 getOrCreateTagWithProperty()

```
int Digikam::TagsCache::getOrCreateTagWithProperty (
    const QString & tagPath,
    const QString & property,
    const QString & value = QString() )
```

If you pass a null string as value, then the value is not checked and not changed.

### 9.1288.2.7 hasProperty()

```
bool Digikam::TagsCache::hasProperty (
    int tagId,
    const QString & property,
    const QString & value = QString() ) const
```

b) has the property with the given value (value not null).

### 9.1288.2.8 parentTags()

```
QList< int > Digikam::TagsCache::parentTags (
    int id ) const
```

If the tag is a toplevel tag or does not exist, an empty list is returned.

### 9.1288.2.9 pickLabelForTag()

```
int Digikam::TagsCache::pickLabelForTag (
    int tagId )
```

see PickLabel values from globals.h. Return -1 if not it's found.

### 9.1288.2.10 pickLabelFromTags()

```
int Digikam::TagsCache::pickLabelFromTags (
    const QList< int > & tagIds )
```

Returns -1 if no tag in the list is a pick label tag.

### 9.1288.2.11 properties()

```
QMap< QString, QString > Digikam::TagsCache::properties (
    int tagId ) const
```

Note: The list and map may be constructed for each call. Prefer [hasProperty\(\)](#) and [property\(\)](#).

### 9.1288.2.12 `propertyValue()`

```
QString Digikam::TagsCache::propertyValue (
    int tagId,
    const QString & property ) const
```

Returning a null string cannot distinguish between the property set with a null value, or the property not set. The first method returns any property, if multiple are set with the same key.

### 9.1288.2.13 `shortenedTagPaths()`

```
QStringList Digikam::TagsCache::shortenedTagPaths (
    const QList< int > & ids,
    LeadingSlashPolicy slashPolicy = IncludeLeadingSlash,
    HiddenTagsPolicy hiddenTagsPolicy = IncludeHiddenTags ) const
```

Orders the given tag paths. If tags begin with the same path (parent tags), the relevant part is cut off in the second line. The second variant allows you to pass a list as return parameter. This list will contain, upon return, the tag id corresponding to each tag in the returned, sorted list of shortened paths.

### 9.1288.2.14 `tagAdded`

```
void Digikam::TagsCache::tagAdded (
    int tagId ) [signal]
```

Use a queued connection if you carry out longer operations from slots connected to these signals.

### 9.1288.2.15 `tagForColorLabel()`

```
int Digikam::TagsCache::tagForColorLabel (
    int label )
```

see `ColorLabel` values from `globals.h`. Return 0 if not it's found.

### 9.1288.2.16 `tagForName()`

```
int Digikam::TagsCache::tagForName (
    const QString & tagName,
    int parentId = 0 ) const
```

If `parentId` is 0, the tag is a toplevel tag. Returns 0 if there is no such tag.

### 9.1288.2.17 `tagForPath()`

```
int Digikam::TagsCache::tagForPath (
    const QString & path ) const
```

The path can be given with or without leading slash. Returns 0 if there is no such tag, or if path is empty. If you want to create the tag if it does not yet exist, use `getOrCreateTag`.

### 9.1288.2.18 tagForPickLabel()

```
int Digikam::TagsCache::tagForPickLabel (
    int label )
```

see PickLabel values from globals.h. Return 0 if not it's found.

### 9.1288.2.19 tagName()

```
QString Digikam::TagsCache::tagName (
    int id ) const
```

For the tag Places/Cities/Paris, this is Paris. If there is no tag for the given id a null string is returned.

### 9.1288.2.20 tagPath()

```
QString Digikam::TagsCache::tagPath (
    int id,
    LeadingSlashPolicy slashPolicy = IncludeLeadingSlash ) const
```

For the tag Places/Cities/Paris, this is Places/Cities/Paris. If there is no tag for the given id a null string is returned.

### 9.1288.2.21 tagsForName()

```
QList< int > Digikam::TagsCache::tagsForName (
    const QString & tagName,
    HiddenTagsPolicy hiddenTagsPolicy = NoHiddenTags ) const
```

For "Paris", this may give "Places/Cities/Paris" and "Places/USA/Texas/Paris". If there is no tag with the given name at all, returns an empty list.

### 9.1288.2.22 tagsWithProperty()

```
QList< int > Digikam::TagsCache::tagsWithProperty (
    const QString & property,
    const QString & value = QString() ) const
```

The tag: a) just has the property. b) has the property with the given value (value not null). Note: The returned list is sorted.

### 9.1288.2.23 tagsWithPropertyCached()

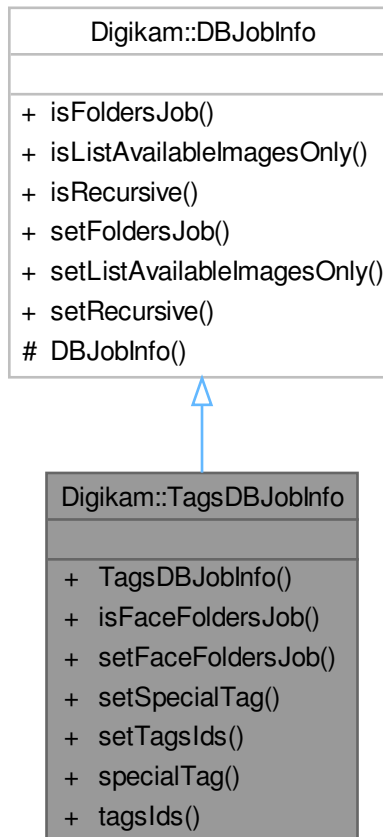
```
QList< int > Digikam::TagsCache::tagsWithPropertyCached (
    const QString & property ) const
```

Use it for queries for which you know that they will be issued very often, so that it's worth caching the result of the already pretty fast [tagsWithProperty\(\)](#).



## 9.1289 Digikam::TagsDBJobInfo Class Reference

Inheritance diagram for Digikam::TagsDBJobInfo:



### Public Member Functions

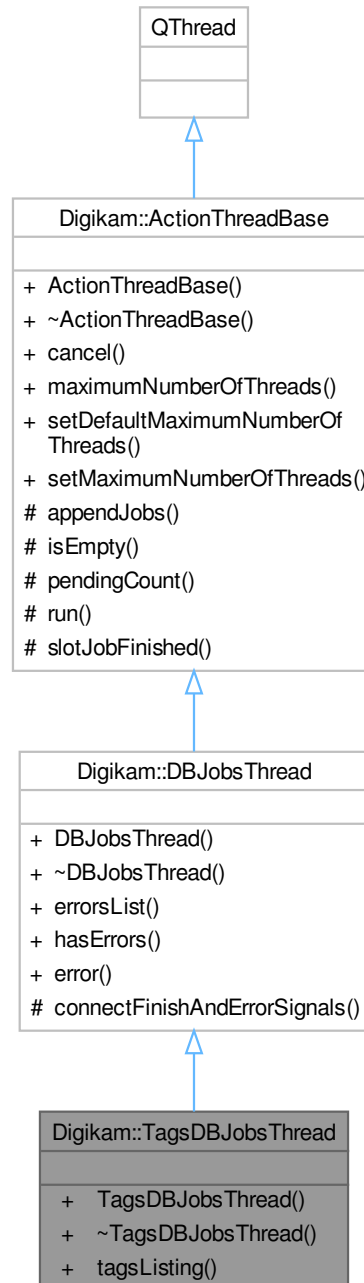
- bool **isFaceFoldersJob** () const
- void **setFaceFoldersJob** ()
- void **setSpecialTag** (const QString &tag)
- void **setTagsIds** (const QList< int > &tagsIds)
- QString **specialTag** () const
- QList< int > **tagsIds** () const

### Public Member Functions inherited from [Digikam::DBJobInfo](#)

- bool **isFoldersJob** () const
- bool **isListAvailableImagesOnly** () const
- bool **isRecursive** () const
- void **setFoldersJob** ()
- void **setListAvailableImagesOnly** ()
- void **setRecursive** ()

## 9.1290 Digikam::TagsDBJobsThread Class Reference

Inheritance diagram for Digikam::TagsDBJobsThread:



### Signals

- void **faceFoldersData** (const QMap< QString, QHash< int, int > > &)
- void **foldersData** (const QHash< int, int > &)

## Signals inherited from [Digikam::DBJobsThread](#)

- void **data** (const QList< [ItemLISTERRecord](#) > &records)
- void **finished** ()

## Public Member Functions

- **TagsDBJobsThread** (QObject \*const parent)
- void **tagsListing** (const [TagsDBJobInfo](#) &info)  
*Starts tags listing and scanning job(s)*

## Public Member Functions inherited from [Digikam::DBJobsThread](#)

- **DBJobsThread** (QObject \*const parent)
- QList< QString > & **errorsList** ()  
*A method to get all errors reported from jobs.*
- bool **hasErrors** ()  
*hasErrors: a method to check for jobs errors*

## Public Member Functions inherited from [Digikam::ActionThreadBase](#)

- **ActionThreadBase** (QObject \*const parent=nullptr)
- void **cancel** (bool isCancel=true)  
*Cancel processing of current jobs under progress.*
- int **maximumNumberOfThreads** () const  
*Return the maximum number of threads used to parallelize collection of job processing.*
- void **setDefaultMaximumNumberOfThreads** ()  
*Reset maximum number of threads used to parallelize collection of job processing to max core detected on computer.*
- void **setMaximumNumberOfThreads** (int n)  
*Adjust maximum number of threads used to parallelize collection of job processing.*

## Additional Inherited Members

## Public Slots inherited from [Digikam::DBJobsThread](#)

- void **error** (const QString &errString)  
*Appends the error string to m\_errorsList.*

## Protected Slots inherited from [Digikam::ActionThreadBase](#)

- void **slotJobFinished** ()

## Protected Member Functions inherited from [Digikam::DBJobsThread](#)

- void **connectFinishAndErrorSignals** (DBJob \*const j)  
*Connects the signals of job to the signals of the thread.*

## Protected Member Functions inherited from [Digikam::ActionThreadBase](#)

- void [appendJobs](#) (const [ActionJobCollection](#) &jobs)  
*Append a collection of jobs to process into QThreadPool.*
- bool [isEmpty](#) () const  
*Return true if list of pending jobs to process is empty.*
- int [pendingCount](#) () const  
*Return the number of pending jobs to process.*
- void [run](#) () override  
*Main thread loop used to process jobs in todo list.*

### 9.1290.1 Member Function Documentation

#### 9.1290.1.1 [tagsListing\(\)](#)

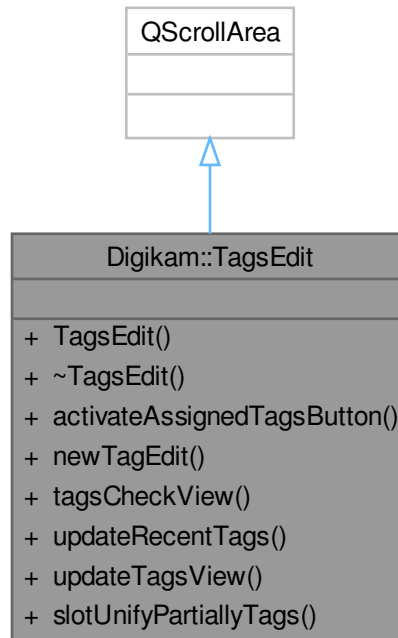
```
void Digikam::TagsDBJobsThread::tagsListing (  
    const TagsDBJobInfo & info )
```

##### Parameters

<i>info</i>	represents the tags job info
-------------	------------------------------

## 9.1291 Digikam::TagsEdit Class Reference

Inheritance diagram for Digikam::TagsEdit:



### Public Slots

- void `slotUnifyPartiallyTags ()`

### Signals

- void `signalChanged ()`
- void `signalImageTagsChanged (qulonglong imageId)`

### Public Member Functions

- `TagsEdit (DisjointMetadata *const hub, QWidget *const parent)`
- void `activateAssignedTagsButton ()`
- `AddTagsLineEdit * newTagEdit () const`
- `TagCheckView * tagsCheckView () const`
- void `updateRecentTags ()`
- void `updateTagsView ()`

## 9.1292 Digikam::TagShortInfo Class Reference

### Public Member Functions

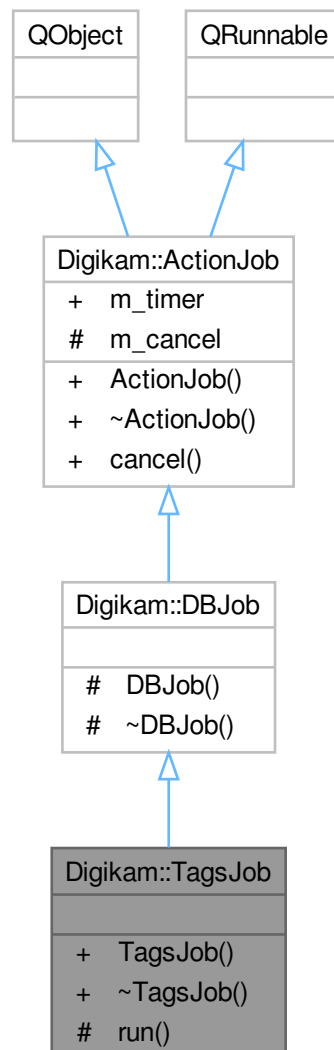
- bool **isNull** () const

### Public Attributes

- int **id** = 0
- QString **name**
- int **pid** = 0

## 9.1293 Digikam::TagsJob Class Reference

Inheritance diagram for Digikam::TagsJob:



## Signals

- void **faceFoldersData** (const QMap< QString, QHash< int, int > > &data)
- void **foldersData** (const QHash< int, int > &data)

## Signals inherited from [Digikam::DBJob](#)

- void **data** (const QList< [ItemListerRecord](#) > &records)
- void **error** (const QString &err)

## Signals inherited from [Digikam::ActionJob](#)

- void **signalDone** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job is done.*
- void **signalProgress** (int)  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job progress.*
- void **signalStarted** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager that job is started.*

## Public Member Functions

- **TagsJob** (const [TagsDBJobInfo](#) &jobInfo)

## Public Member Functions inherited from [Digikam::ActionJob](#)

- **ActionJob** (QObject \*const parent=nullptr)  
*Constructor which delegate deletion of QRunnable instance to [ActionThreadBase](#), not QThreadPool.*
- **~ActionJob** () override  
*Re-implement destructor in you implementation.*

## Protected Member Functions

- void **run** () override

## Additional Inherited Members

## Public Slots inherited from [Digikam::ActionJob](#)

- void **cancel** ()  
*Call this method to cancel job.*

## Public Attributes inherited from [Digikam::ActionJob](#)

- QElapsedTimer **m\_timer**  
*Timer to determine the running time of the job.*

## Protected Attributes inherited from [Digikam::ActionJob](#)

- bool `m_cancel` = false

*You can use this boolean in your implementation to know if job must be canceled.*

## 9.1294 Digikam::TagsLineEditOverlay Class Reference

Inheritance diagram for Digikam::TagsLineEditOverlay:





## Signals

- void **tagEdited** (const QModelIndex &index, const QString &)
- void **tagEdited** (const QModelIndex &index, int rating)

## Signals inherited from Digikam::ItemDelegateOverlay

- void **hideNotification** ()
- void **requestNotification** (const QModelIndex &index, const QString &message)
- void **update** (const QModelIndex &index)

## Public Member Functions

- **TagsLineEditOverlay** (QObject \*const parent)
- [AddTagsLineEdit](#) \* **addTagsLineEdit** () const

## Public Member Functions inherited from Digikam::AbstractWidgetDelegateOverlay

- [AbstractWidgetDelegateOverlay](#) (QObject \*const parent)  
*This class provides functionality for using a widget in an overlay.*

## Public Member Functions inherited from Digikam::ItemDelegateOverlay

- **ItemDelegateOverlay** (QObject \*const parent=nullptr)
- virtual bool **acceptsDelegate** (QAbstractItemDelegate \*) const
- QAbstractItemDelegate \* **delegate** () const
- virtual void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)  
*Only these two methods are implemented as virtual methods.*
- virtual void **paint** (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index)
- void **setDelegate** (QAbstractItemDelegate \*delegate)
- void **setView** (QAbstractItemView \*view)
- QAbstractItemView \* **view** () const

## Protected Slots

- void **slotDataChanged** (const QModelIndex &, const QModelIndex &)
- void **slotTagChanged** (const QString &)
- void **slotTagChanged** (int)

## Protected Slots inherited from Digikam::AbstractWidgetDelegateOverlay

- virtual void **slotLayoutChanged** ()
- virtual void **slotReset** ()  
*Default implementations of these three slots call `hide()`*
- virtual void **slotRowsRemoved** (const QModelIndex &parent, int start, int end)
- virtual void **slotViewportEntered** ()

## Protected Slots inherited from [Digikam::ItemDelegateOverlay](#)

### Protected Member Functions

- QWidget \* [createWidget](#) () override  
*Create your widget here.*
- void [hide](#) () override  
*Called when the widget shall be hidden (mouse cursor left index, viewport, uninstalled etc.).*
- void [setActive](#) (bool) override  
*If active is true, this will call [createWidget\(\)](#), initialize the widget for use, and setup connections for the virtual slots.*
- void [slotEntered](#) (const QModelIndex &index) override  
*Default implementation shows the widget iff the index is valid and [checkIndex](#) returns true.*
- void [updatePosition](#) ()
- void [updateTag](#) ()
- void [visualChange](#) () override  
*Called when any change from the delegate occurs - when the overlay is installed, when size hints, styles or fonts change.*

## Protected Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- virtual bool [checkIndex](#) (const QModelIndex &index) const  
*Return true here if you want to show the overlay for the given index.*
- bool [checkIndexOnEnter](#) (const QModelIndex &index) const  
*Utility method called from [slotEntered](#).*
- bool [eventFilter](#) (QObject \*obj, QEvent \*event) override
- virtual QString [notifyMultipleMessage](#) (const QModelIndex &, int number)
- QWidget \* [parentWidget](#) () const  
*Returns the widget to be used as parent for your widget created in [createWidget\(\)](#)*
- virtual void [viewportLeaveEvent](#) (QObject \*obj, QEvent \*event)  
*Called when a [QEvent::Leave](#) of the viewport is received.*
- virtual void [widgetEnterEvent](#) ()  
*Called when a [QEvent::Enter](#) resp.*
- void [widgetEnterNotifyMultiple](#) (const QModelIndex &index)  
*A sample implementation for above methods.*
- virtual void [widgetLeaveEvent](#) ()
- void [widgetLeaveNotifyMultiple](#) ()

## Protected Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- QList< QModelIndex > [affectedIndexes](#) (const QModelIndex &index) const
- bool [affectsMultiple](#) (const QModelIndex &index) const  
*For the context that an overlay can affect multiple items: Assuming the currently overlaid index is given.*
- int [numberOfAffectedIndexes](#) (const QModelIndex &index) const
- bool [viewHasMultiSelection](#) () const  
*Utility method.*

### Protected Attributes

- QPersistentModelIndex [m\\_index](#)

## Protected Attributes inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- bool `m_mouseButtonPressedOnWidget` = false
- `QWidget * m_widget` = nullptr

## Protected Attributes inherited from [Digikam::ItemDelegateOverlay](#)

- `QAbstractItemDelegate * m_delegate` = nullptr
- `QAbstractItemView * m_view` = nullptr

## 9.1294.1 Member Function Documentation

### 9.1294.1.1 `createWidget()`

```
QWidget * Digikam::TagsLineEditOverlay::createWidget ( ) [override], [protected], [virtual]
```

When creating the object, pass `parentWidget()` as parent widget. Ownership of the object is passed. It will be deleted in `setActive(false)`.

Implements [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.1294.1.2 `hide()`

```
void Digikam::TagsLineEditOverlay::hide ( ) [override], [protected], [virtual]
```

Default implementation `hide()`s `m_widget`.

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.1294.1.3 `setActive()`

```
void Digikam::TagsLineEditOverlay::setActive (
    bool active ) [override], [protected], [virtual]
```

If active is false, this will delete the widget and disconnect all signal from model and view to this object (!)

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.1294.1.4 `slotEntered()`

```
void Digikam::TagsLineEditOverlay::slotEntered (
    const QModelIndex & index ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

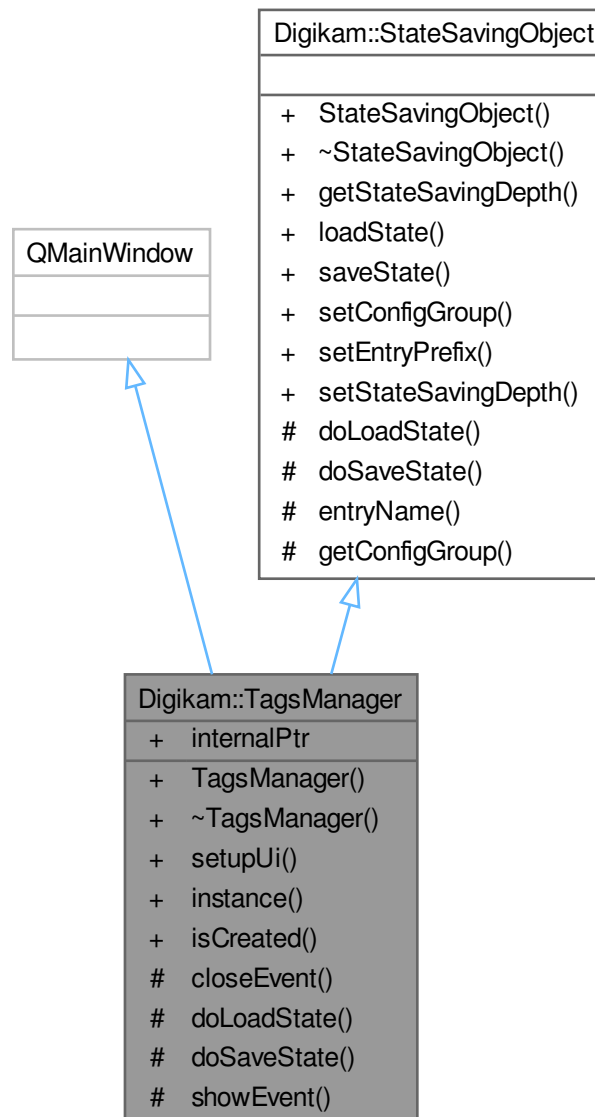
### 9.1294.1.5 visualChange()

```
void Digikam::TagsLineEditOverlay::visualChange ( ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ItemDelegateOverlay](#).

## 9.1295 Digikam::TagsManager Class Reference

Inheritance diagram for Digikam::TagsManager:



### Signals

- void **signalSelectionChanged** ([TAAlbum](#) \*album)

## Public Member Functions

- void **setupUi** ()  
*setupUi setup all gui elements for Tag Manager*

## Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (QObject \*const host)  
*Constructor.*
- virtual **~StateSavingObject** ()  
*Destructor.*
- [StateSavingDepth](#) **getStateSavingDepth** () const  
*Returns the depth used for state saving or loading.*
- void **loadState** ()  
*Invokes loading the class' state.*
- void **saveState** ()  
*Invokes saving the class' state.*
- virtual void **setConfigGroup** (const KConfigGroup &group)  
*Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void **setEntryPrefix** (const QString &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void **setStateSavingDepth** (const [StateSavingDepth](#) depth)  
*Sets the depth used for state saving or loading.*

## Static Public Member Functions

- static [TagsManager](#) \* **instance** ()
- static bool **isCreated** ()

## Static Public Attributes

- static QPointer< [TagsManager](#) > **internalPtr** = QPointer<[TagsManager](#)>()

## Protected Member Functions

- void **closeEvent** (QCloseEvent \*event) override
- void **doLoadState** () override  
*Implement this hook method for state loading.*
- void **doSaveState** () override  
*Implement this hook method for state saving.*
- void **showEvent** (QShowEvent \*event) override

## Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- QString **entryName** (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup **getConfigGroup** () const  
*Returns the config group that must be used for state saving and loading.*

## Additional Inherited Members

### Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }

*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

## 9.1295.1 Member Function Documentation

### 9.1295.1.1 doLoadState()

```
void Digikam::TagsManager::doLoadState ( ) [override], [protected], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

### 9.1295.1.2 doSaveState()

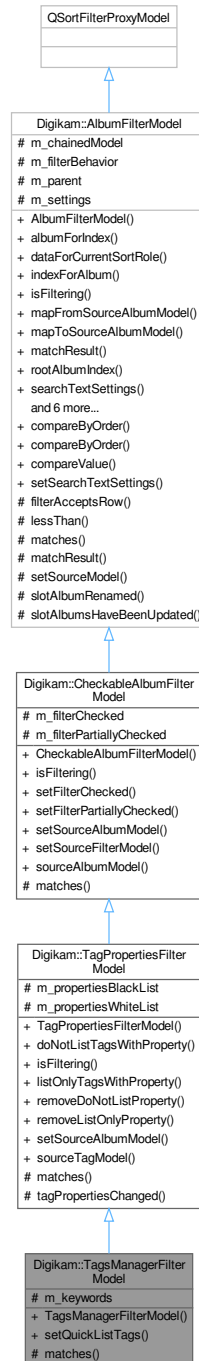
```
void Digikam::TagsManager::doSaveState ( ) [override], [protected], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

## 9.1296 Digikam::TagsManagerFilterModel Class Reference

Inheritance diagram for Digikam::TagsManagerFilterModel:



### Public Member Functions

- **TagsManagerFilterModel** (QObject \*const data=nullptr)
- void **setQuickListTags** (const QList< int > &tags)

## Public Member Functions inherited from [Digikam::TagPropertiesFilterModel](#)

- **TagPropertiesFilterModel** (QObject \*const parent=nullptr)
- void **doNotListTagsWithProperty** (const QString &property)
- bool **isFiltering** () const override
 

*Returns if the currently applied filters will result in any filtering.*
- void **listOnlyTagsWithProperty** (const QString &property)
- void **removeDoNotListProperty** (const QString &property)
- void **removeListOnlyProperty** (const QString &property)
- void **setSourceAlbumModel** ([TagModel](#) \*const source)
- [TagModel](#) \* **sourceTagModel** () const

## Public Member Functions inherited from [Digikam::CheckableAlbumFilterModel](#)

- **CheckableAlbumFilterModel** (QObject \*const parent=nullptr)
- void **setFilterChecked** (bool filter)
- void **setFilterPartiallyChecked** (bool filter)
- void **setSourceAlbumModel** ([AbstractCheckableAlbumModel](#) \*const source)
- void **setSourceFilterModel** ([CheckableAlbumFilterModel](#) \*const source)
- [AbstractCheckableAlbumModel](#) \* **sourceAlbumModel** () const

## Public Member Functions inherited from [Digikam::AlbumFilterModel](#)

- **AlbumFilterModel** (QObject \*const parent=nullptr)
- [Album](#) \* **albumForIndex** (const QModelIndex &index) const
 

*Convenience methods.*
- QVariant **dataForCurrentSortRole** ([Album](#) \*album) const
- QModelIndex **indexForAlbum** ([Album](#) \*album) const
- QModelIndex **mapFromSourceAlbumModel** (const QModelIndex &index) const
- QModelIndex **mapToSourceAlbumModel** (const QModelIndex &index) const
- [MatchResult](#) **matchResult** (const QModelIndex &index) const
 

*Returns the MatchResult of an index of this model.*
- QModelIndex **rootAlbumIndex** () const
- [SearchTextSettings](#) **searchTextSettings** () const
 

*Returns the settings currently used for filtering.*
- void **setFilterBehavior** ([FilterBehavior](#) behavior)
 

*Sets the filter behavior.*
- void **setSourceAlbumModel** ([AbstractAlbumModel](#) \*const source)
 

*Sets the source model.*
- void **setSourceFilterModel** ([AlbumFilterModel](#) \*const source)
 

*Sets a chained filter model.*
- [AbstractAlbumModel](#) \* **sourceAlbumModel** () const
- [AlbumFilterModel](#) \* **sourceFilterModel** () const
- void **updateFilter** ()
 

*Force invalidateFilter() externally.*

## Protected Member Functions

- bool **matches** ([Album](#) \*album) const override
 

*This method provides the basic match checking algorithm.*



## Protected Member Functions inherited from [Digikam::AlbumFilterModel](#)

- bool **filterAcceptsRow** (int source\_row, const QModelIndex &source\_parent) const override
- bool **lessThan** (const QModelIndex &left, const QModelIndex &right) const override
- [MatchResult](#) **matchResult** ([Album](#) \*album) const  
*Returns if the filter matches this album (same logic as filterAcceptsRow).*
- void **setSourceModel** ([QAbstractItemModel](#) \*const model) override  
*Use setSourceAlbumModel.*

## Protected Attributes

- [QSet](#)< int > **m\_keywords**

## Protected Attributes inherited from [Digikam::TagPropertiesFilterModel](#)

- [QSet](#)< [QString](#) > **m\_propertiesBlackList**
- [QSet](#)< [QString](#) > **m\_propertiesWhiteList**

## Protected Attributes inherited from [Digikam::CheckableAlbumFilterModel](#)

- bool **m\_filterChecked** = false
- bool **m\_filterPartiallyChecked** = false

## Protected Attributes inherited from [Digikam::AlbumFilterModel](#)

- [QPointer](#)< [AlbumFilterModel](#) > **m\_chainedModel** = nullptr
- [FilterBehavior](#) **m\_filterBehavior** = [FullFiltering](#)
- [QObject](#) \* **m\_parent** = nullptr
- [SearchTextSettings](#) **m\_settings**

## Additional Inherited Members

## Public Types inherited from [Digikam::AlbumFilterModel](#)

- enum [FilterBehavior](#) { [SimpleFiltering](#) , [FullFiltering](#) , [StrictFiltering](#) }
- enum [MatchResult](#) {  
  [NoMatch](#) = 0 , [DirectMatch](#) , [ParentMatch](#) , [ChildMatch](#) ,  
  [SpecialMatch](#) }

## Public Slots inherited from [Digikam::AlbumFilterModel](#)

- void **setSearchTextSettings** (const [SearchTextSettings](#) &settings)  
*Accepts new settings used for filtering and applies them to the model.*

## Signals inherited from [Digikam::AlbumFilterModel](#)

- void [hasSearchResult](#) (bool hasResult)  
*Indicates whether the newly applied filter results in a search result or not.*
- void [searchTextSettingsAboutToChange](#) (bool searched, bool willSearch)  
*This signal indicates that a new [SearchTextSettings](#) arrived and is about to be applied to the model.*
- void [searchTextSettingsChanged](#) (bool wasSearching, bool searched)  
*Indicates that new search text settings were applied.*
- void **signalFilterChanged** ()  
*Indicates that a new filter was applied to the model.*

## Static Public Member Functions inherited from [Digikam::AlbumFilterModel](#)

- template<typename T >  
static int **compareByOrder** (const T &a, const T &b, Qt::SortOrder sortOrder)
- static int **compareByOrder** (int compareResult, Qt::SortOrder sortOrder)  
*Takes a typical result from a compare method (0 is equal, -1 is less than, 1 is greater than) and applies the given sort order to it.*
- template<typename T >  
static int **compareValue** (const T &a, const T &b)  
*Returns the usual compare result of -1, 0, or 1 for lessThan, equals and greaterThan.*

## Protected Slots inherited from [Digikam::TagPropertiesFilterModel](#)

- void **tagPropertiesChanged** ([TAlbum](#) \*)

## Protected Slots inherited from [Digikam::AlbumFilterModel](#)

- void **slotAlbumRenamed** ([Album](#) \*album)
- void **slotAlbumsHaveBeenUpdated** (int type)

## 9.1296.1 Member Function Documentation

### 9.1296.1.1 matches()

```
bool Digikam::TagsManagerFilterModel::matches (
    Album * album ) const [override], [protected], [virtual]
```

Return true if this single album matches the current criteria. This method can be overridden to provide custom filtering.

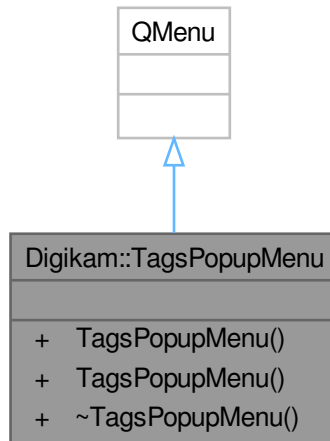
#### Parameters

<i>album</i>	the album to tell if it matches the filter criteria or not.
--------------	---

Reimplemented from [Digikam::TagPropertiesFilterModel](#).

## 9.1297 Digikam::TagsPopupMenu Class Reference

Inheritance diagram for Digikam::TagsPopupMenu:



### Public Types

- enum [Mode](#) { **ASSIGN** = 0 , **REMOVE** , **DISPLAY** , **RECENTLYASSIGNED** }

### Signals

- void **signalPopupMenuView** ()
- void **signalTagActivated** (int id)

### Public Member Functions

- **TagsPopupMenu** (const QList< qlonglong > &selectedImageIDs, [Mode](#) mode, QWidget \*const parent=nullptr)
- **TagsPopupMenu** (qlonglong selectedImageId, [Mode](#) mode, QWidget \*const parent=nullptr)

## 9.1297.1 Member Enumeration Documentation

### 9.1297.1.1 Mode

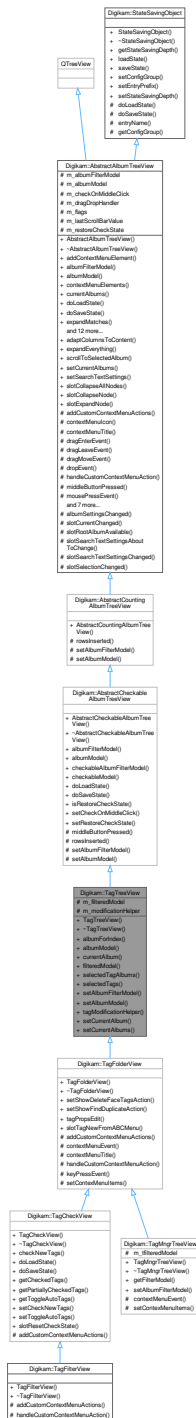
```
enum Digikam::TagsPopupMenu::Mode
```

#### Enumerator

DISPLAY	Used by "GoTo Tag" feature.
---------	-----------------------------

## 9.1298 Digikam::TagTreeView Class Reference

Inheritance diagram for Digikam::TagTreeView:



### Public Slots

- void **setCurrentAlbum** (int tagId, bool selectInAlbumManager=true)
- void **setCurrentAlbums** (const QList< Album \* > &tags, bool selectInAlbumManager=true)

## Public Slots inherited from [Digikam::AbstractAlbumTreeView](#)

- void **adaptColumnsToContent** ()  
*Adapt the column sizes to the contents of the tree view.*
- void **expandEverything** (const QModelIndex &index)  
*Expands the complete tree under the given index.*
- void **scrollToSelectedAlbum** ()  
*Scrolls to the first selected album if there is one.*
- void **setCurrentAlbums** (const QList< Album \* > &albums, bool selectInAlbumManager=true)  
*Selects the given album.*
- void **setSearchTextSettings** (const SearchTextSettings &settings)
- void **slotCollapseAllNodes** ()  
*slotCollapseAllNodes - collapse all nodes without root node*
- void **slotCollapseNode** ()  
*slotCollapseNode - collapse recursively selected nodes*
- void **slotExpandNode** ()  
*slotExpandNode - expands recursively selected nodes*

## Signals

- void **assignTags** (int tagId, const QList< int > &imageIds)

## Signals inherited from [Digikam::AbstractAlbumTreeView](#)

- void **currentAlbumChanged** (Album \*currentAlbum)  
*Emitted when the currently selected album changes.*
- void **selectedAlbumsChanged** (const QList< Album \* > &selectedAlbums)  
*Emitted when the current selection changes.*

## Public Member Functions

- **TagTreeView** (QWidget \*const parent=nullptr, Flags flags=DefaultFlags)
- [TAlbum](#) \* **albumForIndex** (const QModelIndex &index) const
- [TagModel](#) \* **albumModel** () const
- [TAlbum](#) \* **currentAlbum** () const  
*currentAlbum Even if multiple selection is enabled current Album can be only one, the last clicked item if you need selected items, see selectedAlbums() It's NOT the same as AlbumManager::currentAlbums()*
- [TagPropertiesFilterModel](#) \* **filteredModel** () const  
*Contains only the tags filtered by properties - prefer to albumModel()*
- QList< [TAlbum](#) \* > **selectedTagAlbums** ()
- QList< [Album](#) \* > **selectedTags** ()  
*selectedTags - return a list of all selected items in tag model*
- void **setAlbumFilterModel** ([TagPropertiesFilterModel](#) \*const filteredModel, [CheckableAlbumFilterModel](#) \*const filterModel)
- void **setAlbumModel** ([TagModel](#) \*const model)
- [TagModificationHelper](#) \* **tagModificationHelper** () const

## Public Member Functions inherited from [Digikam::AbstractCheckableAlbumTreeView](#)

- [AbstractCheckableAlbumTreeView](#) (QWidget \*const parent, Flags flags)
  - Models of these view can be checkable, they need not.*
- [CheckableAlbumFilterModel](#) \* [albumFilterModel](#) () const
- [AbstractCheckableAlbumModel](#) \* [albumModel](#) () const
  - Manage check state through the model directly.*
- [CheckableAlbumFilterModel](#) \* [checkableAlbumFilterModel](#) () const
- [AbstractCheckableAlbumModel](#) \* [checkableModel](#) () const
- void [doLoadState](#) () override
  - Implements state loading for the album tree view in a somewhat clumsy procedure because the model may not be fully loaded when this method is called.*
- void [doSaveState](#) () override
  - Implement this hook method for state saving.*
- bool [isRestoreCheckState](#) () const
  - Tells if the check state is restored while loading / saving state.*
- void [setCheckOnMiddleClick](#) (bool doThat)
  - Enable checking on middle mouse button click (default: on).*
- void [setRestoreCheckState](#) (bool restore)
  - Set whether to restore check state or not.*

## Public Member Functions inherited from [Digikam::AbstractCountingAlbumTreeView](#)

- [AbstractCountingAlbumTreeView](#) (QWidget \*const parent, Flags flags)

## Public Member Functions inherited from [Digikam::AbstractAlbumTreeView](#)

- [AbstractAlbumTreeView](#) (QWidget \*const parent, Flags flags)
  - Constructs an album tree view.*
- void [addContextMenuElement](#) ([ContextMenuElement](#) \*const element)
- [AlbumFilterModel](#) \* [albumFilterModel](#) () const
- [AbstractSpecificAlbumModel](#) \* [albumModel](#) () const
- QList< [ContextMenuElement](#) \* > [contextMenuElements](#) () const
- template<class A >
  - QList< A \* > [currentAlbums](#) ()
- bool [expandMatches](#) (const QModelIndex &index)
  - Ensures that every current match is visible by expanding all parent entries.*
- QModelIndex [indexVisuallyAt](#) (const QPoint &p)
  - This is a combination of [indexAt\(\)](#) checked with [visualRect\(\)](#).*
- void [removeContextMenuElement](#) ([ContextMenuElement](#) \*const element)
- QList< [Album](#) \* > [selectedItems](#) ()
  - [selectedItems\(\)](#) -*
- void [setAlbumManagerCurrentAlbum](#) (const bool setCurrentAlbum)
  - Some treeviews shall control the global current album kept by [AlbumManager](#).*
- void [setContextMenuIcon](#) (const QPixmap &pixmap)
  - Set the context menu title and icon.*
- void [setContextMenuTitle](#) (const QString &title)
- void [setEnableContextMenu](#) (const bool enable)
  - Determines the global decision to show a popup menu or not.*
- void [setExpandNewCurrentItem](#) (const bool doThat)
  - Expand an item when making it the new current item.*

- void **setExpandOnSingleClick** (const bool doThat)  
*Enable expanding of tree items on single click on the item (default: off)*
- void **setSelectAlbumOnClick** (const bool selectOnClick)  
*Sets whether to select an album on click via the album manager or not.*
- void **setSelectOnContextMenu** (const bool select)  
*Sets whether to select the album under the mouse cursor on a context menu request (so that the album is shown using the album manager) or not.*
- bool **viewportEvent** (QEvent \*event) override  
*For internal use only.*

## Public Member Functions inherited from Digikam::StateSavingObject

- **StateSavingObject** (QObject \*const host)  
*Constructor.*
- virtual **~StateSavingObject** ()  
*Destructor.*
- **StateSavingDepth** **getStateSavingDepth** () const  
*Returns the depth used for state saving or loading.*
- void **loadState** ()  
*Invokes loading the class' state.*
- void **saveState** ()  
*Invokes saving the class' state.*
- virtual void **setConfigGroup** (const KConfigGroup &group)  
*Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void **setEntryPrefix** (const QString &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void **setStateSavingDepth** (const **StateSavingDepth** depth)  
*Sets the depth used for state saving or loading.*

## Protected Attributes

- **TagPropertiesFilterModel** \* **m\_filteredModel** = nullptr
- **TagModificationHelper** \* **m\_modificationHelper** = nullptr

## Protected Attributes inherited from Digikam::AbstractAlbumTreeView

- **AlbumFilterModel** \* **m\_albumFilterModel** = nullptr
- **AbstractSpecificAlbumModel** \* **m\_albumModel** = nullptr
- bool **m\_checkOnMiddleClick** = false
- **AlbumModelDragDropHandler** \* **m\_dragDropHandler** = nullptr
- Flags **m\_flags** = DefaultFlags
- int **m\_lastScrollBarValue** = 0
- bool **m\_restoreCheckState** = false

## Additional Inherited Members

## Public Types inherited from Digikam::AbstractAlbumTreeView

- enum **Flag** {  
    **CreateDefaultModel**, **CreateDefaultFilterModel**, **CreateDefaultDelegate**, **ShowCountAccordingToSettings**,  
    **AlwaysShowInclusiveCounts**, **DefaultFlags** = CreateDefaultFilterModel | CreateDefaultDelegate | Show↔  
    CountAccordingToSettings }  
• typedef QFlags< **Flag** > **Flags**

## Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }

*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

## Protected Slots inherited from [Digikam::AbstractAlbumTreeView](#)

- void [albumSettingsChanged](#) ()
- void [slotCurrentChanged](#) ()
- virtual void [slotRootAlbumAvailable](#) ()  
*override if implemented behavior is not as intended*
- void [slotSearchTextSettingsAboutToChange](#) (bool searched, bool willSearch)
- void [slotSearchTextSettingsChanged](#) (bool wasSearching, bool searching)
- void [slotSelectionChanged](#) ()

## Protected Member Functions inherited from [Digikam::AbstractCheckableAlbumTreeView](#)

- void [middleButtonPressed](#) ([Album](#) \*a) override
- void [rowsInserted](#) (const [QModelIndex](#) &parent, int start, int end) override
- void [setAlbumFilterModel](#) ([CheckableAlbumFilterModel](#) \*const filterModel)
- void [setAlbumModel](#) ([AbstractCheckableAlbumModel](#) \*const model)

## Protected Member Functions inherited from [Digikam::AbstractCountingAlbumTreeView](#)

- void [rowsInserted](#) (const [QModelIndex](#) &parent, int start, int end) override
- void [setAlbumFilterModel](#) ([AlbumFilterModel](#) \*const filterModel)
- void [setAlbumModel](#) ([AbstractCountingAlbumModel](#) \*const model)

## Protected Member Functions inherited from [Digikam::AbstractAlbumTreeView](#)

- virtual void [addCustomContextMenuActions](#) ([ContextMenuHelper](#) &cmh, [Album](#) \*album)  
*Hook method to add custom actions to the generated context menu.*
- virtual [QPixmap](#) [contextMenuIcon](#) () const  
*Hook method that can be implemented to return a special icon used for the context menu.*
- virtual [QString](#) [contextMenuTitle](#) () const  
*Hook method to implement that returns the title for the context menu.*
- void [dragEnterEvent](#) ([QDragEnterEvent](#) \*e) override
- void [dragLeaveEvent](#) ([QDragLeaveEvent](#) \*e) override
- void [dragMoveEvent](#) ([QDragMoveEvent](#) \*e) override
- void [dropEvent](#) ([QDropEvent](#) \*e) override
- virtual void [handleCustomContextMenuAction](#) ([QAction](#) \*action, const [AlbumPointer](#)< [Album](#) > &album)  
*Hook method to handle the custom context menu actions that were added with [addCustomContextMenuActions](#).*
- void [mousePressEvent](#) ([QMouseEvent](#) \*e) override  
*Other helper methods.*
- virtual [QPixmap](#) [pixmapForDrag](#) (const [QStyleOptionViewItem](#) &option, [QList](#)< [QModelIndex](#) > indexes)  
*TODO: Move to delegate, when we have one.*
- void [rowsAboutToBeRemoved](#) (const [QModelIndex](#) &parent, int start, int end) override
- void [rowsInserted](#) (const [QModelIndex](#) &index, int start, int end) override
- void [setAlbumFilterModel](#) ([AlbumFilterModel](#) \*const filterModel)
- void [setAlbumModel](#) ([AbstractSpecificAlbumModel](#) \*const model)
- virtual bool [showContextMenuAt](#) ([QContextMenuEvent](#) \*event, [Album](#) \*albumForEvent)  
*Hook method to implement that determines if a context menu shall be displayed for the given event at the position coded in the event.*
- void [startDrag](#) ([Qt::DropActions](#) supportedActions) override

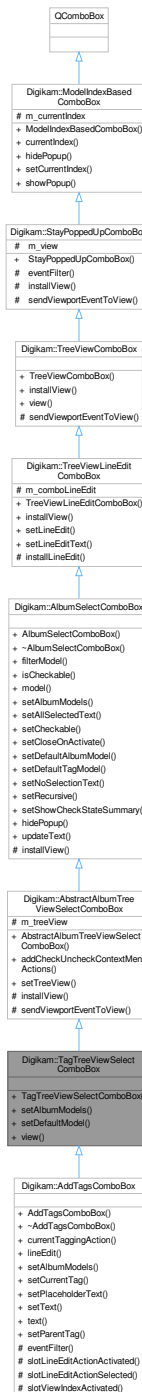


**Protected Member Functions inherited from [Digikam::StateSavingObject](#)**

- QString [entryName](#) (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup [getConfigGroup](#) () const  
*Returns the config group that must be used for state saving and loading.*

## 9.1299 Digikam::TagTreeViewSelectComboBox Class Reference

Inheritance diagram for Digikam::TagTreeViewSelectComboBox:



### Public Member Functions

- **TagTreeViewSelectComboBox** (QWidget \*const parent=nullptr)
- void **setAlbumModels** (TagModel \*model, TagPropertiesFilterModel \*filteredModel=nullptr, CheckableAlbumFilterModel \*filterModel=nullptr)

- void **setDefaultModel** ()
- [TagTreeView](#) \* **view** () const

## Public Member Functions inherited from [Digikam::AbstractAlbumTreeViewSelectComboBox](#)

- [AbstractAlbumTreeViewSelectComboBox](#) (QWidget \*const parent=nullptr)  
*Abstract class.*
- void [addCheckUncheckContextMenuActions](#) ()  
*Enables a context menu which contains options to check or uncheck groups of albums, given you have a checkable model.*
- void [setTreeView](#) ([AbstractAlbumTreeView](#) \*const treeView)  
*Set a tree view created by you instead of creating a default view (in the subclasses).*

## Public Member Functions inherited from [Digikam::AlbumSelectComboBox](#)

- **AlbumSelectComboBox** (QWidget \*const parent=nullptr)
- QSortFilterProxyModel \* **filterModel** () const  
*Return the filter model in use.*
- bool **isCheckable** () const
- [AbstractCheckableAlbumModel](#) \* **model** () const  
*Returns the source model.*
- void **setAlbumModels** ([AbstractCheckableAlbumModel](#) \*model, [AlbumFilterModel](#) \*filterModel=nullptr)
- void **setAllSelectedText** (bool all)  
*Enable or disable the text used to describe the status when all album is selected.*
- void [setCheckable](#) (bool checkable)  
*Enable checkboxes next to the items.*
- void [setCloseOnActivate](#) (bool close)  
*Enable closing when an item was activated (clicked).*
- void [setDefaultAlbumModel](#) ()  
*Once after creation, call one of these three methods.*
- void **setDefaultTagModel** ()
- void [setNoSelectionText](#) (const QString &text)  
*Sets the text that is used to describe the state when no album is selected.*
- void **setRecursive** (bool recursive)  
*If all subalbums shall be selected when parent will be selected.*
- void [setShowCheckStateSummary](#) (bool show)  
*If the box is checkable, enable showing a resume a la "3 Albums checked" in the combo box text.*

## Public Member Functions inherited from [Digikam::TreeViewLineEditComboBox](#)

- [TreeViewLineEditComboBox](#) (QWidget \*const parent=nullptr)  
*This class provides a [TreeViewComboBox](#) with a read-only line edit.*
- void [installView](#) (QAbstractItemView \*view=nullptr) override  
*Replace the standard combo box list view with a QTreeView.*
- void **setLineEdit** (QLineEdit \*edit)
- void [setLineEditText](#) (const QString &text)  
*Set the text of the line edit (the text that is visible if the popup is not opened).*

## Public Member Functions inherited from [Digikam::TreeViewComboBox](#)

- [TreeViewComboBox](#) (QWidget \*parent=nullptr)  
*This class provides a QComboBox with a QTreeView instead of the usual QListView.*
- [QTreeView](#) \* [view](#) () const  
*Returns the QTreeView of this class.*

## Public Member Functions inherited from [Digikam::StayPoppedUpComboBox](#)

- [StayPoppedUpComboBox](#) (QWidget \*const parent=nullptr)  
*This class provides an abstract QComboBox with a custom view (which is created by implementing subclasses) instead of the usual QListView.*

## Public Member Functions inherited from [Digikam::ModelIndexBasedComboBox](#)

- [ModelIndexBasedComboBox](#) (QWidget \*const parent=nullptr)  
*QComboBox has a current index based on a single integer.*
- [QModelIndex](#) [currentIndex](#) () const
- void [hidePopup](#) () override
- void [setCurrentIndex](#) (const QModelIndex &index)
- void [showPopup](#) () override

## Additional Inherited Members

## Public Slots inherited from [Digikam::AlbumSelectComboBox](#)

- void [hidePopup](#) () override
- virtual void [updateText](#) ()  
*Updates the text describing the selection ("3 Albums selected").*

## Protected Member Functions inherited from [Digikam::AbstractAlbumTreeViewSelectComboBox](#)

- void [installView](#) (QAbstractItemView \*view=nullptr) override  
*Replace the standard combo box list view with a QTreeView.*
- void [sendViewportEventToView](#) (QEvent \*e) override  
*Implement in subclass: Send the given event to the viewportEvent() method of m\_view.*

## Protected Member Functions inherited from [Digikam::AlbumSelectComboBox](#)

- void [installView](#) (QAbstractItemView \*view=nullptr) override  
*Replace the standard combo box list view with a QTreeView.*

## Protected Member Functions inherited from [Digikam::TreeViewLineEditComboBox](#)

- virtual void [installLineEdit](#) ()  
*Sets a line edit.*

### Protected Member Functions inherited from [Digikam::TreeViewComboBox](#)

- void [sendViewportEventToView](#) (QEvent \*e) override

*Implement in subclass: Send the given event to the viewportEvent() method of m\_view.*

### Protected Member Functions inherited from [Digikam::StayPoppedUpComboBox](#)

- bool [eventFilter](#) (QObject \*watched, QEvent \*event) override
- void [installView](#) (QAbstractItemView \*view)

*Replace the standard combo box list view with the given view.*

### Protected Attributes inherited from [Digikam::AbstractAlbumTreeViewSelectComboBox](#)

- [AbstractAlbumTreeView](#) \* [m\\_treeView](#) = nullptr

### Protected Attributes inherited from [Digikam::TreeViewLineEditComboBox](#)

- QLineEdit \* [m\\_comboLineEdit](#) = nullptr

### Protected Attributes inherited from [Digikam::StayPoppedUpComboBox](#)

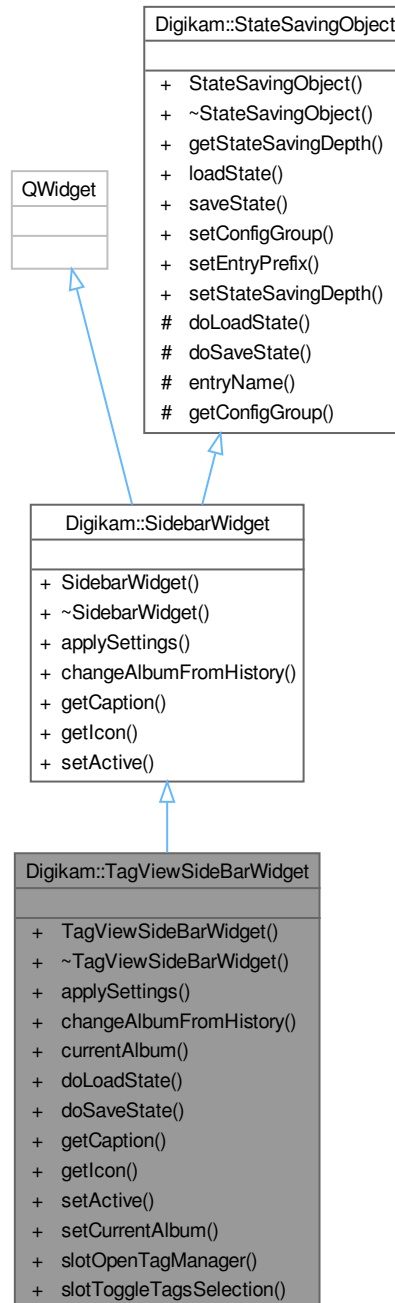
- QAbstractItemView \* [m\\_view](#) = nullptr

### Protected Attributes inherited from [Digikam::ModelIndexBasedComboBox](#)

- QPersistentModelIndex [m\\_currentIndex](#)

## 9.1300 Digikam::TagViewSideBarWidget Class Reference

Inheritance diagram for Digikam::TagViewSideBarWidget:



### Public Slots

- void **setCurrentAlbum** ([TAlbum](#) \*album)
- void **slotOpenTagManager** ()
- void **slotToggleTagsSelection** (int radioClicked)

## Signals

- void **signalFindDuplicates** (const QList< [TAlbum](#) \* > &albums)

## Signals inherited from [Digikam::SidebarWidget](#)

- void **requestActiveTab** ([SidebarWidget](#) \*)  
*This signal can be emitted if this sidebar widget wants to be the one that is active.*
- void **signalNotificationError** (const QString &message, int type)  
*To dispatch error message to temporized pop-up notification widget hosted with icon-view.*

## Public Member Functions

- **TagViewSideBarWidget** (QWidget \*const parent, [TagModel](#) \*const model)
- void **applySettings** () override  
*This method is invoked when the application settings should be (re-) applied to this widget.*
- void **changeAlbumFromHistory** (const QList< [Album](#) \* > &album) override  
*This is called on this widget when the history requires to move back to the specified album.*
- [AlbumPointer](#)< [TAlbum](#) > **currentAlbum** () const
- void **doLoadState** () override  
*Implement this hook method for state loading.*
- void **doSaveState** () override  
*Implement this hook method for state saving.*
- const QString **getCaption** () override  
*Must be implemented to return the title of this sidebar's tab.*
- const QIcon **getIcon** () override  
*Must be implemented and return the icon that shall be visible for this sidebar widget.*
- void **setActive** (bool active) override  
*This method is called if the visible sidebar widget is changed.*

## Public Member Functions inherited from [Digikam::SidebarWidget](#)

- [SidebarWidget](#) (QWidget \*const parent)  
*Constructor.*
- **~SidebarWidget** () override=default  
*Destructor.*

## Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (QObject \*const host)  
*Constructor.*
- virtual **~StateSavingObject** ()  
*Destructor.*
- [StateSavingDepth](#) **getStateSavingDepth** () const  
*Returns the depth used for state saving or loading.*
- void **loadState** ()  
*Invokes loading the class' state.*
- void **saveState** ()  
*Invokes saving the class' state.*
- virtual void **setConfigGroup** (const KConfigGroup &group)  
*Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void **setEntryPrefix** (const QString &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void **setStateSavingDepth** (const [StateSavingDepth](#) depth)  
*Sets the depth used for state saving or loading.*

## Additional Inherited Members

## Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }

*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

## Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- [QString](#) [entryName](#) (const [QString](#) &base) const

*Always use this method to create config group entry names.*

- [KConfigGroup](#) [getConfigGroup](#) () const

*Returns the config group that must be used for state saving and loading.*

## 9.1300.1 Member Function Documentation

### 9.1300.1.1 [applySettings\(\)](#)

```
void Digikam::TagViewSideBarWidget::applySettings ( ) [override], [virtual]
```

Implements [Digikam::SidebarWidget](#).

### 9.1300.1.2 [changeAlbumFromHistory\(\)](#)

```
void Digikam::TagViewSideBarWidget::changeAlbumFromHistory (
    const QList< Album * > & album ) [override], [virtual]
```

Implements [Digikam::SidebarWidget](#).

### 9.1300.1.3 [doLoadState\(\)](#)

```
void Digikam::TagViewSideBarWidget::doLoadState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

### 9.1300.1.4 [doSaveState\(\)](#)

```
void Digikam::TagViewSideBarWidget::doSaveState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).



### 9.1300.1.5 `getCaption()`

```
const QString Digikam::TagViewSideBarWidget::getCaption ( ) [override], [virtual]
```

#### Returns

localized title string

Implements [Digikam::SidebarWidget](#).

### 9.1300.1.6 `getIcon()`

```
const QIcon Digikam::TagViewSideBarWidget::getIcon ( ) [override], [virtual]
```

#### Returns

pixmap icon

Implements [Digikam::SidebarWidget](#).

### 9.1300.1.7 `setActive()`

```
void Digikam::TagViewSideBarWidget::setActive (
    bool active ) [override], [virtual]
```

#### Parameters

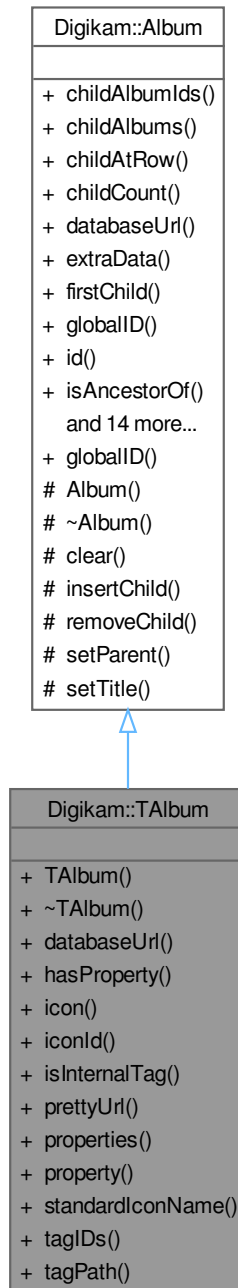
<i>active</i>	if true, this widget is the new active widget, if false another widget is active
---------------	--

Implements [Digikam::SidebarWidget](#).

## 9.1301 Digikam::TAlbum Class Reference

A Tag [Album](#) representation.

Inheritance diagram for Digikam::TAlbum:



### Public Member Functions

- **TAlbum** (const QString &title, int id, bool root=false)
- [CoreDbUrl databaseUrl](#) () const override
- bool **hasProperty** (const QString &key) const
- QString **icon** () const
- qlonglong **iconId** () const

- bool **isInternalTag** () const
- QString **prettyUrl** () const
- QMap< QString, QString > **properties** () const
- QString **property** (const QString &key) const
- QString **standardIconName** () const
- QList< int > **tagIDs** () const
- QString **tagPath** (bool leadingSlash=true) const

## Public Member Functions inherited from Digikam::Album

- QList< int > **childAlbumIds** (bool recursive=false)
- AlbumList **childAlbums** (bool recursive=false)
- Album \* **childAtRow** (int row) const
- int **childCount** () const
- void \* **extraData** (const void \*const key) const  
*Retrieve the associated extra data associated with key.*
- Album \* **firstChild** () const
- int **globalID** () const  
*An album ID is only unique among the set of all Albums of its Type.*
- int **id** () const  
*Each album has a ID uniquely identifying it in the set of Albums of a Type.*
- bool **isAncestorOf** (Album \*const album) const
- bool **isRoot** () const
- bool **isTrashAlbum** () const
- bool **isUsedByLabelsTree** () const
- Album \* **lastChild** () const
- Album \* **next** () const
- Album \* **parent** () const
- void **prepareForDeletion** ()  
*For secure deletion in an album model, call this function beforehand.*
- Album \* **prev** () const
- void **removeExtraData** (const void \*const key)  
*Remove the associated extra data associated with key.*
- int **rowFromAlbum** () const
- void **setExtraData** (const void \*const key, void \*const value)  
*This allows to associate some "extra" data to a Album.*
- void **setUsedByLabelsTree** (bool isUsed)  
*Sets the property m\_usedByLabelsTree to true if the search album was created using the Colors and labels tree view.*
- QString **title** () const
- Type **type** () const

## Friends

- class **AlbumManager**

## Additional Inherited Members

## Public Types inherited from Digikam::Album

- enum **Type** {  
  **PHYSICAL** = 0 , **TAG** , **DATE** , **SEARCH** ,  
  **FACE** }

## Static Public Member Functions inherited from [Digikam::Album](#)

- static int [globalID](#) ([Type](#) type, int id)  
*Produces the global id.*

## Protected Member Functions inherited from [Digikam::Album](#)

- **Album** ([Album::Type](#) type, int id, bool root)  
*Constructor.*
- virtual [~Album](#) ()  
*Destructor.*
- void **clear** ()  
*Delete all child albums and also remove any associated extra data.*
- void **insertChild** ([Album](#) \*const child)
- void **removeChild** ([Album](#) \*const child)
- void **setParent** ([Album](#) \*const parent)
- void **setTitle** (const QString &title)

### 9.1301.1 Member Function Documentation

#### 9.1301.1.1 [databaseUrl\(\)](#)

```
CoreDbUrl Digikam::TAlbum::databaseUrl ( ) const [override], [virtual]
```

##### Returns

the kde url of the album

Implements [Digikam::Album](#).

#### 9.1301.1.2 [tagPath\(\)](#)

```
QString Digikam::TAlbum::tagPath (
    bool leadingSlash = true ) const
```

##### Returns

The tag path, e.g. "/People/Friend/John" if leadingSlash is true, "People/Friend/John" if leadingSlash if false.  
The root [TAlbum](#) returns "/" resp. "".

## 9.1302 Digikam::Template Class Reference

### Public Member Functions

- QStringList **authors** () const
- QString **authorsPosition** () const
- [IptcCoreContactInfo](#) **contactInfo** () const
- [MetaEngine::AltLangMap](#) **copyright** () const
- QString **credit** () const
- QString **instructions** () const
- QStringList **iptcSubjects** () const
- bool **isEmpty** () const
  - Return true if [Template](#) contents is empty.*
- bool **isNull** () const
  - Return true if [Template](#) title is null.*
- [IptcCoreLocationInfo](#) **locationInfo** () const
- void **merge** (const [Template](#) &t)
  - Merge the metadata from another [Template](#).*
- bool **operator==** (const [Template](#) &t) const
  - Compare for metadata equality, not including "templateTitle" value.*
- [MetaEngine::AltLangMap](#) **rightUsageTerms** () const
- void **setAuthors** (const QStringList &authors)
- void **setAuthorsPosition** (const QString &authorPosition)
- void **setContactInfo** (const [IptcCoreContactInfo](#) &inf)
- void **setCopyright** (const [MetaEngine::AltLangMap](#) &copyright)
- void **setCredit** (const QString &credit)
- void **setInstructions** (const QString &instructions)
- void **setIptcSubjects** (const QStringList &subjects)
- void **setLocationInfo** (const [IptcCoreLocationInfo](#) &inf)
- void **setRightUsageTerms** (const [MetaEngine::AltLangMap](#) &rightUsageTerms)
- void **setSource** (const QString &source)
- void **setTemplateTitle** (const QString &title)
- QString **source** () const
- QString **templateTitle** () const

### Static Public Member Functions

- static QString **removeTemplateTitle** ()

### Protected Attributes

- QStringList **m\_authors**
  - List of author names.*
- QString **m\_authorsPosition**
  - Description of authors position.*
- [IptcCoreContactInfo](#) **m\_contactInfo**
  - IPTC Contact Information.*
- [MetaEngine::AltLangMap](#) **m\_copyright**
  - Language alternative copyright notices.*
- QString **m\_credit**
  - Credit description.*

- **QString `m_instructions`**  
*Special instructions to process with contents.*
- **[IptcCoreLocationInfo](#) `m_locationInfo`**  
*IPTC Location Information.*
- **[MetaEngine::AltLangMap](#) `m_rightUsageTerms`**  
*Language alternative right term usages.*
- **QString `m_source`**  
*Descriptions of contents source.*
- **QStringList `m_subjects`**  
*IPTC Subjects Information.*
- **QString `m_templateTitle`**  
*Template title used internally.*

## 9.1302.1 Member Data Documentation

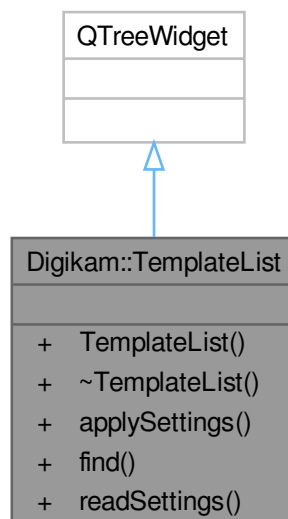
### 9.1302.1.1 `m_templateTitle`

QString Digikam::Template::m\_templateTitle [protected]

This value always exist and cannot be empty.

## 9.1303 Digikam::TemplateList Class Reference

Inheritance diagram for Digikam::TemplateList:

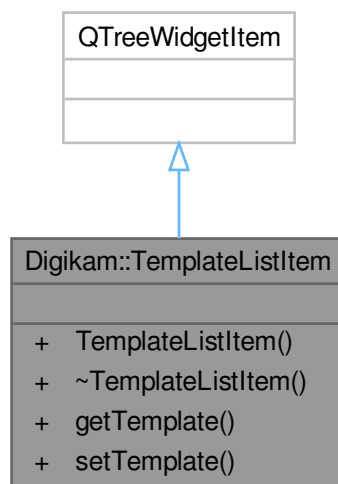


### Public Member Functions

- **TemplateList** (QWidget \*const parent=nullptr)
- void **applySettings** ()
- [TemplateListItem](#) \* **find** (const QString &title)
- void **readSettings** ()

## 9.1304 Digikam::TemplateListItem Class Reference

Inheritance diagram for Digikam::TemplateListItem:

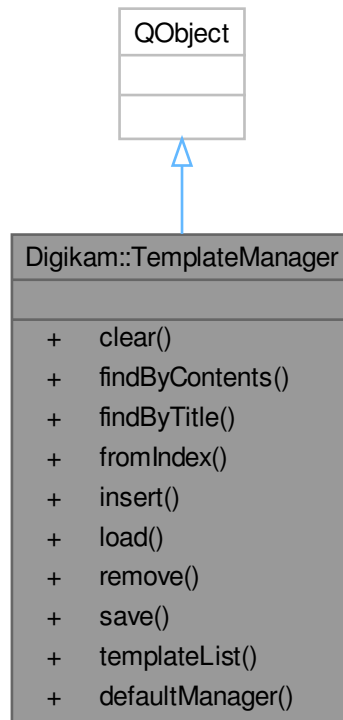


### Public Member Functions

- **TemplateListItem** (QTreeWidgetItem \*const parent, const [Template](#) &t)
- [Template](#) **getTemplate** () const
- void **setTemplate** (const [Template](#) &t)

## 9.1305 Digikam::TemplateManager Class Reference

Inheritance diagram for Digikam::TemplateManager:



### Signals

- void **signalTemplateAdded** (const [Template](#) &)
- void **signalTemplateRemoved** (const [Template](#) &)

### Public Member Functions

- void **clear** ()
- [Template](#) **findByContents** (const [Template](#) &tref) const
- [Template](#) **findByTitle** (const QString &title) const
- [Template](#) **fromIndex** (int index) const
- void **insert** (const [Template](#) &t)
- bool **load** ()
- void **remove** (const [Template](#) &t)
- bool **save** ()
- QList< [Template](#) > **templateList** () const

### Static Public Member Functions

- static [TemplateManager](#) \* **defaultManager** ()

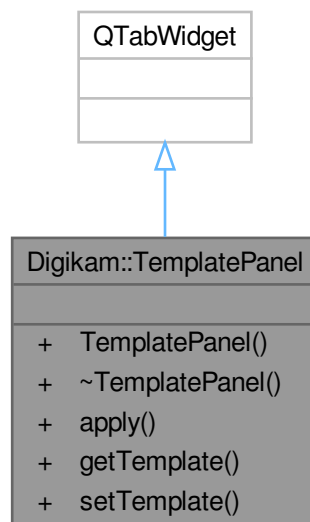


## Friends

- class **TemplateManagerCreator**

## 9.1306 Digikam::TemplatePanel Class Reference

Inheritance diagram for Digikam::TemplatePanel:



## Public Types

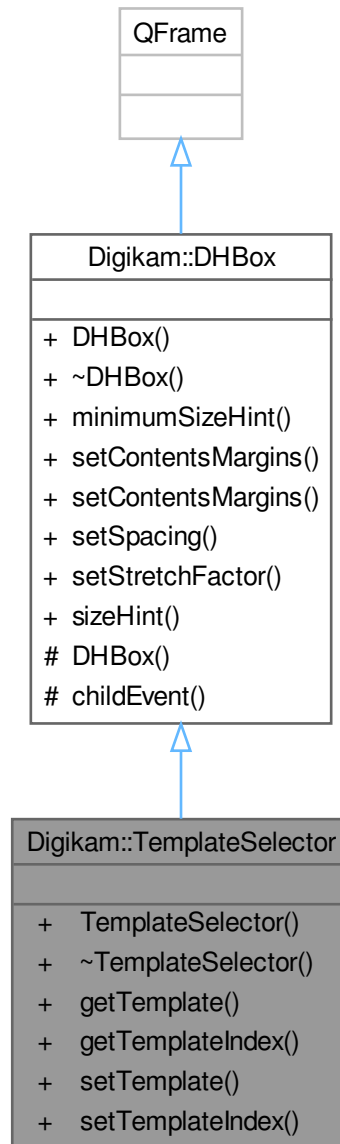
- enum **TemplateTab** { **RIGHTS** =0 , **LOCATION** , **CONTACT** , **SUBJECTS** }

## Public Member Functions

- **TemplatePanel** (`QWidget *const parent=nullptr`)
- void **apply** ()
- **Template** **getTemplate** () const
- void **setTemplate** (const **Template** &t)

## 9.1307 Digikam::TemplateSelector Class Reference

Inheritance diagram for Digikam::TemplateSelector:



### Public Types

- enum **SelectorItems** { **REMOVETEMPLATE** = 0 , **DONTCHANGE** = 1 }

### Signals

- void **signalTemplateSelected** ()

### Public Member Functions

- **TemplateSelector** (QWidget \*const parent=nullptr)
- **Template** **getTemplate** () const
- int **getTemplateIndex** () const
- void **setTemplate** (const **Template** &t)
- void **setTemplateIndex** (int i)

### Public Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (QWidget \*const parent=nullptr)
- QSize **minimumSizeHint** () const override
- void **setContentsMargins** (const QMargins &margins)
- void **setContentsMargins** (int left, int top, int right, int bottom)
- void **setSpacing** (int space)
- void **setStretchFactor** (QWidget \*const widget, int stretch)
- QSize **sizeHint** () const override

### Additional Inherited Members

### Protected Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (bool vertical, QWidget \*const parent)
- void **childEvent** (QChildEvent \*e) override

## 9.1308 Digikam::TemplateViewer Class Reference

Inheritance diagram for Digikam::TemplateViewer:



### Public Member Functions

- **TemplateViewer** (`QWidget *const parent=nullptr`)
- void **setTemplate** (`const Template &t`)

### Public Member Functions inherited from [Digikam::DExpanderBox](#)

- **DExpanderBox** (`QWidget *const parent=nullptr`)

- void **addItem** (QWidget \*const w, const QIcon &icon, const QString &txt, const QString &objName, bool expandBydefault)
- Add [DLabelExpander](#) item at end of box layout with these settings : 'w' : the widget hosted by [DLabelExpander](#).*
- void **addItem** (QWidget \*const w, const QString &txt, const QString &objName, bool expandBydefault)
- void **addStretch** ()
- bool **buttonIsVisible** (int index) const
- bool **checkboxIsVisible** (int index) const
- int **count** () const
- int **indexOf** ([DLabelExpander](#) \*const widget) const
- void **insertItem** (int index, QWidget \*const w, const QIcon &icon, const QString &txt, const QString &objName, bool expandBydefault)
- Insert [DLabelExpander](#) item at box layout index with these settings : 'w' : the widget hosted by [DLabelExpander](#).*
- void **insertItem** (int index, QWidget \*const w, const QString &txt, const QString &objName, bool expandBydefault)
- void **insertStretch** (int index)
- bool **isChecked** (int index) const
- bool **isItemEnabled** (int index) const
- bool **isItemExpanded** (int index) const
- QIcon **itemIcon** (int index) const
- QString **itemText** (int index) const
- QString **itemToolTip** (int index) const
- virtual void **readSettings** (KConfigGroup &group)
- void **removeItem** (int index)
- void **setButtonIcon** (int index, const QIcon &icon)
- void **setButtonVisible** (int index, bool b)
- void **setCheckBoxVisible** (int index, bool b)
- void **setChecked** (int index, bool b)
- void **setItemEnabled** (int index, bool enabled)
- void **setItemExpanded** (int index, bool b)
- void **setItemIcon** (int index, const QIcon &icon)
- void **setItemText** (int index, const QString &txt)
- void **setItemToolTip** (int index, const QString &tip)
- [DLabelExpander](#) \* **widget** (int index) const
- virtual void **writeSettings** (KConfigGroup &group)

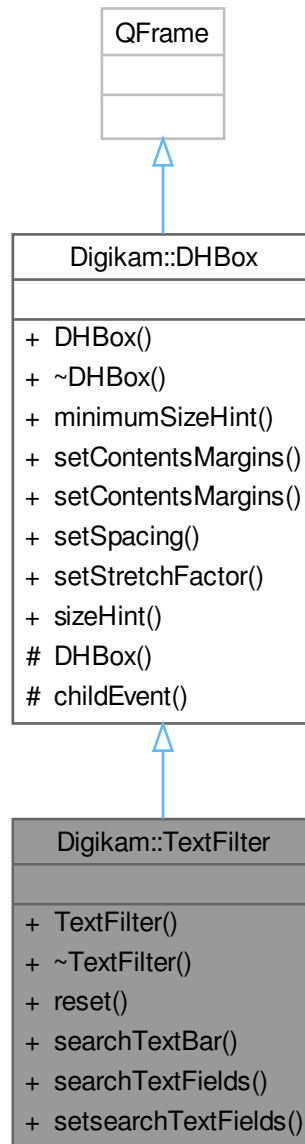
### Additional Inherited Members

### Signals inherited from [Digikam::DExpanderBox](#)

- void **signalItemButtonPressed** (int index)
- void **signalItemExpanded** (int index, bool b)
- void **signalItemToggled** (int index, bool b)

## 9.1309 Digikam::TextFilter Class Reference

Inheritance diagram for Digikam::TextFilter:



### Signals

- void **signalSearchTextFilterSettings** (const [SearchTextFilterSettings](#) &)

### Public Member Functions

- **TextFilter** (QWidget \*const parent)

- void **reset** ()
- [SearchTextBar](#) \* **searchTextBar** () const
- SearchTextFilterSettings::TextFilterFields **searchTextFields** ()
- void **setsearchTextFields** (SearchTextFilterSettings::TextFilterFields fields)

### Public Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (QWidget \*const parent=nullptr)
- QSize **minimumSizeHint** () const override
- void **setContentsMargins** (const QMargins &margins)
- void **setContentsMargins** (int left, int top, int right, int bottom)
- void **setSpacing** (int space)
- void **setStretchFactor** (QWidget \*const widget, int stretch)
- QSize **sizeHint** () const override

### Additional Inherited Members

### Protected Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (bool vertical, QWidget \*const parent)
- void **childEvent** (QChildEvent \*e) override

## 9.1310 Digikam::TextureContainer Class Reference

### Public Types

- enum **TextureTypes** {  
    **PaperTexture** = 0 , **Paper2Texture** , **FabricTexture** , **BurlapTexture** ,  
    **BricksTexture** , **Bricks2Texture** , **CanvasTexture** , **MarbleTexture** ,  
    **Marble2Texture** , **BlueJeanTexture** , **CellWoodTexture** , **MetalWireTexture** ,  
    **ModernTexture** , **WallTexture** , **MossTexture** , **StoneTexture** }

### Static Public Member Functions

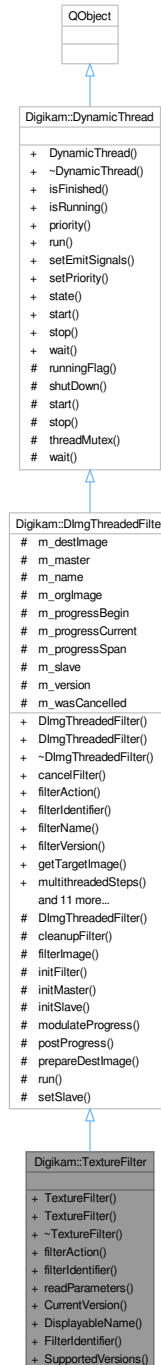
- static QString **getTexturePath** (int texture)

### Public Attributes

- int **blendGain** = 200
- int **textureType** = MarbleTexture

## 9.1311 Digikam::TextureFilter Class Reference

Inheritance diagram for Digikam::TextureFilter:



### Public Member Functions

- `TextureFilter` (`DImg *const orgImage`, `QObject *const parent=nullptr`, `const TextureContainer &settings=TextureContainer()`)
- `TextureFilter` (`QObject *const parent=nullptr`)



- [FilterAction filterAction](#) () override  
*Returns the action description corresponding to currently set options.*
- [QString filterIdentifier](#) () const override  
*Return the identifier for this filter in the image history.*
- void [readParameters](#) (const [FilterAction](#) &action) override

## Public Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImg](#) \*const orgImage, [QObject](#) \*const parent, const [QString](#) &name=[QString](#)())  
*Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter](#) ([QObject](#) \*const parent=nullptr, const [QString](#) &name=[QString](#)())  
*Constructs a filter without argument.*
- virtual void [cancelFilter](#) ()  
*Cancel the threaded computation.*
- const [QString](#) & [filterName](#) ()
- int [filterVersion](#) () const
- [DImg](#) [getTargetImage](#) ()
- [QList](#)< int > [multithreadedSteps](#) (int stop, int start=0) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool [parametersSuccessfullyRead](#) () const  
*Optional: error handling for readParameters.*
- virtual [QString](#) [readParametersError](#) (const [FilterAction](#) &actionThatFailed) const
- void [setFilterName](#) (const [QString](#) &name)
- void [setFilterVersion](#) (int version)  
*Replaying a filter action: Set the filter version.*
- void [setOriginalImage](#) (const [DImg](#) &orgImage)
- void [setupAndStartDirectly](#) (const [DImg](#) &orgImage, [DImgThreadedFilter](#) \*const master, int progress←Begin=0, int progressEnd=100)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void [setupFilter](#) (const [DImg](#) &orgImage)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void [startFilter](#) ()  
*Start the threaded computation.*
- virtual void [startFilterDirectly](#) ()  
*Start computation of this filter, directly in this thread.*
- virtual [QList](#)< int > [supportedVersions](#) () const

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread](#) ([QObject](#) \*const parent=nullptr)  
*This class extends [QRunnable](#), so you have to reimplement virtual void [run\(\)](#).*
- [~DynamicThread](#) () override  
*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool [isFinished](#) () const
- bool [isRunning](#) () const
- [QThread::Priority](#) [priority](#) () const
- void [setEmitSignals](#) (bool emitThem)
- void [setPriority](#) ([QThread::Priority](#) priority)  
*Sets the priority for this dynamic thread.*
- State [state](#) () const

### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from Digikam::DImgThreadedFilter

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from Digikam::DynamicThread

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from Digikam::DImgThreadedFilter

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

### 9.1311.1 Member Function Documentation

#### 9.1311.1.1 filterAction()

`FilterAction` Digikam::TextureFilter::filterAction ( ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

#### 9.1311.1.2 filterIdentifier()

`QString` Digikam::TextureFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

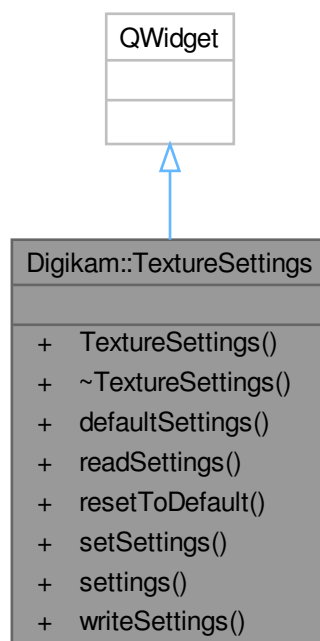
#### 9.1311.1.3 readParameters()

`void` Digikam::TextureFilter::readParameters (   
     const `FilterAction` & *action* ) [override], [virtual]

Implements [Digikam::DImgThreadedFilter](#).

## 9.1312 Digikam::TextureSettings Class Reference

Inheritance diagram for Digikam::TextureSettings:



## Signals

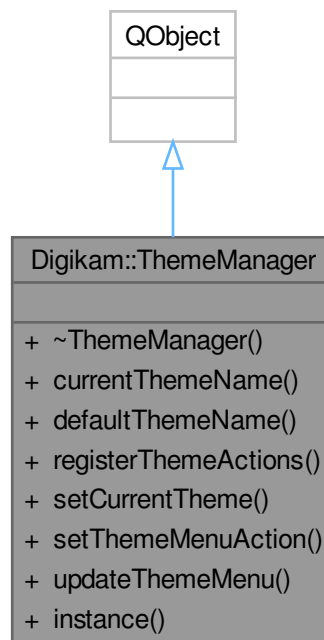
- void **signalSettingsChanged** ()

## Public Member Functions

- **TextureSettings** (QWidget \*const parent)
- **TextureContainer defaultSettings** () const
- void **readSettings** (const KConfigGroup &group)
- void **resetToDefault** ()
- void **setSettings** (const **TextureContainer** &settings)
- **TextureContainer settings** () const
- void **writeSettings** (KConfigGroup &group)

## 9.1313 Digikam::ThemeManager Class Reference

Inheritance diagram for Digikam::ThemeManager:



## Signals

- void **signalThemeChanged** ()

**Public Member Functions**

- QString **currentThemeName** () const
- QString **defaultThemeName** () const
- void **registerThemeActions** (DXmlGuiWindow \*const win)
- void **setCurrentTheme** (const QString &name)
- void **setThemeMenuAction** (QMenu \*const action)
- void **updateThemeMenu** ()

**Static Public Member Functions**

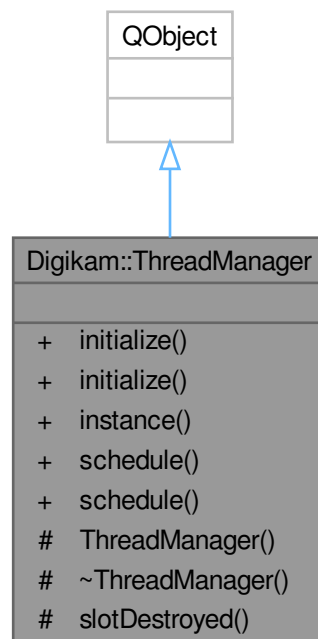
- static ThemeManager \* **instance** ()

**Friends**

- class ThemeManagerCreator

**9.1314 Digikam::ThreadManager Class Reference**

Inheritance diagram for Digikam::ThreadManager:

**Public Slots**

- void **schedule** (QRunnable \*runnable)
- void **schedule** (WorkerObject \*object)

**Public Member Functions**

- void **initialize** ([DynamicThread](#) \*const dynamicThread)
- void **initialize** ([WorkerObject](#) \*const object)

**Static Public Member Functions**

- static [ThreadManager](#) \* **instance** ()

**Protected Slots**

- void **slotDestroyed** (QObject \*object)

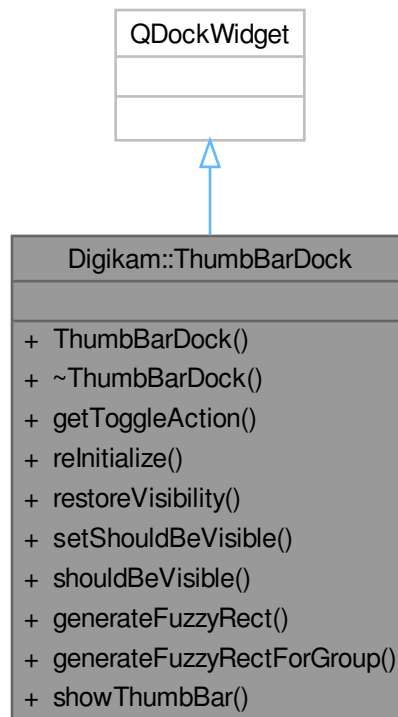
**Friends**

- class **ThreadManagerCreator**

## 9.1315 Digikam::ThumbBarDock Class Reference

A dock widget specifically designed for thumbnail bars (class `ThumbNailView` or one of its descendants).

Inheritance diagram for `Digikam::ThumbBarDock`:



## Public Types

- enum **Visibility** { **WAS\_HIDDEN** , **WAS\_SHOWN** , **SHOULD\_BE\_HIDDEN** , **SHOULD\_BE\_SHOWN** }

## Public Slots

- void **showThumbBar** (bool)

## Public Member Functions

- **ThumbBarDock** (QWidget \*const parent=nullptr, Qt::WindowFlags flags=Qt::WindowFlags())
- QAction \* **getToggleAction** (QObject \*const parent, const QString &caption=QString()) const  
*Return an Action to show and hide the thumbnail bar.*
- void **reinitialize** ()  
*Measure the orientation and size of the widget and adjust the containing thumbnail bar accordingly.*
- void **restoreVisibility** ()
- void **setShouldBeVisible** (bool)
- bool **shouldBeVisible** () const  
*The normal show() and hide() functions don't apply that well, because there are two orthogonal reasons to hide the thumbbar: the user doesn't want it, and the window with the thumbbar isn't shown.*

## Static Public Member Functions

- static QPixmap **generateFuzzyRect** (const QSize &size, const QColor &color, int radius, const QColor &fill←  
Color=Qt::transparent)
- static QPixmap **generateFuzzyRectForGroup** (const QSize &size, const QColor &color, int radius)

### 9.1315.1 Detailed Description

It provides the same look as a toolbar.

### 9.1315.2 Member Function Documentation

#### 9.1315.2.1 reinitialize()

```
void Digikam::ThumbBarDock::reInitialize ( )
```

Normally not needed, but useful when the dock widget has changed location and/or size and the appropriate signals aren't emitted.

#### 9.1315.2.2 shouldBeVisible()

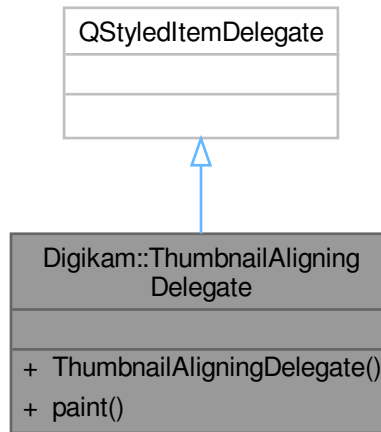
```
bool Digikam::ThumbBarDock::shouldBeVisible ( ) const
```

The restoreVisibility() function will set the visibility status to what it should be according to the user setting. The set←ShouldBeVisible() function can change this setting. showThumbBar() can be used to hide and show the thumbbar according to the user preference. shouldBeVisible() tells whether the thumbbar should be shown according to the user.



## 9.1316 Digikam::ThumbnailAligningDelegate Class Reference

Inheritance diagram for Digikam::ThumbnailAligningDelegate:



### Public Member Functions

- **ThumbnailAligningDelegate** (QObject \*const parent=nullptr)
- void **paint** (QPainter \*painter, const QStyleOptionViewItem &option, const QModelIndex &index) const override

## 9.1317 Digikam::ThumbnailCreator Class Reference

### Public Types

- enum **StorageMethod** { **NoThumbnailStorage** , **FreeDesktopStandard** , **ThumbnailDatabase** }

### Public Member Functions

- **ThumbnailCreator** (int [thumbnailSize](#), StorageMethod method)  
*Create a thumbnail creator object, and set the thumbnail size.*
- **ThumbnailCreator** (StorageMethod method)  
*Create a thumbnail creator object.*
- void **deleteThumbnailsFromDisk** (const QString &filePath) const  
*Deletes all available thumbnails from the on-disk thumbnail cache.*
- QString **errorString** () const  
*Returns the last error that occurred.*
- QImage **load** (const [ThumbnailIdentifier](#) &identifier, bool onlyStorage=false) const  
*Create a thumbnail for the specified file.*

- QImage **loadDetail** (const [ThumbnailIdentifier](#) &identifier, const QRect &detailRect, bool onlyStorage=false) const  
*Creates a thumbnail for the specified detail of the file.*
- void **pregenerate** (const [ThumbnailIdentifier](#) &identifier) const  
*Ensures that the thumbnail is pregenerated in the database, but does not load it from there.*
- void **pregenerateDetail** (const [ThumbnailIdentifier](#) &identifier, const QRect &detailRect) const
- void **setExifRotate** (bool rotate)  
*Set the Exif rotation property.*
- void **setLoadingProperties** ([DImgLoaderObserver](#) \*const observer, const [DRawDecoding](#) &settings)  
*If you plan to load thumbnail from the context of the threadimageio framework, you can specify the relevant parameters.*
- void **setOnlyLargeThumbnails** (bool onlyLarge)  
*If you enable this property, the thumbnail creator will create only large thumbnails on disk (256x256 as described in FreeDesktop paper).*
- void **setRemoveAlphaChannel** (bool removeAlpha)  
*If you enable this property, the returned QImage objects will not have an alpha channel.*
- void **setThumbnailInfoProvider** ([ThumbnailInfoProvider](#) \*const provider)  
*Set a [ThumbnailInfoProvider](#) to provide custom ThumbnailInfos.*
- void **setThumbnailSize** (int [thumbnailSize](#))  
*Sets the thumbnail size.*
- void **store** (const QString &path, const QImage &image) const  
*Store the given image as thumbnail of the given path.*
- void **storeDetailThumbnail** (const QString &path, const QRect &detailRect, const QImage &image) const
- int **storedSize** () const  
*Return the stored image size, the size of the image that is stored on disk (according to Storage Method).*
- int **thumbnailSize** () const  
*Return the thumbnail size, the maximum size of the QImage returned by load.*

### Static Public Member Functions

- static [ThumbnailInfo](#) **fileThumbnailInfo** (const QString &path)  
*Creates a default [ThumbnailInfo](#) for the given path using QFileInfo only.*
- static QString **identifierForDetail** (const [ThumbnailInfo](#) &info, const QRect &rect)  
*Returns the customIdentifier for the detail thumbnail.*

## 9.1317.1 Constructor & Destructor Documentation

### 9.1317.1.1 ThumbnailCreator()

```
Digikam::ThumbnailCreator::ThumbnailCreator (
    StorageMethod method ) [explicit]
```

You must call `setThumbnailSize` before load.

## 9.1317.2 Member Function Documentation

### 9.1317.2.1 deleteThumbnailsFromDisk()

```
void Digikam::ThumbnailCreator::deleteThumbnailsFromDisk (
    const QString & filePath ) const
```

A subsequent call to `load()` will recreate the thumbnail.

### 9.1317.2.2 errorString()

```
QString Digikam::ThumbnailCreator::errorString ( ) const
```

It is valid if load returned a null QImage object.

### 9.1317.2.3 loadDetail()

```
QImage Digikam::ThumbnailCreator::loadDetail (
    const ThumbnailIdentifier & identifier,
    const QRect & detailRect,
    bool onlyStorage = false ) const
```

A suitable custom identifier (for cache key etc.) is inserted as `image.text("customIdentifier")`.

### 9.1317.2.4 setExifRotate()

```
void Digikam::ThumbnailCreator::setExifRotate (
    bool rotate )
```

If `exifRotate` is true, the thumbnail will be rotated according to the Exif information. Default value is true.

### 9.1317.2.5 setLoadingProperties()

```
void Digikam::ThumbnailCreator::setLoadingProperties (
    DImgLoaderObserver *const observer,
    const DRawDecoding & settings )
```

They will be passed if a thumbnail is created by loading with `DImg`. Note that `DImg` is not used in most cases (Raw files, JPEG)

### 9.1317.2.6 setOnlyLargeThumbnails()

```
void Digikam::ThumbnailCreator::setOnlyLargeThumbnails (
    bool onlyLarge )
```

Normally, for requested sizes below 128, thumbnails of 128x128 will be cached on disk. Default value is false.

### 9.1317.2.7 setRemoveAlphaChannel()

```
void Digikam::ThumbnailCreator::setRemoveAlphaChannel (
    bool removeAlpha )
```

Images with transparency will be blended over an opaque background.

**9.1317.2.8 setThumbnailSize()**

```
void Digikam::ThumbnailCreator::setThumbnailSize (
    int thumbnailSize )
```

This is the maximum size of the QImage returned by load.

**9.1317.2.9 store()**

```
void Digikam::ThumbnailCreator::store (
    const QString & path,
    const QImage & image ) const
```

Image should at least have [storedSize\(\)](#).

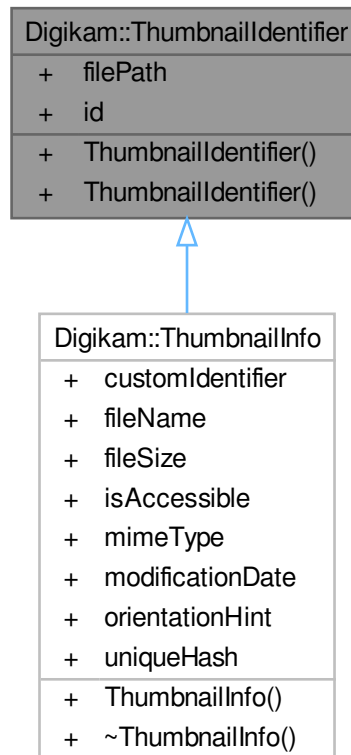
**9.1317.2.10 storedSize()**

```
int Digikam::ThumbnailCreator::storedSize ( ) const
```

This size is possibly larger than thumbnailSize. Possible values: 128 or 256.

**9.1318 Digikam::ThumbnailIdentifier Class Reference**

Inheritance diagram for Digikam::ThumbnailIdentifier:



### Public Member Functions

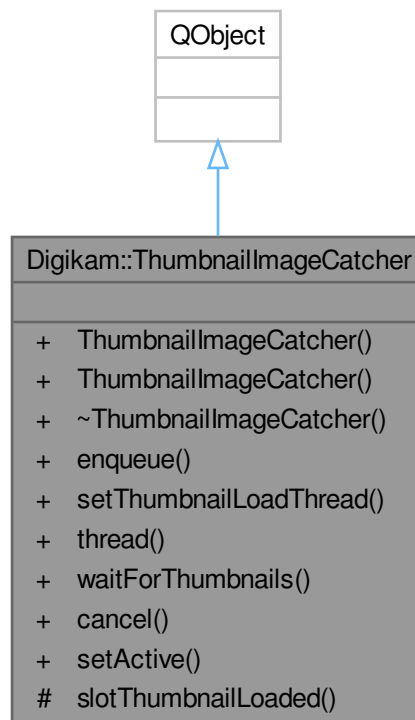
- **ThumbnailIdentifier** (const QString &path)

### Public Attributes

- QString **filePath**  
*The file path from which the thumbnail shall be generated.*
- qlonglong **id** = 0  
*The database id, which needs to be translated to uniqueHash + fileSize.*

## 9.1319 Digikam::ThumbnailImageCatcher Class Reference

Inheritance diagram for Digikam::ThumbnailImageCatcher:



### Public Slots

- void `cancel` ()  
*If the catcher is waiting in `waitForThumbnails()` in a different thread, cancels the waiting.*
- void `setActive` (bool active)  
*The catcher is active per default after construction.*

## Public Member Functions

- [ThumbnailImageCatcher](#) (QObject \*const parent=nullptr)  
*Use this class to get a thumbnail synchronously.*
- **ThumbnailImageCatcher** ([ThumbnailLoadThread](#) \*const thread, QObject \*const parent=nullptr)
- int [enqueue](#) ()  
*After requesting a thumbnail from the thread, call [enqueue\(\)](#) each time.*
- void **setThumbnailLoadThread** ([ThumbnailLoadThread](#) \*const thread)
- [ThumbnailLoadThread](#) \* **thread** () const
- QList< QImage > **waitForThumbnails** ()

## Protected Slots

- void **slotThumbnailLoaded** (const [LoadingDescription](#) &, const QImage &)

## 9.1319.1 Constructor & Destructor Documentation

### 9.1319.1.1 ThumbnailImageCatcher()

```
Digikam::ThumbnailImageCatcher::ThumbnailImageCatcher (
    QObject *const parent = nullptr ) [explicit]
```

1. Create the [ThumbnailImageCatcher](#) object with your [ThumbnailLoadThread](#)
2. a) Request a thumbnail b) Call [enqueue\(\)](#)
3. Call [waitForThumbnails](#) which returns the thumbnail QImage(s).

Note: Not meant for loading QPixmap thumbnails.

## 9.1319.2 Member Function Documentation

### 9.1319.2.1 cancel

```
void Digikam::ThumbnailImageCatcher::cancel ( ) [slot]
```

The results will be returned as received so far.

### 9.1319.2.2 enqueue()

```
int Digikam::ThumbnailImageCatcher::enqueue ( )
```

Enqueue records the requested loading operation in an internal list. A loading operation can result in the return of more than one thumbnail, so [enqueue\(\)](#) returns the number of expected results. Then call [waitForThumbnails](#). The returned list is the sum of previous calls to [enqueue](#), one entry per expected result, in order. If stopped prematurely or loading failed, the respective entries will be null.

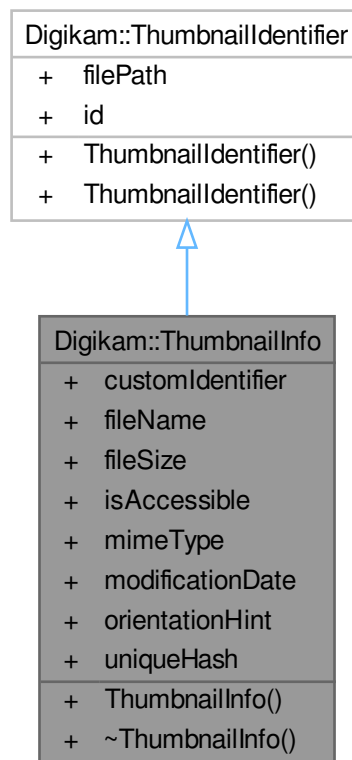
### 9.1319.2.3 setActive

```
void Digikam::ThumbnailImageCatcher::setActive (
    bool active ) [slot]
```

Deactivate it if you use the catcher as a longer-lived object and do not use it for some time, then activate it before you request a thumbnail from the thread again.

## 9.1320 Digikam::ThumbnailInfo Class Reference

Inheritance diagram for Digikam::ThumbnailInfo:



### Public Attributes

- QString **customIdentifier**  
A custom identifier, if neither `filePath` nor `uniqueHash` are applicable.
- QString **fileName**  
The file name (the name, not the directory)
- qlonglong **fileSize** = 0
- bool **isAccessible** = false  
If the original file is at all accessible on disk.

- QString `mimeType`  
*The mime type of the original file.*
- QDateTime `modificationDate`  
*The modification date of the original file.*
- int `orientationHint` = `DMetadadata::ORIENTATION_UNSPECIFIED`  
*Gives a hint at the orientation of the image.*
- QString `uniqueHash`  
*If available, the uniqueHash + fileSize pair for identification of the original file by content.*

## Public Attributes inherited from `Digikam::ThumbnailIdentifier`

- QString `filePath`  
*The file path from which the thumbnail shall be generated.*
- qlonglong `id` = 0  
*The database id, which needs to be translated to uniqueHash + fileSize.*

## Additional Inherited Members

## Public Member Functions inherited from `Digikam::ThumbnailIdentifier`

- `ThumbnailIdentifier` (const QString &path)

## 9.1320.1 Member Data Documentation

### 9.1320.1.1 `isAccessible`

```
bool Digikam::ThumbnailInfo::isAccessible = false
```

May be false if a file on a removable device is used.

### 9.1320.1.2 `mimeType`

```
QString Digikam::ThumbnailInfo::mimeType
```

Currently "image" or "video" otherwise empty.

### 9.1320.1.3 `modificationDate`

```
QDateTime Digikam::ThumbnailInfo::modificationDate
```

Thumbnail will be regenerated if thumb's modification date is older than this.

### 9.1320.1.4 `orientationHint`

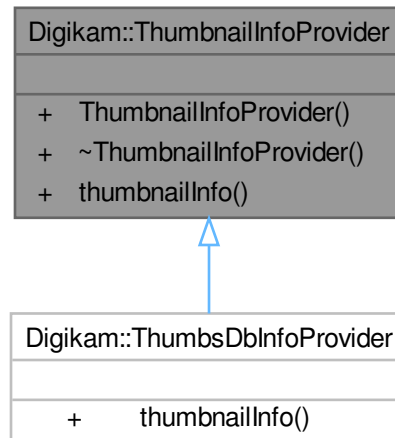
```
int Digikam::ThumbnailInfo::orientationHint = DMetadadata::ORIENTATION_UNSPECIFIED
```

This can be used to supersede the Exif information in the file. Will not be used if `DMetadadata::ORIENTATION_↔ UNSPECIFIED` (default value)



## 9.1321 Digikam::ThumbnailInfoProvider Class Reference

Inheritance diagram for Digikam::ThumbnailInfoProvider:

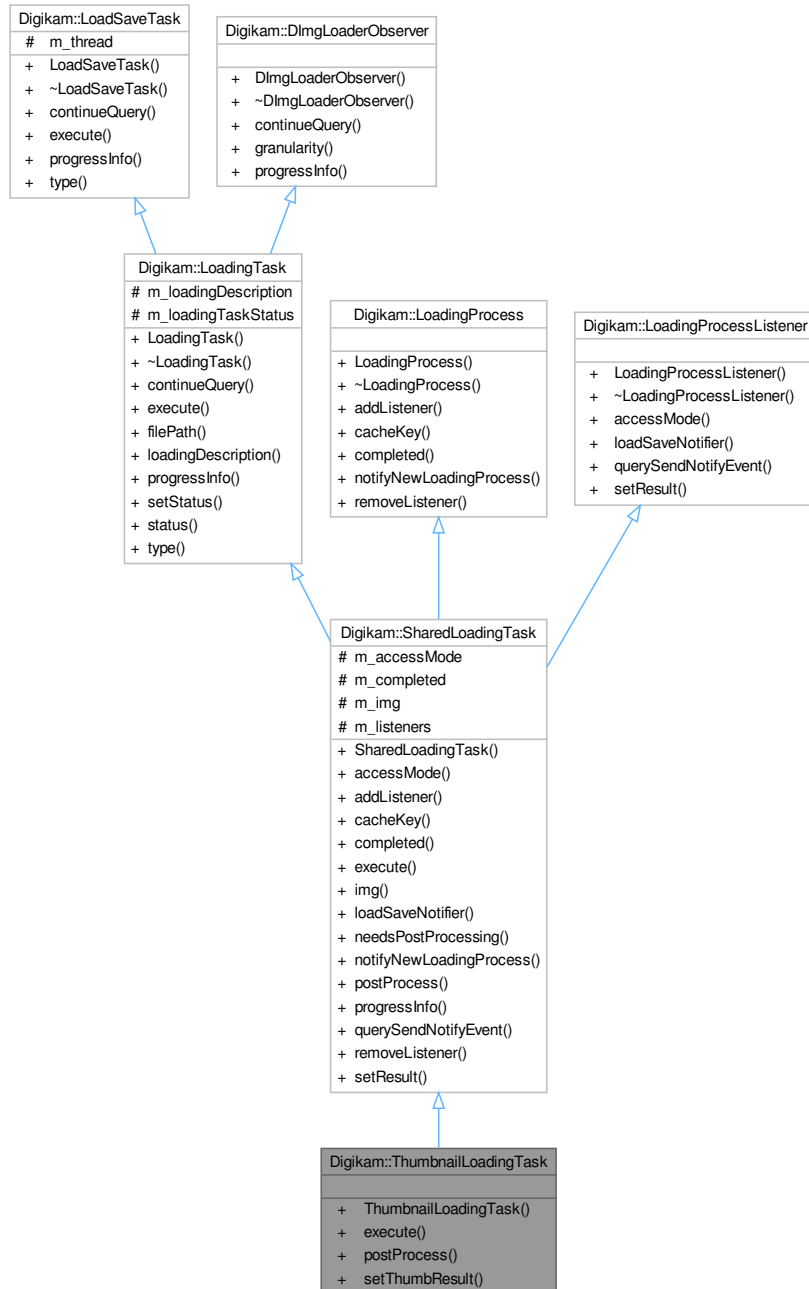


### Public Member Functions

- virtual `ThumbnailInfo thumbnailInfo` (const `ThumbnailIdentifier &`)=0

## 9.1322 Digikam::ThumbnailLoadingTask Class Reference

Inheritance diagram for Digikam::ThumbnailLoadingTask:



### Public Member Functions

- **ThumbnailLoadingTask** (`LoadSaveThread *const thread`, `const LoadingDescription &description`)
- void `execute()` override
- void `postProcess()` override
- void **setThumbResult** (`const LoadingDescription &loadingDescription`, `const QImage &qimage`)

## Public Member Functions inherited from Digikam::SharedLoadingTask

- **SharedLoadingTask** ([LoadSaveThread](#) \*const thread, const [LoadingDescription](#) &description, [LoadSaveThread::AccessMode](#) mode=[LoadSaveThread::AccessModeReadWrite](#), [LoadingTaskStatus](#) loadingTaskStatus=[LoadingTaskStatusLoading](#))
- [LoadSaveThread::AccessMode](#) **accessMode** () const override
- void **addListener** ([LoadingProcessListener](#) \*const listener) override
- [QString](#) **cacheKey** () const override
- bool **completed** () const override
- [DImg](#) **img** () const
- [LoadSaveNotifier](#) \* **loadSaveNotifier** () const override
- bool **needsPostProcessing** () const
- void **notifyNewLoadingProcess** ([LoadingProcess](#) \*const process, const [LoadingDescription](#) &description) override
- void **progressInfo** (float progress) override
- bool **querySendNotifyEvent** () const override
- void **removeListener** ([LoadingProcessListener](#) \*const listener) override
- void **setResult** (const [LoadingDescription](#) &loadingDescription, const [DImg](#) &img) override

## Public Member Functions inherited from Digikam::LoadingTask

- **LoadingTask** ([LoadSaveThread](#) \*const thread, const [LoadingDescription](#) &description, [LoadingTaskStatus](#) loadingTaskStatus=[LoadingTaskStatusLoading](#))
- bool **continueQuery** () override
- [QString](#) **filePath** () const
- const [LoadingDescription](#) & **loadingDescription** () const
- void **setStatus** ([LoadingTaskStatus](#) status)
- [LoadingTaskStatus](#) **status** () const
- [TaskType](#) **type** () override

## Public Member Functions inherited from Digikam::LoadSaveTask

- **LoadSaveTask** ([LoadSaveThread](#) \*const thread)

## Public Member Functions inherited from Digikam::DImgLoaderObserver

- virtual float **granularity** ()  
*Return a relative value which determines the granularity, the frequency with which the [DImgLoaderObserver](#) is checked and progress is posted.*

## Additional Inherited Members

## Public Types inherited from Digikam::LoadingTask

- enum **LoadingTaskStatus** { [LoadingTaskStatusLoading](#) , [LoadingTaskStatusPreloading](#) , [LoadingTaskStatusStopping](#) }

## Public Types inherited from Digikam::LoadSaveTask

- enum **TaskType** { [TaskTypeLoading](#) , [TaskTypeSaving](#) }

### Protected Attributes inherited from [Digikam::SharedLoadingTask](#)

- [LoadSaveThread::AccessMode](#) **m\_accessMode** = [LoadSaveThread::AccessModeReadWrite](#)
- volatile bool **m\_completed** = false
- [DImg](#) **m\_img**
- [QList](#)< [LoadingProcessListener](#) \* > **m\_listeners**

### Protected Attributes inherited from [Digikam::LoadingTask](#)

- [LoadingDescription](#) **m\_loadingDescription**
- volatile [LoadingTaskStatus](#) **m\_loadingTaskStatus** = [LoadingTaskStatusLoading](#)

### Protected Attributes inherited from [Digikam::LoadSaveTask](#)

- [LoadSaveThread](#) \* **m\_thread** = nullptr

## 9.1322.1 Member Function Documentation

### 9.1322.1.1 execute()

```
void Digikam::ThumbnailLoadingTask::execute ( ) [override], [virtual]
```

Reimplemented from [Digikam::SharedLoadingTask](#).

### 9.1322.1.2 postProcess()

```
void Digikam::ThumbnailLoadingTask::postProcess ( ) [override], [virtual]
```

Reimplemented from [Digikam::SharedLoadingTask](#).



## Signals inherited from `Digikam::LoadSaveThread`

- void `signalImageLoaded` (const `LoadingDescription` &loadingDescription, const `DImg` &img)  
*This signal is emitted when the loading process has finished.*
- void `signalImageSaved` (const `QString` &filePath, bool success)
- void `signalImageStartedLoading` (const `LoadingDescription` &loadingDescription)  
*All signals are delivered to the thread from where the `LoadSaveThread` object has been created.*
- void `signalImageStartedSaving` (const `QString` &filePath)
- void `signalLoadingProgress` (const `LoadingDescription` &loadingDescription, float progress)  
*This signal is emitted whenever new progress info is available and the notification policy allows emitting the signal.*
- void `signalMoreCompleteLoadingAvailable` (const `LoadingDescription` &oldLoadingDescription, const `LoadingDescription` &newLoadingDescription)  
*This signal is emitted if.*
- void `signalSavingProgress` (const `QString` &filePath, float progress)
- void `signalThumbnailLoaded` (const `LoadingDescription` &loadingDescription, const `QImage` &img)

## Signals inherited from `Digikam::DynamicThread`

- void `finished` ()
- void `starting` ()  
*Emitted if `emitSignals` is enabled.*

## Public Member Functions

- `ThumbnailLoadThread` (`QObject` \*const parent=nullptr)
- void `find` (const `ThumbnailIdentifier` &identifier)  
*Find a thumbnail.*
- void `find` (const `ThumbnailIdentifier` &identifier, const `QRect` &rect)
- void `find` (const `ThumbnailIdentifier` &identifier, const `QRect` &rect, int size)
- bool `find` (const `ThumbnailIdentifier` &identifier, const `QRect` &rect, `QPixmap` &pixmap)  
*All tastes of `find()` methods, for loading the thumbnail of a detail.*
- bool `find` (const `ThumbnailIdentifier` &identifier, const `QRect` &rect, `QPixmap` &pixmap, int size, bool only↔  
Storage=false)
- void `find` (const `ThumbnailIdentifier` &identifier, int size)  
*Same as above, but does not use the global size, but an extra specified size.*
- bool `find` (const `ThumbnailIdentifier` &identifier, `QPixmap` &pixmap)  
*Find a thumbnail.*
- bool `find` (const `ThumbnailIdentifier` &identifier, `QPixmap` &pixmap, int size, bool onlyStorage=false)  
*Same as above, but does not use the global size, but an extra specified size.*
- bool `findBuffered` (const `ThumbnailIdentifier` &identifier, const `QRect` &rect, `QPixmap` &pixmap, int size)  
*Find the thumbnail pixmap in the buffered cache to avoid flickering while loading a new thumbnail.*
- void `findGroup` (const `QList`< `QPair`< `ThumbnailIdentifier`, `QRect` > > &filePathAndRects)
- void `findGroup` (const `QList`< `QPair`< `ThumbnailIdentifier`, `QRect` > > &filePathsAndRects, int size)
- void `findGroup` (`QList`< `ThumbnailIdentifier` > &identifiers)  
*Find a group of thumbnails.*
- void `findGroup` (`QList`< `ThumbnailIdentifier` > &identifiers, int size)
- `QList`< `LoadingDescription` > `lastDescriptions` () const  
*Returns the descriptions used by the last call to any of the above methods.*
- void `load` (const `LoadingDescription` &description)  
*Load a thumbnail.*
- int `pixmapToThumbnailSize` (int size) const

- Computes the thumbnail size for the give pixmap size.*

  - void [pregenerateGroup](#) (const QList< [ThumbnailIdentifier](#) > &identifiers)

*Pregenerate the thumbnail group.*
- void **pregenerateGroup** (const QList< [ThumbnailIdentifier](#) > &identifiers, int size)
- void [preload](#) (const [ThumbnailIdentifier](#) &identifier)
- Preload the thumbnail or thumbnail group.*

  - void **preload** (const [ThumbnailIdentifier](#) &identifier, int size)
  - void **preloadGroup** (QList< [ThumbnailIdentifier](#) > &identifiers)
  - void **preloadGroup** (QList< [ThumbnailIdentifier](#) > &identifiers, int size)
- void [setHighlightPixmap](#) (bool highlight)
- If you enable this, a highlighting border will be drawn around the pixmap.*

  - void [setPixmapRequested](#) (bool wantPixmap)

*If you enable this, the signal thumbnailLoaded(LoadingDescription, QPixmap) will be emitted.*
- void [setSendSurrogatePixmap](#) (bool send)
- If you enable this, the thread will try hard to send a pixmap if thumbnail loading failed.*

  - void [setThumbnailSize](#) (int size, bool forFace=false)

*Set the requested thumbnail size.*
- void [storeDetailThumbnail](#) (const QString &filePath, const QRect &detailRect, const QImage &image, bool isFace=false)
- Stores the given detail thumbnail on disk.*

  - int **storedSize** () const
  - [ThumbnailCreator](#) \* [thumbnailCreator](#) () const
  - int [thumbnailToPixmapSize](#) (int size) const

*Computes the pixmap size for the give thumbnail size.*

## Public Member Functions inherited from [Digikam::ManagedLoadSaveThread](#)

- **ManagedLoadSaveThread** (QObject \*const parent=nullptr)
- Termination is controlled by setting the TerminationPolicy Default is TerminationPolicyTerminateLoading.*

  - void [load](#) (const [LoadingDescription](#) &description)

*Append a task to load the given file to the task list.*
- void **load** (const [LoadingDescription](#) &description, [LoadingPolicy](#) policy)
- [LoadingPolicy](#) **loadingPolicy** () const
- void **save** (const [DImg](#) &image, const QString &filePath, const QString &format)
- Append a task to save the image to the task list.*

  - void [setLoadingPolicy](#) ([LoadingPolicy](#) policy)

*Set the loading policy.*
- void **setTerminationPolicy** ([TerminationPolicy](#) terminationPolicy)
- void **stopAllTasks** ()
- void **stopLoading** (const [LoadingDescription](#) &desc, [LoadingTaskFilter](#) filter=[LoadingTaskFilterAll](#))
- Same than previous method, but Stop and remove tasks filtered by LoadingDescription.*

  - void [stopLoading](#) (const QString &filePath=QString(), [LoadingTaskFilter](#) filter=[LoadingTaskFilterAll](#))

*Stop and remove tasks filtered by filePath and policy.*
- void [stopSaving](#) (const QString &filePath=QString())
- Stop and remove saving tasks filtered by filePath.*

  - [TerminationPolicy](#) **terminationPolicy** () const

## Public Member Functions inherited from [Digikam::LoadSaveThread](#)

- **LoadSaveThread** (QObject \*const parent=nullptr)
- **~LoadSaveThread** () override
  - Destructor: The thread will execute all pending tasks and wait for this upon destruction.*
- void **imageLoaded** (const [LoadingDescription](#) &loadingDescription, const [DImg](#) &img) override
- void **imageSaved** (const QString &filePath, bool success) override
- void **imageStartedLoading** (const [LoadingDescription](#) &loadingDescription) override
- void **imageStartedSaving** (const QString &filePath) override
- void **load** (const [LoadingDescription](#) &description)
  - Append a task to load the given file to the task list.*
- void **loadingProgress** (const [LoadingDescription](#) &loadingDescription, float progress) override
- void **moreCompleteLoadingAvailable** (const [LoadingDescription](#) &oldLoadingDescription, const [LoadingDescription](#) &newLoadingDescription) override
- virtual bool **querySendNotifyEvent** () const
- void **save** (const [DImg](#) &image, const QString &filePath, const QString &format)
  - Append a task to save the image to the task list.*
- void **savingProgress** (const QString &filePath, float progress) override
- void **setNotificationPolicy** ([NotificationPolicy](#) notificationPolicy)
- virtual void **taskHasFinished** ()

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- **DynamicThread** (QObject \*const parent=nullptr)
  - This class extends QRunnable, so you have to reimplement virtual void run().*
- **~DynamicThread** () override
  - The destructor calls stop() and wait(), but if you, in your destructor, delete any data that is accessed by your run() method, you must call stop() and wait() before yourself.*
- bool **isFinished** () const
- bool **isRunning** () const
- QThread::Priority **priority** () const
- void **setEmitSignals** (bool emitThem)
- void **setPriority** (QThread::Priority priority)
  - Sets the priority for this dynamic thread.*
- State **state** () const

## Static Public Member Functions

- static void **cleanUp** ()
- static [ThumbnailLoadThread](#) \* **defaultIconViewThread** ()
- static [ThumbnailLoadThread](#) \* **defaultThread** ()
  - Return application-wide default thumbnail threads.*
- static void **deleteThumbnail** (const QString &filePath)
  - This is a tool to force regeneration of thumbnails.*
- static void **initializeNoThumbnailStorage** ()
  - Disable storing thumbnails in the disk cache.*
- static void **initializeThumbnailDatabase** (const [DbEngineParameters](#) &params, [ThumbnailInfoProvider](#) \*const provider=nullptr)
  - Enable loading of thumbnails from a thumbnail database.*
- static int **maximumThumbnailPixmapSize** (bool withHighlighting)
- static int **maximumThumbnailSize** ()
  - Returns the maximum possible size of a thumbnail.*
- static void **setDisplayingWidget** (QWidget \*const widget)
  - For color management, this sets the widget the thumbnails will be color managed for.*
- static int **thumbnailToPixmapSize** (bool withHighlight, int size)



## Static Public Member Functions inherited from Digikam::LoadSaveThread

- static int **exifOrientation** (const QString &filePath, const [DMetadata](#) &metadata, bool isRaw, bool fromRaw↔ EmbeddedPreview)  
*Retrieves the Exif orientation, either from the info provider if available, or from the metadata.*
- static [LoadSaveFileInfoProvider](#) \* **infoProvider** ()
- static void **setInfoProvider** ([LoadSaveFileInfoProvider](#) \*const infoProvider)

## Protected Member Functions

- void **thumbnailLoaded** (const [LoadingDescription](#) &loadingDescription, const QImage &img) override  
*virtual method overridden from [LoadSaveNotifier](#), implemented first by [LoadSaveThread](#) called by [ThumbnailTask](#) from working thread*

## Protected Member Functions inherited from Digikam::ManagedLoadSaveThread

- void **load** (const [LoadingDescription](#) &description, [LoadingMode](#) loadingMode, [AccessMode](#) mode=[AccessModeReadWrite](#))
- void **load** (const [LoadingDescription](#) &description, [LoadingMode](#) loadingMode, [LoadingPolicy](#) policy, [AccessMode](#) mode=[AccessModeReadWrite](#))
- void **loadPreview** (const [LoadingDescription](#) &description, [LoadingPolicy](#) policy)
- void **loadThumbnail** (const [LoadingDescription](#) &description)
- void **preloadThumbnail** (const [LoadingDescription](#) &description)
- void **preloadThumbnailGroup** (const QList< [LoadingDescription](#) > &descriptions)
- void **prependThumbnailGroup** (const QList< [LoadingDescription](#) > &descriptions)
- void **shutDown** ()

## Protected Member Functions inherited from Digikam::LoadSaveThread

- void **notificationReceived** ()
- void **run** () override  
*Implement this pure virtual function in your subclass.*

## Protected Member Functions inherited from Digikam::DynamicThread

- bool **runningFlag** () const volatile  
*In you [run\(\)](#) method, you shall regularly check for [runningFlag\(\)](#) and cleanup and return if false.*
- void **shutDown** ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call [stop\(\)](#) and [wait\(\)](#), knowing that nothing will call [start\(\)](#) anymore after this 3) Be sure the thread will never be running at destruction.*
- void **start** (QMutexLocker< QMutex > &locker)  
*Doing the same as [start\(\)](#), [stop\(\)](#) and [wait](#) above, provide it with a locked QMutexLocker on mutex().*
- void **stop** (const QMutexLocker< QMutex > &locker)
- QMutex \* **threadMutex** () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void **wait** (QMutexLocker< QMutex > &locker)

## Additional Inherited Members

### Public Types inherited from [Digikam::ManagedLoadSaveThread](#)

- enum [LoadingMode](#) { [LoadingModeNormal](#) , [LoadingModeShared](#) }  
*used by [SharedLoadSaveThread](#) only*
- enum [LoadingPolicy](#) { [LoadingPolicyFirstRemovePrevious](#) , [LoadingPolicyPrepend](#) , [LoadingPolicySimplePrepend](#) , [LoadingPolicyAppend](#) ,  
[LoadingPolicySimpleAppend](#) , [LoadingPolicyPreload](#) }
- enum [LoadingTaskFilter](#) { [LoadingTaskFilterAll](#) , [LoadingTaskFilterPreloading](#) }
- enum [TerminationPolicy](#) { [TerminationPolicyTerminateLoading](#) , [TerminationPolicyTerminatePreloading](#) ,  
[TerminationPolicyWait](#) , [TerminationPolicyTerminateAll](#) }

### Public Types inherited from [Digikam::LoadSaveThread](#)

- enum [AccessMode](#) { [AccessModeRead](#) , [AccessModeReadWrite](#) }  
*used by [SharedLoadSaveThread](#) only*
- enum [NotificationPolicy](#) { [NotificationPolicyDirect](#) , [NotificationPolicyTimeLimited](#) }

### Public Types inherited from [Digikam::DynamicThread](#)

- enum [State](#) { [Inactive](#) , [Scheduled](#) , [Running](#) , [Deactivating](#) }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void [start](#) ()
- void [stop](#) ()  
*Stop computation, sets the running flag to false.*
- void [wait](#) ()  
*Waits until the thread finishes.*

### Protected Attributes inherited from [Digikam::ManagedLoadSaveThread](#)

- [LoadingPolicy](#) [m\\_loadingPolicy](#) = [LoadingPolicyAppend](#)
- [TerminationPolicy](#) [m\\_terminationPolicy](#) = [TerminationPolicyTerminateLoading](#)

### Protected Attributes inherited from [Digikam::LoadSaveThread](#)

- [LoadSaveTask](#) \* [m\\_currentTask](#) = nullptr
- [QMutex](#) [m\\_mutex](#)
- [NotificationPolicy](#) [m\\_notificationPolicy](#) = [NotificationPolicyTimeLimited](#)
- [QList](#)< [LoadSaveTask](#) \* > [m\\_todo](#)

## 9.1323.1 Member Function Documentation

### 9.1323.1.1 defaultThread()

```
ThumbnailLoadThread * Digikam::ThumbnailLoadThread::defaultThread ( ) [static]
```

It is perfectly all right to create an extra object of the class, but it is useful to have default object

### 9.1323.1.2 deleteThumbnail()

```
void Digikam::ThumbnailLoadThread::deleteThumbnail (
    const QString & filePath ) [static]
```

All thumbnail files for the given file will be removed from disk, and the cached instances will be removed as well. Use this method if you know that the contents of the file has changed. This method works independently from the multithreaded thumbnail loading.

### 9.1323.1.3 find() [1/2]

```
void Digikam::ThumbnailLoadThread::find (
    const ThumbnailIdentifier & identifier )
```

This method sends the signals and does not return values like the method above. If you certainly need asynchronous return, connect with Qt::QueuedConnection to the signals. If you connect directly, the signals may be sent from within the method call.

### 9.1323.1.4 find() [2/2]

```
bool Digikam::ThumbnailLoadThread::find (
    const ThumbnailIdentifier & identifier,
    QPixmap & pixmap )
```

If the pixmap is found in the cache, returns true and sets pixmap to the found QPixmap. If the pixmap is not found in the cache, [load\(\)](#) is called to start the loading process, false is returned and pixmap is not touched.

### 9.1323.1.5 findGroup()

```
void Digikam::ThumbnailLoadThread::findGroup (
    QList< ThumbnailIdentifier > & identifiers )
```

The items will be loaded in order and signals will be sent. Can be used to ensure that thumbnails are loaded in a particular order

### 9.1323.1.6 initializeNoThumbnailStorage()

```
void Digikam::ThumbnailLoadThread::initializeNoThumbnailStorage ( ) [static]
```

This shall be called once at application startup. This need not be called, then the FreeDesktop standard is used.

### 9.1323.1.7 initializeThumbnailDatabase()

```
void Digikam::ThumbnailLoadThread::initializeThumbnailDatabase (
    const DbEngineParameters & params,
    ThumbnailInfoProvider *const provider = nullptr ) [static]
```

This shall be called once at application startup. This need not be called, then the FreeDesktop standard is used. You can optionally provide a thumbnail info provider.

### 9.1323.1.8 lastDescriptions()

```
QList< LoadingDescription > Digikam::ThumbnailLoadThread::lastDescriptions ( ) const
```

After calling single-thumbnail methods (find, preload) the list will have size 1, after the group methods (findGroup, preloadGroup, pregenerateGroup) the list can be larger than 1. There is no information if the description was ever scheduled in the thread, already processed, skipped or canceled.

### 9.1323.1.9 load()

```
void Digikam::ThumbnailLoadThread::load (
    const LoadingDescription & description )
```

You do not need to use this method directly, it will not access the pixmap cache. Use [find\(\)](#). The [LoadingDescription](#) shall be constructed with the constructor for preview/thumbnail jobs. (in the description constructor, you need to specify file path, thumbnail size and Exif rotation)

### 9.1323.1.10 maximumThumbnailSize()

```
int Digikam::ThumbnailLoadThread::maximumThumbnailSize ( ) [static]
```

If you request a larger size, the thumbnail will not load. The size of the pixmap can slightly differ, especially when highlighting.

### 9.1323.1.11 pregenerateGroup()

```
void Digikam::ThumbnailLoadThread::pregenerateGroup (
    const QList< ThumbnailIdentifier > & identifiers )
```

No signals will be emitted when these are loaded.

### 9.1323.1.12 preload()

```
void Digikam::ThumbnailLoadThread::preload (
    const ThumbnailIdentifier & identifier )
```

This is essentially the same as loading, but with a lower priority.

### 9.1323.1.13 setDisplayingWidget()

```
void Digikam::ThumbnailLoadThread::setDisplayingWidget (
    QWidget *const widget ) [static]
```

(currently it is only possible to set one global widget)

### 9.1323.1.14 setHighlightPixmap()

```
void Digikam::ThumbnailLoadThread::setHighlightPixmap (
    bool highlight )
```

This option has only an effect if  `pixmapRequested`  is true. Default value: Enabled.

### 9.1323.1.15 setPixmapRequested()

```
void Digikam::ThumbnailLoadThread::setPixmapRequested (
    bool wantPixmap )
```

If you do not enable this, only the QImage-based signal (see [LoadSaveThread](#)) will be emitted.

If you disable this, pay attention to the (global) setting of the [LoadingCache](#), which per default does not cache the images !!

Default value: Enabled.

### 9.1323.1.16 setSendSurrogatePixmap()

```
void Digikam::ThumbnailLoadThread::setSendSurrogatePixmap (
    bool send )
```

It will use standard system icons to replace the real thumbnail. If you disable this, a null QPixmap will be sent. This does not influence the QImage-based signal; this signal will be emitted with a null QImage regardless of this setting here, if the loading failed. Default value: Enabled.

### 9.1323.1.17 setThumbnailSize()

```
void Digikam::ThumbnailLoadThread::setThumbnailSize (
    int size,
    bool forFace = false )
```

#### Note

If the thread is currently loading thumbnails, there is no guarantee as to when the property change by one of the following methods takes effect. Default value: 128

### 9.1323.1.18 `signalThumbnailLoaded`

```
void Digikam::ThumbnailLoadThread::signalThumbnailLoaded (
    const LoadingDescription & loadingDescription,
    const QPixmap & pix ) [signal]
```

#### Note

See [LoadSaveThread](#) for a QImage-based `thumbnailLoaded()` signal.

### 9.1323.1.19 `storeDetailThumbnail()`

```
void Digikam::ThumbnailLoadThread::storeDetailThumbnail (
    const QString & filePath,
    const QRect & detailRect,
    const QImage & image,
    bool isFace = false )
```

Use this if possible because generation of detail thumbnails is potentially slower. The image should at least have `storedSize()`.

### 9.1323.1.20 `thumbnailCreator()`

```
ThumbnailCreator * Digikam::ThumbnailLoadThread::thumbnailCreator ( ) const
```

#### Note

For internal use - may only be used from the thread

### 9.1323.1.21 `thumbnailLoaded()`

```
void Digikam::ThumbnailLoadThread::thumbnailLoaded (
    const LoadingDescription & loadingDescription,
    const QImage & img ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::LoadSaveThread](#).

### 9.1323.1.22 `thumbnailsAvailable`

```
void Digikam::ThumbnailLoadThread::thumbnailsAvailable ( ) [signal]
```

#### Note

For internal use only.

### 9.1323.1.23 thumbnailToPixmapSize()

```
int Digikam::ThumbnailLoadThread::thumbnailToPixmapSize (
    int size ) const
```

These can differ when highlighting is turned on.

## 9.1324 Digikam::ThumbnailSize Class Reference

### Public Types

- enum [Size](#) {
 **Step** = 8 , **Tiny** = 32 , **VerySmall** = 64 , **MediumSmall** = 80 ,
 **Small** = 100 , **Medium** = 142 , **Large** = 160 , **Huge** = 256 ,
 **HD** = 512 , **MAX** = 1024 }

### Public Member Functions

- **ThumbnailSize** (const [ThumbnailSize](#) &thumbsize)
- **ThumbnailSize** (int size)
- bool **operator!=** (const [ThumbnailSize](#) &thumbsize) const
- [ThumbnailSize](#) & **operator=** (const [ThumbnailSize](#) &thumbsize)
- bool **operator==** (const [ThumbnailSize](#) &thumbsize) const
- int **size** () const

### Static Public Member Functions

- static bool **getUseLargeThumbs** ()
- static int **maxThumbsSize** ()
- static void **readSettings** (const KConfigGroup &group)
- static void **saveSettings** (KConfigGroup &group, bool val)
- static void **setUseLargeThumbs** (bool val)

## 9.1324.1 Member Enumeration Documentation

### 9.1324.1.1 Size

```
enum Digikam::ThumbnailSize::Size
```

#### Enumerator

Small	Most usable small size of thumbnails to prevent overloaded overlays show under thumbs (as Pick label and Group indicator) See bugs #321337 and #275381 for details.
-------	---

## 9.1325 Digikam::ThumbsDb Class Reference

### Public Member Functions

- `QList< int > findAll ()`  
*Returns the thumbnail ids of all thumbnails in the database.*
- `ThumbsDbInfo findByCustomIdentifier (const QString &id)`
- `ThumbsDbInfo findByFilePath (const QString &path)`
- `ThumbsDbInfo findByFilePath (const QString &path, const QString &uniqueHash)`  
*This is findByFilePath with extra security: Pass the uniqueHash which you have.*
- `ThumbsDbInfo findByHash (const QString &uniqueHash, qlonglong fileSize)`
- `QHash< QString, int > getFilePathsWithThumbnail ()`
- `QString getLegacySetting (const QString &keyword)`
- `QString getSetting (const QString &keyword)`
- `BdEngineBackend::QueryState insertCustomIdentifier (const QString &id, int thumbId)`
- `BdEngineBackend::QueryState insertFilePath (const QString &path, int thumbId)`
- `BdEngineBackend::QueryState insertThumbnail (const ThumbsDbInfo &info, QVariant *const lastInsertId=nullptr)`
- `BdEngineBackend::QueryState insertUniqueHash (const QString &uniqueHash, qlonglong fileSize, int thumbId)`
- `bool integrityCheck ()`  
*Returns true if the integrity of the database is preserved.*
- `BdEngineBackend::QueryState remove (int thumbId)`
- `BdEngineBackend::QueryState removeByCustomIdentifier (const QString &id)`
- `BdEngineBackend::QueryState removeByFilePath (const QString &path)`  
*Removes thumbnail data associated to the given file path.*
- `BdEngineBackend::QueryState removeByUniqueHash (const QString &uniqueHash, qlonglong fileSize)`  
*Removes thumbnail data associated to the given uniqueHash/fileSize.*
- `BdEngineBackend::QueryState renameByFilePath (const QString &oldPath, const QString &newPath)`
- `BdEngineBackend::QueryState replaceThumbnail (const ThumbsDbInfo &info)`
- `void replaceUniqueHash (const QString &oldUniqueHash, int oldFileSize, const QString &newUniqueHash, int newFileSize)`
- `bool setSetting (const QString &keyword, const QString &value)`
- `BdEngineBackend::QueryState updateModificationDate (int thumbId, const QDateTime &modificationDate)`
- `void vacuum ()`  
*Shrinks the database.*

### Friends

- class `ThumbsDbAccess`

### 9.1325.1 Member Function Documentation

#### 9.1325.1.1 findByFilePath()

```
ThumbsDbInfo Digikam::ThumbsDb::findByFilePath (
    const QString & path,
    const QString & uniqueHash )
```

If an entry is found by file path, and the entry is referenced by any uniqueHash, which is different from the given hash, a null info is returned. If uniqueHash is null, equivalent to the simple findByFilePath.



## 9.1326 Digikam::ThumbsDbAccess Class Reference

### Public Member Functions

- [ThumbsDbAccess](#) ()  
*This class is written in analogy to [CoreDbAccess](#) (some features stripped off).*
- [ThumbsDbBackend](#) \* **backend** () const
- [ThumbsDb](#) \* **db** () const
- QString **lastError** () const
- void [setLastError](#) (const QString &error)  
*Set the "last error" message.*

### Static Public Member Functions

- static bool **checkReadyForUse** ([InitializationObserver](#) \*const observer)
- static void **cleanUpDatabase** ()
- static void **initDbEngineErrorHandler** ([DbEngineErrorHandler](#) \*const errorHandler)
- static bool **isInitialized** ()
- static [DbEngineParameters](#) **parameters** ()
- static void **setParameters** (const [DbEngineParameters](#) &parameters)

### 9.1326.1 Constructor & Destructor Documentation

#### 9.1326.1.1 ThumbsDbAccess()

```
Digikam::ThumbsDbAccess::ThumbsDbAccess ( )
```

For documentation, see `coredbaccess.h`

### 9.1326.2 Member Function Documentation

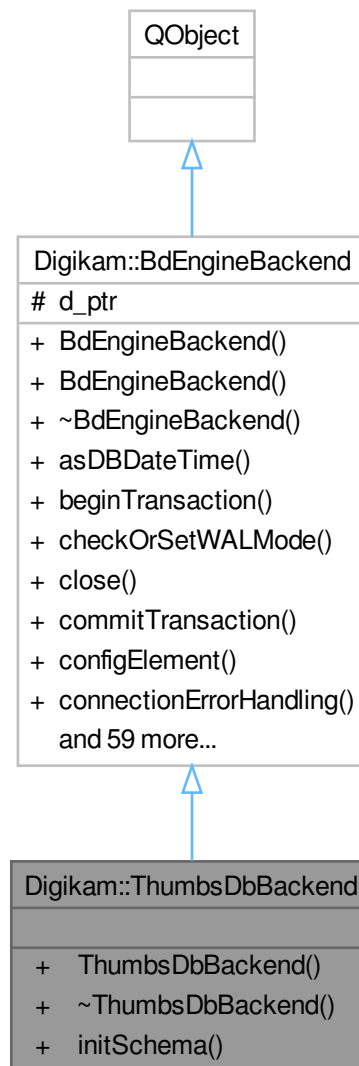
#### 9.1326.2.1 setLastError()

```
void Digikam::ThumbsDbAccess::setLastError (
    const QString & error )
```

This method is not for public use.

## 9.1327 Digikam::ThumbsDbBackend Class Reference

Inheritance diagram for Digikam::ThumbsDbBackend:



### Public Member Functions

- **ThumbsDbBackend** ([DbEngineLocking](#) \*const locking, const QString &backendName=QLatin1↳String("thumbnailDatabase-"))
- bool **initSchema** ([ThumbsDbSchemaUpdater](#) \*const updater)

*Initialize the database schema to the current version, carry out upgrades if necessary.*

## Public Member Functions inherited from Digikam::BdEngineBackend

- [BdEngineBackend](#) (const QString &backendName, [DbEngineLocking](#) \*const locking)
 

*Creates a database backend.*
- **BdEngineBackend** (const QString &backendName, [DbEngineLocking](#) \*const locking, [BdEngineBackend](#)←Private &dd)
- QDateTime [asDBDateTime](#) (const QDateTime &dateTime) const
 

*Depending on the database backend return a local or UTC date format.*
- [BdEngineBackend::QueryState](#) **beginTransaction** ()
 

*Begin a database transaction.*
- bool [checkOrSetWALMode](#) ()
 

*Check or set WAL mode for SQLite database if enabled in settings.*
- void **close** ()
 

*Close the database connection.*
- [BdEngineBackend::QueryState](#) **commitTransaction** ()
 

*Commit the current database transaction.*
- [DbEngineConfigSettings](#) **configElement** () const
 

*Return config read from XML, corresponding to this backend's database type.*
- bool [connectionErrorHandling](#) (int retries)
 

*Called when an attempted connection to the database failed.*
- [DbEngineSqlQuery](#) **copyQuery** (const [DbEngineSqlQuery](#) &old)
 

*Creates a faithful copy of the passed query, with the current db connection.*
- DbType **databaseType** () const
 

*Return the database type.*
- bool **exec** ([DbEngineSqlQuery](#) &query)
 

*Calls exec/execBatch on the query, and handles debug output if something went wrong.*
- bool **execBatch** ([DbEngineSqlQuery](#) &query)
- [QueryState](#) **execDBAction** (const [DbEngineAction](#) &action, const QMap< QString, QVariant > &bindingMap, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
 

*Performs the database action on the current database.*
- [QueryState](#) **execDBAction** (const [DbEngineAction](#) &action, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
 

*Performs the database action on the current database.*
- [QueryState](#) **execDBAction** (const QString &action, const QMap< QString, QVariant > &bindingMap, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
 

*Performs the database action on the current database.*
- [QueryState](#) **execDBAction** (const QString &action, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
 

*Performs the database action on the current database.*
- QSqlQuery [execDBActionQuery](#) (const [DbEngineAction](#) &action, const QMap< QString, QVariant > &bindingMap)
 

*Performs the database action on the current database.*
- QSqlQuery **execDBActionQuery** (const QString &action, const QMap< QString, QVariant > &bindingMap)
- [QueryState](#) **execDirectSql** (const QString &query)
 

*Calls exec on the query, and handles debug output if something went wrong.*
- [QueryState](#) **execDirectSqlWithResult** (const QString &query, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
 

*Calls exec on the query, and handles debug output if something went wrong.*
- [DbEngineSqlQuery](#) **execQuery** (const QString &sql)
 

*Executes the statement and returns the query object.*
- [DbEngineSqlQuery](#) **execQuery** (const QString &sql, const QList< QVariant > &boundValues)
- [DbEngineSqlQuery](#) **execQuery** (const QString &sql, const QMap< QString, QVariant > &bindingMap)
 

*Method which accept a hashmap with key, values which are used for named binding.*
- [DbEngineSqlQuery](#) **execQuery** (const QString &sql, const QVariant &bindValue1)

- [DbEngineSqlQuery](#) **execQuery** (const QString &sql, const QVariant &bindValue1, const QVariant &bindValue2)
- [DbEngineSqlQuery](#) **execQuery** (const QString &sql, const QVariant &bindValue1, const QVariant &bindValue2, const QVariant &bindValue3)
- [DbEngineSqlQuery](#) **execQuery** (const QString &sql, const QVariant &bindValue1, const QVariant &bindValue2, const QVariant &bindValue3, const QVariant &bindValue4)
- void **execQuery** ([DbEngineSqlQuery](#) &preparedQuery, const QList< QVariant > &boundValues)
- void **execQuery** ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &bindValue1)
  - Binds the values and executes the prepared query.*
- void **execQuery** ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &bindValue1, const QVariant &bindValue2)
- void **execQuery** ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &bindValue1, const QVariant &bindValue2, const QVariant &bindValue3)
- void **execQuery** ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &bindValue1, const QVariant &bindValue2, const QVariant &bindValue3, const QVariant &bindValue4)
- [QueryState](#) **execSql** (const QString &sql, const QList< QVariant > &boundValues, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** (const QString &sql, const QMap< QString, QVariant > &bindingMap, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
  - Method which accepts a map for named binding.*
- [QueryState](#) **execSql** (const QString &sql, const QVariant &bindValue1, const QVariant &bindValue2, const QVariant &bindValue3, const QVariant &bindValue4, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** (const QString &sql, const QVariant &bindValue1, const QVariant &bindValue2, const QVariant &bindValue3, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** (const QString &sql, const QVariant &bindValue1, const QVariant &bindValue2, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** (const QString &sql, const QVariant &bindValue1, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** (const QString &sql, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
  - Executes the SQL statement, and write the returned data into the values list.*
- [QueryState](#) **execSql** ([DbEngineSqlQuery](#) &preparedQuery, const QList< QVariant > &boundValues, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &bindValue1, const QVariant &bindValue2, const QVariant &bindValue3, const QVariant &bindValue4, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &bindValue1, const QVariant &bindValue2, const QVariant &bindValue3, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &bindValue1, const QVariant &bindValue2, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** ([DbEngineSqlQuery](#) &preparedQuery, const QVariant &bindValue1, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execSql** ([DbEngineSqlQuery](#) &preparedQuery, QList< QVariant > \*const values=nullptr, QVariant \*const lastInsertId=nullptr)
- [QueryState](#) **execUpsertDBAction** (const [DbEngineAction](#) &action, const QVariant &id, const QStringList &fieldNames, const QList< QVariant > &values)
  - Performs a special DBAction that is usually needed to "INSERT or UPDATE" entries in a table.*
- [QueryState](#) **execUpsertDBAction** (const QString &action, const QVariant &id, const QStringList &fieldNames, const QList< QVariant > &values)
- [DbEngineAction](#) **getDBAction** (const QString &actionName) const
  - Returns a database action with name, specified in actionName, for the current database.*
- [DbEngineSqlQuery](#) **getQuery** ()

- Creates an empty query object waiting for the statement.*

  - [QueryState handleQueryResult](#) ([DbEngineSqlQuery](#) &query, [QList< QVariant >](#) \*const values, [QVariant](#) \*const lastInsertId)
- Checks if there was a connection error.*

  - bool **isCompatible** (const [DbEngineParameters](#) &parameters)
- Checks if the parameters can be used for this database backend.*

  - bool **isInTransaction** () const
- Returns if the database is in a different thread in a transaction.*

  - bool **isOpen** () const
  - bool **isReady** () const
  - [QString lastError](#) ()
- Returns a description of the last error that occurred on this database.*

  - [QSqlError lastSQLError](#) ()
- Returns the last error that occurred on this database.*

  - int **maximumBoundValues** () const
- Returns the maximum number of bound parameters allowed per query.*

  - bool **open** (const [DbEngineParameters](#) &parameters)
- Open the database connection.*

  - [DbEngineSqlQuery prepareQuery](#) (const [QString](#) &sql)
- Creates a query object prepared with the statement, waiting for bound values.*

  - bool **queryErrorHandling** ([DbEngineSqlQuery](#) &query, int retries)
- Called with a failed query.*

  - [QList< QVariant >](#) **readToList** ([DbEngineSqlQuery](#) &query)
- Reads data of returned result set into a list which is returned.*

  - void **rollbackTransaction** ()
- Rollback the current database transaction.*

  - void **setDbEngineErrorHandler** ([DbEngineErrorHandler](#) \*const handler)
- Add a [DbEngineErrorHandler](#).*

  - void **setForeignKeyChecks** (bool check)
- Enables or disables FOREIGN\_KEY\_CHECKS for the database.*

  - [Status status](#) () const
- Returns the current status of the database backend.*

  - [QStringList tables](#) ()
- Returns a list with the names of tables in the database.*

  - bool **transactionErrorHandling** (const [QSqlError](#) &[lastError](#), int retries)

### Additional Inherited Members

### Public Types inherited from [Digikam::BdEngineBackend](#)

- enum **DbType** { [SQLite](#) , [MySQL](#) }
- enum **QueryOperationStatus** { [ExecuteNormal](#) , [Wait](#) , [AbortQueries](#) }
- enum **QueryStateEnum** { [NoErrors](#) , [SQLError](#) , [ConnectionError](#) }
- enum **Status** { [Unavailable](#) , [Open](#) , [OpenSchemaChecked](#) }

### Protected Attributes inherited from [Digikam::BdEngineBackend](#)

- [BdEngineBackendPrivate](#) \*const **d\_ptr** = nullptr

## 9.1327.1 Member Function Documentation

### 9.1327.1.1 initSchema()

```
bool Digikam::ThumbsDbBackend::initSchema (
    ThumbsDbSchemaUpdater *const updater )
```

Shall only be called from the thread that called [open\(\)](#).

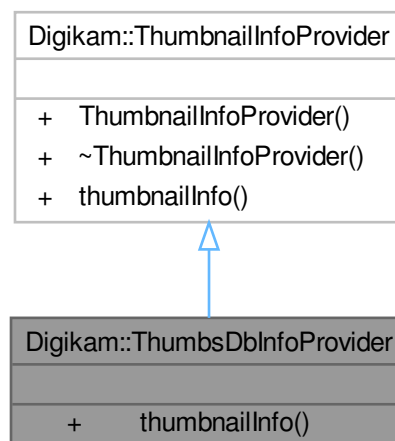
## 9.1328 Digikam::ThumbsDbInfo Class Reference

### Public Attributes

- QByteArray **data**
- int **id** = -1
- QDateTime **modificationDate**
- int **orientationHint** = 0
- DatabaseThumbnail::Type **type** = DatabaseThumbnail::UndefinedType

## 9.1329 Digikam::ThumbsDbInfoProvider Class Reference

Inheritance diagram for Digikam::ThumbsDbInfoProvider:



### Public Member Functions

- [ThumbnailInfo thumbnailInfo](#) (const [ThumbnailIdentifier](#) &identifier) override

## 9.1329.1 Member Function Documentation

### 9.1329.1.1 thumbnailInfo()

```
ThumbnailInfo Digikam::ThumbsDbInfoProvider::thumbnailInfo (  
    const ThumbnailIdentifier & identifier ) [override], [virtual]
```

Implements [Digikam::ThumbnailInfoProvider](#).

## 9.1330 Digikam::ThumbsDbSchemaUpdater Class Reference

### Public Member Functions

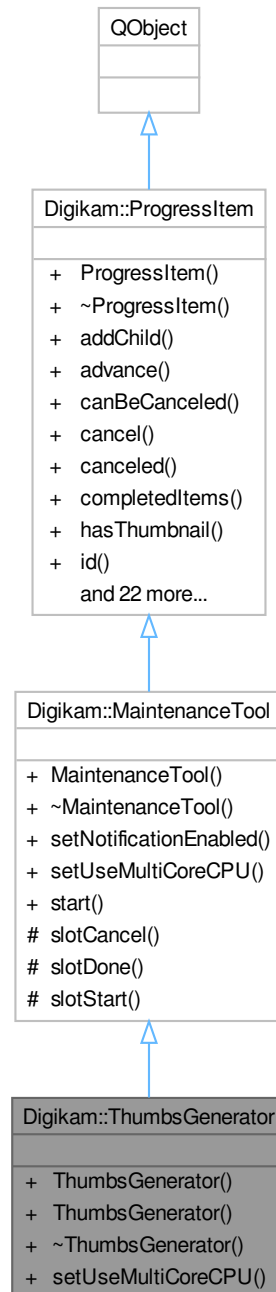
- **ThumbsDbSchemaUpdater** ([ThumbsDbAccess](#) \*const dbAccess)
- void **setObserver** ([InitializationObserver](#) \*const observer)
- bool **update** ()

### Static Public Member Functions

- static int **schemaVersion** ()

## 9.1331 Digikam::ThumbsGenerator Class Reference

Inheritance diagram for Digikam::ThumbsGenerator:



### Public Member Functions

- [ThumbsGenerator](#) (const bool rebuildAll, const AlbumList &list, [ProgressItem](#) \*const parent=nullptr)  
*Constructor using AlbumList as argument.*



- **ThumbsGenerator** (const bool rebuildAll, int albumId, [ProgressItem](#) \*const parent=nullptr)  
*Constructor using Album Id as argument.*
- void **setUseMultiCoreCPU** (bool b) override  
*Re-implement this method if your tool is able to use multi-core CPU to process item in parallel.*

### Public Member Functions inherited from [Digikam::MaintenanceTool](#)

- **MaintenanceTool** (const QString &id, [ProgressItem](#) \*const parent=nullptr)
- void **setNotificationEnabled** (bool b)  
*If true, show a notification message on desktop notification manager with time elapsed to run process.*

### Public Member Functions inherited from [Digikam::ProgressItem](#)

- **ProgressItem** ([ProgressItem](#) \*const parent, const QString &id, const QString &label, const QString &status, bool canBeCanceled, bool hasThumb)
- void **addChild** ([ProgressItem](#) \*const kiddo)
- bool **advance** (unsigned int v)  
*Advance total items processed by n values and update percentage in progressbar.*
- bool **canBeCanceled** () const
- void **cancel** ()
- bool **canceled** () const
- unsigned int **completedItems** () const
- bool **hasThumbnail** () const
- const QString & **id** () const
- bool **incCompletedItems** (unsigned int v=1)
- void **incTotalItems** (unsigned int v=1)
- const QString & **label** () const
- [ProgressItem](#) \* **parent** () const
- unsigned int **progress** () const
- void **removeChild** ([ProgressItem](#) \*const kiddo)
- void **reset** ()  
*Reset the progress value of this item to 0 and the status string to the empty string.*
- void **setComplete** ()  
*Tell the item it has finished.*
- bool **setCompletedItems** (unsigned int v)
- void **setLabel** (const QString &v)
- void **setProgress** (unsigned int v)  
*Set the progress (percentage of completion) value of this item.*
- void **setShowAtStart** (bool showAtStart)  
*Set the property to pop-up item when it's added in progress manager.*
- void **setStatus** (const QString &v)  
*Set the string to be used for showing this item's current status.*
- void **setThumbnail** (const QIcon &icon)  
*Sets whether this item has a thumbnail.*
- void **setTotalItems** (unsigned int v)
- void **setUsesBusyIndicator** (bool useBusyIndicator)  
*Sets whether this item uses a busy indicator instead of real progress for its progress bar.*
- bool **showAtStart** () const
- const QString & **status** () const
- bool **totalCompleted** () const
- unsigned int **totalItems** () const
- void **updateProgress** ()  
*Recalculate progress according to total/completed items and update.*
- bool **usesBusyIndicator** () const

## Additional Inherited Members

### Public Slots inherited from [Digikam::MaintenanceTool](#)

- void **start** ()

### Signals inherited from [Digikam::MaintenanceTool](#)

- void **signalCanceled** ()  
*Emit when process is canceled.*
- void **signalComplete** ()  
*Emit when process is done (not canceled).*

### Signals inherited from [Digikam::ProgressItem](#)

- void [progressItemAdded](#) ([ProgressItem](#) \*item)  
*Emitted when a new [ProgressItem](#) is added.*
- void [progressItemCanceled](#) ([ProgressItem](#) \*item)  
*Emitted when an item was canceled.*
- void **progressItemCanceledById** (const QString &id)
- void [progressItemCompleted](#) ([ProgressItem](#) \*item)  
*Emitted when a progress item was completed.*
- void [progressItemLabel](#) ([ProgressItem](#) \*item, const QString &label)  
*Emitted when the label of an item changed.*
- void [progressItemProgress](#) ([ProgressItem](#) \*item, unsigned int v)  
*Emitted when the progress value of an item changes.*
- void [progressItemStatus](#) ([ProgressItem](#) \*item, const QString &mess)  
*Emitted when the status message of an item changed.*
- void [progressItemThumbnail](#) ([ProgressItem](#) \*item, const QPixmap &thumb)  
*Emitted when the thumbnail data must be set in item.*
- void [progressItemUsesBusyIndicator](#) ([ProgressItem](#) \*item, bool value)  
*Emitted when the busy indicator state of an item changes.*

### Protected Slots inherited from [Digikam::MaintenanceTool](#)

- virtual void **slotCancel** ()
- virtual void **slotDone** ()
- virtual void **slotStart** ()

## 9.1331.1 Constructor & Destructor Documentation

### 9.1331.1.1 ThumbsGenerator() [1/2]

```
Digikam::ThumbsGenerator::ThumbsGenerator (
    const bool rebuildAll,
    int albumId,
    ProgressItem *const parent = nullptr ) [explicit]
```

If Id = -1, whole Albums collection is processed.

## 9.1331.1.2 ThumbsGenerator() [2/2]

```
Digikam::ThumbsGenerator::ThumbsGenerator (
    const bool rebuildAll,
    const AlbumList & list,
    ProgressItem *const parent = nullptr )
```

If list is empty, whole Albums collection is processed.

## 9.1331.2 Member Function Documentation

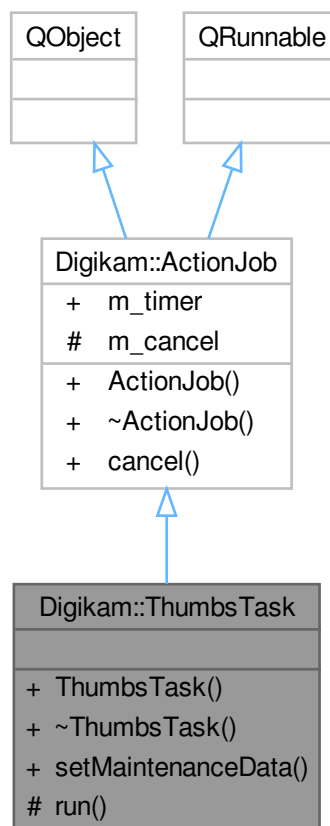
## 9.1331.2.1 setUseMultiCoreCPU()

```
void Digikam::ThumbsGenerator::setUseMultiCoreCPU (
    bool ) [override], [virtual]
```

Reimplemented from [Digikam::MaintenanceTool](#).

## 9.1332 Digikam::ThumbsTask Class Reference

Inheritance diagram for Digikam::ThumbsTask:



## Signals

- void **signalFinished** (const [ItemInfo](#) &, const QImage &)

## Signals inherited from [Digikam::ActionJob](#)

- void **signalDone** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job is done.*
- void **signalProgress** (int)  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job progress.*
- void **signalStarted** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager that job is started.*

## Public Member Functions

- void **setMaintenanceData** ([MaintenanceData](#) \*const data=nullptr)

## Public Member Functions inherited from [Digikam::ActionJob](#)

- **ActionJob** (QObject \*const parent=nullptr)  
*Constructor which delegate deletion of QRunnable instance to [ActionThreadBase](#), not QThreadPool.*
- **~ActionJob** () override  
*Re-implement destructor in you implementation.*

## Protected Member Functions

- void **run** () override

## Additional Inherited Members

## Public Slots inherited from [Digikam::ActionJob](#)

- void **cancel** ()  
*Call this method to cancel job.*

## Public Attributes inherited from [Digikam::ActionJob](#)

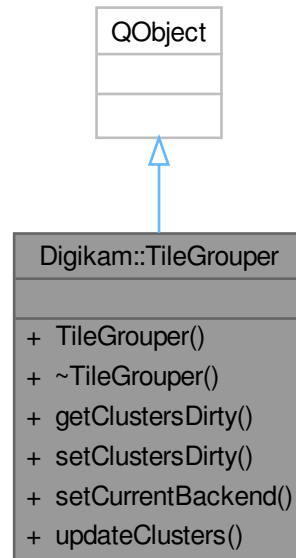
- QElapsedTimer **m\_timer**  
*Timer to determine the running time of the job.*

## Protected Attributes inherited from [Digikam::ActionJob](#)

- bool **m\_cancel** = false  
*You can use this boolean in your implementation to know if job must be canceled.*

## 9.1333 Digikam::TileGrouper Class Reference

Inheritance diagram for Digikam::TileGrouper:



### Public Member Functions

- **TileGrouper** (const QExplicitlySharedDataPointer< [GeofaceSharedData](#) > &sharedData, QObject \*const parent)
- bool **getClustersDirty** () const
- void **setClustersDirty** ()
- void **setCurrentBackend** ([MapBackend](#) \*const backend)
- void **updateClusters** ()

### 9.1333.1 Member Function Documentation

#### 9.1333.1.1 updateClusters()

```
void Digikam::TileGrouper::updateClusters ( )
```

## 9.1334 Digikam::TileIndex Class Reference

### Public Types

- enum **Constants** { **MaxLevel** = 9 , **MaxIndexCount** = MaxLevel+1 , **Tiling** = 10 , **MaxLinearIndex** = Tiling\*↔ Tiling }
- enum **CornerPosition** { **CornerNW** = 1 , **CornerSW** = 2 , **CornerNE** = 3 , **CornerSE** = 4 }
- typedef QList< [TileIndex](#) > **List**

### Public Member Functions

- void **appendLatLonIndex** (const int latIndex, const int lonIndex)
- void **appendLinearIndex** (const int newIndex)
- int **at** (const int getLevel) const
- void **clear** ()
- int **indexCount** () const
- int **indexLat** (const int getLevel) const
- int **indexLon** (const int getLevel) const
- int **lastIndex** () const
- QPoint **latLonIndex** (const int getLevel) const
- void **latLonIndex** (const int getLevel, int \*const latIndex, int \*const lonIndex) const
- int **level** () const
- int **linearIndex** (const int getLevel) const
- [TileIndex](#) **mid** (const int first, const int len) const
- void **oneUp** ()
- [GeoCoordinates](#) **toCoordinates** () const
- [GeoCoordinates](#) **toCoordinates** (const CornerPosition ofCorner) const
- QList< int > **toIntList** () const

### Static Public Member Functions

- static [TileIndex](#) **fromCoordinates** (const [Digikam::GeoCoordinates](#) &coordinate, const int getLevel)
- static [TileIndex](#) **fromIntList** (const QList< int > &intList)
- static bool **indicesEqual** (const [TileIndex](#) &a, const [TileIndex](#) &b, const int upToLevel)
- static QList< QList< int > > **listToIntListList** (const QList< [TileIndex](#) > &tileIndexList)

## 9.1335 Digikam::TimeAdjustContainer Class Reference

Container that store all timestamp adjustments.

### Public Types

- enum **AdjType** { COPYVALUE = 0 , ADDVALUE , SUBVALUE , INTERVAL }
- enum **UseDataSource** { APPDATE = 0 , FILENAME , FILEDATE , METADATADATE , CUSTOMDATE }
- enum **UseFileType** { FILELASTMOD = 0 , FILECREATED }
- enum **UseMetaDataType** { EXIFIPTCXMP = 0 , EXIFCREATED , EXIFORIGINAL , EXIFDIGITIZED , IPTCCREATED , XMPCREATED , FUZZYCREATED , FUZZYORIGINAL , FUZZYDIGITIZED }

### Public Member Functions

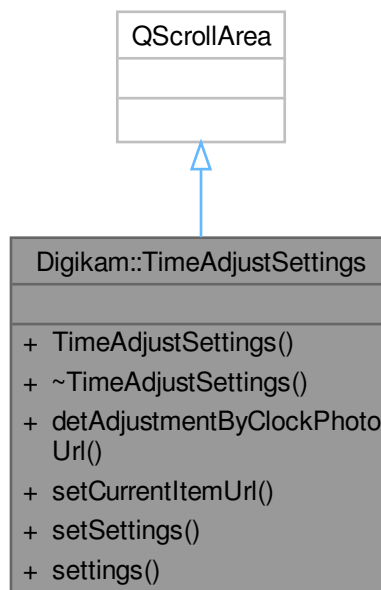
- bool **atLeastOneUpdateToProcess** () const  
*Check if at least one option is selected.*
- QDateTime **calculateAdjustedDate** (const QDateTime &originalTime, int index=0)
- QDateTime **getDateFromTimeFromString** (const QString &dateStr) const
- QMap< QString, bool > **getDateFromTimeTagsMap** () const

**Public Attributes**

- int **adjustmentDays** = 0
- QDateTime **adjustmentTime** = QDateTime()
- int **adjustmentType** = COPYVALUE
- QDateTime **customDate** = QDateTime::currentDateTime()
- QDateTime **customTime** = QDateTime::currentDateTime()
- int **dateSource** = APPDATE
- bool **enableExifTool** = false
  - Only a temporary variable, will not be saved.*
- int **fileDateSource** = FILELASTMOD
- int **metadataSource** = EXIFIPTCXMP
- bool **updEXIFDigDate** = false
- bool **updEXIFModDate** = false
- bool **updEXIFOriDate** = false
- bool **updEXIFThmDate** = false
- bool **updFileModDate** = false
- bool **updlfAvailable** = true
- bool **updIPTCDate** = false
- bool **updUseExifTool** = false
- bool **updXMPDate** = false
- bool **updXMPVideo** = false

**9.1336 Digikam::TimeAdjustSettings Class Reference**

Inheritance diagram for Digikam::TimeAdjustSettings:



## Signals

- void **signalSettingsChanged** ()
- void **signalSettingsChangedTool** ()
- void **signalSrcTimestampChanged** ()

## Public Member Functions

- **TimeAdjustSettings** (QWidget \*const parent, bool timeAdjustTool=false)
- void **detAdjustmentByClockPhotoUrl** (const QUrl &url)
- void **setCurrentItemUrl** (const QUrl &url)
- void **setSettings** (const [TimeAdjustContainer](#) &settings)
- [TimeAdjustContainer](#) **settings** () const

### 9.1336.1 Member Function Documentation

#### 9.1336.1.1 detAdjustmentByClockPhotoUrl()

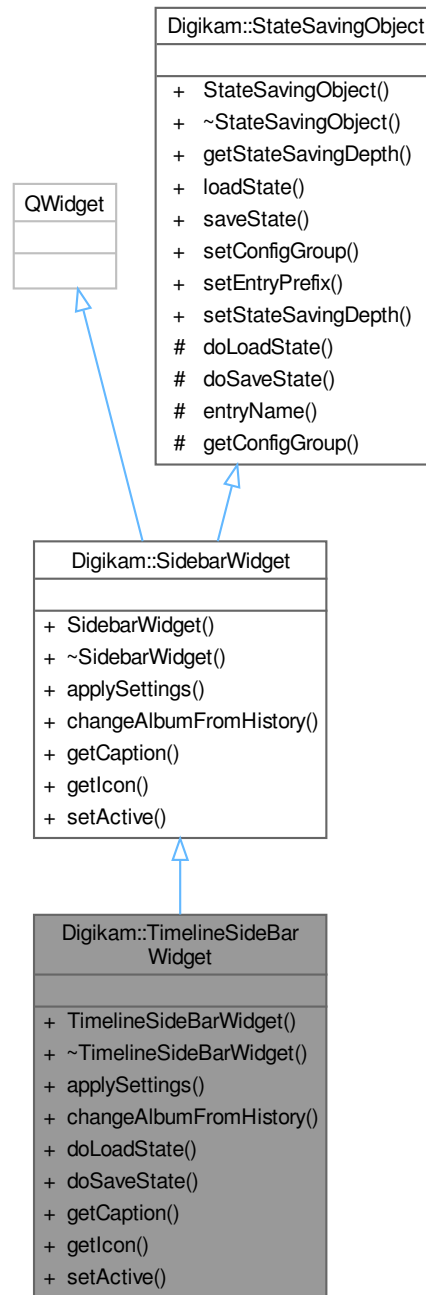
```
void Digikam::TimeAdjustSettings::detAdjustmentByClockPhotoUrl (
    const QUrl & url )
```

When user press the clock photo button, a dialog is displayed and set the results to the proper widgets.



## 9.1337 Digikam::TimelineSideBarWidget Class Reference

Inheritance diagram for Digikam::TimelineSideBarWidget:



### Public Member Functions

- **TimelineSideBarWidget** (QWidget \*const parent, [searchModel](#) \*const searchModel, [SearchModificationHelper](#) \*const searchModificationHelper)

- void [applySettings](#) () override  
*This method is invoked when the application settings should be (re-) applied to this widget.*
- void [changeAlbumFromHistory](#) (const QList< [Album](#) \* > &album) override  
*This is called on this widget when the history requires to move back to the specified album.*
- void [doLoadState](#) () override  
*Implement this hook method for state loading.*
- void [doSaveState](#) () override  
*Implement this hook method for state saving.*
- const QString [getCaption](#) () override  
*Must be implemented to return the title of this sidebar's tab.*
- const QIcon [getIcon](#) () override  
*Must be implemented and return the icon that shall be visible for this sidebar widget.*
- void [setActive](#) (bool active) override  
*This method is called if the visible sidebar widget is changed.*

## Public Member Functions inherited from [Digikam::SidebarWidget](#)

- [SidebarWidget](#) (QWidget \*const parent)  
*Constructor.*
- [~SidebarWidget](#) () override=default  
*Destructor.*

## Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (QObject \*const host)  
*Constructor.*
- virtual [~StateSavingObject](#) ()  
*Destructor.*
- [StateSavingDepth](#) [getStateSavingDepth](#) () const  
*Returns the depth used for state saving or loading.*
- void [loadState](#) ()  
*Invokes loading the class' state.*
- void [saveState](#) ()  
*Invokes saving the class' state.*
- virtual void [setConfigGroup](#) (const KConfigGroup &group)  
*Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void [setEntryPrefix](#) (const QString &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void [setStateSavingDepth](#) (const [StateSavingDepth](#) depth)  
*Sets the depth used for state saving or loading.*

## Additional Inherited Members

## Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }  
*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

## Signals inherited from [Digikam::SidebarWidget](#)

- void **requestActiveTab** ([SidebarWidget](#) \*)  
*This signal can be emitted if this sidebar widget wants to be the one that is active.*
- void **signalNotificationError** (const QString &message, int type)  
*To dispatch error message to temporized pop-up notification widget hosted with icon-view.*

## Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- QString **entryName** (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup **getConfigGroup** () const  
*Returns the config group that must be used for state saving and loading.*

### 9.1337.1 Member Function Documentation

#### 9.1337.1.1 **applySettings()**

```
void Digikam::TimelineSideBarWidget::applySettings ( ) [override], [virtual]
```

Implements [Digikam::SidebarWidget](#).

#### 9.1337.1.2 **changeAlbumFromHistory()**

```
void Digikam::TimelineSideBarWidget::changeAlbumFromHistory (
    const QList< Album * > & album ) [override], [virtual]
```

Implements [Digikam::SidebarWidget](#).

#### 9.1337.1.3 **doLoadState()**

```
void Digikam::TimelineSideBarWidget::doLoadState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

#### 9.1337.1.4 **doSaveState()**

```
void Digikam::TimelineSideBarWidget::doSaveState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

### 9.1337.1.5 `getCaption()`

```
const QString Digikam::TimelineSideBarWidget::getCaption ( ) [override], [virtual]
```

#### Returns

localized title string

Implements [Digikam::SidebarWidget](#).

### 9.1337.1.6 `getIcon()`

```
const QIcon Digikam::TimelineSideBarWidget::getIcon ( ) [override], [virtual]
```

#### Returns

pixmap icon

Implements [Digikam::SidebarWidget](#).

### 9.1337.1.7 `setActive()`

```
void Digikam::TimelineSideBarWidget::setActive (
    bool active ) [override], [virtual]
```

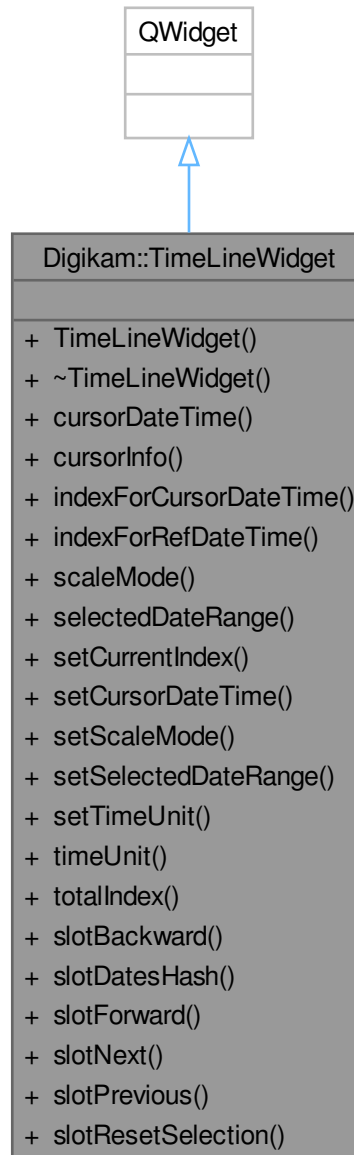
#### Parameters

<i>active</i>	if true, this widget is the new active widget, if false another widget is active
---------------	--

Implements [Digikam::SidebarWidget](#).

## 9.1338 Digikam::TimeLineWidget Class Reference

Inheritance diagram for Digikam::TimeLineWidget:



### Public Types

- enum `ScaleMode` { `LinScale` = 0 , `LogScale` }
- enum `SelectionMode` { `Unselected` = 0 , `FuzzySelection` , `Selected` }
- enum `TimeUnit` { `Day` = 0 , `Week` , `Month` , `Year` }

## Public Slots

- void **slotBackward** ()
- void **slotDatesHash** (const QHash< QDateTime, int > &)
- void **slotForward** ()
- void **slotNext** ()
- void **slotPrevious** ()
- void **slotResetSelection** ()

## Signals

- void **signalCursorPositionChanged** ()
- void **signalDateMapChanged** ()
- void **signalRefDateTimeChanged** ()
- void **signalSelectionChanged** ()

## Public Member Functions

- **TimeLineWidget** (QWidget \*const parent=nullptr)
- QDateTime **cursorDateTime** () const
- int **cursorInfo** (QString &infoDate) const
- int **indexForCursorDateTime** () const
- int **indexForRefDateTime** () const
- [ScaleMode](#) **scaleMode** () const
- [DateRangeList](#) **selectedDateRange** (int &totalCount) const  
*Return a list of Date-Range based on selection performed on days-map.*
- void **setCurrentIndex** (int index)
- void **setCursorDateTime** (const QDateTime &dateTime)
- void **setScaleMode** ([ScaleMode](#) scaleMode)
- void **setSelectedDateRange** (const [DateRangeList](#) &list)
- void **setTimeUnit** (TimeUnit timeUnit)
- TimeUnit **timeUnit** () const
- int **totalIndex** () const

## 9.1338.1 Member Enumeration Documentation

### 9.1338.1.1 ScaleMode

```
enum Digikam::TimeLineWidget::ScaleMode
```

#### Enumerator

LinScale	Linear scale.
LogScale	Logarithmic scale.

### 9.1338.1.2 SelectionMode

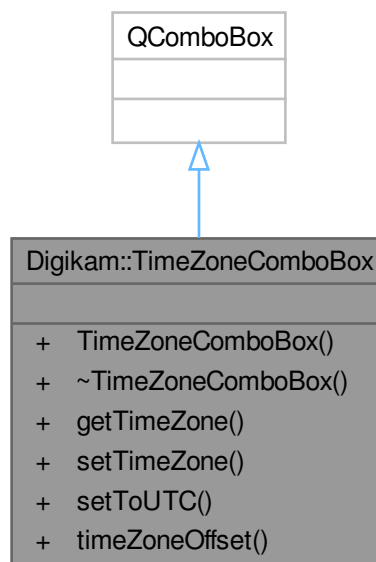
```
enum Digikam::TimeLineWidget::SelectionMode
```

## Enumerator

Unselected	No selection.
FuzzySelection	Partially selected.
Selected	Fully selected.

## 9.1339 Digikam::TimeZoneComboBox Class Reference

Inheritance diagram for Digikam::TimeZoneComboBox:



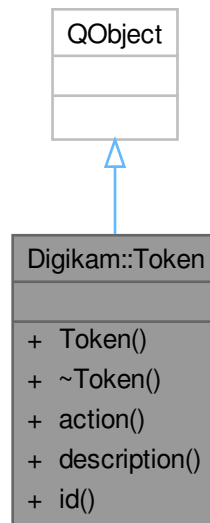
### Public Member Functions

- **TimeZoneComboBox** (QWidget \*const parent)
- QString **getTimeZone** () const
- void **setTimeZone** (const QString &timeStr)
- void **setToUTC** ()
- int **timeZoneOffset** () const

## 9.1340 Digikam::Token Class Reference

Token is the smallest parsing unit in AdvancedRename utility

Inheritance diagram for Digikam::Token:



## Signals

- void **signalTokenTriggered** (const QString &)  
*This signal is emitted when the action of the token is triggered.*

## Public Member Functions

- **Token** (const QString &id, const QString &description)
- QAction \* **action** () const
- QString **description** () const
- QString **id** () const

### 9.1340.1 Detailed Description

The Token class represents the smallest parsing unit for the [Parser](#) class. Every string you enter as a renaming pattern is a combination of tokens and literal text. For example

```
"[file]{upper}_###_abc.[ext]{lower}"
```

is composed of five tokens

```
[file]
{upper}
###
.[ext]
{lower}
```

and two literals

```
_
_abc
```

A rule must assign at least one token object, to make parsing work. More than one token can be assigned to a Rule.

See also

[Rule::addToken\(\)](#)



## 9.1340.2 Member Function Documentation

### 9.1340.2.1 action()

```
QAction * Digikam::Token::action ( ) const
```

#### Returns

The action of the token. This action can be connected to a button or menu item. If triggered, high-level classes like [AdvancedRenameWidget](#) can connect to the signal and display the emitted text in the line edit input widget.

### 9.1340.2.2 description()

```
QString Digikam::Token::description ( ) const
```

#### Returns

The description of the token. It can be used for example in the tooltip of the [AdvancedRenameWidget](#).

### 9.1340.2.3 id()

```
QString Digikam::Token::id ( ) const
```

#### Returns

The ID of the token. This is the actual token string, for example  
"`[file]`"

This id will be emitted as a signal by `slotTriggered()`.

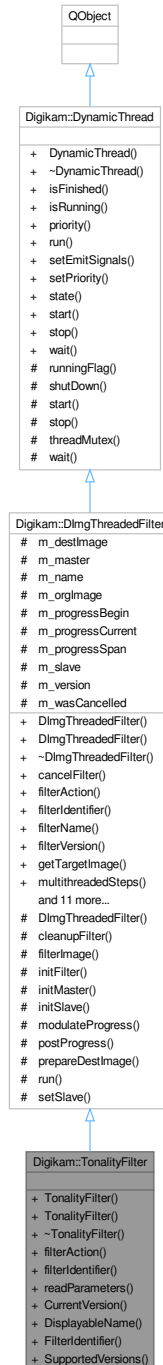
## 9.1341 Digikam::TonalityContainer Class Reference

### Public Attributes

- int **blueMask** = 0
- int **greenMask** = 0
- int **redMask** = 0

## 9.1342 Digikam::TonalityFilter Class Reference

Inheritance diagram for Digikam::TonalityFilter:



### Public Member Functions

- **TonalityFilter** ([DImg](#) \*const orgImage, QObject \*const parent=nullptr, const [TonalityContainer](#) &settings=[TonalityContainer](#)())
- **TonalityFilter** (QObject \*const parent=nullptr)

- [FilterAction filterAction](#) () override  
*Returns the action description corresponding to currently set options.*
- [QString filterIdentifier](#) () const override  
*Return the identifier for this filter in the image history.*
- void [readParameters](#) (const [FilterAction](#) &action) override

## Public Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImg](#) \*const orgImage, [QObject](#) \*const parent, const [QString](#) &name=[QString](#)())  
*Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter](#) ([QObject](#) \*const parent=nullptr, const [QString](#) &name=[QString](#)())  
*Constructs a filter without argument.*
- virtual void [cancelFilter](#) ()  
*Cancel the threaded computation.*
- const [QString](#) & [filterName](#) ()
- int [filterVersion](#) () const
- [DImg](#) [getTargetImage](#) ()
- [QList](#)< int > [multithreadedSteps](#) (int stop, int start=0) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool [parametersSuccessfullyRead](#) () const  
*Optional: error handling for readParameters.*
- virtual [QString](#) [readParametersError](#) (const [FilterAction](#) &actionThatFailed) const
- void [setFilterName](#) (const [QString](#) &name)
- void [setFilterVersion](#) (int version)  
*Replaying a filter action: Set the filter version.*
- void [setOriginalImage](#) (const [DImg](#) &orgImage)
- void [setupAndStartDirectly](#) (const [DImg](#) &orgImage, [DImgThreadedFilter](#) \*const master, int progress←Begin=0, int progressEnd=100)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void [setupFilter](#) (const [DImg](#) &orgImage)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void [startFilter](#) ()  
*Start the threaded computation.*
- virtual void [startFilterDirectly](#) ()  
*Start computation of this filter, directly in this thread.*
- virtual [QList](#)< int > [supportedVersions](#) () const

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread](#) ([QObject](#) \*const parent=nullptr)  
*This class extends [QRunnable](#), so you have to reimplement virtual void [run\(\)](#).*
- [~DynamicThread](#) () override  
*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool [isFinished](#) () const
- bool [isRunning](#) () const
- [QThread::Priority](#) [priority](#) () const
- void [setEmitSignals](#) (bool emitThem)
- void [setPriority](#) ([QThread::Priority](#) priority)  
*Sets the priority for this dynamic thread.*
- State [state](#) () const

### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

## Protected Member Functions inherited from Digikam::DImgThreadedFilter

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

## Protected Member Functions inherited from Digikam::DynamicThread

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

## Protected Attributes inherited from Digikam::DImgThreadedFilter

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.1342.1 Member Function Documentation

### 9.1342.1.1 filterAction()

```
FilterAction Digikam::TonalityFilter::filterAction ( ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

### 9.1342.1.2 filterIdentifier()

```
QString Digikam::TonalityFilter::filterIdentifier ( ) const [inline], [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

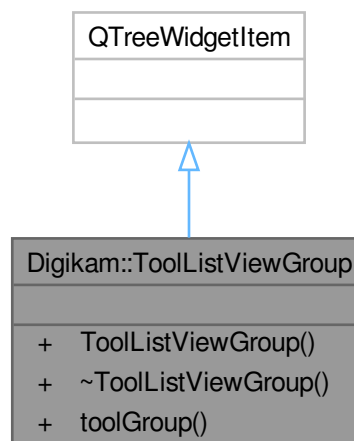
### 9.1342.1.3 readParameters()

```
void Digikam::TonalityFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

## 9.1343 Digikam::ToolListViewGroup Class Reference

Inheritance diagram for Digikam::ToolListViewGroup:

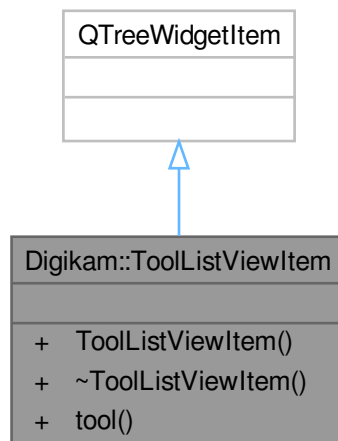


### Public Member Functions

- **ToolListViewGroup** (`QTreeWidgetItem *const parent`, [BatchTool::BatchToolGroup](#) group)
- [BatchTool::BatchToolGroup](#) **toolGroup** () const

## 9.1344 Digikam::ToolListViewItem Class Reference

Inheritance diagram for Digikam::ToolListViewItem:

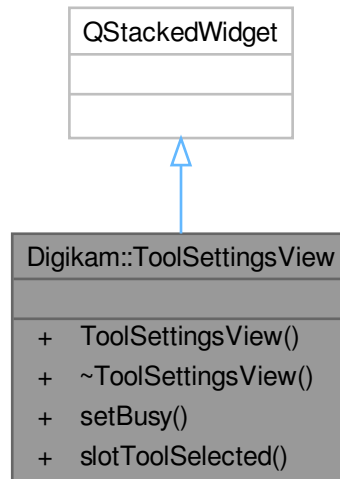


### Public Member Functions

- `ToolListViewItem` (`ToolListViewGroup` \*const parent, `BatchTool` \*const tool)
- `BatchTool` \* `tool` () const

## 9.1345 Digikam::ToolSettingsView Class Reference

Inheritance diagram for Digikam::ToolSettingsView:



### Public Slots

- void **slotToolSelected** (const [BatchToolSet](#) &)

### Signals

- void **signalSettingsChanged** (const [BatchToolSet](#) &)

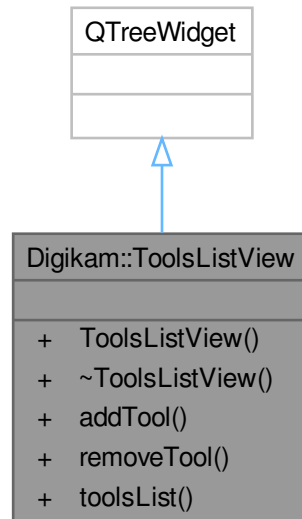
### Public Member Functions

- **ToolSettingsView** (QWidget \*const parent=nullptr)
- void **setBusy** (bool b)



## 9.1346 Digikam::ToolsListView Class Reference

Inheritance diagram for Digikam::ToolsListView:



### Signals

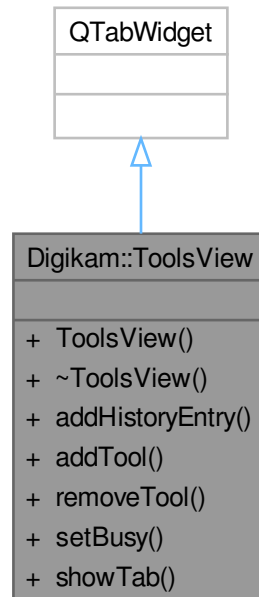
- void **signalAssignTools** (const QMap< int, QString > &)

### Public Member Functions

- **ToolsListView** (QWidget \*const parent)
- void **addTool** ([BatchTool](#) \*const tool)
- bool **removeTool** ([BatchTool](#) \*const tool)
- [BatchToolsList](#) **toolsList** ()

## 9.1347 Digikam::ToolsView Class Reference

Inheritance diagram for Digikam::ToolsView:



### Public Types

- enum `ViewTabs` { `TOOLS = 0` , `WORKFLOW` , `HISTORY` }

### Signals

- void `signalAssignQueueSettings` (QString)
- void `signalAssignTools` (const QMap< int, QString > &)
- void `signalHistoryEntryClicked` (int, qlonglong)
- void `signalUpdateQueueSettings` (QString)

### Public Member Functions

- `ToolsView` (QWidget \*const parent=nullptr)
- void `addHistoryEntry` (const QString &msg, DHistoryView::EntryType type, int queueId=-1, qlonglong itemId=-1)
- void `addTool` (`BatchTool` \*const tool)
- bool `removeTool` (`BatchTool` \*const tool)
- void `setBusy` (bool b)
- void `showTab` (ViewTabs t)

## 9.1348 Digikam::TooltipCreator Class Reference

### Public Member Functions

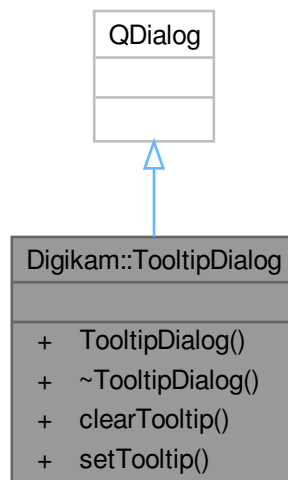
- QString **tooltip** (const Parser \*const parser)

### Static Public Member Functions

- static TooltipCreator & **getInstance** ()

## 9.1349 Digikam::TooltipDialog Class Reference

Inheritance diagram for Digikam::TooltipDialog:

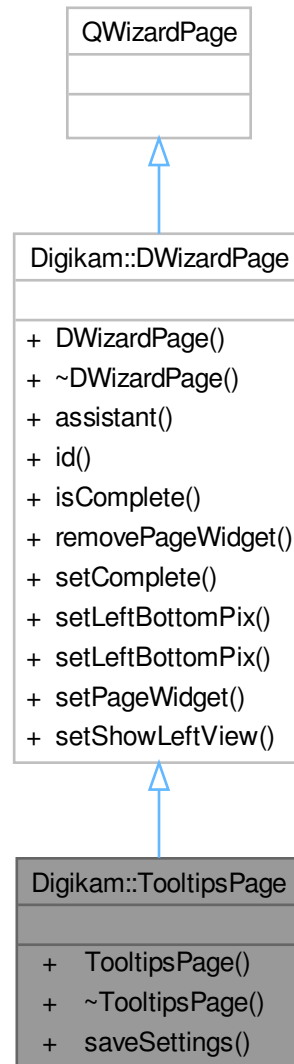


### Public Member Functions

- **TooltipDialog** (QWidget \*const parent)
- void **clearTooltip** ()
- void **setTooltip** (const QString &tooltip)

## 9.1350 Digikam::TooltipsPage Class Reference

Inheritance diagram for Digikam::TooltipsPage:



### Public Member Functions

- **TooltipsPage** (QWizard \*const dlg)
- void **saveSettings** ()

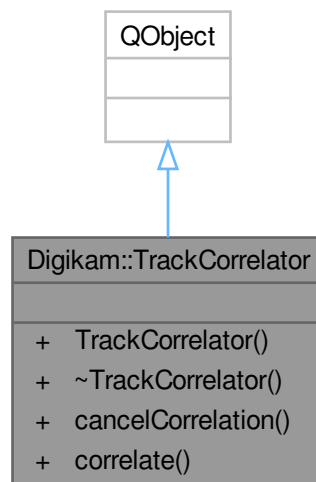
### Public Member Functions inherited from [Digikam::DWizardPage](#)

- **DWizardPage** (QWizard \*const dlg, const QString &title)
- QWizard \* **assistant** () const

- int **id** () const
- bool **isComplete** () const override
- void **removePageWidget** (QWidget \*const w)
- void **setComplete** (bool b)
- void **setLeftBottomPix** (const QIcon &icon)
- void **setLeftBottomPix** (const QPixmap &pix)
- void **setPageWidget** (QWidget \*const w)
- void **setShowLeftView** (bool v)

## 9.1351 Digikam::TrackCorrelator Class Reference

Inheritance diagram for Digikam::TrackCorrelator:



### Classes

- class [Correlation](#)
- class [CorrelationOptions](#)

### Public Types

- enum **CorrelationFlags** { **CorrelationFlagCoordinates** = 1 , **CorrelationFlagInterpolated** = 2 , **CorrelationFlagAltitude** = 3 }

### Signals

- void **signalAllItemsCorrelated** ()
- void **signalCorrelationCanceled** ()
- void **signalItemsCorrelated** (const Digikam::TrackCorrelator::Correlation::List &correlatedItems)

## Public Member Functions

- **TrackCorrelator** ([TrackManager](#) \*const trackManager, QObject \*const parent=nullptr)
- void **cancelCorrelation** ()
- void **correlate** (const Correlation::List &itemsToCorrelate, const [CorrelationOptions](#) &options)  
*GPS-correlate items.*

## 9.1352 Digikam::TrackCorrelator::Correlation Class Reference

### Public Types

- typedef QList< [Correlation](#) > **List**

### Public Attributes

- [GeoCoordinates](#) **coordinates**
- QDateTime **dateTime**
- int **fixType** = -1
- CorrelationFlags **flags** = CorrelationFlagCoordinates
- qreal **hDop** = -1.0
- int **nSatellites** = -1
- qreal **pDop** = -1.0
- qreal **speed** = -1.0
- QVariant **userData**

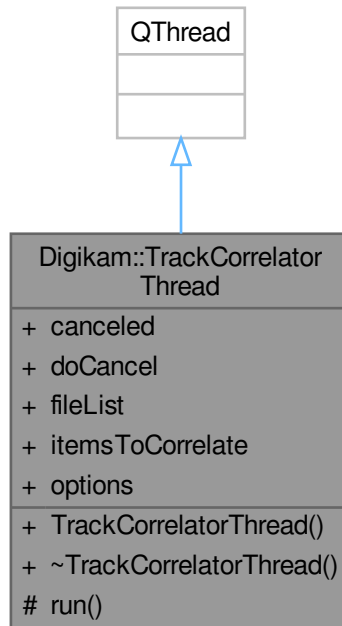
## 9.1353 Digikam::TrackCorrelator::CorrelationOptions Class Reference

### Public Attributes

- bool **interpolate** = false
- int **interpolationDstTime** = 0
- int **maxGapTime** = 0
- int **secondsOffset** = 0
- int **timeZoneOffset** = 0

## 9.1354 Digikam::TrackCorrelatorThread Class Reference

Inheritance diagram for Digikam::TrackCorrelatorThread:



### Signals

- void **signalItemsCorrelated** (const Digikam::TrackCorrelator::Correlation::List &correlatedItems)

### Public Member Functions

- **TrackCorrelatorThread** (QObject \*const parent=nullptr)

### Public Attributes

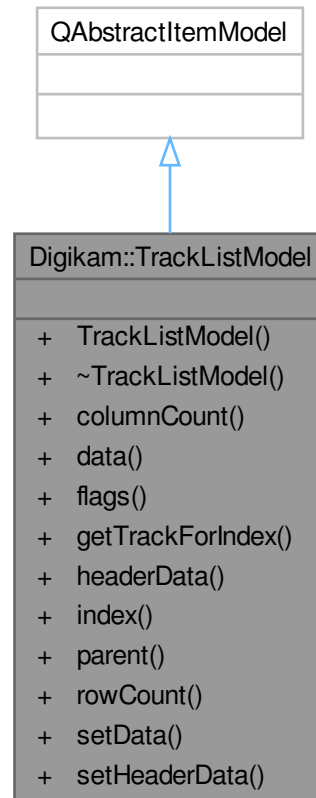
- bool **canceled** = false
- bool **doCancel** = false
- TrackManager::Track::List **fileList**
- TrackCorrelator::Correlation::List **itemsToCorrelate**
- [TrackCorrelator::CorrelationOptions](#) **options**

### Protected Member Functions

- void **run** () override

## 9.1355 Digikam::TrackListModel Class Reference

Inheritance diagram for Digikam::TrackListModel:



### Public Member Functions

- **TrackListModel** ([TrackManager](#) \*const trackManager, QObject \*const parent)
- int **columnCount** (const QModelIndex &parent=QModelIndex()) const override
- QVariant **data** (const QModelIndex &index, int role=Qt::DisplayRole) const override
- Qt::ItemFlags **flags** (const QModelIndex &index) const override
- [TrackManager::Track](#) **getTrackForIndex** (const QModelIndex &index) const
- QVariant **headerData** (int section, Qt::Orientation orientation, int role) const override
- QModelIndex **index** (int row, int column, const QModelIndex &parent=QModelIndex()) const override
- QModelIndex **parent** (const QModelIndex &index) const override
- int **rowCount** (const QModelIndex &parent=QModelIndex()) const override
- bool **setData** (const QModelIndex &index, const QVariant &value, int role) override
- bool **setHeaderData** (int section, Qt::Orientation orientation, const QVariant &value, int role) override



## 9.1355.1 Member Function Documentation

### 9.1355.1.1 headerData()

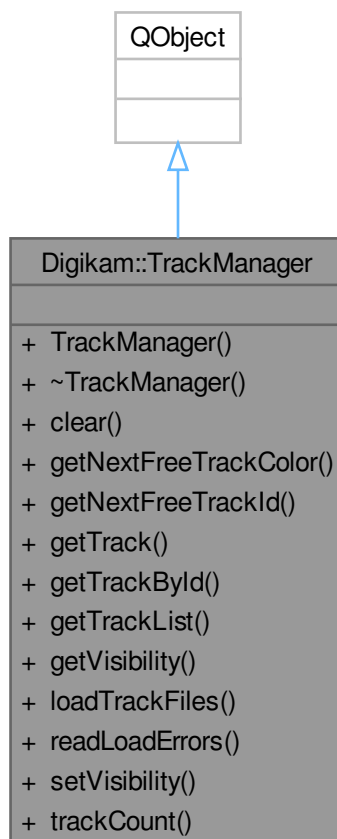
```
QVariant Digikam::TrackListModel::headerData (
    int section,
    Qt::Orientation orientation,
    int role ) const [override]
```

### 9.1355.1.2 index()

```
QModelIndex Digikam::TrackListModel::index (
    int row,
    int column,
    const QModelIndex & parent = QModelIndex() ) const [override]
```

## 9.1356 Digikam::TrackManager Class Reference

Inheritance diagram for Digikam::TrackManager:



## Classes

- class [Track](#)
- class [TrackPoint](#)

## Public Types

- enum **ChangeFlag** { **ChangeTrackPoints** = 1 , **ChangeMetadata** = 2 , **ChangeRemoved** = 4 , **ChangeAdd** = ChangeTrackPoints | ChangeMetadata }
- typedef quint32 **Id**
- typedef QPair< **Id**, ChangeFlag > **TrackChanges**

## Signals

- void **signalAllTrackFilesReady** ()
- void **signalTrackFilesReadyAt** (const int startIndex, const int endIndex)
- void **signalTracksChanged** (const QList< TrackManager::TrackChanges > &trackChanges)
- void **signalVisibilityChanged** (const bool newValue)

## Public Member Functions

- **TrackManager** (QObject \*const parent=nullptr)
- void **clear** ()
- QColor **getNextFreeTrackColor** ()
- quint64 **getNextFreeTrackId** ()
- const [Track](#) & **getTrack** (const int index) const
- [Track](#) **getTrackById** (const quint64 trackId) const
- Track::List **getTrackList** () const
- bool **getVisibility** () const
- void **loadTrackFiles** (const QList< QUrl > &urls)
- QList< QPair< QUrl, QString > > **readLoadErrors** ()
- void **setVisibility** (const bool value)
- int **trackCount** () const

## 9.1356.1 Member Typedef Documentation

### 9.1356.1.1 Id

```
typedef quint32 Digikam::TrackManager::Id
```

#### Note

we assume here that we will never load more than uint32\_max tracks.

## 9.1356.2 Member Function Documentation

### 9.1356.2.1 clear()

```
void Digikam::TrackManager::clear ( )
```

## 9.1357 Digikam::TrackManager::Track Class Reference

### Public Types

- enum **Flags** { **FlagVisible** = 1 , **FlagDefault** = FlagVisible }
- typedef QList< [Track](#) > **List**

### Public Attributes

- QColor **color** = Qt::red
- Flags **flags** = FlagDefault
- [Id](#) **id** = 0  
*0 means no track id assigned yet*
- QList< [TrackPoint](#) > **points**
- QUrl **url**

## 9.1358 Digikam::TrackManager::TrackPoint Class Reference

### Public Types

- typedef QList< [TrackPoint](#) > **List**

### Static Public Member Functions

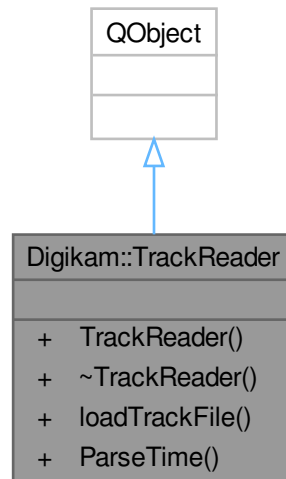
- static bool **EarlierThan** (const [TrackPoint](#) &a, const [TrackPoint](#) &b)

### Public Attributes

- [GeoCoordinates](#) **coordinates**
- QDateTime **dateTime**
- int **fixType** = -1
- qreal **hDop** = -1.0
- int **nSatellites** = -1
- qreal **pDop** = -1.0
- qreal **speed** = -1.0

## 9.1359 Digikam::TrackReader Class Reference

Inheritance diagram for Digikam::TrackReader:



### Classes

- class [TrackReadResult](#)

### Public Member Functions

- **TrackReader** ([TrackReadResult](#) \*const dataTarget)

### Static Public Member Functions

- static [TrackReadResult](#) **loadTrackFile** (const QUrl &url)
- static QDateTime **ParseTime** (const QString &tstring)

### Friends

- class `::TestTracks`

## 9.1360 Digikam::TrackReader::TrackReadResult Class Reference

### Public Types

- typedef QList< [TrackReadResult](#) > **List**

**Public Attributes**

- bool **isValid** = false
- QString **loadError**
- [TrackManager::Track](#) **track**

**9.1361 Digikam::TrainerWorker Class Reference**

Inheritance diagram for Digikam::TrainerWorker:



## Public Slots

- void **process** (const FacePipelineExtendedPackage::Ptr &package)  
*TODO: investigate this method.*

## Public Slots inherited from [Digikam::WorkerObject](#)

- void [deactivate](#) ([DeactivatingMode](#) mode=[FlushSignals](#))  
*Quits execution of this worker object.*
- void **schedule** ()  
*Starts execution of this worker object: The object is moved to a thread and an event loop started, so that queued signals will be received.*

## Signals

- void **processed** (const FacePipelineExtendedPackage::Ptr &package)

## Signals inherited from [Digikam::WorkerObject](#)

- void **finished** ()
- void **started** ()

## Public Member Functions

- **TrainerWorker** (FacePipeline::Private \*const dd)

## Public Member Functions inherited from [Digikam::WorkerObject](#)

- [WorkerObject](#) ()  
*Deriving from a worker object allows you to execute your slots in a thread.*
- bool [connectAndSchedule](#) (const QObject \*sender, const char \*signal, const char \*method, Qt::↔ ConnectionType type=Qt::AutoConnection) const  
*You must normally call [schedule\(\)](#) to ensure that the object is active when you send a signal with work data.*
- QThread::Priority **priority** () const
- void [setPriority](#) (QThread::Priority priority)  
*Sets the priority for this dynamic thread.*
- State **state** () const
- void **wait** ()

## Protected Member Functions

- void [aboutToDeactivate](#) () override  
*Called from [deactivate\(\)](#), typically from a different thread than the worker thread, possibly the UI thread.*

## Protected Member Functions inherited from [Digikam::WorkerObject](#)

- virtual void [aboutToQuitLoop](#) ()
  - Called from within thread's event loop to quit processing.*
- void **addRunnable** (WorkerObjectRunnable \*loop)
- bool **event** (QEvent \*e) override
- void **removeRunnable** (WorkerObjectRunnable \*loop)
- void **run** ()
- void **setEventLoop** (QEventLoop \*loop)
- void [shutDown](#) ()
  - If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void **transitionToInactive** ()
- bool **transitionToRunning** ()

## Protected Attributes

- FacePipeline::Private \*const **d** = nullptr
- [FaceltemRetriever](#) **imageRetriever**
- [FacialRecognitionWrapper](#) **recognizer**

## Additional Inherited Members

## Public Types inherited from [Digikam::WorkerObject](#)

- enum [DeactivatingMode](#) { [FlushSignals](#) , [KeepSignals](#) , [PhaseOut](#) }
- enum **State** { [Inactive](#) , [Scheduled](#) , [Running](#) , [Deactivating](#) }

## Static Public Member Functions inherited from [Digikam::WorkerObject](#)

- static bool **connectAndSchedule** (const QObject \*sender, const char \*signal, const [WorkerObject](#) \*receiver, const char \*method, Qt::ConnectionType type=Qt::AutoConnection)
- static bool **disconnectAndSchedule** (const QObject \*sender, const char \*signal, const [WorkerObject](#) \*receiver, const char \*method)

### 9.1361.1 Member Function Documentation

#### 9.1361.1.1 [aboutToDeactivate\(\)](#)

```
void Digikam::TrainerWorker::aboutToDeactivate ( ) [override], [protected], [virtual]
```

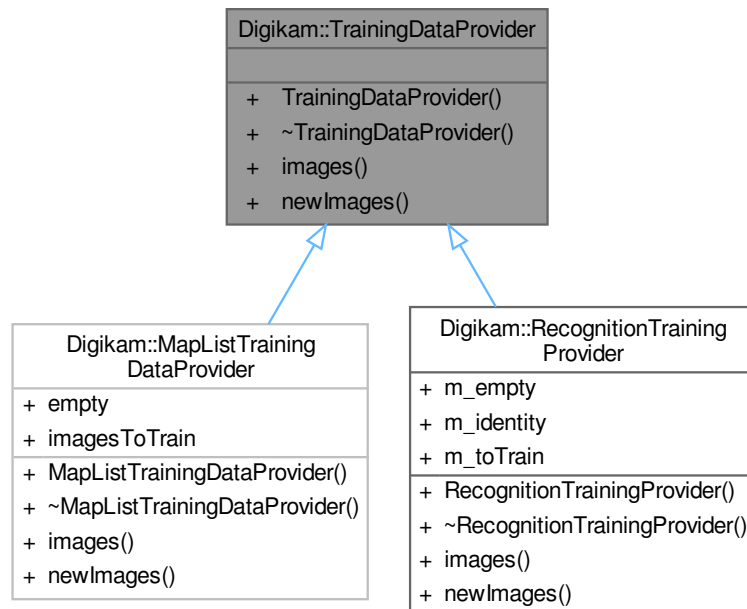
You can stop any extra controlled threads here. Immediately afterwards, an event will be sent to the working thread which will cause the event loop to quit. ([aboutToQuitLoop\(\)](#))

Reimplemented from [Digikam::WorkerObject](#).

## 9.1362 Digikam::TrainingDataProvider Class Reference

A [TrainingDataProvider](#) provides a call-back interface for the training process to retrieve the necessary information.

Inheritance diagram for Digikam::TrainingDataProvider:



### Public Member Functions

- virtual [ImageListProvider](#) \* [images](#) (const [Identity](#) &identity)=0  
*Provides all images known for the given identity.*
- virtual [ImageListProvider](#) \* [newImages](#) (const [Identity](#) &identity)=0  
*Provides those images for the given identity that have not yet been supplied for training.*

### 9.1362.1 Detailed Description

It is not specified, but depends on the backend which of the methods in which order and for which identities will be called.

### 9.1362.2 Member Function Documentation

#### 9.1362.2.1 [images\(\)](#)

```
virtual ImageListProvider * Digikam::TrainingDataProvider::images (
    const Identity & identity ) [pure virtual]
```

Ownership of the returned object stays with the [TrainingDataProvider](#).

Implemented in [Digikam::RecognitionTrainingProvider](#).



### 9.1362.2.2 newImages()

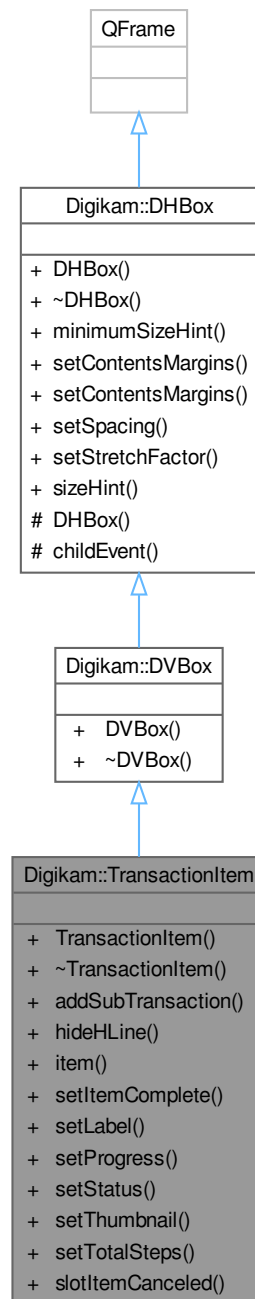
```
virtual ImageListProvider * Digikam::TrainingDataProvider::newImages (
    const Identity & identity ) [pure virtual]
```

Ownership of the returned object stays with the [TrainingDataProvider](#).

Implemented in [Digikam::RecognitionTrainingProvider](#).

## 9.1363 Digikam::TransactionItem Class Reference

Inheritance diagram for Digikam::TransactionItem:



### Public Slots

- void `slotItemCanceled()`

## Public Member Functions

- **TransactionItem** (QWidget \*const parent, [ProgressItem](#) \*const item, bool first)
- void **addSubTransaction** ([ProgressItem](#) \*const item)
- void **hideHLine** ()
- [ProgressItem](#) \* **item** () const
- void **setItemComplete** ()
  - The progressitem is deleted immediately, we take 5s to go out, so better not use mItem during this time.*
- void **setLabel** (const QString &)
- void **setProgress** (int progress)
- void **setStatus** (const QString &)
- void **setThumbnail** (const QPixmap &)
- void **setTotalSteps** (int totalSteps)

## Public Member Functions inherited from [Digikam::DVBox](#)

- **DVBox** (QWidget \*const parent=nullptr)

## Public Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (QWidget \*const parent=nullptr)
- QSize **minimumSizeHint** () const override
- void **setContentMargins** (const QMargins &margins)
- void **setContentMargins** (int left, int top, int right, int bottom)
- void **setSpacing** (int space)
- void **setStretchFactor** (QWidget \*const widget, int stretch)
- QSize **sizeHint** () const override

## Additional Inherited Members

## Protected Member Functions inherited from [Digikam::DHBox](#)

- **DHBox** (bool vertical, QWidget \*const parent)
- void **childEvent** (QChildEvent \*e) override

## 9.1363.1 Member Function Documentation

### 9.1363.1.1 setStatus()

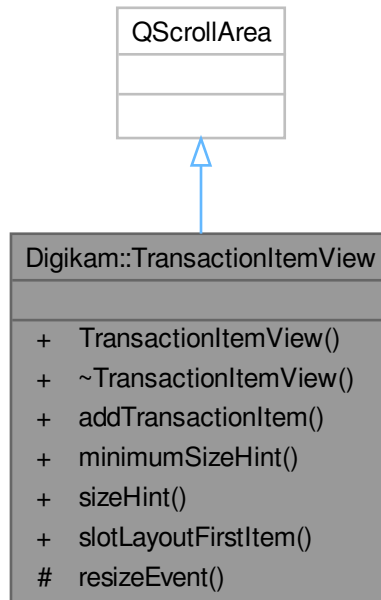
```
void Digikam::TransactionItem::setStatus (
    const QString & status )
```

#### Note

the given text is interpreted as RichText, so you might need to use `.toHtmlEscaped()` it before passing.

## 9.1364 Digikam::TransactionItemView Class Reference

Inheritance diagram for Digikam::TransactionItemView:



### Public Slots

- void `slotLayoutFirstItem ()`

### Signals

- void `signalTransactionViewsEmpty ()`

### Public Member Functions

- `TransactionItemView` (`QWidget *const parent=nullptr, const QString &name=QString()`)
- `TransactionItem * addTransactionItem` (`ProgressItem *item, bool first`)
- `QSize minimumSizeHint ()` const override
- `QSize sizeHint ()` const override

### Protected Member Functions

- void `resizeEvent` (`QResizeEvent *event`) override

## 9.1365 Digikam::TransitionMngr Class Reference

### Public Types

- enum **TransType** {  
**None** = 0 , **ChessBoard** , **MeltDown** , **Sweep** ,  
**Mosaic** , **Cubism** , **Growing** , **HorizontalLines** ,  
**VerticalLines** , **CircleOut** , **MultiCircleOut** , **SpiralIn** ,  
**Blobs** , **Fade** , **SlideL2R** , **SlideR2L** ,  
**SlideT2B** , **SlideB2T** , **PushL2R** , **PushR2L** ,  
**PushT2B** , **PushB2T** , **SwapL2R** , **SwapR2L** ,  
**SwapT2B** , **SwapB2T** , **BlurIn** , **BlurOut** ,  
**Random** }

### Public Member Functions

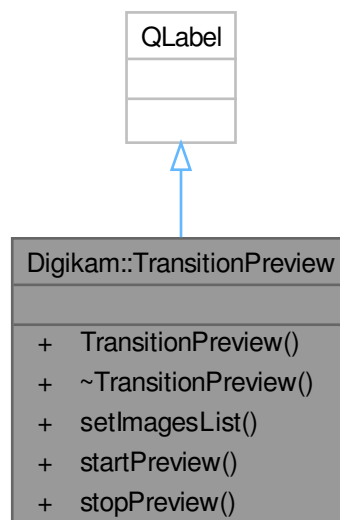
- QImage **currentFrame** (int &tmout)
- void **setInImage** (const QImage &iimg)
- void **setOutImage** (const QImage &oimg)
- void **setOutputSize** (const QSize &size)
- void **setTransition** (TransType type)

### Static Public Member Functions

- static QMap< TransType, QString > **transitionNames** ()

## 9.1366 Digikam::TransitionPreview Class Reference

Inheritance diagram for Digikam::TransitionPreview:

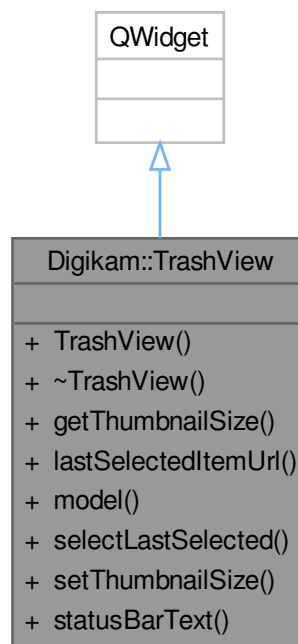


### Public Member Functions

- **TransitionPreview** (QWidget \*const parent=nullptr)
- void **setImagesList** (const QList< QUrl > &images)
- void **startPreview** (TransitionMngr::TransType eff)
- void **stopPreview** ()

## 9.1367 Digikam::TrashView Class Reference

Inheritance diagram for Digikam::TrashView:



### Signals

- void **selectionChanged** ()
- void **signalEmptytrash** ()

### Public Member Functions

- **TrashView** (QWidget \*const parent=nullptr)
- **ThumbnailSize** [getThumbnailSize](#) () const
- **QUrl** [lastSelectedItemUrl](#) () const
- **DTrashItemModel** \* [model](#) () const
- void **selectLastSelected** ()
  - Highlights the last selected item when the view gets focus.*
- void [setThumbnailSize](#) (const [ThumbnailSize](#) &thumbSize)
  - set thumbnail size to give to model*
- **QString** [statusBarText](#) () const

## 9.1367.1 Member Function Documentation

### 9.1367.1.1 getThumbnailSize()

```
ThumbnailSize Digikam::TrashView::getThumbnailSize ( ) const
```

#### Returns

current thumbnail size

### 9.1367.1.2 lastSelectedItemUrl()

```
QUrl Digikam::TrashView::lastSelectedItemUrl ( ) const
```

#### Returns

QUrl to the last selected item in view

### 9.1367.1.3 model()

```
DTrashItemModel * Digikam::TrashView::model ( ) const
```

#### Returns

model used for the view

### 9.1367.1.4 setThumbnailSize()

```
void Digikam::TrashView::setThumbnailSize (
    const ThumbnailSize & thumbSize )
```

#### Parameters

<i>thumbSize</i>	size to set
------------------	-------------

### 9.1367.1.5 statusBarText()

```
QString Digikam::TrashView::statusBarText ( ) const
```

#### Returns

text for the main status bar

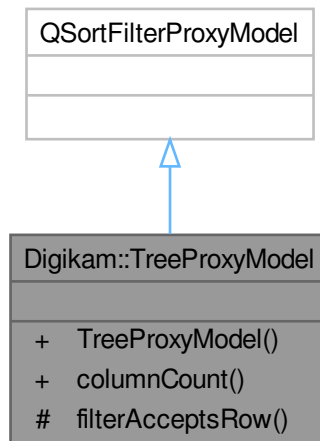
## 9.1368 Digikam::TreeBranch Class Reference

### Public Attributes

- QString **data**
- QString **help**
- QList< TreeBranch \* > **newChildren**
- QList< TreeBranch \* > **oldChildren**
- TreeBranch \* **parent** = nullptr
- QPersistentModelIndex **sourceIndex**
- QList< TreeBranch \* > **spacerChildren**
- Type **type** = TypeChild

## 9.1369 Digikam::TreeProxyModel Class Reference

Inheritance diagram for Digikam::TreeProxyModel:



### Signals

- void **signalFilterAccepts** (bool)

### Public Member Functions

- **TreeProxyModel** (QObject \*const parent=nullptr)
- int **columnCount** (const QModelIndex &) const override

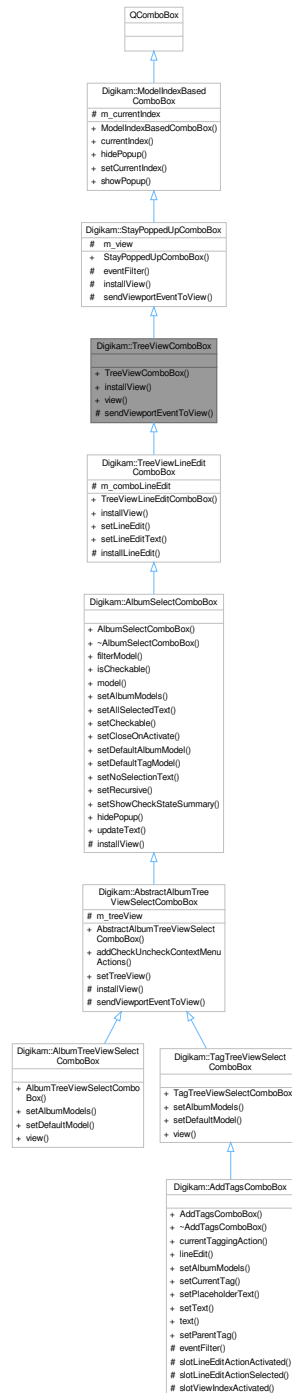
### Protected Member Functions

- bool **filterAcceptsRow** (int srow, const QModelIndex &sparent) const override



## 9.1370 Digikam::TreeViewComboBox Class Reference

Inheritance diagram for Digikam::TreeViewComboBox:



### Public Member Functions

- [TreeViewComboBox](#) (QWidget \*parent=nullptr)

*This class provides a QComboBox with a QTreeView instead of the usual QListView.*

- virtual void `installView` (QAbstractItemView \*view=nullptr)  
*Replace the standard combo box list view with a QTreeView.*
- QTreeView \* `view` () const  
*Returns the QTreeView of this class.*

### Public Member Functions inherited from [Digikam::StayPoppedUpComboBox](#)

- [StayPoppedUpComboBox](#) (QWidget \*const parent=nullptr)  
*This class provides an abstract QComboBox with a custom view (which is created by implementing subclasses) instead of the usual QListView.*

### Public Member Functions inherited from [Digikam::ModelIndexBasedComboBox](#)

- [ModelIndexBasedComboBox](#) (QWidget \*const parent=nullptr)  
*QComboBox has a current index based on a single integer.*
- QModelIndex `currentIndex` () const
- void `hidePopup` () override
- void `setCurrentIndex` (const QModelIndex &index)
- void `showPopup` () override

### Protected Member Functions

- void `sendViewportEventToView` (QEvent \*e) override  
*Implement in subclass: Send the given event to the viewportEvent() method of m\_view.*

### Protected Member Functions inherited from [Digikam::StayPoppedUpComboBox](#)

- bool `eventFilter` (QObject \*watched, QEvent \*event) override
- void `installView` (QAbstractItemView \*view)  
*Replace the standard combo box list view with the given view.*

### Additional Inherited Members

### Protected Attributes inherited from [Digikam::StayPoppedUpComboBox](#)

- QAbstractItemView \* `m_view` = nullptr

### Protected Attributes inherited from [Digikam::ModelIndexBasedComboBox](#)

- QPersistentModelIndex `m_currentIndex`

## 9.1370.1 Constructor & Destructor Documentation

### 9.1370.1.1 `TreeViewComboBox()`

```
Digikam::TreeViewComboBox::TreeViewComboBox (
    QWidget * parent = nullptr ) [explicit]
```

You need three steps: Construct the object, call `setModel()` with an appropriate `QAbstractItemModel`, then call `installView()` to replace the standard combo box view with a `QTreeView`.

## 9.1370.2 Member Function Documentation

### 9.1370.2.1 installView()

```
void Digikam::TreeViewComboBox::installView (
    QAbstractItemView * view = nullptr ) [virtual]
```

Call this after installing an appropriate model.

Reimplemented in [Digikam::AlbumSelectComboBox](#), [Digikam::AbstractAlbumTreeViewSelectComboBox](#), and [Digikam::TreeViewLineEditComboBox](#).

### 9.1370.2.2 sendViewportEventToView()

```
void Digikam::TreeViewComboBox::sendViewportEventToView (
    QEvent * e ) [override], [protected], [virtual]
```

This method is protected for a usual QAbstractItemView. You can override, pass a view, and call parent implementation. The existing view will be used. You must then also reimplement sendViewportEventToView.

Implements [Digikam::StayPoppedUpComboBox](#).

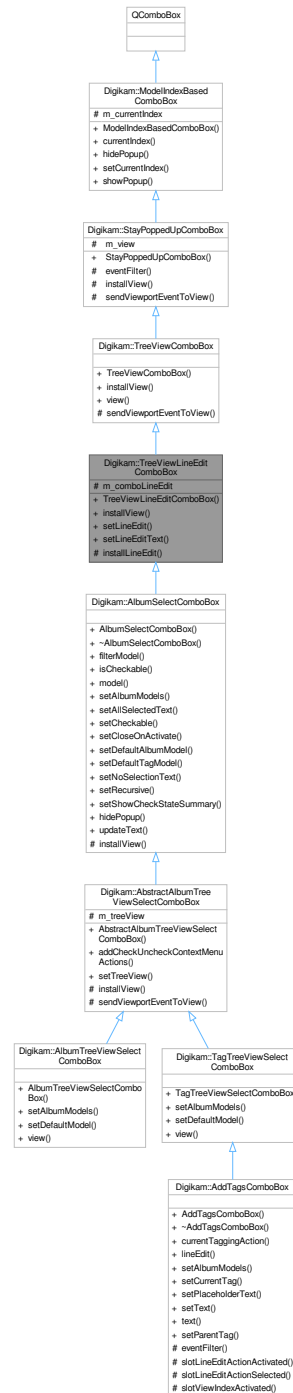
### 9.1370.2.3 view()

```
QTreeView * Digikam::TreeViewComboBox::view ( ) const
```

Valid after [installView\(\)](#) has been called

## 9.1371 Digikam::TreeViewLineEditComboBox Class Reference

Inheritance diagram for Digikam::TreeViewLineEditComboBox:



### Public Member Functions

- [TreeViewLineEditComboBox](#) (QWidget \*const parent=nullptr)

This class provides a [TreeViewComboBox](#) with a read-only line edit.

- void [installView](#) (QAbstractItemView \*view=nullptr) override  
*Replace the standard combo box list view with a QTreeView.*
- void [setLineEdit](#) (QLineEdit \*edit)
- void [setLineEditText](#) (const QString &text)  
*Set the text of the line edit (the text that is visible if the popup is not opened).*

### Public Member Functions inherited from [Digikam::TreeViewComboBox](#)

- [TreeViewComboBox](#) (QWidget \*parent=nullptr)  
*This class provides a QComboBox with a QTreeView instead of the usual QListView.*
- QTreeView \* [view](#) () const  
*Returns the QTreeView of this class.*

### Public Member Functions inherited from [Digikam::StayPoppedUpComboBox](#)

- [StayPoppedUpComboBox](#) (QWidget \*const parent=nullptr)  
*This class provides an abstract QComboBox with a custom view (which is created by implementing subclasses) instead of the usual QListView.*

### Public Member Functions inherited from [Digikam::ModelIndexBasedComboBox](#)

- [ModelIndexBasedComboBox](#) (QWidget \*const parent=nullptr)  
*QComboBox has a current index based on a single integer.*
- QModelIndex [currentIndex](#) () const
- void [hidePopup](#) () override
- void [setCurrentIndex](#) (const QModelIndex &index)
- void [showPopup](#) () override

### Protected Member Functions

- virtual void [installLineEdit](#) ()  
*Sets a line edit.*

### Protected Member Functions inherited from [Digikam::TreeViewComboBox](#)

- void [sendViewportEventToView](#) (QEvent \*e) override  
*Implement in subclass: Send the given event to the viewportEvent() method of m\_view.*

### Protected Member Functions inherited from [Digikam::StayPoppedUpComboBox](#)

- bool [eventFilter](#) (QObject \*watched, QEvent \*event) override
- void [installView](#) (QAbstractItemView \*view)  
*Replace the standard combo box list view with the given view.*

### Protected Attributes

- QLineEdit \* [m\\_comboLineEdit](#) = nullptr

## Protected Attributes inherited from [Digikam::StayPoppedUpComboBox](#)

- `QAbstractItemView * m_view = nullptr`

## Protected Attributes inherited from [Digikam::ModelIndexBasedComboBox](#)

- `QPersistentModelIndex m_currentIndex`

### 9.1371.1 Constructor & Destructor Documentation

#### 9.1371.1.1 [TreeViewLineEditComboBox\(\)](#)

```
Digikam::TreeViewLineEditComboBox::TreeViewLineEditComboBox (
    QWidget *const parent = nullptr ) [explicit]
```

The text in the line edit can be adjusted. The combo box will open on a click on the line edit. You need three steps: Construct the object, call `setModel()` with an appropriate `QAbstractItemModel`, then call [installView\(\)](#) to replace the standard combo box view with a `QTreeView`.

### 9.1371.2 Member Function Documentation

#### 9.1371.2.1 [installLineEdit\(\)](#)

```
void Digikam::TreeViewLineEditComboBox::installLineEdit ( ) [protected], [virtual]
```

Called by [installView\(\)](#). The default implementation is described above. An empty implementation will keep the default `QComboBox` line edit.

#### 9.1371.2.2 [installView\(\)](#)

```
void Digikam::TreeViewLineEditComboBox::installView (
    QAbstractItemView * view = nullptr ) [override], [virtual]
```

Call this after installing an appropriate model.

Reimplemented from [Digikam::TreeViewComboBox](#).

#### 9.1371.2.3 [setLineEditText\(\)](#)

```
void Digikam::TreeViewLineEditComboBox::setLineEditText (
    const QString & text )
```

Applicable only for default [installLineEdit\(\)](#) implementation.

## 9.1372 Digikam::TrimmedModifier Class Reference

Inheritance diagram for Digikam::TrimmedModifier:



### Public Member Functions

- QString [parseOperation](#) ([ParseSettings](#) &settings, const QRegularExpressionMatch &match) override  
*TODO: describe me.*

## Public Member Functions inherited from [Digikam::Modifier](#)

- **Modifier** (const QString &name, const QString &description)
- **Modifier** (const QString &name, const QString &description, const QString &icon)

## Public Member Functions inherited from [Digikam::Rule](#)

- **Rule** (const QString &name)
- **Rule** (const QString &name, const QString &icon)
- QString **description** () const
- QPixmap **icon** (Rule::IconType type=Rule::Action) const
- bool **isValid** () const

*Checks the validity of the parse object.*

- **ParseResults parse** ([ParseSettings](#) &settings)
- QRegularExpression & **regExp** () const
- TODO: This is probably not needed anymore.*
- QPushButton \* **registerButton** (QWidget \*parent)
- Register a button in the parent object.*
- QAction \* **registerMenu** (QMenu \*parent)
- Register a menu action in the parent object.*
- virtual void **reset** ()
- Resets the parser to its initial state.*
- TokenList & **tokens** () const
- bool **useTokenMenu** () const
- Returns true if a token menu is used.*

## Additional Inherited Members

## Public Types inherited from [Digikam::Rule](#)

- enum **IconType** { **Action** = 0 , **Dialog** }

## Signals inherited from [Digikam::Rule](#)

- void **signalTokenTriggered** (const QString &)

## Static Public Member Functions inherited from [Digikam::Rule](#)

- static QString **escapeToken** (const QString &token)
- Escape the token characters to make them work in regular expressions.*

## Protected Slots inherited from [Digikam::Rule](#)

- virtual void **slotTokenTriggered** (const QString &)



## Protected Member Functions inherited from [Digikam::Rule](#)

- bool [addToken](#) (const QString &id, const QString &description, const QString &actionName=QString())  
*add a token to the parser, every parser should at least assign one token object*
- void [setDescription](#) (const QString &desc)
- void [setIcon](#) (const QString &pixmap)
- void [setRegExp](#) (const QRegularExpression &regExp)
- void [setUseTokenMenu](#) (bool value)  
*If multiple tokens have been assigned to a rule, a menu will be created.*

### 9.1372.1 Member Function Documentation

#### 9.1372.1.1 [parseOperation\(\)](#)

```
QString Digikam::TrimmedModifier::parseOperation (
    ParseSettings & settings,
    const QRegularExpressionMatch & match ) [override], [virtual]
```

##### Parameters

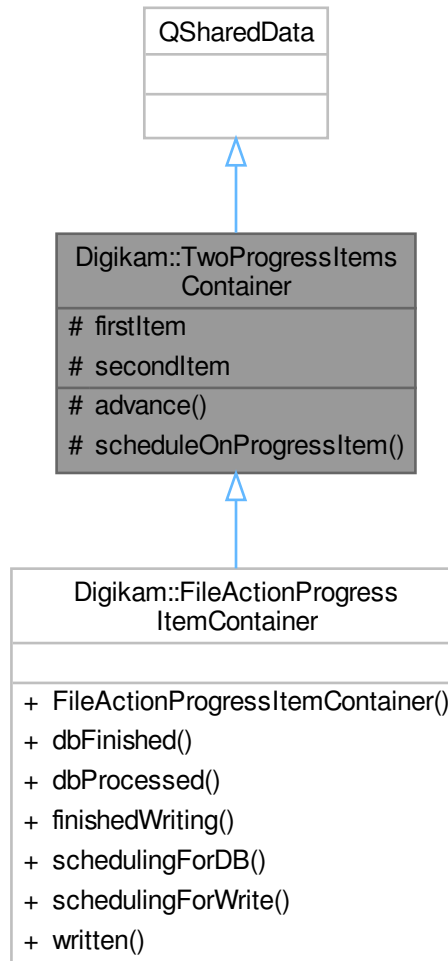
<i>settings</i>	contains settings
<i>match</i>	result of the regular expression match done in <a href="#">Option::parse()</a>

##### Returns

Implements [Digikam::Modifier](#).

## 9.1373 Digikam::TwoProgressItemsContainer Class Reference

Inheritance diagram for Digikam::TwoProgressItemsContainer:



### Protected Member Functions

- void **advance** (QAtomicPointer< [ProgressItem](#) > &ptr, int n)
- void **scheduleOnProgressItem** (QAtomicPointer< [ProgressItem](#) > &ptr, int total, const QString &action, [FileActionProgressItemCreator](#) \*const creator)

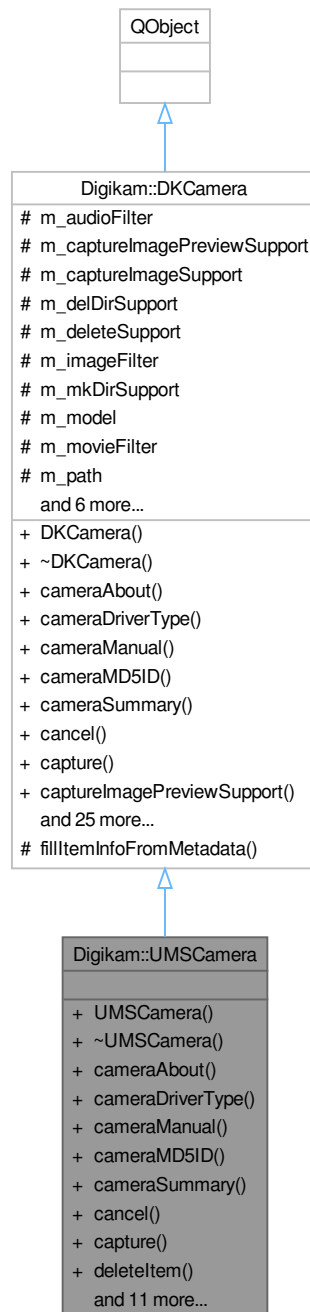
### Protected Attributes

- QAtomicPointer< [ProgressItem](#) > **firstItem**
- QAtomicPointer< [ProgressItem](#) > **secondItem**

## 9.1374 Digikam::UMSCamera Class Reference

USB Mass Storage camera Implementation of abstract type [DKCamera](#).

Inheritance diagram for Digikam::UMSCamera:



### Public Member Functions

- **UMSCamera** (const QString &title, const QString &model, const QString &port, const QString &path)

- bool [cameraAbout](#) (QString &about) override
- DKCamera::CameraDriverType [cameraDriverType](#) () override
- bool [cameraManual](#) (QString &>manual) override
- QByteArray [cameraMD5ID](#) () override
- bool [cameraSummary](#) (QString &summary) override
- void [cancel](#) () override
- bool [capture](#) (CamItemInfo &itemInfo) override
  - Method not supported by UMS camera.*
- bool [deleteItem](#) (const QString &folder, const QString &itemName) override
- bool [doConnect](#) () override
- bool [downloadItem](#) (const QString &folder, const QString &itemName, const QString &saveFile) override
- bool [getFolders](#) (const QString &folder) override
- bool [getFreeSpace](#) (qint64 &bytesSize, qint64 &bytesAvail) override
- void [getItemInfo](#) (const QString &folder, const QString &itemName, [CamItemInfo](#) &info, bool useMetadata) override
- bool [getItemsInfoList](#) (const QString &folder, bool useMetadata, [CamItemInfoList](#) &infoList) override
  - If getImageDimensions is false, the camera shall set width and height to -1 if the values are not immediately available.*
- bool [getMetadata](#) (const QString &folder, const QString &itemName, [DMetadata](#) &meta) override
- bool [getPreview](#) (QImage &preview) override
  - Method not supported by UMS camera.*
- bool [getThumbnail](#) (const QString &folder, const QString &itemName, QImage &thumbnail) override
- bool [setLockItem](#) (const QString &folder, const QString &itemName, bool lock) override
- bool [uploadItem](#) (const QString &folder, const QString &itemName, const QString &localFile, [CamItemInfo](#) &info) override

## Public Member Functions inherited from [Digikam::DKCamera](#)

- **DKCamera** (const QString &title, const QString &model, const QString &port, const QString &path)
- bool **captureImagePreviewSupport** () const
- bool **captureImageSupport** () const
- bool **delDirSupport** () const
- bool **deleteSupport** () const
- QString **mime** (const QString &fileext) const
- bool **mkDirSupport** () const
- QString **model** () const
- QString **path** () const
- QString **port** () const
- void **printSupportedFeatures** ()
- bool **thumbnailSupport** () const
- QString **title** () const
- bool **uploadSupport** () const
- QString **uuid** () const

## Additional Inherited Members

## Public Types inherited from [Digikam::DKCamera](#)

- enum **CameraDriverType** { **GPhotoDriver** = 0 , **UMSDriver** }

## Signals inherited from [Digikam::DKCamera](#)

- void **signalFolderList** (const QStringList &)

## Protected Member Functions inherited from [Digikam::DKCamera](#)

- void `fillItemInfoFromMetadata` ([CamItemInfo](#) &item, const [DMetadata](#) &meta) const

## Protected Attributes inherited from [Digikam::DKCamera](#)

- `QString m_audioFilter`
- `bool m_captureImagePreviewSupport = false`
- `bool m_captureImageSupport = false`
- `bool m_delDirSupport = false`
- `bool m_deleteSupport = false`
- `QString m_imageFilter`
- `bool m_mkDirSupport = false`
- `QString m_model`
- `QString m_movieFilter`
- `QString m_path`
- `QString m_port`
- `QString m_rawFilter`
- `bool m_thumbnailSupport = false`
- `QString m_title`
- `bool m_uploadSupport = false`
- `QString m_uuid`

### 9.1374.1 Member Function Documentation

#### 9.1374.1.1 `cameraAbout()`

```
bool Digikam::UMSCamera::cameraAbout (
    QString & about ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

#### 9.1374.1.2 `cameraDriverType()`

```
DKCamera::CameraDriverType Digikam::UMSCamera::cameraDriverType ( ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

#### 9.1374.1.3 `cameraManual()`

```
bool Digikam::UMSCamera::cameraManual (
    QString & manual ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

#### 9.1374.1.4 `cameraMD5ID()`

```
QByteArray Digikam::UMSCamera::cameraMD5ID ( ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

### 9.1374.1.5 cameraSummary()

```
bool Digikam::UMSCamera::cameraSummary (
    QString & summary ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

### 9.1374.1.6 cancel()

```
void Digikam::UMSCamera::cancel ( ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

### 9.1374.1.7 capture()

```
bool Digikam::UMSCamera::capture (
    CamItemInfo & itemInfo ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

### 9.1374.1.8 deleteItem()

```
bool Digikam::UMSCamera::deleteItem (
    const QString & folder,
    const QString & itemName ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

### 9.1374.1.9 doConnect()

```
bool Digikam::UMSCamera::doConnect ( ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

### 9.1374.1.10 downloadItem()

```
bool Digikam::UMSCamera::downloadItem (
    const QString & folder,
    const QString & itemName,
    const QString & saveFile ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

### 9.1374.1.11 getFolders()

```
bool Digikam::UMSCamera::getFolders (
    const QString & folder ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

### 9.1374.1.12 getFreeSpace()

```
bool Digikam::UMSCamera::getFreeSpace (
    qint64 & bytesSize,
    qint64 & bytesAvail ) [override], [virtual]
```

#### Note

implemented in gui, outside the camera thread.

Implements [Digikam::DKCamera](#).

### 9.1374.1.13 getItemInfo()

```
void Digikam::UMSCamera::getItemInfo (
    const QString & folder,
    const QString & itemName,
    CamItemInfo & info,
    bool useMetadata ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

### 9.1374.1.14 getItemsInfoList()

```
bool Digikam::UMSCamera::getItemsInfoList (
    const QString & folder,
    bool useMetadata,
    CamItemInfoList & infoList ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

### 9.1374.1.15 getMetadata()

```
bool Digikam::UMSCamera::getMetadata (
    const QString & folder,
    const QString & itemName,
    DMetadata & meta ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

### 9.1374.1.16 getPreview()

```
bool Digikam::UMSCamera::getPreview (
    QImage & preview ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

**9.1374.1.17 getThumbnail()**

```
bool Digikam::UMSCamera::getThumbnail (
    const QString & folder,
    const QString & itemName,
    QImage & thumbnail ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

**9.1374.1.18 setLockItem()**

```
bool Digikam::UMSCamera::setLockItem (
    const QString & folder,
    const QString & itemName,
    bool lock ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).

**9.1374.1.19 uploadItem()**

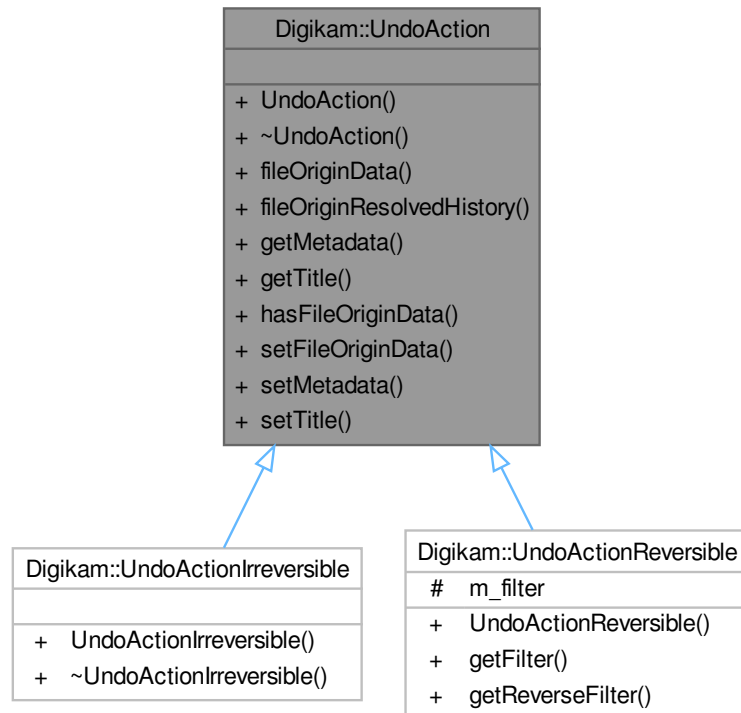
```
bool Digikam::UMSCamera::uploadItem (
    const QString & folder,
    const QString & itemName,
    const QString & localFile,
    CamItemInfo & info ) [override], [virtual]
```

Implements [Digikam::DKCamera](#).



## 9.1375 Digikam::UndoAction Class Reference

Inheritance diagram for Digikam::UndoAction:

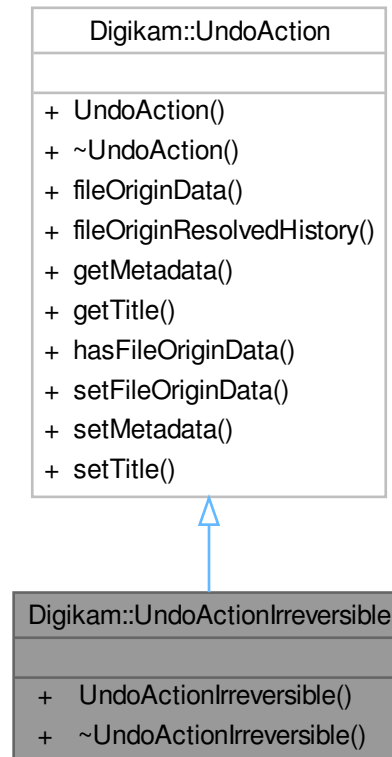


### Public Member Functions

- **UndoAction** (const [EditorCore](#) \*const core)
- QVariant **fileOriginData** () const
- [DImageHistory](#) **fileOriginResolvedHistory** () const
- [UndoMetadataContainer](#) **getMetadata** () const
- QString **getTitle** () const
- bool **hasFileOriginData** () const
- void **setFileOriginData** (const QVariant &data, const [DImageHistory](#) &resolvedInitialHistory)
- void **setMetadata** (const [UndoMetadataContainer](#) &)
- void **setTitle** (const QString &title)

## 9.1376 Digikam::UndoActionIrreversible Class Reference

Inheritance diagram for Digikam::UndoActionIrreversible:



### Public Member Functions

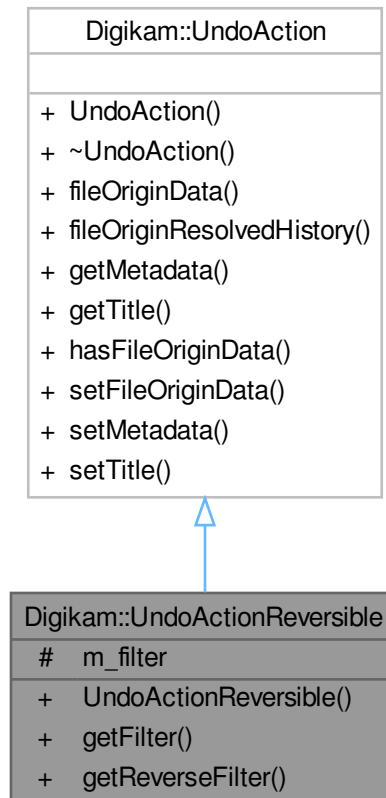
- **UndoActionIrreversible** (const [EditorCore](#) \*const core, const QString &title=QString())

### Public Member Functions inherited from [Digikam::UndoAction](#)

- **UndoAction** (const [EditorCore](#) \*const core)
- QVariant **fileOriginData** () const
- [DImageHistory](#) **fileOriginResolvedHistory** () const
- [UndoMetadataContainer](#) **getMetadata** () const
- QString **getTitle** () const
- bool **hasFileOriginData** () const
- void **setFileOriginData** (const QVariant &data, const [DImageHistory](#) &resolvedInitialHistory)
- void **setMetadata** (const [UndoMetadataContainer](#) &)
- void **setTitle** (const QString &title)

## 9.1377 Digikam::UndoActionReversible Class Reference

Inheritance diagram for Digikam::UndoActionReversible:



### Public Member Functions

- **UndoActionReversible** (const [EditorCore](#) \*const core, const [DImageBuiltinFilter](#) &reversibleFilter)
- [DImageBuiltinFilter](#) **getFilter** () const
- [DImageBuiltinFilter](#) **getReverseFilter** () const

### Public Member Functions inherited from [Digikam::UndoAction](#)

- **UndoAction** (const [EditorCore](#) \*const core)
- QVariant **fileOriginData** () const
- [DImageHistory](#) **fileOriginResolvedHistory** () const
- [UndoMetadataContainer](#) **getMetadata** () const
- QString **getTitle** () const
- bool **hasFileOriginData** () const
- void **setFileOriginData** (const QVariant &data, const [DImageHistory](#) &resolvedInitialHistory)
- void **setMetadata** (const [UndoMetadataContainer](#) &)
- void **setTitle** (const QString &title)

## Protected Attributes

- [DImgBuiltinFilter](#) `m_filter`

## 9.1378 Digikam::UndoCache Class Reference

### Public Member Functions

- void **clear** ()  
*Delete all cache files.*
- void **clearFrom** (int level)  
*Delete all cache files starting from the given level upwards.*
- [DImg](#) **getData** (int level) const  
*Get the image data from a cache file.*
- bool **putData** (int level, const [DImg](#) &img) const  
*Write the image data into a cache file.*

## 9.1379 Digikam::UndoManager Class Reference

### Public Member Functions

- **UndoManager** ([EditorCore](#) \*const core)
- void **addAction** ([UndoAction](#) \*const action)
- bool **anyMoreRedo** () const
- bool **anyMoreUndo** () const
- int **availableRedoSteps** () const
- int **availableUndoSteps** () const
- void **clear** (bool clearCache=true)
- void **clearPreviousOriginData** ()
- [DImageHistory](#) **getImageHistoryOfFullRedo** () const  
*The history if all available redo steps are redone.*
- QStringList **getRedoHistory** () const
- QStringList **getUndoHistory** () const
- bool **hasChanges** () const
- bool **isAtOrigin** () const
- bool **putImageDataAndHistory** ([DImg](#) \*const img, int stepsBack) const
- void **redo** ()
- void **rollbackToOrigin** ()
- void **setOrigin** () const
- void **undo** ()

## 9.1380 Digikam::UndoMetadataContainer Class Reference

### Public Member Functions

- bool **changesIccProfile** (const [DImg](#) &target) const
- void **toImage** ([DImg](#) &img) const  
*Write this container's values to the [DImg](#).*

### Static Public Member Functions

- static [UndoMetadataContainer](#) **fromImage** (const [DImg](#) &img)  
*Fill a container from the [DImg](#).*

### Public Attributes

- [DImageHistory](#) **history**
- [IccProfile](#) **profile**

## 9.1381 Digikam::UndoState Class Reference

### Public Attributes

- bool **hasChanges** = false
- bool **hasRedo** = false
- bool **hasUndo** = false
- bool **hasUndoableChanges** = false

## 9.1382 Digikam::UniqueModifier Class Reference

Inheritance diagram for Digikam::UniqueModifier:



### Public Member Functions

- QString `parseOperation` (`ParseSettings` &settings, const QRegularExpressionMatch &match) override  
*TODO: describe me.*
- void `reset` () override  
*Resets the parser to its initial state.*

## Public Member Functions inherited from Digikam::Modifier

- **Modifier** (const QString &name, const QString &description)
- **Modifier** (const QString &name, const QString &description, const QString &icon)

## Public Member Functions inherited from Digikam::Rule

- **Rule** (const QString &name)
- **Rule** (const QString &name, const QString &icon)
- QString **description** () const
- QPixmap **icon** (Rule::IconType type=Rule::Action) const
- bool **isValid** () const

*Checks the validity of the parse object.*

- **ParseResults parse** (ParseSettings &settings)
- QRegularExpression & **regExp** () const
- QPushButton \* **registerButton** (QWidget \*parent)

*TODO: This is probably not needed anymore.*

*Register a button in the parent object.*

- QAction \* **registerMenu** (QMenu \*parent)
- TokenList & **tokens** () const
- bool **useTokenMenu** () const

*Returns true if a token menu is used.*

## Additional Inherited Members

## Public Types inherited from Digikam::Rule

- enum **IconType** { **Action** = 0 , **Dialog** }

## Signals inherited from Digikam::Rule

- void **signalTokenTriggered** (const QString &)

## Static Public Member Functions inherited from Digikam::Rule

- static QString **escapeToken** (const QString &token)  
*Escape the token characters to make them work in regular expressions.*

## Protected Slots inherited from Digikam::Rule

- virtual void **slotTokenTriggered** (const QString &)

## Protected Member Functions inherited from [Digikam::Rule](#)

- bool [addToken](#) (const QString &id, const QString &description, const QString &actionName=QString())  
*add a token to the parser, every parser should at least assign one token object*
- void [setDescription](#) (const QString &desc)
- void [setIcon](#) (const QString &pixmap)
- void [setRegExp](#) (const QRegularExpression &regExp)
- void [setUseTokenMenu](#) (bool value)  
*If multiple tokens have been assigned to a rule, a menu will be created.*

### 9.1382.1 Member Function Documentation

#### 9.1382.1.1 [parseOperation\(\)](#)

```
QString Digikam::UniqueModifier::parseOperation (
    ParseSettings & settings,
    const QRegularExpressionMatch & match ) [override], [virtual]
```

##### Parameters

<i>settings</i>	contains settings
<i>match</i>	result of the regular expression match done in <a href="#">Option::parse()</a>

##### Returns

Implements [Digikam::Modifier](#).

#### 9.1382.1.2 [reset\(\)](#)

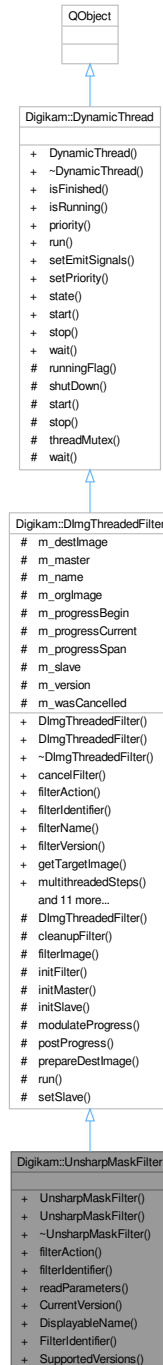
```
void Digikam::UniqueModifier::reset ( ) [override], [virtual]
```

Reimplemented from [Digikam::Rule](#).



## 9.1383 Digikam::UnsharpMaskFilter Class Reference

Inheritance diagram for Digikam::UnsharpMaskFilter:



### Public Member Functions

- **UnsharpMaskFilter** (`Dlmg` \*const orgImage, `QObject` \*const parent=nullptr, double radius=1.0, double amount=1.0, double threshold=0.05, bool luma=false)

- **UnsharpMaskFilter** (QObject \*const parent=nullptr)
- **FilterAction filterAction** () override  
*Returns the action description corresponding to currently set options.*
- QString **filterIdentifier** () const override  
*Return the identifier for this filter in the image history.*
- void **readParameters** (const **FilterAction** &action) override

## Public Member Functions inherited from **Digikam::DImgThreadedFilter**

- **DImgThreadedFilter** (DImg \*const orgImage, QObject \*const parent, const QString &name=QString())  
*Constructs a filter with all arguments (ready to use).*
- **DImgThreadedFilter** (QObject \*const parent=nullptr, const QString &name=QString())  
*Constructs a filter without argument.*
- virtual void **cancelFilter** ()  
*Cancel the threaded computation.*
- const QString & **filterName** ()
- int **filterVersion** () const
- **DImg getTargetImage** ()
- QList< int > **multithreadedSteps** (int stop, int start=0) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool **parametersSuccessfullyRead** () const  
*Optional: error handling for readParameters.*
- virtual QString **readParametersError** (const **FilterAction** &actionThatFailed) const
- void **setFilterName** (const QString &name)
- void **setFilterVersion** (int version)  
*Replaying a filter action: Set the filter version.*
- void **setOriginalImage** (const **DImg** &orgImage)
- void **setupAndStartDirectly** (const **DImg** &orgImage, **DImgThreadedFilter** \*const master, int progress←Begin=0, int progressEnd=100)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void **setupFilter** (const **DImg** &orgImage)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void **startFilter** ()  
*Start the threaded computation.*
- virtual void **startFilterDirectly** ()  
*Start computation of this filter, directly in this thread.*
- virtual QList< int > **supportedVersions** () const

## Public Member Functions inherited from **Digikam::DynamicThread**

- **DynamicThread** (QObject \*const parent=nullptr)  
*This class extends QRunnable, so you have to reimplement virtual void [run\(\)](#).*
- **~DynamicThread** () override  
*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool **isFinished** () const
- bool **isRunning** () const
- QThread::Priority **priority** () const
- void **setEmitSignals** (bool emitThem)
- void **setPriority** (QThread::Priority priority)  
*Sets the priority for this dynamic thread.*
- State **state** () const

### Static Public Member Functions

- static int **CurrentVersion** ()
- static QString **DisplayName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

### Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

### Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

### Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()  
*Emitted if emitSignals is enabled.*

### Protected Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void [cleanupFilter](#) ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void [initFilter](#) ()  
*Start filter operation before threaded method.*
- void [initMaster](#) ()
- void [initSlave](#) ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int [modulateProgress](#) (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void [postProgress](#) (int progress)  
*Emit progress info.*
- virtual void [prepareDestImage](#) ()
- void [run](#) () override  
*List of threaded operations by filter.*
- void [setSlave](#) ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

### Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool [runningFlag](#) () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void [start](#) (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void [stop](#) (const QMutexLocker< QMutex > &locker)
- QMutex \* [threadMutex](#) () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void [wait](#) (QMutexLocker< QMutex > &locker)

### Protected Attributes inherited from [Digikam::DImgThreadedFilter](#)

- [DImg](#) [m\\_destImage](#)  
*Output image data.*
- [DImgThreadedFilter](#) \* [m\\_master](#) = nullptr  
*The master of this slave filter.*
- QString [m\\_name](#)  
*Filter name.*
- [DImg](#) [m\\_orgImage](#)  
*Copy of original Image data.*
- int [m\\_progressBegin](#) = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int [m\\_progressCurrent](#) = 0  
*To prevent signals bombarding with progress indicator value in postProgress().*
- int [m\\_progressSpan](#) = 0
- [DImgThreadedFilter](#) \* [m\\_slave](#) = nullptr  
*The current slave.*
- int [m\\_version](#) = 1
- bool [m\\_wasCancelled](#) = false

## 9.1383.1 Member Function Documentation

### 9.1383.1.1 filterAction()

`FilterAction` Digikam::UnsharpMaskFilter::filterAction ( ) [override], [virtual]

Implements `Digikam::DImgThreadedFilter`.

### 9.1383.1.2 filterIdentifier()

`QString` Digikam::UnsharpMaskFilter::filterIdentifier ( ) const [inline], [override], [virtual]

Implements `Digikam::DImgThreadedFilter`.

### 9.1383.1.3 readParameters()

```
void Digikam::UnsharpMaskFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements `Digikam::DImgThreadedFilter`.

## 9.1384 Digikam::VersionFileInfo Class Reference

### Public Member Functions

- `VersionFileInfo` (const `QString` &path, const `QString` &fileName, const `QString` &format)
- `QString` `filePath` ( ) const
- `QUrl` `fileUrl` ( ) const
- bool `isNull` ( ) const

### Public Attributes

- `QString` `fileName`
- `QString` `format`
- `QString` `path`

## 9.1385 Digikam::VersionFileOperation Class Reference

### Public Types

- enum `Task` {  
`NewFile` = 1 << 0, `Replace` = 1 << 1, `SaveAndDelete` = 1 << 2, `MoveToIntermediate` = 1 << 3,  
`StoreIntermediates` = 1 << 4 }
- typedef `QFlags`< `Task` > `Tasks`

## Public Member Functions

- [VersionFileOperation](#) ()=default  
*This class describes an operation necessary for storing an image under version control.*
- `QStringList` **allFilePaths** () const  
*Returns a list with all saving locations, for main result or intermediates.*

## Public Attributes

- [VersionFileInfo](#) **intermediateForLoadedFile**
- `QMap< int, VersionFileInfo >` **intermediates**
- [VersionFileInfo](#) **loadedFile**
- [VersionFileInfo](#) **saveFile**
- Tasks **tasks**

## 9.1385.1 Member Enumeration Documentation

### 9.1385.1.1 Task

```
enum Digikam::VersionFileOperation::Task
```

#### Enumerator

NewFile	saveFile is a new file. Excludes Replace.
Replace	loadedFile and saveFile are the same - replace. Excludes NewFile.
SaveAndDelete	Similar to Replace, but the new file name differs from the old one, which should be removed.
MoveToIntermediate	Move loadedFile to loadedFileToIntermediate.
StoreIntermediates	Store additional snapshots from within history.

## 9.1385.2 Constructor & Destructor Documentation

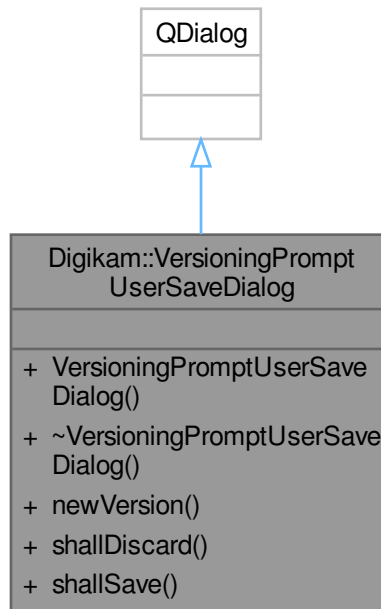
### 9.1385.2.1 VersionFileOperation()

```
Digikam::VersionFileOperation::VersionFileOperation ( ) [explicit], [default]
```

The loadedFile and current history is given to the [VersionManager](#). The saveFile is the destination of the save operation. If the loadedFile shall be moved to an intermediate, the name is given in intermediateForLoadedFile. The intermediates map may contain name of intermediates to save the state after action i of the history (initial← ResolvedHistory.size() <= i < currentHistory.size() - 1).

## 9.1386 Digikam::VersioningPromptUserSaveDialog Class Reference

Inheritance diagram for Digikam::VersioningPromptUserSaveDialog:



### Public Member Functions

- **VersioningPromptUserSaveDialog** (`QWidget *const parent`)
- `bool newVersion () const`
- `bool shallDiscard () const`
- `bool shallSave () const`

## 9.1387 Digikam::VersionItemFilterSettings Class Reference

### Public Member Functions

- **VersionItemFilterSettings** (`const VersionManagerSettings &settings`)
- `bool isExemptedBySettings` (`const ItemInfo &info`) `const`
- `bool isFiltering () const`  
*Returns if images will be filtered by these criteria at all.*
- `bool isFilteringByTags () const`  
*Returns if the tag is a filter criteria.*
- `bool isHiddenBySettings` (`const ItemInfo &info`) `const`
- `bool matches` (`const ItemInfo &info`) `const`  
*Returns true if the given ItemInfo matches the filter criteria.*
- `bool operator==` (`const VersionItemFilterSettings &other`) `const`
- `void setExceptionList` (`const QList< qulonglong > &idlist`, `const QString &id`)  
*Add list with exceptions: These images will be exempted from filtering by this filter.*
- `void setVersionManagerSettings` (`const VersionManagerSettings &settings`)  
*— Tags filter —*

### Protected Attributes

- QHash< QString, QList< qlonglong > > **m\_exceptionLists**
- int **m\_exceptionTagFilter** = 0
- QList< int > **m\_excludeTagFilter**  
*DatabaseFields::Set watchFlags() const: Would return 0.*
- int **m\_includeTagFilter** = 0

## 9.1388 Digikam::VersionManager Class Reference

### Public Types

- enum **FileNameType** { **CurrentVersionName** , **NewVersionName** }

### Public Member Functions

- [VersionNamingScheme](#) \* **namingScheme** () const
- [VersionFileOperation](#) **operation** (FileNameType request, const [VersionFileInfo](#) &loadedFile, const [DImageHistory](#) &initialResolvedHistory, const [DImageHistory](#) &currentHistory)
- [VersionFileOperation](#) **operationNewVersionAs** (const [VersionFileInfo](#) &loadedFile, const [VersionFileInfo](#) &saveLocation, const [DImageHistory](#) &initialResolvedHistory, const [DImageHistory](#) &currentHistory)
- [VersionFileOperation](#) **operationNewVersionInFormat** (const [VersionFileInfo](#) &loadedFile, const QString &format, const [DImageHistory](#) &initialResolvedHistory, const [DImageHistory](#) &currentHistory)
- void **setNamingScheme** ([VersionNamingScheme](#) \*scheme)
- void **setSettings** (const [VersionManagerSettings](#) &settings)
- [VersionManagerSettings](#) **settings** () const
- virtual QString **toplevelDirectory** (const QString &path)
- virtual QStringList **workspaceFileFormats** () const

## 9.1389 Digikam::VersionManagerSettings Class Reference

### Public Types

- enum **EditorClosingMode** { **AlwaysAsk** , **AutoSave** }
- typedef QFlags< IntermediateSavepoint > **IntermediateBehavior**
- enum **IntermediateSavepoint** { **NoIntermediates** = 0 , **AfterEachSession** = 1 << 0 , **AfterRawConversion** = 1 << 1 , **WhenNotReproducible** = 1 << 2 }
- enum **ShowInViewFlag** { **OnlyShowCurrent** = 0 , **ShowOriginal** = 1 << 0 , **ShowIntermediates** = 1 << 1 }
- typedef QFlags< ShowInViewFlag > **ShowInViewFlags**

### Public Member Functions

- void **readFromConfig** (const KConfigGroup &group)
- void **writeToConfig** (KConfigGroup &group) const

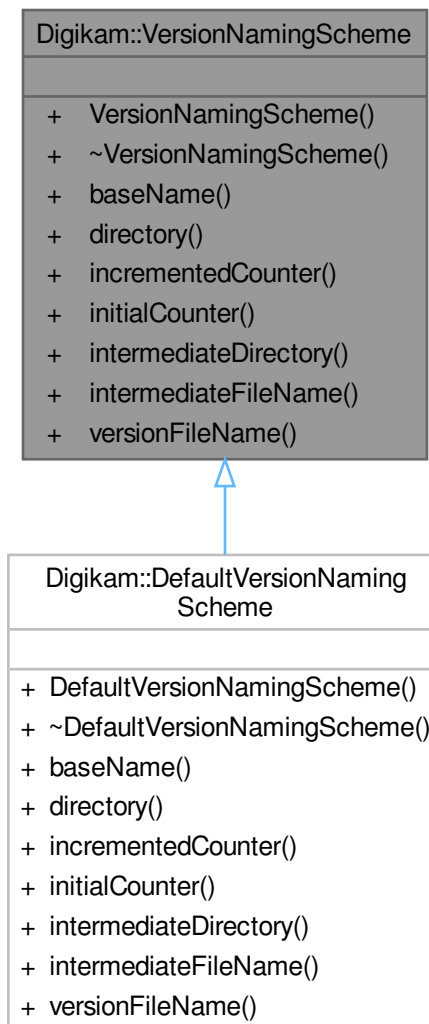


**Public Attributes**

- EditorClosingMode **editorClosingMode** = AlwaysAsk
- bool **enabled** = true
- QString **format** = QLatin1String("JPG")
  - Image format string as defined for database, in upper case.*
- IntermediateBehavior **saveIntermediateVersions** = NoIntermediates
- ShowInViewFlags **showInViewFlags** = ShowOriginal

**9.1390 Digikam::VersionNamingScheme Class Reference**

Inheritance diagram for Digikam::VersionNamingScheme:



## Public Member Functions

- **VersionNamingScheme** ()=default  
*Creates and analyzes file names of versioned files.*
- virtual QString **baseName** (const QString &path, const QString &filename, QVariant \*counter=nullptr, QVariant \*intermediateCounter=nullptr)=0  
*Analyzes the given file name.*
- virtual QString **directory** (const QString &path, const QString &filename)=0  
*For a loaded file in directory path and with file name filename, returns the directory in which a new version (a new intermediate version, resp.) shall be stored.*
- virtual QVariant **incrementedCounter** (const QVariant &counter)=0  
*Returns the given counter "incremented", that is, changed in a steady, repeatable fashion.*
- virtual QVariant **initialCounter** ()=0  
*Returns an initial counter value for version and intermediate number counters.*
- virtual QString **intermediateDirectory** (const QString &currentPath, const QString &fileName)=0
- virtual QString **intermediateFileName** (const QString &path, const QString &filename, const QVariant &version, const QVariant &counter)=0  
*Creates a version file name for an intermediate file in given directory, as previously returned by [directory\(\)](#), given baseName, as previously returned by [baseName](#), version and intermediate number counter.*
- virtual QString **versionFileName** (const QString &path, const QString &baseName, const QVariant &counter)=0  
*Creates a version file name for a file in given directory, as previously returned by [directory\(\)](#), given baseName, as previously returned by [baseName](#), and version counter.*

## 9.1390.1 Member Function Documentation

### 9.1390.1.1 baseName()

```
virtual QString Digikam::VersionNamingScheme::baseName (
    const QString & path,
    const QString & filename,
    QVariant * counter = nullptr,
    QVariant * intermediateCounter = nullptr ) [pure virtual]
```

Returns the basename in the sense of stripping the file name of all added version information: A scheme that appends a number, like "MyFile-1.jpg", shall return "MyFile". Path is the directory, filename the file name, so path + filename is the file path. If counter is given, and the given file name has a version number, write it to counter. If intermediateCounter is given, and the given file name has an intermediate counter number, write it to counter. If not available, do not touch the given counters. See [initialCounter\(\)](#) for the valid counter formats.

Implemented in [Digikam::DefaultVersionNamingScheme](#).

### 9.1390.1.2 directory()

```
virtual QString Digikam::VersionNamingScheme::directory (
    const QString & path,
    const QString & filename ) [pure virtual]
```

Implemented in [Digikam::DefaultVersionNamingScheme](#).

### 9.1390.1.3 incrementedCounter()

```
virtual QVariant Digikam::VersionNamingScheme::incrementedCounter (
    const QVariant & counter ) [pure virtual]
```

You shall never return the given counter.

Implemented in [Digikam::DefaultVersionNamingScheme](#).

### 9.1390.1.4 initialCounter()

```
virtual QVariant Digikam::VersionNamingScheme::initialCounter ( ) [pure virtual]
```

There are two places where you shall generate counters You will receive the given QVariant in [incrementedCounter\(\)](#), [versionFileName\(\)](#) and [baseName\(\)](#), and you shall read a counter value from a generated file name in [baseName\(\)](#).

Implemented in [Digikam::DefaultVersionNamingScheme](#).

### 9.1390.1.5 intermediateFileName()

```
virtual QString Digikam::VersionNamingScheme::intermediateFileName (
    const QString & path,
    const QString & filename,
    const QVariant & version,
    const QVariant & counter ) [pure virtual]
```

Do not append a file suffix. You do not need to check if the file exists.

Implemented in [Digikam::DefaultVersionNamingScheme](#).

### 9.1390.1.6 versionFileName()

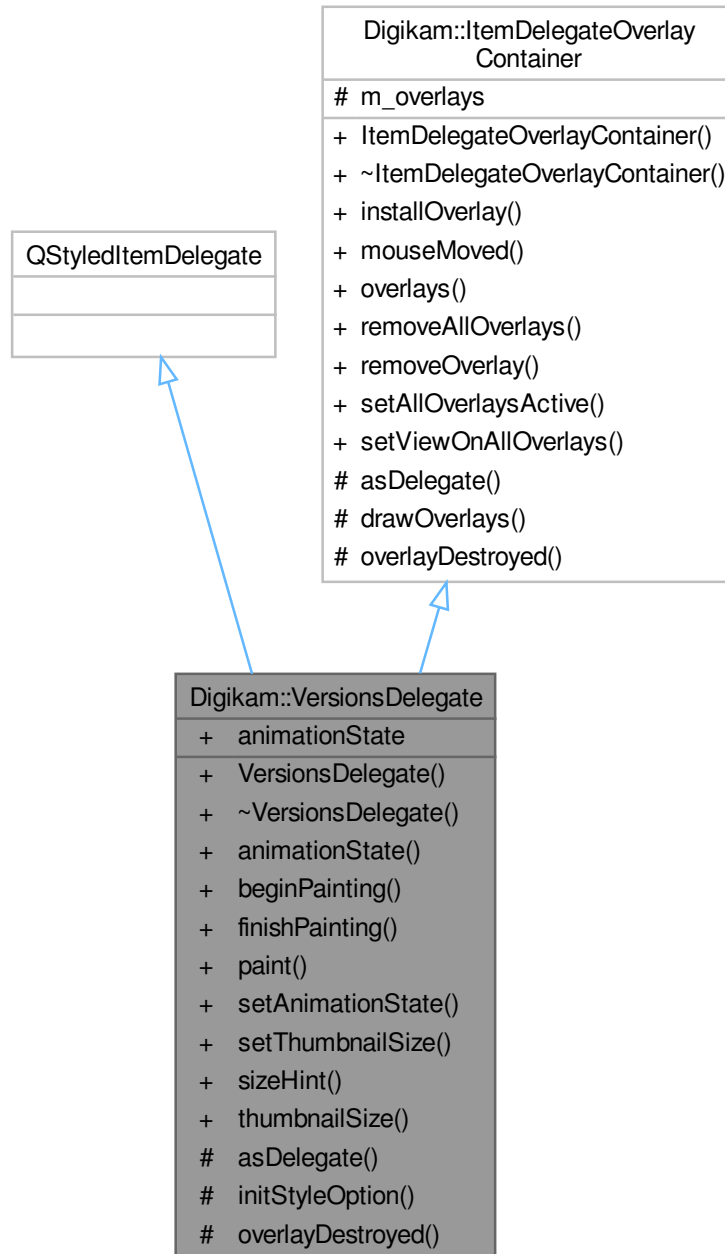
```
virtual QString Digikam::VersionNamingScheme::versionFileName (
    const QString & path,
    const QString & baseName,
    const QVariant & counter ) [pure virtual]
```

Do not append a file suffix. You do not need to check if the file exists.

Implemented in [Digikam::DefaultVersionNamingScheme](#).

## 9.1391 Digikam::VersionsDelegate Class Reference

Inheritance diagram for Digikam::VersionsDelegate:



### Signals

- void **animationStateChanged** ()
- void **hideNotification** ()
- void **requestNotification** (const QModelIndex &index, const QString &message)
- void **visualChange** ()

**Public Member Functions**

- **VersionsDelegate** (QObject \*const parent)
- int **animationState** () const
- void **beginPainting** ()
- void **finishPainting** ()
- void **paint** (QPainter \*painter, const QStyleOptionViewItem &option, const QModelIndex &index) const override
- void **setAnimationState** (int animationState)
- void **setThumbnailSize** (int size) const
- QSize **sizeHint** (const QStyleOptionViewItem &option, const QModelIndex &index) const override
- int **thumbnailSize** () const

**Public Member Functions inherited from [Digikam::ItemDelegateOverlayContainer](#)**

- [ItemDelegateOverlayContainer](#) ()=default  
*This is a sample implementation for delegate management methods, to be inherited by a delegate.*
- void **installOverlay** ([ItemDelegateOverlay](#) \*overlay)
- void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)
- QList< [ItemDelegateOverlay](#) \* > **overlays** () const
- void **removeAllOverlays** ()
- void **removeOverlay** ([ItemDelegateOverlay](#) \*overlay)
- void **setAllOverlaysActive** (bool active)
- void **setViewOnAllOverlays** (QAbstractItemView \*view)

**Protected Slots**

- void **overlayDestroyed** (QObject \*o) override

**Protected Member Functions**

- QAbstractItemDelegate \* **asDelegate** () override  
*Returns the delegate, typically, the derived class.*
- void **initStyleOption** (QStyleOptionViewItem \*option, const QModelIndex &index) const override

**Protected Member Functions inherited from [Digikam::ItemDelegateOverlayContainer](#)**

- virtual void **drawOverlays** (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index) const
- virtual void **overlayDestroyed** (QObject \*o)  
*Declare as slot in the derived class calling this method.*

**Properties**

- int **animationState**

## Additional Inherited Members

### Protected Attributes inherited from [Digikam::ItemDelegateOverlayContainer](#)

- `QList< ItemDelegateOverlay * > m_overlays`

## 9.1391.1 Member Function Documentation

### 9.1391.1.1 `asDelegate()`

```
QAbstractItemDelegate * Digikam::VersionsDelegate::asDelegate ( ) [inline], [override], [protected], [virtual]
```

Implements [Digikam::ItemDelegateOverlayContainer](#).

### 9.1391.1.2 `requestNotification`

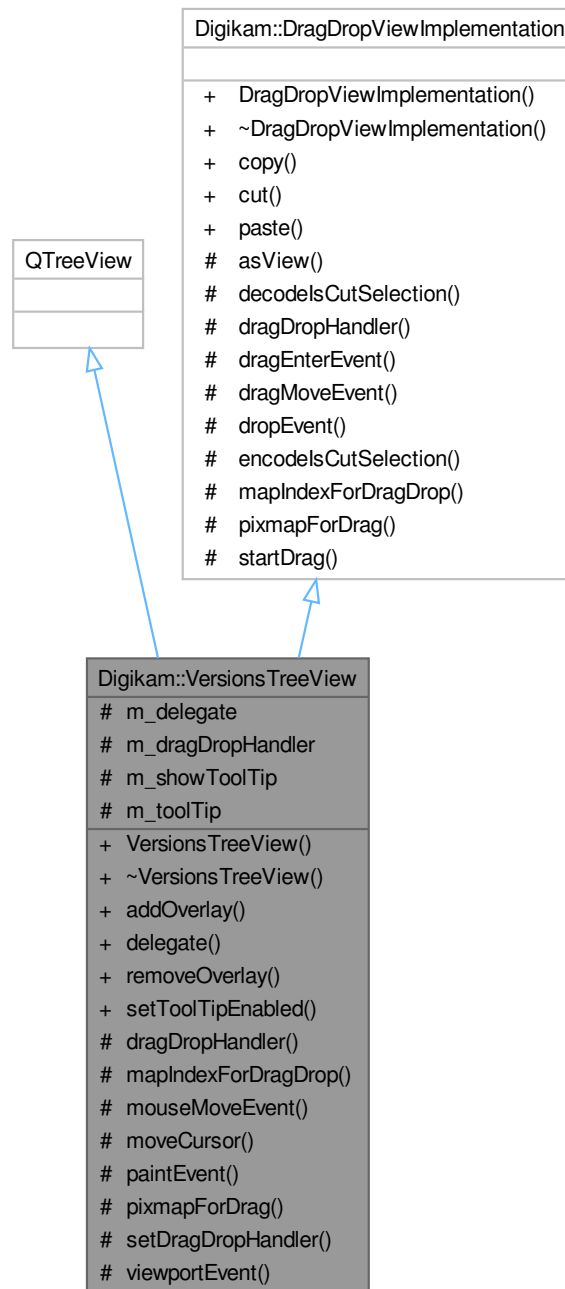
```
void Digikam::VersionsDelegate::requestNotification (
    const QModelIndex & index,
    const QString & message ) [signal]
```

Note

for [ItemDelegateOverlayContainer](#), unimplemented:

## 9.1392 Digikam::VersionsTreeView Class Reference

Inheritance diagram for Digikam::VersionsTreeView:



## Public Member Functions

- **VersionsTreeView** (QWidget \*const parent=nullptr)
- `~VersionsTreeView` () override
- void **addOverlay** (ItemDelegateOverlay \*overlay)
- **VersionsDelegate** \* **delegate** () const
- void **removeOverlay** (ItemDelegateOverlay \*overlay)
- void **setToolTipEnabled** (bool on)

## Public Member Functions inherited from [Digikam::DragDropViewImplementation](#)

- virtual void **copy** ()
- virtual void **cut** ()
- virtual void **paste** ()

## Protected Member Functions

- [AbstractItemDragDropHandler](#) \* **dragDropHandler** () const override  
*You need to implement these three methods Returns the drag drop handler.*
- QModelIndex **mapIndexForDragDrop** (const QModelIndex &index) const override  
*Maps the given index of the view's model to an index of the handler's model, which can be a source model of the view's model.*
- void **mouseMoveEvent** (QMouseEvent \*event) override
- QModelIndex **moveCursor** (CursorAction cursorAction, Qt::KeyboardModifiers modifiers) override
- void **paintEvent** (QPaintEvent \*e) override
- QPixmap **pixmapForDrag** (const QList< QModelIndex > &indexes) const override  
*Creates a pixmap for dragging the given indexes.*
- virtual void **setDragDropHandler** ([AbstractItemDragDropHandler](#) \*handler)
- bool **viewportEvent** (QEvent \*event) override

## Protected Member Functions inherited from [Digikam::DragDropViewImplementation](#)

- virtual QAbstractItemView \* **asView** ()=0  
*This one is implemented by DECLARE\_VIEW\_DRAG\_DROP\_METHODS.*
- bool **decodelsCutSelection** (const QMimeData \*mimeData)
- void **dragEnterEvent** (QDragEnterEvent \*event)  
*Implements the relevant QAbstractItemView methods for drag and drop.*
- void **dragMoveEvent** (QDragMoveEvent \*e)
- void **dropEvent** (QDropEvent \*e)
- void **encodelsCutSelection** (QMimeData \*mime, bool isCutSelection)
- void **startDrag** (Qt::DropActions supportedActions)

## Protected Attributes

- [VersionsDelegate](#) \* **m\_delegate** = nullptr
- [AbstractItemDragDropHandler](#) \* **m\_dragDropHandler** = nullptr
- bool **m\_showToolTip** = false
- ToolTip \* **m\_toolTip** = nullptr



## 9.1392.1 Constructor & Destructor Documentation

### 9.1392.1.1 ~VersionsTreeView()

```
Digikam::VersionsTreeView::~VersionsTreeView ( ) [override]
```

#### Note

All overlay management code in a sophisticated form can be studied in [ItemCategorizedView](#)

## 9.1392.2 Member Function Documentation

### 9.1392.2.1 dragDropHandler()

```
AbstractItemDragDropHandler * Digikam::VersionsTreeView::dragDropHandler ( ) const [override],  
[protected], [virtual]
```

Implements [Digikam::DragDropViewImplementation](#).

### 9.1392.2.2 mapIndexForDragDrop()

```
QModelIndex Digikam::VersionsTreeView::mapIndexForDragDrop (  
    const QModelIndex & index ) const [override], [protected], [virtual]
```

Implements [Digikam::DragDropViewImplementation](#).

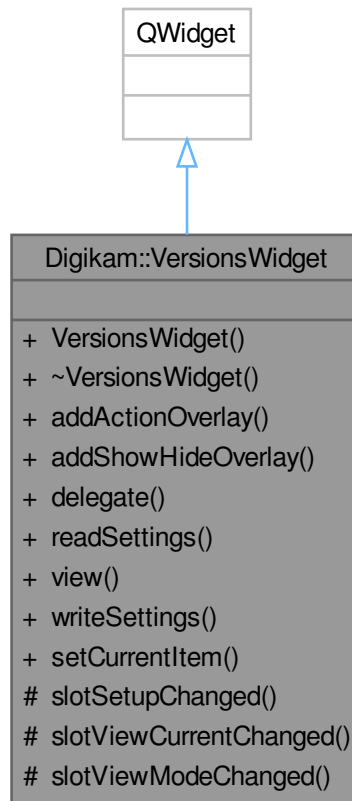
### 9.1392.2.3 pixmapForDrag()

```
QPixmap Digikam::VersionsTreeView::pixmapForDrag (  
    const QList< QModelIndex > & indexes ) const [override], [protected], [virtual]
```

Implements [Digikam::DragDropViewImplementation](#).

## 9.1393 Digikam::VersionsWidget Class Reference

Inheritance diagram for Digikam::VersionsWidget:



### Public Slots

- void **setCurrentItem** (const [ItemInfo](#) &info)

### Signals

- void **imageSelected** (const [ItemInfo](#) &info)

### Public Member Functions

- **VersionsWidget** (`QWidget *const parent=nullptr`)
- [ActionVersionsOverlay](#) \* **addActionOverlay** (const `QIcon` &icon, const `QString` &text, const `QString` &tip=`QString()`)
- [ShowHideVersionsOverlay](#) \* **addShowHideOverlay** ()
- [VersionsDelegate](#) \* **delegate** () const
- void **readSettings** (const `KConfigGroup` &group)
- [VersionsTreeView](#) \* **view** () const
- void **writeSettings** (`KConfigGroup` &group)

### Protected Slots

- void **slotSetupChanged** ()
- void **slotViewCurrentChanged** (const QModelIndex &current, const QModelIndex &previous)
- void **slotViewModeChanged** (int mode)

## 9.1394 Digikam::VideoFrame Class Reference

### Public Member Functions

- **VideoFrame** (int width, int height, int lineSize)

### Public Attributes

- QVector< quint8 > **frameData**
- quint32 **height** = 0
- quint32 **lineSize** = 0
- quint32 **width** = 0

## 9.1395 Digikam::VideoInfoContainer Class Reference

### Public Member Functions

- bool **isEmpty** () const
- bool **isNull** () const
- bool **operator==** (const [VideoInfoContainer](#) &t) const

### Public Attributes

- QString **aspectRatio**
- QString **audioBitRate**
- QString **audioChannelType**
- QString **audioCodec**
- QString **duration**
- QString **frameRate**
- QString **videoCodec**

## 9.1396 Digikam::VideoMetadataContainer Class Reference

### Public Attributes

- bool **allFieldsNull** = true
- QString **aspectRatio**
- QString **audioBitRate**
- QString **audioChannelType**
- QString **audioCodec**
- QString **duration**
- QString **frameRate**
- QString **videoCodec**

## 9.1397 Digikam::VideoStripFilter Class Reference

### Public Member Functions

- void **process** ([VideoFrame](#) &videoFrame)

## 9.1398 Digikam::VideoThumbDecoder Class Reference

### Public Member Functions

- **VideoThumbDecoder** (const QString &filename)
- bool **decodeVideoFrame** () const
- void **destroy** ()
- QString **getCodec** () const
- int **getDuration** () const
- int **getHeight** () const
- bool **getInitialized** () const
- void **getScaledVideoFrame** (int scaledSize, bool maintainAspectRatio, [VideoFrame](#) &videoFrame)
- int **getWidth** () const
- void **initialize** (const QString &filename)
- void **seek** (int timeInSeconds)

## 9.1399 Digikam::VideoThumbnailer Class Reference

### Public Member Functions

- **VideoThumbnailer** (int thumbnailSize, bool workaroundIssues, bool maintainAspectRatio, bool smart←  
FrameSelection)
- void **addFilter** ([VideoStripFilter](#) \*const filter)
- void **clearFilters** ()
- void **generateThumbnail** (const QString &videoFile, QImage &image)
- void **removeFilter** (const [VideoStripFilter](#) \*const filter)
- void **setMaintainAspectRatio** (bool enabled)
- void **setSeekPercentage** (int percentage)
- void **setSeekTime** (const QString &seekTime)
- void **setSmartFrameSelection** (bool enabled)
- void **setThumbnailSize** (int size)
- void **setWorkAroundIssues** (bool workAround)

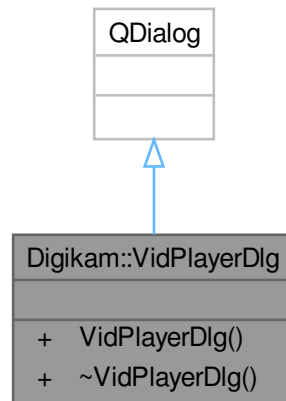
## 9.1400 Digikam::VideoThumbWriter Class Reference

### Public Member Functions

- void **writeFrame** ([VideoFrame](#) &frame, QImage &image)

## 9.1401 Digikam::VidPlayerDlg Class Reference

Inheritance diagram for Digikam::VidPlayerDlg:



### Public Member Functions

- **VidPlayerDlg** (const QString &file, QWidget \*const parent=nullptr)

## 9.1402 Digikam::VidSlideSettings Class Reference

### Public Types

- enum `Selection` { **IMAGES** = 0 , **ALBUMS** }  
*Images selection mode.*
- enum `VidBitRate` {  
`VBR04` = 0 , `VBR05` , `VBR10` , `VBR12` ,  
`VBR15` , `VBR20` , `VBR25` , `VBR30` ,  
`VBR40` , `VBR45` , `VBR50` , `VBR60` ,  
`VBR80` }  
*Video rates in bits per seconds.*
- enum `VidCodec` {  
`X264` = 0 , `MPEG4` , `MPEG2` , `MJPEG` ,  
`FLASH` , `WEBMVP8` , `THEORA` , `WMV7` ,  
`WMV8` , `WMV9` }  
*Video Codecs.*
- enum `VidFormat` { `AVI` = 0 , `MKV` , `MP4` , `MPG` }  
*Video Container Formats.*
- enum `VidPlayer` { **NOPLAYER** = 0 , **INTERNAL** , **DESKTOP** }  
*Video player to use.*
- enum `VidStd` { `PAL` = 0 , `NTSC` }  
*Video Standards.*

- enum `VidType` {  
`QVGA = 0`, `VCD1`, `VCD2`, `CVD1`,  
`CVD2`, `HVGA`, `SVCD1`, `SDTV1`,  
`SDTV2`, `EDTV1`, `SVCD2`, `EGA`,  
`VGA`, `SDTV3`, `EDTV2`, `DVD1`,  
`DVD2`, `WVGA`, `SVGA`, `DVGA`,  
`XVGA`, `HDTV`, `WXGA1`, `WXGA2`,  
`SXGA`, `SXGAPLUS`, `WSXGA`, `HDPLUS`,  
`UXGA`, `WSXGAPLUS`, `BLUERAY`, `WUXGA`,  
`TXGA`, `QXGA`, `UWFHD`, `WQHD`,  
`WQXGA`, `QSXGA`, `QSXGAPLUS`, `WQXGAPLUS`,  
`WQSXGA`, `QUXGA`, `UHD4K`, `WQUXGA`,  
`HXGA`, `UHD5K`, `WHXGA`, `HSXGA`,  
`UHD6K`, `WHSXGA`, `HUXGA`, `UHD8K`,  
`WHUXGA`, `UW10K`, `UW16K` }

*Video types (size of target screen) See [https://en.wikipedia.org/wiki/List\\_of\\_common\\_resolutions#Digital\\_TV\\_standards](https://en.wikipedia.org/wiki/List_of_common_resolutions#Digital_TV_standards) [https://en.wikipedia.org/wiki/Aspect\\_ratio\\_\(image\)](https://en.wikipedia.org/wiki/Aspect_ratio_(image))*

### Public Member Functions

- `QStringList defaultFFmpegSearchPaths ()` const
- void `readSettings` (const `KConfigGroup &group`)  
*Read and write settings in config file between sessions.*
- int `videoBitRate ()` const  
*Return the current video bit rate.*
- `QString videoCodec ()` const  
*Return the current video ffmpeg codec name.*
- `QString videoFormat ()` const  
*Return the current video format extension.*
- `qreal videoFrameRate ()` const  
*Return the current video frame rate.*
- `QSize videoSize ()` const  
*Return the current video size.*
- void `writeSettings` (`KConfigGroup &group`)

### Static Public Member Functions

- static bool `isVideoTVFormat (VidType type)`  
*Return true if type is a video TV format. If false is returned type is computer graphics screen format.*
- static `QMap< VidBitRate, QString > videoBitRateNames ()`
- static `QMap< VidCodec, QString > videoCodecNames ()`
- static `QMap< VidFormat, QString > videoFormatNames ()`
- static `QMap< VidPlayer, QString > videoPlayerNames ()`
- static `QSize videoSizeFromType (VidType type)`  
*Return the current size from a type of video.*
- static `QMap< VidStd, QString > videoStdNames ()`
- static `QMap< VidType, QString > videoTypeNames ()`  
*Helper methods to fill combobox from GUI.*

## Public Attributes

- int **abitRate** = 64000  
*Encoded Audio stream bit rate in bit/s.*
- QString **audioTrack**  
*Soundtrack stream.*
- FileSaveConflictBox::ConflictRule **conflictRule** = FileSaveConflictBox::OVERWRITE  
*Rule to follow if video file already exists.*
- bool **equalize** = false  
*Equalize filter to applying while encoding video from frames.*
- QMap< QString, QString > **ffmpegCodecs**  
*Map of FFmpeg codec names and features.*
- QMap< QString, QString > **ffmpegFormats**  
*Map of FFmpeg format names and features.*
- QString **ffmpegPath**  
*Path to FFmpeg binary.*
- QString **filesList**  
*Path to list of frame files to encode.*
- [DInfoInterface](#) \* **iface** = nullptr  
*Plugin host interface to handle item properties.*
- int **imgFrames** = 125  
*Amount of frames by image to encode in video (ex: 125 frames = 5 s at 25 img/s).*
- QList< QUrl > **inputImages**  
*Images stream.*
- [FrameOsdSettings](#) **osdSettings**  
*On Screen Display parameters.*
- QString **outputDir** = QStandardPaths::writableLocation(QStandardPaths::MoviesLocation)  
*Encoded video stream directory.*
- QString **outputFile**  
*Path to encoded video.*
- [VidPlayer](#) **outputPlayer** = INTERNAL  
*Open video stream in player at end.*
- QString **outputVideo**  
*Target video file encoded at end.*
- [Selection](#) **selMode** = IMAGES  
*Items selection mode.*
- QTime **soundtrackLength**  
*Duration of the soundtrack.*
- int **strength** = 5  
*Equalization strength factor.*
- QString **tempDir**  
*To store temporary frames.*
- TransitionMngr::TransType **transition** = TransitionMngr::None  
*Transition type between images.*
- [VidBitRate](#) **vbitRate** = VBR12  
*Encoded Video stream bit rate in bit/s.*
- [VidCodec](#) **vCodec** = X264  
*Encoded video codec.*
- [EffectMngr::EffectType](#) **vEffect** = [EffectMngr::None](#)  
*Encoded video effect while displaying images.*
- [VidFormat](#) **vFormat** = MP4

*Encoded video container format.*

- **VidStd vStandard** = PAL

*Encoded Video standard => frame rate in img/s.*

- **VidType vType** = BLUERAY

*Encoded video type.*

## 9.1402.1 Member Enumeration Documentation

### 9.1402.1.1 VidBitRate

enum `Digikam::VidSlideSettings::VidBitRate`

Enumerator

VBR04	400000
VBR05	500000
VBR10	1000000
VBR12	1200000
VBR15	1500000
VBR20	2000000
VBR25	2500000
VBR30	3000000
VBR40	4000000
VBR45	4500000
VBR50	5000000
VBR60	6000000
VBR80	8000000

### 9.1402.1.2 VidCodec

enum `Digikam::VidSlideSettings::VidCodec`

Enumerator

X264	<a href="https://en.wikipedia.org/wiki/X264">https://en.wikipedia.org/wiki/X264</a>
MPEG4	<a href="https://en.wikipedia.org/wiki/MPEG-4">https://en.wikipedia.org/wiki/MPEG-4</a>
MPEG2	<a href="https://en.wikipedia.org/wiki/MPEG-2">https://en.wikipedia.org/wiki/MPEG-2</a>
MJPEG	<a href="https://en.wikipedia.org/wiki/Motion_JPEG">https://en.wikipedia.org/wiki/Motion_JPEG</a>
FLASH	<a href="https://en.wikipedia.org/wiki/Adobe_Flash">https://en.wikipedia.org/wiki/Adobe_Flash</a>
WEBMVP8	<a href="https://en.wikipedia.org/wiki/VP8">https://en.wikipedia.org/wiki/VP8</a>
THEORA	<a href="https://en.wikipedia.org/wiki/Theora">https://en.wikipedia.org/wiki/Theora</a>
WMV7	<a href="https://en.wikipedia.org/wiki/Windows_Media_Video">https://en.wikipedia.org/wiki/Windows_Media_Video</a>
WMV8	<a href="https://en.wikipedia.org/wiki/Windows_Media_Video">https://en.wikipedia.org/wiki/Windows_Media_Video</a>
WMV9	<a href="https://en.wikipedia.org/wiki/Windows_Media_Video">https://en.wikipedia.org/wiki/Windows_Media_Video</a>



### 9.1402.1.3 VidFormat

enum `Digikam::VidSlideSettings::VidFormat`

#### Enumerator

AVI	<a href="https://en.wikipedia.org/wiki/Audio_Video_Interleave">https://en.wikipedia.org/wiki/Audio_Video_Interleave</a>
MKV	<a href="https://en.wikipedia.org/wiki/Matroska">https://en.wikipedia.org/wiki/Matroska</a>
MP4	<a href="https://en.wikipedia.org/wiki/MPEG-4_Part_14">https://en.wikipedia.org/wiki/MPEG-4_Part_14</a>
MPG	<a href="https://en.wikipedia.org/wiki/MPEG-2">https://en.wikipedia.org/wiki/MPEG-2</a>

### 9.1402.1.4 VidStd

enum `Digikam::VidSlideSettings::VidStd`

#### Enumerator

PAL	25 FPS
NTSC	29.97 FPS

### 9.1402.1.5 VidType

enum `Digikam::VidSlideSettings::VidType`

#### Enumerator

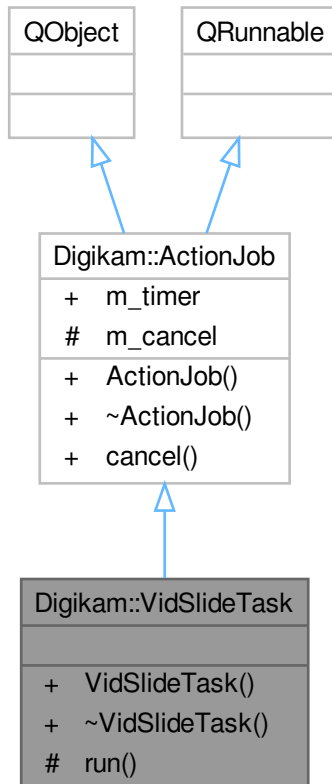
QVGA	320 x 180 - 16:9 - Computer Graphics
VCD1	352 x 240 - 7:5 - Digital TV
VCD2	352 x 288 - 6:5 - Digital TV
CVD1	352 x 480 - 11:15 - Digital TV
CVD2	352 x 576 - 11:18 - Digital TV
HVGA	480 x 270 - 16:9 - Computer Graphics
SVCD1	480 x 480 - 1:1 - Digital TV
SDTV1	528 x 480 - 11:10 - Digital TV
SDTV2	544 x 480 - 17:15 - Digital TV
EDTV1	544 x 576 - 17:18 - Digital TV
SVCD2	480 x 576 - 5:6 - Digital TV
EGA	640 x 350 - 16:9 - Computer Graphics
VGA	640 x 480 - 4:3 - Computer Graphics
SDTV3	704 x 480 - 22:15 - Digital TV
EDTV2	704 x 576 - 11:9 - Digital TV
DVD1	720 x 480 - 3:2 - Digital TV
DVD2	720 x 576 - 5:4 - Digital TV
WVGA	800 x 450 - 16:9 - Computer Graphics
SVGA	800 x 600 - 4:3 - Computer Graphics
DVGA	960 x 640 - 3:2 - Computer Graphics
XVGA	1024 x 576 - 16:9 - Computer Graphics

## Enumerator

HDTV	1280 x 720 - 16:9 - Digital TV
WXGA1	1280 x 768 - 5:3 - Computer Graphics
WXGA2	1280 x 800 - 8:5 - Computer Graphics
SXGA	1280 x 1024 - 5:4 - Computer Graphics
SXGAPLUS	1400 x 1050 - 4:3 - Computer Graphics
WSXGA	1440 x 900 - 8:5 - Computer Graphics
HDPLUS	1600 x 900 - 16:9 - Digital TV
UXGA	1600 x 1200 - 4:3 - Computer Graphics
WSXGAPLUS	1680 x 1050 - 8:5 - Computer Graphics
BLUERAY	1920 x 1080 - 19:9 - Digital TV
WUXGA	1920 x 1200 - 8:5 - Computer Graphics
TXGA	1920 x 1440 - 7:5 - Computer Graphics
QXGA	2048 x 1536 - 4:3 - Computer Graphics
UWFHD	2560 < 1080 - 21:9 - Computer Graphics
WQHD	2560 x 1440 - 16:9 - Computer Graphics
WQXGA	2560 x 1600 - 8:5 - Computer Graphics
QSXGA	2560 x 2048 - 5:4 - Computer Graphics
QSXGAPLUS	2800 x 2100 - 4:3 - Computer Graphics
WQXGAPLUS	3200 x 1800 - 16:9 - Computer Graphics
WQSXGA	3200 x 2048 - 25:16 - Computer Graphics
QUXGA	3200 x 2400 - 4:3 - Computer Graphics
UHD4K	3840 x 2160 - 19:9 - Digital TV
WQUXGA	3840 x 2400 - 8:5 - Computer Graphics
HXGA	4096 x 3072 - 4:3 - Computer Graphics
UHD5K	5120 x 2880 - 16:9 - Computer Graphics
WHXGA	5120 x 3200 - 8:5 - Computer Graphics
HSXGA	5120 x 4096 - 5:4 - Computer Graphics
UHD6K	6016 x 3384 - 16:9 - Computer Graphics
WHSXGA	6400 x 4096 - 25:16 - Computer Graphics
HUXGA	6400 x 4800 - 4:3 - Computer Graphics
UHD8K	7680 x 4320 - 16:9 - Digital TV
WHUXGA	7680 x 4800 - 8:5 - Computer Graphics
UW10K	10240 x 4320 - 21:9 - Computer Graphics
UW16K	15360 x 8640 - 16:9 - Computer Graphics

## 9.1403 Digikam::VidSlideTask Class Reference

Inheritance diagram for Digikam::VidSlideTask:



### Signals

- void **signalDone** (bool)
- void **signalMessage** (const QString &, bool)

### Signals inherited from [Digikam::ActionJob](#)

- void **signalDone** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job is done.*
- void **signalProgress** (int)  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager the job progress.*
- void **signalStarted** ()  
*Use this signal in your implementation to inform [ActionThreadBase](#) manager that job is started.*

### Public Member Functions

- **VidSlideTask** ([VidSlideSettings](#) \*const settings)

## Public Member Functions inherited from [Digikam::ActionJob](#)

- **ActionJob** (QObject \*const parent=nullptr)  
*Constructor which delegate deletion of QRunnable instance to [ActionThreadBase](#), not QThreadPool.*
- [~ActionJob](#) () override  
*Re-implement destructor in you implementation.*

## Protected Member Functions

- void **run** () override

## Additional Inherited Members

## Public Slots inherited from [Digikam::ActionJob](#)

- void **cancel** ()  
*Call this method to cancel job.*

## Public Attributes inherited from [Digikam::ActionJob](#)

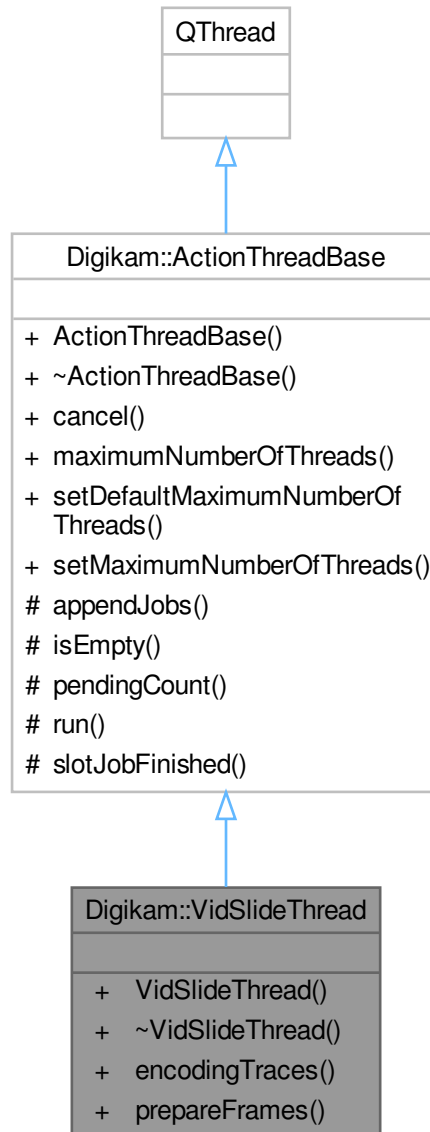
- QElapsedTimer **m\_timer**  
*Timer to determine the running time of the job.*

## Protected Attributes inherited from [Digikam::ActionJob](#)

- bool **m\_cancel** = false  
*You can use this boolean in your implementation to know if job must be canceled.*

## 9.1404 Digikam::VidSlideThread Class Reference

Inheritance diagram for Digikam::VidSlideThread:



### Signals

- void **signalDone** (bool)
- void **signalMessage** (const QString &, bool)
- void **signalProgress** (int)

## Public Member Functions

- **VidSlideThread** (QObject \*const parent)
- QString **encodingTraces** () const
- void **prepareFrames** (VidSlideSettings \*const settings)

*Stage 1: prepare frames in temporary directory.*

## Public Member Functions inherited from [Digikam::ActionThreadBase](#)

- **ActionThreadBase** (QObject \*const parent=nullptr)
- void **cancel** (bool isCancel=true)  
*Cancel processing of current jobs under progress.*
- int **maximumNumberOfThreads** () const  
*Return the maximum number of threads used to parallelize collection of job processing.*
- void **setDefaultMaximumNumberOfThreads** ()  
*Reset maximum number of threads used to parallelize collection of job processing to max core detected on computer.*
- void **setMaximumNumberOfThreads** (int n)  
*Adjust maximum number of threads used to parallelize collection of job processing.*

## Additional Inherited Members

## Protected Slots inherited from [Digikam::ActionThreadBase](#)

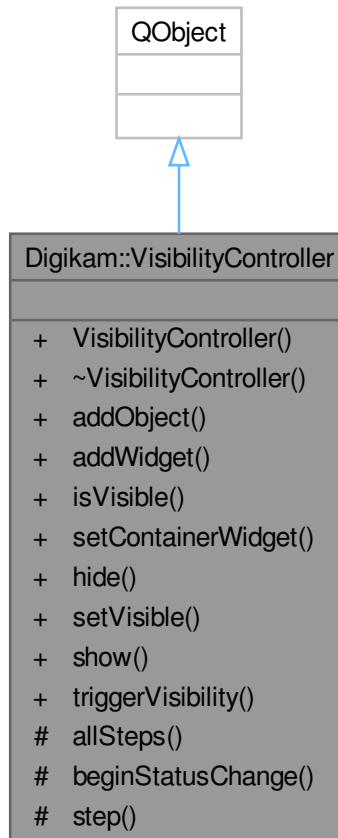
- void **slotJobFinished** ()

## Protected Member Functions inherited from [Digikam::ActionThreadBase](#)

- void **appendJobs** (const [ActionJobCollection](#) &jobs)  
*Append a collection of jobs to process into QThreadPool.*
- bool **isEmpty** () const  
*Return true if list of pending jobs to process is empty.*
- int **pendingCount** () const  
*Return the number of pending jobs to process.*
- void **run** () override  
*Main thread loop used to process jobs in todo list.*

## 9.1405 Digikam::VisibilityController Class Reference

Inheritance diagram for Digikam::VisibilityController:



### Public Types

- enum **Status** {  
    **Unknown** , **Hidden** , **Showing** , **Shown** ,  
    **Hiding** }

### Public Slots

- void **hide** ()
- void **setVisible** (bool visible)  
    *Shows/hides all added objects.*
- void **show** ()
- void **triggerVisibility** ()  
    *Shows if hidden and hides if visible.*

### Public Member Functions

- **VisibilityController** (QObject \*const parent)
- void **addObject** (VisibilityObject \*const object)
 

*Add an object implementing the VisibilityObject interface.*
- void **addWidget** (QWidget \*const widget)
 

*Add a widget to this controller.*
- bool **isVisible** () const
 

*Returns true if the contained objects are visible or becoming visible.*
- void **setContainerWidget** (QWidget \*const widget)
 

*Set the widget containing the widgets added to this controller.*

### Protected Member Functions

- void **allSteps** ()
- virtual void **beginStatusChange** ()
- void **step** ()

## 9.1405.1 Member Function Documentation

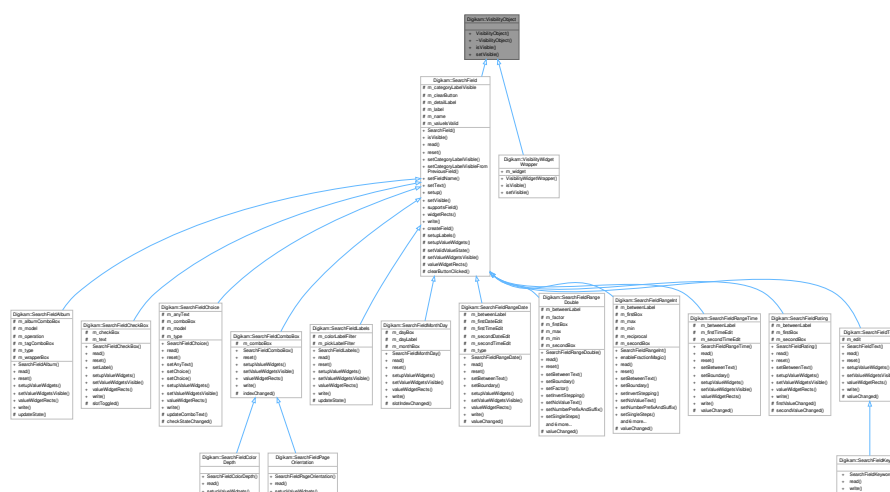
### 9.1405.1.1 addObject()

```
void Digikam::VisibilityController::addObject (
    VisibilityObject *const object )
```

You can use this if you have your widgets grouped in intermediate objects.

## 9.1406 Digikam::VisibilityObject Class Reference

Inheritance diagram for Digikam::VisibilityObject:





### Public Member Functions

- virtual bool **isVisible** ()=0
- virtual void **setVisible** (bool visible)=0

## 9.1407 Digikam::WBContainer Class Reference

### Public Member Functions

- bool **isDefault** () const
- bool **operator==** (const [WBContainer](#) &other) const
- void **writeToFilterAction** ([FilterAction](#) &action, const QString &prefix=QString()) const

### Static Public Member Functions

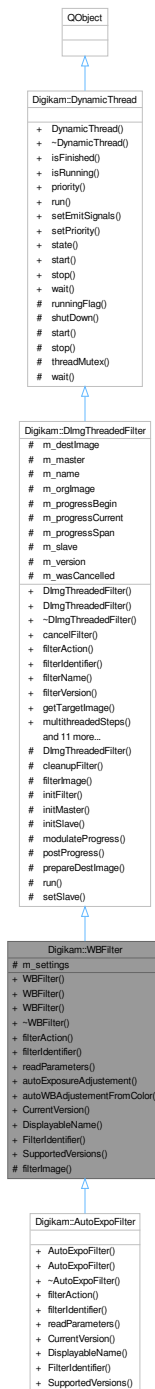
- static [WBContainer](#) **fromFilterAction** (const [FilterAction](#) &action, const QString &prefix=QString())

### Public Attributes

- double **black** = 0.0  
*Neutral color temperature settings.*
- double **dark** = 0.0
- double **expositionFine** = 0.0
- double **expositionMain** = 0.0
- double **gamma** = 1.0
- double **green** = 1.0
- double **saturation** = 1.0
- double **temperature** = 6500.0

## 9.1408 Digikam::WBFilter Class Reference

Inheritance diagram for Digikam::WBFilter:



### Public Member Functions

- **WBFilter** (const [WBContainer](#) &settings, [DimgThreadedFilter](#) \*const master, const [Dimg](#) &orgImage, const [Dimg](#) &destImage, int progressBegin=0, int progressEnd=100)

- **WBFilter** (*DImg* \*const orgImage, *QObject* \*const parent=nullptr, const [WBContainer](#) &settings=[WBContainer](#)())
- **WBFilter** (*QObject* \*const parent=nullptr)
- [FilterAction](#) **filterAction** () override  
*Returns the action description corresponding to currently set options.*
- *QString* **filterIdentifier** () const override  
*Return the identifier for this filter in the image history.*
- void [readParameters](#) (const [FilterAction](#) &action) override

## Public Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) (*DImg* \*const orgImage, *QObject* \*const parent, const *QString* &name=*QString*())  
*Constructs a filter with all arguments (ready to use).*
- [DImgThreadedFilter](#) (*QObject* \*const parent=nullptr, const *QString* &name=*QString*())  
*Constructs a filter without argument.*
- virtual void **cancelFilter** ()  
*Cancel the threaded computation.*
- const *QString* & **filterName** ()
- int **filterVersion** () const
- [DImg](#) **getTargetImage** ()
- *QList*< int > **multithreadedSteps** (int stop, int start=0) const  
*This method return a list of steps to process parallelized operation in filter using QtConcurrents API.*
- virtual bool **parametersSuccessfullyRead** () const  
*Optional: error handling for readParameters.*
- virtual *QString* **readParametersError** (const [FilterAction](#) &actionThatFailed) const
- void **setFilterName** (const *QString* &name)
- void **setFilterVersion** (int version)  
*Replaying a filter action: Set the filter version.*
- void **setOriginalImage** (const [DImg](#) &orgImage)
- void **setupAndStartDirectly** (const [DImg](#) &orgImage, [DImgThreadedFilter](#) \*const master, int progress←Begin=0, int progressEnd=100)  
*Initializes the filter for use as a slave and directly starts computation (in-thread)*
- void **setupFilter** (const [DImg](#) &orgImage)  
*You need to call this and then start filter of you used the constructor not setting an original image.*
- virtual void **startFilter** ()  
*Start the threaded computation.*
- virtual void **startFilterDirectly** ()  
*Start computation of this filter, directly in this thread.*
- virtual *QList*< int > **supportedVersions** () const

## Public Member Functions inherited from [Digikam::DynamicThread](#)

- [DynamicThread](#) (*QObject* \*const parent=nullptr)  
*This class extends [QRunnable](#), so you have to reimplement virtual void [run\(\)](#).*
- ~[DynamicThread](#) () override  
*The destructor calls [stop\(\)](#) and [wait\(\)](#), but if you, in your destructor, delete any data that is accessed by your [run\(\)](#) method, you must call [stop\(\)](#) and [wait\(\)](#) before yourself.*
- bool **isFinished** () const
- bool **isRunning** () const
- *QThread*::Priority **priority** () const
- void **setEmitSignals** (bool emitThem)
- void **setPriority** (*QThread*::Priority priority)  
*Sets the priority for this dynamic thread.*
- State **state** () const

### Static Public Member Functions

- static void **autoExposureAdjustement** (const [DImg](#) \*const img, double &black, double &expo)
- static void **autoWBAdjustementFromColor** (const QColor &tc, double &temperature, double &green)
- static int **CurrentVersion** ()
- static QString **DisplayableName** ()
- static QString **FilterIdentifier** ()
- static QList< int > **SupportedVersions** ()

### Protected Member Functions

- void **filterImage** () override  
*Main image filter method.*

### Protected Member Functions inherited from [Digikam::DImgThreadedFilter](#)

- [DImgThreadedFilter](#) ([DImgThreadedFilter](#) \*const master, const [DImg](#) &orgImage, const [DImg](#) &destImage, int progressBegin=0, int progressEnd=100, const QString &name=QString())  
*Support for chaining two filters as master and thread.*
- virtual void **cleanupFilter** ()  
*Clean up filter data if necessary, called by stopComputation() method.*
- virtual void **initFilter** ()  
*Start filter operation before threaded method.*
- void **initMaster** ()
- void **initSlave** ([DImgThreadedFilter](#) \*const master, int progressBegin=0, int progressEnd=100)  
*Initialize the filter for use as a slave - reroutes progress info to master.*
- virtual int **modulateProgress** (int progress)  
*This method modulates the progress value from the 0..100 span to the span of this slave.*
- void **postProgress** (int progress)  
*Emit progress info.*
- virtual void **prepareDestImage** ()
- void **run** () override  
*List of threaded operations by filter.*
- void **setSlave** ([DImgThreadedFilter](#) \*const slave)  
*Inform the master that there is currently a slave.*

### Protected Member Functions inherited from [Digikam::DynamicThread](#)

- bool **runningFlag** () const volatile  
*In you run() method, you shall regularly check for runningFlag() and cleanup and return if false.*
- void **shutDown** ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void **start** (QMutexLocker< QMutex > &locker)  
*Doing the same as start(), stop() and wait above, provide it with a locked QMutexLocker on mutex().*
- void **stop** (const QMutexLocker< QMutex > &locker)
- QMutex \* **threadMutex** () const  
*This is the non-recursive mutex used to protect state variables and waiting in this class.*
- void **wait** (QMutexLocker< QMutex > &locker)

## Protected Attributes

- [WBContainer](#) **m\_settings**

## Protected Attributes inherited from [Digikam::DImgThreadedFilter](#)

- [DImg](#) **m\_destImage**  
*Output image data.*
- [DImgThreadedFilter](#) \* **m\_master** = nullptr  
*The master of this slave filter.*
- [QString](#) **m\_name**  
*Filter name.*
- [DImg](#) **m\_orgImage**  
*Copy of original Image data.*
- int **m\_progressBegin** = 0  
*The progress span that a slave filter uses in the parent filter's progress.*
- int **m\_progressCurrent** = 0  
*To prevent signals bombarding with progress indicator value in [postProgress\(\)](#).*
- int **m\_progressSpan** = 0
- [DImgThreadedFilter](#) \* **m\_slave** = nullptr  
*The current slave.*
- int **m\_version** = 1
- bool **m\_wasCancelled** = false

## Additional Inherited Members

## Public Types inherited from [Digikam::DynamicThread](#)

- enum **State** { **Inactive** , **Scheduled** , **Running** , **Deactivating** }

## Public Slots inherited from [Digikam::DynamicThread](#)

- void **start** ()
- void **stop** ()  
*Stop computation, sets the running flag to false.*
- void **wait** ()  
*Waits until the thread finishes.*

## Signals inherited from [Digikam::DImgThreadedFilter](#)

- void **finished** (bool success)  
*Emitted when the computation has completed.*
- void **progress** (int progress)  
*Emitted when progress info from the calculation is available.*
- void **started** ()  
*This signal is emitted when image data is available and the computation has started.*

## Signals inherited from [Digikam::DynamicThread](#)

- void **finished** ()
- void **starting** ()

*Emitted if emitSignals is enabled.*

### 9.1408.1 Member Function Documentation

#### 9.1408.1.1 autoWBAdjustementFromColor()

```
void Digikam::WBFilter::autoWBAdjustementFromColor (
    const QColor & tc,
    double & temperature,
    double & green ) [static]
```

This is a dichotomic search based on Blue and Red layers ratio to find the matching temperature adapted from ufraw (0.12.1) RGB\_to\_Temperature

#### 9.1408.1.2 filterAction()

```
FilterAction Digikam::WBFilter::filterAction ( ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

#### 9.1408.1.3 filterIdentifier()

```
QString Digikam::WBFilter::filterIdentifier ( ) const [inline], [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

#### 9.1408.1.4 filterImage()

```
void Digikam::WBFilter::filterImage( ) [override], [protected], [virtual]
```

Override in subclass.

Implements [Digikam::DImgThreadedFilter](#).

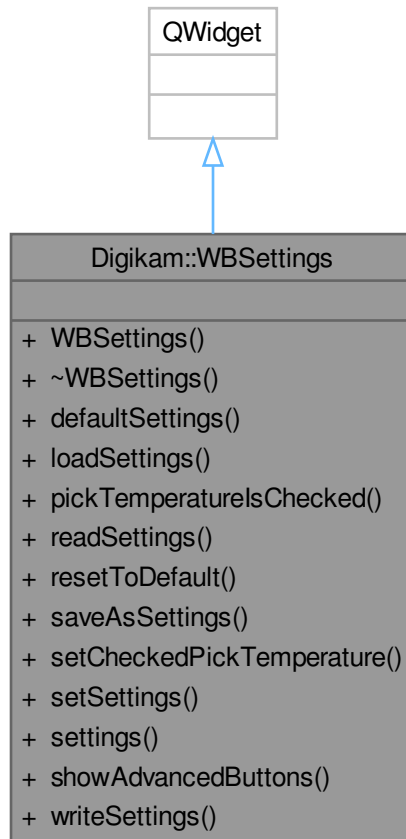
#### 9.1408.1.5 readParameters()

```
void Digikam::WBFilter::readParameters (
    const FilterAction & action ) [override], [virtual]
```

Implements [Digikam::DImgThreadedFilter](#).

## 9.1409 Digikam::WBSettings Class Reference

Inheritance diagram for Digikam::WBSettings:



### Signals

- void `signalAutoAdjustExposure ()`
- void `signalPickerColorButtonActivated ()`
- void `signalSettingsChanged ()`

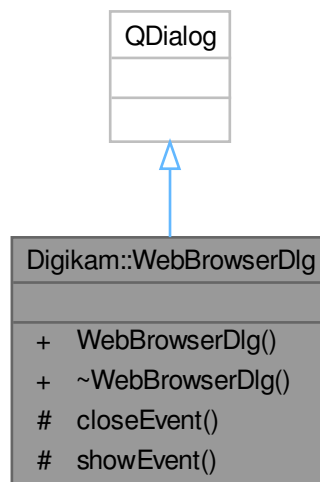
### Public Member Functions

- **`WBSettings`** (`QWidget *const parent`)
- `WBContainer` **`defaultSettings`** () const
- void **`loadSettings`** ()
- bool **`pickTemperaturesIsChecked`** ()
- void **`readSettings`** (const `KConfigGroup &group`)
- void **`resetToDefault`** ()
- void **`saveAsSettings`** ()
- void **`setCheckedPickTemperature`** (bool `b`)

- void **setSettings** (const [WBContainer](#) &settings)
- [WBContainer](#) **settings** () const
- void **showAdvancedButtons** (bool b)
- void **writeSettings** (KConfigGroup &group)

## 9.1410 Digikam::WebBrowserDlg Class Reference

Inheritance diagram for Digikam::WebBrowserDlg:



### Signals

- void **closeView** (bool val)
- void **urlChanged** (const [QUrl](#) &url)

### Public Member Functions

- **WebBrowserDlg** (const [QUrl](#) &url, [QWidget](#) \*const parent, bool hideDeskBrowser=false)

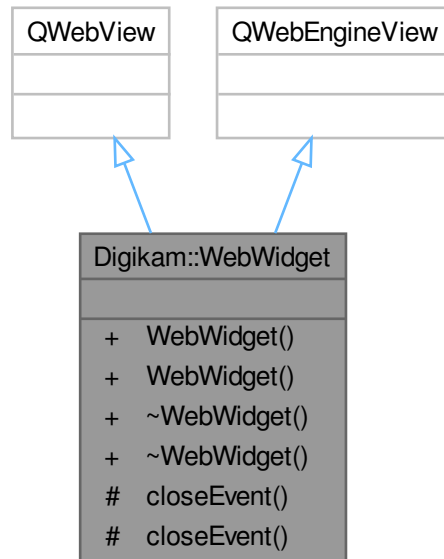
### Protected Member Functions

- void **closeEvent** ([QCloseEvent](#) \*) override
- void **showEvent** ([QShowEvent](#) \*) override



## 9.1411 Digikam::WebWidget Class Reference

Inheritance diagram for Digikam::WebWidget:



### Signals

- void **closeView** (bool val)
- void **closeView** (bool val)

### Public Member Functions

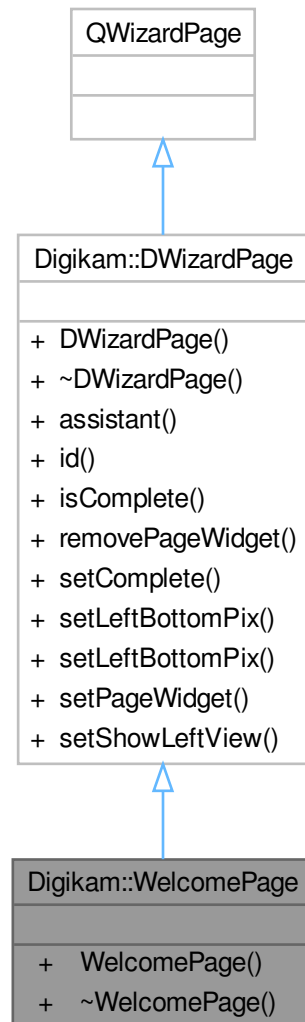
- **WebWidget** (QWidget \*const parent=nullptr)
- **WebWidget** (QWidget \*const parent=nullptr)

### Protected Member Functions

- void **closeEvent** (QCloseEvent \*event) override
- void **closeEvent** (QCloseEvent \*event) override

## 9.1412 Digikam::WelcomePage Class Reference

Inheritance diagram for Digikam::WelcomePage:



### Public Member Functions

- `WelcomePage` (`QWizard *const dlg`)

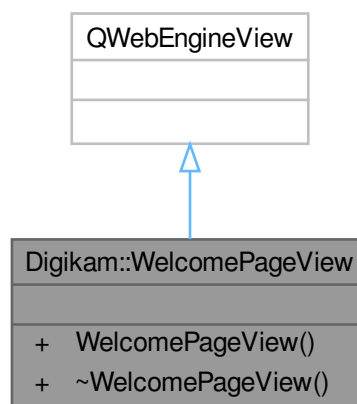
### Public Member Functions inherited from [Digikam::DWizardPage](#)

- `DWizardPage` (`QWizard *const dlg, const QString &title`)
- `QWizard * assistant () const`
- `int id () const`
- `bool isComplete () const` override

- void **removePageWidget** (QWidget \*const w)
- void **setComplete** (bool b)
- void **setLeftBottomPix** (const QIcon &icon)
- void **setLeftBottomPix** (const QPixmap &pix)
- void **setPageWidget** (QWidget \*const w)
- void **setShowLeftView** (bool v)

## 9.1413 Digikam::WelcomePageView Class Reference

Inheritance diagram for Digikam::WelcomePageView:

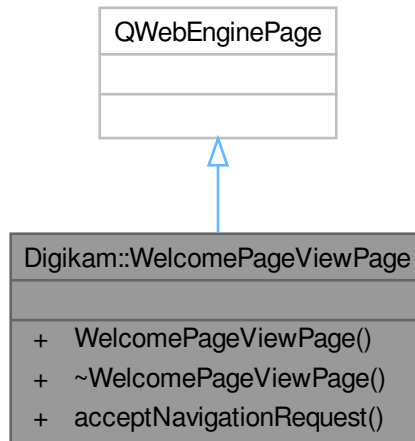


### Public Member Functions

- **WelcomePageView** (QWidget \*const parent)

## 9.1414 Digikam::WelcomePageViewPage Class Reference

Inheritance diagram for Digikam::WelcomePageViewPage:



### Signals

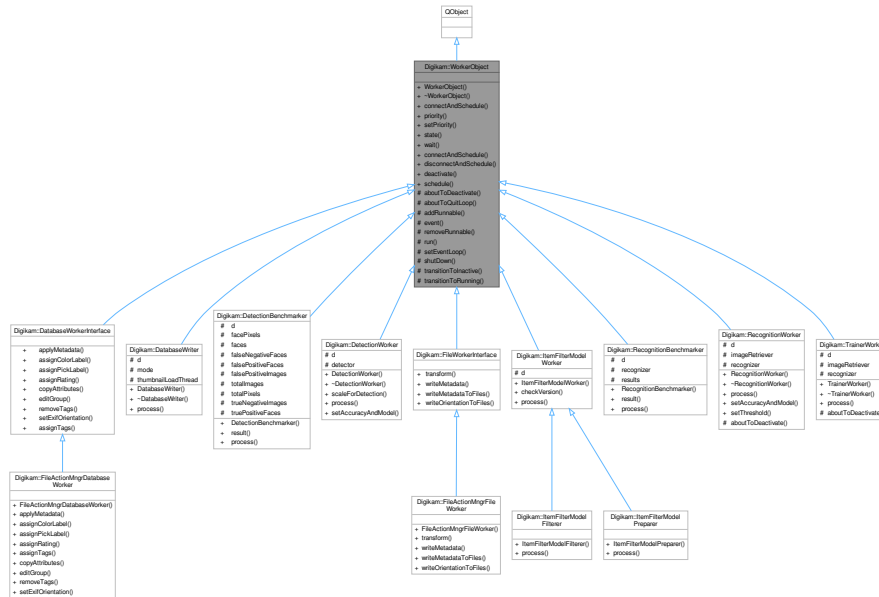
- void **linkClicked** (const QUrl &)

### Public Member Functions

- **WelcomePageViewPage** (QObject \*const parent=nullptr)
- bool **acceptNavigationRequest** (const QUrl &, QWebEnginePage::NavigationType, bool) override

## 9.1415 Digikam::WorkerObject Class Reference

Inheritance diagram for Digikam::WorkerObject:



### Public Types

- enum [DeactivatingMode](#) { [FlushSignals](#) , [KeepSignals](#) , [PhaseOut](#) }
- enum [State](#) { [Inactive](#) , [Scheduled](#) , [Running](#) , [Deactivating](#) }

### Public Slots

- void [deactivate](#) ([DeactivatingMode](#) mode=[FlushSignals](#))  
*Quits execution of this worker object.*
- void [schedule](#) ()  
*Starts execution of this worker object: The object is moved to a thread and an event loop started, so that queued signals will be received.*

### Signals

- void [finished](#) ()
- void [started](#) ()

### Public Member Functions

- [WorkerObject](#) ()  
*Deriving from a worker object allows you to execute your slots in a thread.*
- bool [connectAndSchedule](#) (const QObject \*sender, const char \*signal, const char \*method, Qt::ConnectionType type=Qt::AutoConnection) const  
*You must normally call [schedule\(\)](#) to ensure that the object is active when you send a signal with work data.*
- QThread::Priority [priority](#) () const
- void [setPriority](#) (QThread::Priority priority)  
*Sets the priority for this dynamic thread.*
- State [state](#) () const
- void [wait](#) ()

## Static Public Member Functions

- static bool **connectAndSchedule** (const QObject \*sender, const char \*signal, const [WorkerObject](#) \*receiver, const char \*method, Qt::ConnectionType type=Qt::AutoConnection)
- static bool **disconnectAndSchedule** (const QObject \*sender, const char \*signal, const [WorkerObject](#) \*receiver, const char \*method)

## Protected Member Functions

- virtual void [aboutToDeactivate](#) ()  
*Called from [deactivate\(\)](#), typically from a different thread than the worker thread, possibly the UI thread.*
- virtual void [aboutToQuitLoop](#) ()  
*Called from within thread's event loop to quit processing.*
- void **addRunnable** (WorkerObjectRunnable \*loop)
- bool **event** (QEvent \*e) override
- void **removeRunnable** (WorkerObjectRunnable \*loop)
- void **run** ()
- void **setEventLoop** (QEventLoop \*loop)
- void [shutDown](#) ()  
*If you are deleting data in your destructor which is accessed from the thread, do one of the following from your destructor to guarantee a safe shutdown: 1) Call this method 2) Call stop() and wait(), knowing that nothing will call start() anymore after this 3) Be sure the thread will never be running at destruction.*
- void **transitionToInactive** ()
- bool **transitionToRunning** ()

## Friends

- class **ThreadManager**
- class **WorkerObjectRunnable**

## 9.1415.1 Member Enumeration Documentation

### 9.1415.1.1 DeactivatingMode

enum [Digikam::WorkerObject::DeactivatingMode](#)

#### Enumerator

FlushSignals	Already sent signals are cleared.
KeepSignals	The thread is stopped, but already sent signals remain in the queue.
PhaseOut	The thread is stopped when all signals emitted until now have been processed.

## 9.1415.2 Constructor & Destructor Documentation

### 9.1415.2.1 WorkerObject()

[Digikam::WorkerObject::WorkerObject](#) ( ) [explicit]

Implement any slots and connect signals just as usual. Call [schedule\(\)](#) before or when signals are emitted. The object will have moved to a thread when the signals are received by the slots. Call [deactivate\(\)](#) to stop computation. Note that without calling [schedule\(\)](#), no signal will ever be processed. You can use the `connectAndSchedule` convenience connection to avoid having to call [schedule\(\)](#) directly. Note that you cannot make this QObject the child of another QObject. Please check if you need to call `shutdown` from your destructor (see below).

## 9.1415.3 Member Function Documentation

### 9.1415.3.1 [aboutToDeactivate\(\)](#)

```
void Digikam::WorkerObject::aboutToDeactivate ( ) [protected], [virtual]
```

You can stop any extra controlled threads here. Immediately afterwards, an event will be sent to the working thread which will cause the event loop to quit. ([aboutToQuitLoop\(\)](#))

Reimplemented in [Digikam::RecognitionWorker](#), and [Digikam::TrainerWorker](#).

### 9.1415.3.2 [aboutToQuitLoop\(\)](#)

```
void Digikam::WorkerObject::aboutToQuitLoop ( ) [protected], [virtual]
```

Quit any blocking operation. Immediately afterwards, the event loop will be quit.

### 9.1415.3.3 [connectAndSchedule\(\)](#)

```
bool Digikam::WorkerObject::connectAndSchedule (
    const QObject * sender,
    const char * signal,
    const char * method,
    Qt::ConnectionType type = Qt::AutoConnection ) const
```

Instead, you can use these `connect()` methods when connecting your signal to this object, the signal that carries work data. Then the object will be scheduled each time you emit the signal.

### 9.1415.3.4 [deactivate](#)

```
void Digikam::WorkerObject::deactivate (
    DeactivatingMode mode = FlushSignals ) [slot]
```

If mode is `FlushSignals`, all already emitted signals will be cleared. If mode is `KeepSignals`, already emitted signals are not cleared and will be kept in the event queue until destruction or [schedule\(\)](#) is called. If mode is `PhaseOut`, already emitted signals will be processed and the thread quit immediately afterwards.

### 9.1415.3.5 [setPriority\(\)](#)

```
void Digikam::WorkerObject::setPriority (
    QThread::Priority priority )
```

Can be set anytime. If the thread is currently not running, the priority will be set when it is run next time. When you set `QThread::InheritPriority` (default), the priority is not changed but inherited from the thread pool.

### 9.1415.3.6 shutDown()

```
void Digikam::WorkerObject::shutDown ( ) [protected]
```

Note: This irrevocably stops this object. Note: It is not sufficient that your parent class does this. Calling this method, or providing one of the above mentioned equivalent guarantees, must be done by every single last class in the hierarchy with an implemented destructor deleting data. (the base class destructor is always called after the derived class)

## 9.1416 Digikam::Workflow Class Reference

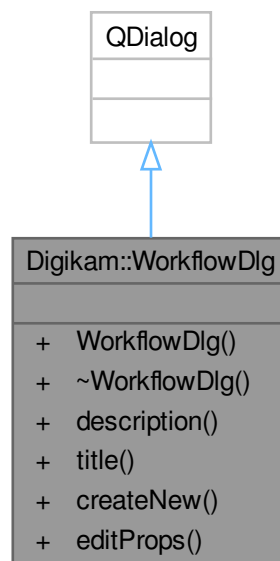
This container group all queue common settings plus all assigned batch tools.

### Public Attributes

- [BatchSetList](#) **aTools**
- `QString` **desc**
- [QueueSettings](#) **qSettings**
- `QString` **title**

## 9.1417 Digikam::WorkflowDlg Class Reference

Inheritance diagram for Digikam::WorkflowDlg:





**Public Member Functions**

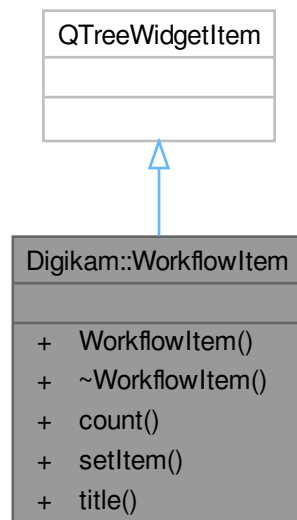
- **WorkflowDlg** (const [Workflow](#) &wf, bool create=false)
- QString **description** () const
- QString **title** () const

**Static Public Member Functions**

- static bool **createNew** ([Workflow](#) &wf)
- static bool **editProps** ([Workflow](#) &wf)

**9.1418 Digikam::WorkflowItem Class Reference**

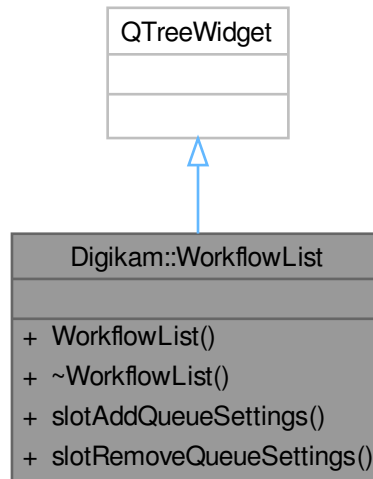
Inheritance diagram for Digikam::WorkflowItem:

**Public Member Functions**

- **WorkflowItem** ([WorkflowList](#) \*const parent, const QString &name)
- int **count** () const
- void **setItem** (const QString &title=QString(), const QString &desc=QString(), int count=0)
- QString **title** () const

## 9.1419 Digikam::WorkflowList Class Reference

Inheritance diagram for Digikam::WorkflowList:



### Public Slots

- void **slotAddQueueSettings** (const QString &title)
- void **slotRemoveQueueSettings** (const QString &title)

### Signals

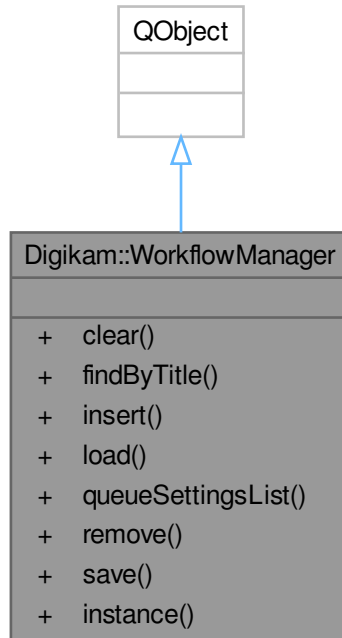
- void **signalAssignQueueSettings** (const QString &)
- void **signalUpdateQueueSettings** (const QString &)

### Public Member Functions

- **WorkflowList** (QWidget \*const parent)

## 9.1420 Digikam::WorkflowManager Class Reference

Inheritance diagram for Digikam::WorkflowManager:



### Signals

- void **signalQueueSettingsAdded** (const QString &)
- void **signalQueueSettingsRemoved** (const QString &)

### Public Member Functions

- void **clear** ()
- [Workflow](#) **findByTitle** (const QString &title) const
- void **insert** (const [Workflow](#) &q)
- bool **load** (QStringList &failed)
  - Load all [Workflow](#) from XML settings file.*
- QList<[Workflow](#)> **queueSettingsList** () const
- void **remove** (const [Workflow](#) &q)
- bool **save** ()
  - Save all [Workflow](#) to XML settings file.*

### Static Public Member Functions

- static [WorkflowManager](#) \* **instance** ()

**Friends**

- class **WorkflowManagerCreator**

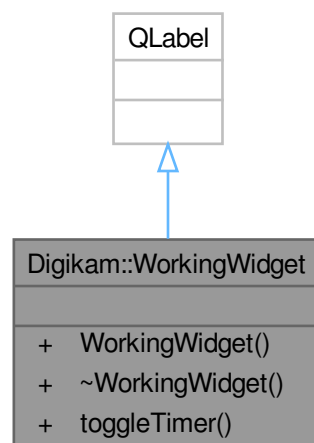
**9.1420.1 Member Function Documentation****9.1420.1.1 load()**

```
bool Digikam::WorkflowManager::load (
    QStringList & failed )
```

Fill 'failed' list with incompatible [Workflow](#) title/description not loaded.

**9.1421 Digikam::WorkingWidget Class Reference**

Inheritance diagram for Digikam::WorkingWidget:

**Public Slots**

- void **toggleTimer** (bool turnOn=false)

**Signals**

- void **animationStep** ()

**Public Member Functions**

- **WorkingWidget** (QWidget \*const parent=nullptr)

## 9.1422 Digikam::WSAlbum Class Reference

### Public Member Functions

- void **setBaseAlbum** (const [WSAlbum](#) &album)

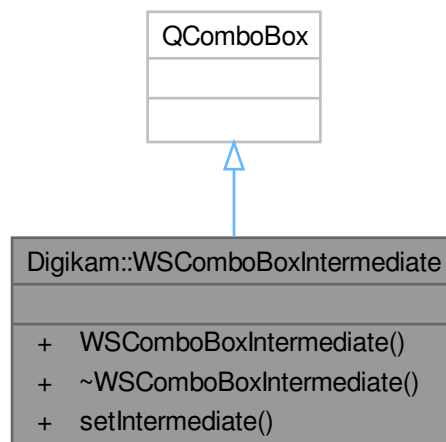
*This method is used by derived class of WSAlbum, to set the attributes inherited from [WSAlbum](#), knowing a [WSAlbum](#).*

### Public Attributes

- QString **description**
- QString **id**
- bool **isRoot** = true
- QString **location**
- QString **parentID**
- QString **title**
- bool **uploadable** = true
- QString **url**

## 9.1423 Digikam::WSComboBoxIntermediate Class Reference

Inheritance diagram for Digikam::WSComboBoxIntermediate:



### Public Member Functions

- **WSComboBoxIntermediate** (QWidget \*const =nullptr, const QString &=QString())

*Initialize the combobox with a parent and a string to indicate the intermediate state.*

- void **setIntermediate** (bool)

*Set the state of the combobox to intermediate.*

## 9.1423.1 Member Function Documentation

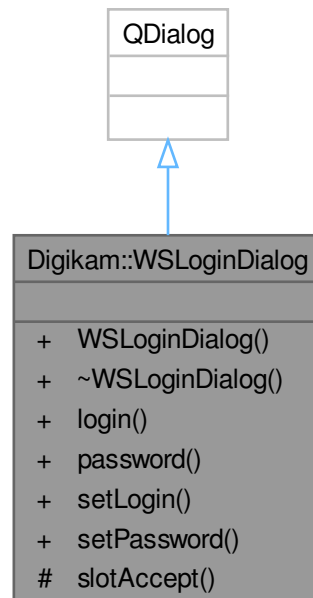
### 9.1423.1.1 setIntermediate()

```
void Digikam::WSComboBoxIntermediate::setIntermediate (
    bool state )
```

The intermediate state is 'unset' when another index is selected.

## 9.1424 Digikam::WSLoginDialog Class Reference

Inheritance diagram for Digikam::WSLoginDialog:



### Public Member Functions

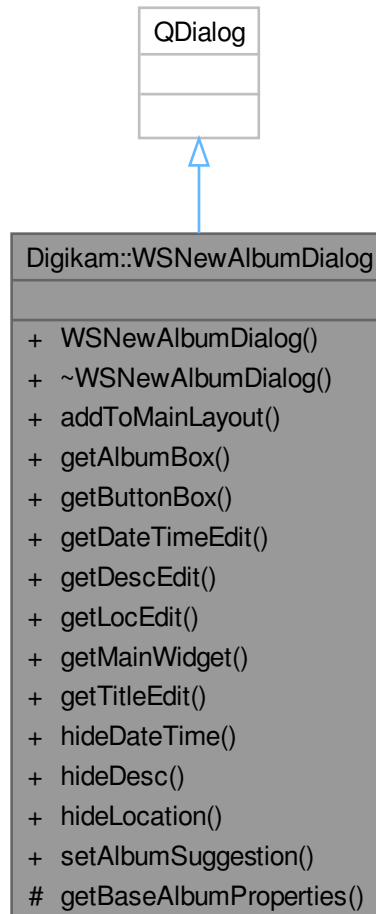
- **WSLoginDialog** (QWidget \*const parent, const QString &prompt, const QString &header=QString(), const QString &passwd=QString())
- QString **login** () const
- QString **password** () const
- void **setLogin** (const QString &)
- void **setPassword** (const QString &)

### Protected Slots

- void **slotAccept** ()

## 9.1425 Digikam::WSNewAlbumDialog Class Reference

Inheritance diagram for Digikam::WSNewAlbumDialog:



### Public Member Functions

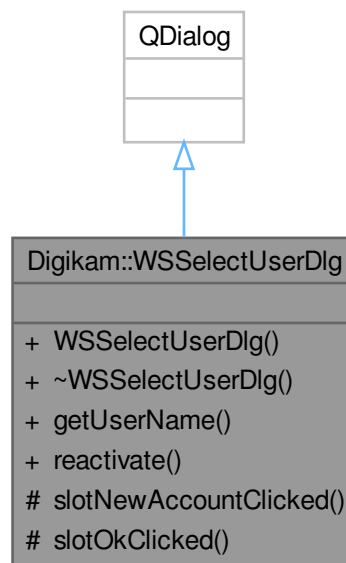
- **WSNewAlbumDialog** (QWidget \*const parent, const QString &toolName)
- void **addToMainLayout** (QWidget \*const widget)
- QGroupBox \* **getAlbumBox** () const
- QDialogButtonBox \* **getButtonBox** () const
- QDateTimeEdit \* **getDateTimeEdit** () const
- [DTextEdit](#) \* **getDescEdit** () const
- [DTextEdit](#) \* **getLocEdit** () const
- QWidget \* **getMainWidget** () const
- [DTextEdit](#) \* **getTitleEdit** () const
- void **hideDateTime** ()
- void **hideDesc** ()
- void **hideLocation** ()
- void **setAlbumSuggestion** (const QString &title)

### Protected Member Functions

- void **getBaseAlbumProperties** ([WSAlbum](#) &baseAlbum)

## 9.1426 Digikam::WSSelectUserDlg Class Reference

Inheritance diagram for Digikam::WSSelectUserDlg:



### Public Member Functions

- **WSSelectUserDlg** (`QWidget *const parent, const QString &serviceName`)
- `QString` **getUserName** () const
- void **reactivate** ()

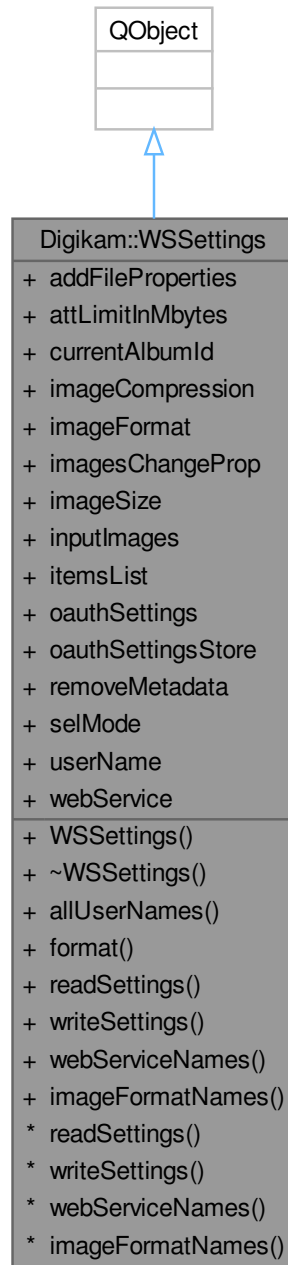
### Protected Slots

- void **slotNewAccountClicked** ()
- void **slotOkClicked** ()



## 9.1427 Digikam::WSSettings Class Reference

Inheritance diagram for Digikam::WSSettings:



### Public Types

- enum **ImageFormat** { **JPEG** = 0 , **PNG** }
- enum **Selection** { **EXPORT** = 0 , **IMPORT** }

*Images selection mode.*

- enum **WebService** {  
**FLICKR** = 0 , **DROPBOX** , **IMGUR** , **FACEBOOK** ,  
**SMUGMUG** , **GDRIVE** , **GPHOTO** }

## Public Member Functions

- **WSSettings** (QObject \*const parent=nullptr)
- QList **allUserNames** (const QString &serviceName)  
*Helper method to list all user accounts (of all web service) that user logged in before.*
- QString **format** () const
  
- void **readSettings** (const KConfigGroup &group)  
*Read and write settings in config file between sessions.*
- void **writeSettings** (KConfigGroup &group)

## Static Public Member Functions

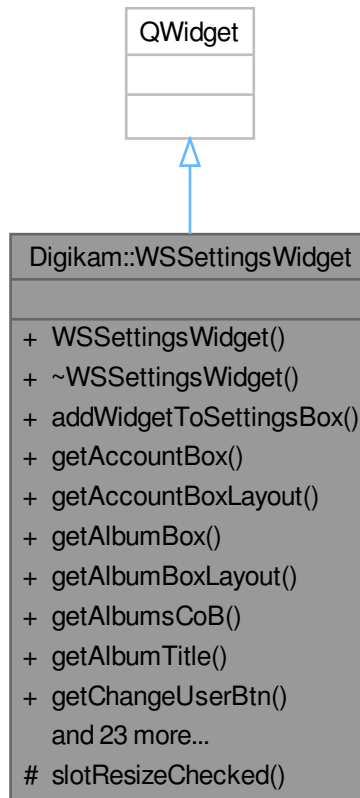
- static QMap< WebService, QString > **webServiceNames** ()  
*Helper methods to fill settings from GUI.*
- static QMap< ImageFormat, QString > **imageFormatNames** ()

## Public Attributes

- bool **addFileProperties** = false
- qint64 **attLimitInMbytes** = 17
- QString **currentAlbumId**  
*Selected album to upload to.*
- int **imageCompression** = 75
- ImageFormat **imageFormat** = JPEG
- bool **imagesChangeProp** = false
- int **imageSize** = 1024
- QList< QUrl > **inputImages**  
*Selected items to upload.*
- QMap< QUrl, QUrl > **itemsList**  
*Map of original item and attached item (can be resized).*
- QSettings \* **oauthSettings** = nullptr
- O0SettingsStore \* **oauthSettingsStore** = nullptr
- bool **removeMetadata** = false
- Selection **selMode** = EXPORT  
*Items selection mode.*
- QString **userName**
- WebService **webService** = FLICKR

## 9.1428 Digikam::WSSettingsWidget Class Reference

Inheritance diagram for Digikam::WSSettingsWidget:



### Public Member Functions

- **WSSettingsWidget** (`QWidget *const parent`, [DInfoInterface](#) \*const iface, `const QString &toolName`)
- void **addWidgetToSettingsBox** (`QWidget *const widget`)
- `QGroupBox * getAccountBox () const`
- `QGridLayout * getAccountBoxLayout () const`
- `QGroupBox * getAlbumBox () const`
- `QGridLayout * getAlbumBoxLayout () const`
- `QComboBox * getAlbumsCoB () const`
- `QString getAlbumTitle () const`
- `QPushButton * getChangeUserBtn () const`
- `QString getDestinationPath () const`
- `QComboBox * getDimensionCoB () const`
- `QSpinBox * getDimensionSpB () const`
- `QLabel * getHeaderLbl () const`
- `QSpinBox * getImgQualitySpB () const`
- `QPushButton * getNewAlbmBtn () const`
- `QGroupBox * getOptionsBox () const`

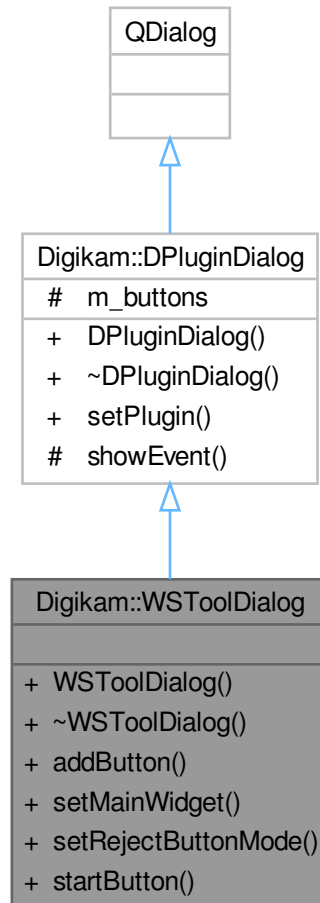
- QGridLayout \* **getOptionsBoxLayout** () const
- QCheckBox \* **getOriginalCheckBox** () const
- QCheckBox \* **getPhotoldCheckBox** () const
- QPushButton \* **getReloadBtn** () const
- QCheckBox \* **getResizeCheckBox** () const
- QWidget \* **getSettingsBox** () const
- QVBoxLayout \* **getSettingsBoxLayout** () const
- QGroupBox \* **getSizeBox** () const
- QVBoxLayout \* **getSizeBoxLayout** () const
- QGroupBox \* **getUploadBox** () const
- QVBoxLayout \* **getUploadBoxLayout** () const
- QLabel \* **getUserNameLabel** () const
- [DItemsList](#) \* **imagesList** () const
- [DProgressWdg](#) \* **progressBar** () const
- void **replacelmageList** (QWidget \*const widget)
- virtual void **updateLabels** (const QString &name=QString(), const QString &url=QString())=0

### Protected Slots

- void **slotResizeChecked** ()

## 9.1429 Digikam::WSToolDialog Class Reference

Inheritance diagram for Digikam::WSToolDialog:



### Signals

- void **cancelClicked** ()

### Public Member Functions

- **WSToolDialog** (QWidget \*const parent, const QString &objName)
- void **addButton** (QAbstractButton \*button, QDialogButtonBox::ButtonRole role)
- void **setMainWidget** (QWidget \*const widget)
- void **setRejectButtonMode** (QDialogButtonBox::StandardButton button)
- QPushButton \* **startButton** () const

### Public Member Functions inherited from [Digikam::DPluginDialog](#)

- **DPluginDialog** (QWidget \*const parent, const QString &objName)
- void **setPlugin** ([DPlugin](#) \*const tool)

### Additional Inherited Members

### Protected Member Functions inherited from [Digikam::DPluginDialog](#)

- void **showEvent** (QShowEvent \*) override

### Protected Attributes inherited from [Digikam::DPluginDialog](#)

- QDialogButtonBox \* **m\_buttons** = nullptr

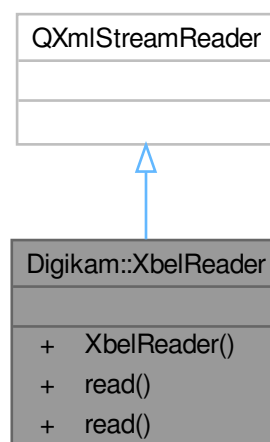
## 9.1430 Digikam::WSToolUtils Class Reference

### Static Public Member Functions

- static void **clearToken** (const QString &name)
- static QString **decodeKey** (const QString &key)
- static QSettings \* **getOAuthSettings** (QObject \*const parent)
- static QDir **makeTemporaryDir** (const char \*prefix)
- static QString **randomString** (const int &length)  
*Generates random string.*
- static QString **readToken** (const QString &name)
- static void **removeTemporaryDir** (const char \*prefix)
- static void **saveToken** (const QString &name, const QString &token)

## 9.1431 Digikam::XbelReader Class Reference

Inheritance diagram for Digikam::XbelReader:



### Public Member Functions

- [BookmarkNode](#) \* **read** (const QString &fileName)
- [BookmarkNode](#) \* **read** (QIODevice \*const device, bool addRootFolder=false)

## 9.1432 Digikam::XbelWriter Class Reference

Inheritance diagram for Digikam::XbelWriter:

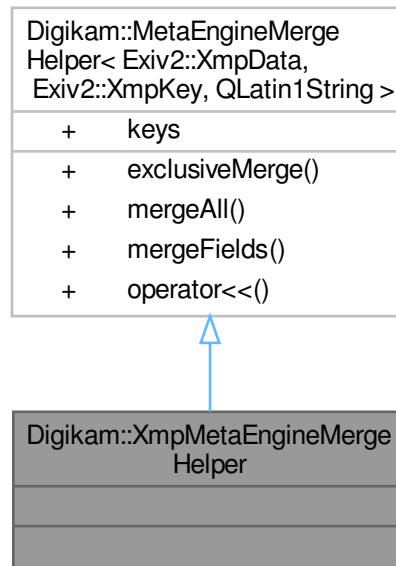


### Public Member Functions

- bool **write** (const QString &fileName, const [BookmarkNode](#) \*const root)
- bool **write** (QIODevice \*const device, const [BookmarkNode](#) \*const root)

## 9.1433 Digikam::XmpMetaEngineMergeHelper Class Reference

Inheritance diagram for Digikam::XmpMetaEngineMergeHelper:



### Additional Inherited Members

#### Public Member Functions inherited from

#### [Digikam::MetaEngineMergeHelper< Exiv2::XmpData, Exiv2::XmpKey, QLatin1String >](#)

- void [exclusiveMerge](#) (const Exiv2::XmpData &src, Exiv2::XmpData &dest)  
*Merge two (Exif,IPTC,Xmp) Data packages, the result is stored in dest.*
- void [mergeAll](#) (const Exiv2::XmpData &src, Exiv2::XmpData &dest)  
*Merge two (Exif,IPTC,Xmp) Data packages, where the result is stored in dest and fields from src take precedence over existing data from dest.*
- void [mergeFields](#) (const Exiv2::XmpData &src, Exiv2::XmpData &dest)  
*Merge two (Exif,IPTC,Xmp) Data packages, the result is stored in dest.*
- [MetaEngineMergeHelper](#) & [operator<<](#) (const QLatin1String &key)

#### Public Attributes inherited from

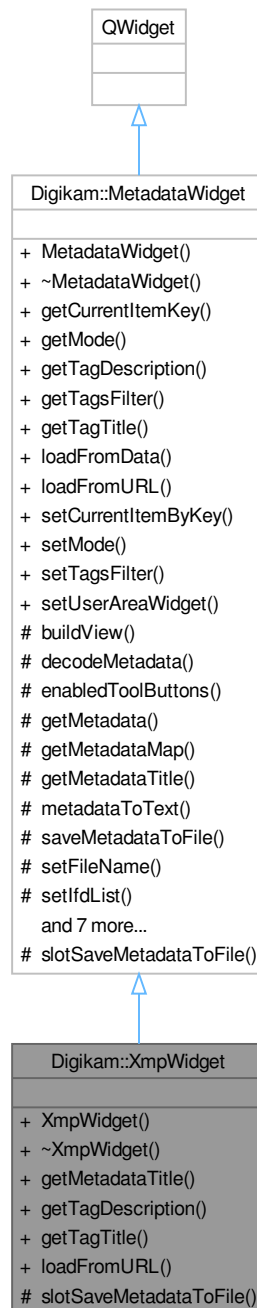
#### [Digikam::MetaEngineMergeHelper< Exiv2::XmpData, Exiv2::XmpKey, QLatin1String >](#)

- QList< QLatin1String > [keys](#)



## 9.1434 Digikam::XmpWidget Class Reference

Inheritance diagram for Digikam::XmpWidget:



### Public Member Functions

- **XmpWidget** (QWidget \*const parent, const QString &name=QString())
- QString [getMetadataTitle](#) () const override

- QString [getTagDescription](#) (const QString &key) override
- QString [getTagTitle](#) (const QString &key) override
- bool [loadFromURL](#) (const QUrl &url) override

### Public Member Functions inherited from [Digikam::MetadataWidget](#)

- **MetadataWidget** (QWidget \*const parent, const QString &name=QString())
- QString [getCurrentItemKey](#) () const
- int [getMode](#) () const
- QStringList [getTagsFilter](#) () const
- virtual bool [loadFromData](#) (const QString &fileName, const [DMetadadata](#) &data=[DMetadadata](#)())
- void [setCurrentItemByKey](#) (const QString &itemKey)
- void [setMode](#) (int mode)
- void [setTagsFilter](#) (const QStringList &list)
- void [setUserAreaWidget](#) (QWidget \*const w)

### Protected Slots

- void [slotSaveMetadataToFile](#) () override

### Protected Slots inherited from [Digikam::MetadataWidget](#)

- virtual void [slotSaveMetadataToFile](#) ()=0

### Additional Inherited Members

### Public Types inherited from [Digikam::MetadataWidget](#)

- enum [TagFilters](#) { NONE = 0 , PHOTO , CUSTOM }

### Signals inherited from [Digikam::MetadataWidget](#)

- void [signalSetupMetadataFilters](#) ()

### Protected Member Functions inherited from [Digikam::MetadataWidget](#)

- void [enabledToolButtons](#) (bool)
  - [DMetadadata](#) \* [getMetadata](#) () const
  - const [DMetadadata::MetaDatumMap](#) & [getMetadataMap](#) ()
  - QString [metadataToText](#) () const
  - QUrl [saveMetadataToFile](#) (const QString &caption, const QString &fileFilter)
  - void [setFileName](#) (const QString &fileName)
  - void [setIfdList](#) (const [DMetadadata::MetaDatumMap](#) &ifds, const QStringList &keysFilter, const QStringList &tagsFilter)
  - void [setIfdList](#) (const [DMetadadata::MetaDatumMap](#) &ifds, const QStringList &tagsFilter=QStringList())
  - bool [setMetadata](#) (const [DMetadadata](#) &data=[DMetadadata](#)())
  - virtual void [setMetadataEmpty](#) ()
  - void [setMetadataMap](#) (const [DMetadadata::MetaDatumMap](#) &data=[DMetadadata::MetaDatumMap](#)())
  - void [setup](#) ()
- Call this method in children class constructors to init signal/slots connections.*
- bool [storeMetadataToFile](#) (const QUrl &url, const QByteArray &metaData)
  - [MetadataListView](#) \* [view](#) () const

## 9.1434.1 Member Function Documentation

### 9.1434.1.1 getMetadataTitle()

```
QString Digikam::XmpWidget::getMetadataTitle ( ) const [override], [virtual]
```

Implements [Digikam::MetadataWidget](#).

### 9.1434.1.2 getTagDescription()

```
QString Digikam::XmpWidget::getTagDescription (
    const QString & key ) [override], [virtual]
```

Reimplemented from [Digikam::MetadataWidget](#).

### 9.1434.1.3 getTagTitle()

```
QString Digikam::XmpWidget::getTagTitle (
    const QString & key ) [override], [virtual]
```

Reimplemented from [Digikam::MetadataWidget](#).

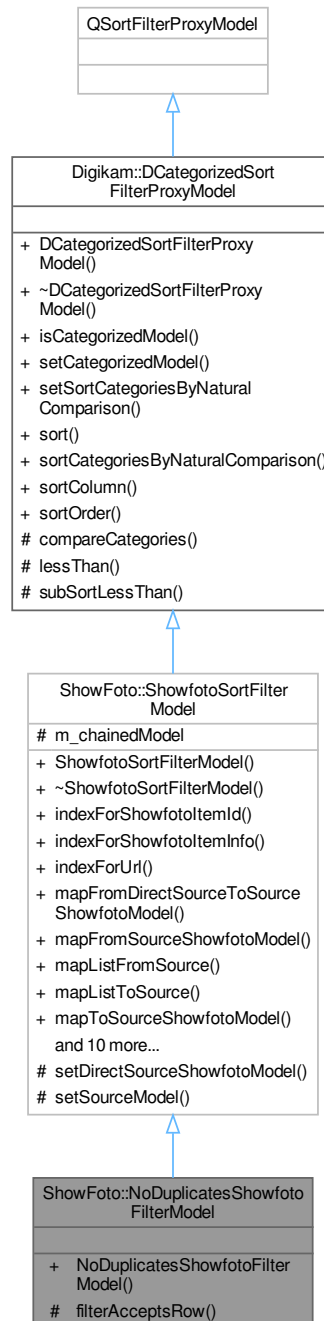
### 9.1434.1.4 loadFromURL()

```
bool Digikam::XmpWidget::loadFromURL (
    const QUrl & url ) [override], [virtual]
```

Implements [Digikam::MetadataWidget](#).

## 9.1435 ShowFoto::NoDuplicatesShowfotoFilterModel Class Reference

Inheritance diagram for ShowFoto::NoDuplicatesShowfotoFilterModel:



### Public Member Functions

- **NoDuplicatesShowfotoFilterModel** (QObject \*const parent=nullptr)

## Public Member Functions inherited from ShowFoto::ShowfotoSortFilterModel

- **ShowfotoSortFilterModel** (QObject \*const parent=nullptr)
  - QModelIndex **indexForShowfotoItemId** (qulonglong id) const
  - QModelIndex **indexForShowfotoItemInfo** (const ShowfotoItemInfo &info) const
  - QModelIndex **indexForUrl** (const QUrl &fileUrl) const
  - QModelIndex **mapFromDirectSourceToSourceShowfotoModel** (const QModelIndex &sourceModelIndex) const
  - QModelIndex **mapFromSourceShowfotoModel** (const QModelIndex &showfotoModelIndex) const
  - QList< QModelIndex > **mapListFromSource** (const QList< QModelIndex > &sourceIndexes) const
  - QList< QModelIndex > **mapListToSource** (const QList< QModelIndex > &indexes) const
  - QModelIndex **mapToSourceShowfotoModel** (const QModelIndex &proxyIndex) const
- Convenience methods mapped to ShowfotoItemModel.*
- void **setSourceFilterModel** (ShowfotoSortFilterModel \*const sourceModel)
  - void **setSourceShowfotoModel** (ShowfotoItemModel \*const sourceModel)
  - virtual ShowfotoFilterModel \* **showfotoFilterModel** () const
- Returns this, any chained ShowfotoFilterModel, or 0.*
- qulonglong **showfotoItemId** (const QModelIndex &index) const
  - QList< qulonglong > **showfotoItemIds** (const QList< QModelIndex > &indexes) const
  - ShowfotoItemInfo **showfotoItemInfo** (const QModelIndex &index) const
  - QList< ShowfotoItemInfo > **showfotoItemInfos** (const QList< QModelIndex > &indexes) const
  - QList< ShowfotoItemInfo > **showfotoItemInfosSorted** () const
- Returns a list of all showfoto infos, sorted according to this model.*
- ShowfotoSortFilterModel \* **sourceFilterModel** () const
  - ShowfotoItemModel \* **sourceShowfotoModel** () const

## Public Member Functions inherited from Digikam::DCategorizedSortFilterProxyModel

- **DCategorizedSortFilterProxyModel** (QObject \*const parent=nullptr)
  - bool **isCategorizedModel** () const
  - void **setCategorizedModel** (bool categorizedModel)
- Enables or disables the categorization feature.*
- void **setSortCategoriesByNaturalComparison** (bool sortCategoriesByNaturalComparison)
- Set if the sorting using CategorySortRole will use a natural comparison in the case that strings were returned.*
- void **sort** (int column, Qt::SortOrder order=Qt::AscendingOrder) override
- Overridden from QSortFilterProxyModel.*
- bool **sortCategoriesByNaturalComparison** () const
  - int **sortColumn** () const
  - Qt::SortOrder **sortOrder** () const

## Protected Member Functions

- bool **filterAcceptsRow** (int source\_row, const QModelIndex &source\_parent) const override

## Protected Member Functions inherited from ShowFoto::ShowfotoSortFilterModel

- virtual void **setDirectSourceShowfotoModel** (ShowfotoItemModel \*const sourceModel)
- Reimplement if needed. Called only when model shall be set as (direct) sourceModel.*
- void **setSourceModel** (QAbstractItemModel \*sourceModel) override

## Protected Member Functions inherited from [Digikam::DCategorizedSortFilterProxyModel](#)

- virtual int [compareCategories](#) (const QModelIndex &left, const QModelIndex &right) const  
*This method compares the category of the `left` index with the category of the `right` index.*
- bool [lessThan](#) (const QModelIndex &left, const QModelIndex &right) const override  
*Overridden from `QSortFilterProxyModel`.*
- virtual bool [subSortLessThan](#) (const QModelIndex &left, const QModelIndex &right) const  
*This method has a similar purpose as [lessThan\(\)](#) has on `QSortFilterProxyModel`.*

## Additional Inherited Members

## Public Types inherited from [Digikam::DCategorizedSortFilterProxyModel](#)

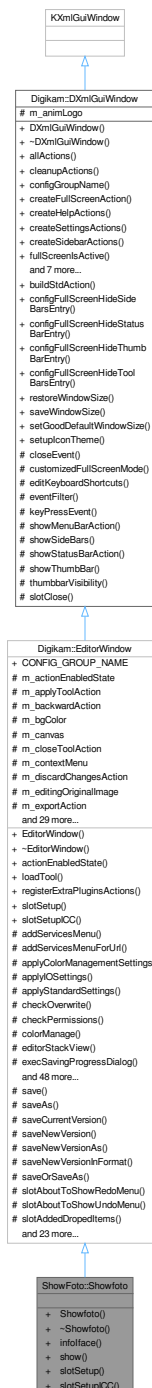
- enum [AdditionalRoles](#) { [CategoryDisplayRole](#) = 0x17CE990A , [CategorySortRole](#) = 0x27857E60 }

## Protected Attributes inherited from [ShowFoto::ShowfotoSortFilterModel](#)

- [ShowfotoSortFilterModel](#) \* [m\\_chainedModel](#) = nullptr

## 9.1436 ShowFoto::Showfoto Class Reference

Inheritance diagram for ShowFoto::Showfoto:



### Public Slots

- void `slotSetup ()` override
- void `slotSetupICC ()` override

## Public Slots inherited from [Digikam::EditorWindow](#)

- void **slotSetup** () override=0
- virtual void **slotSetupICC** ()=0

## Signals

- void **signalInfoList** (const ShowfotoItemInfoList &)
- void **signalLoadCurrentItem** (const QList< QUrl > &urlList)
- void **signalOpenFile** (const QList< QUrl > &urls)
- void **signalOpenFolder** (const QUrl &)

## Signals inherited from [Digikam::EditorWindow](#)

- void **signalNoCurrentItem** ()
- void **signalPreviewModeChanged** (int)
- void **signalSelectionChanged** (const QRect &)
- void **signalToolApplied** ()

## Public Member Functions

- **Showfoto** (const QList< QUrl > &urlList, QWidget \*const parent=nullptr)
- DInfoInterface \* **infoface** (DPluginAction \*const ac) override  
*Return the interface instance to access to items information.*
- virtual void **show** ()

## Public Member Functions inherited from [Digikam::EditorWindow](#)

- **EditorWindow** (const QString &name, QWidget \*const parent=nullptr)
- bool **actionEnabledState** () const
- void **loadTool** ([EditorTool](#) \*const tool)
- void **registerExtraPluginsActions** (QString &dom) override

## Public Member Functions inherited from [Digikam::DXmlGuiWindow](#)

- **DXmlGuiWindow** (QWidget \*const parent=nullptr, Qt::WindowFlags f=Qt::WindowFlags())
- QList< QAction \* > **allActions** () const  
*Return all actions from internal collection.*
- void **cleanupActions** ()  
*Cleanup unwanted actions from action collection.*
- QString **configGroupName** () const
- void **createFullscreenAction** (const QString &name)  
*Create Full-screen action to action collection instance from managed window set through setManagedWindow().*
- void **createHelpActions** (const QString &handbookSection, bool coreOptions=true)  
*Create common actions from Help menu for all digiKam main windows.*
- void **createSettingsActions** ()  
*Create common actions to setup all digiKam main windows.*
- void **createSidebarActions** ()  
*Create common actions to handle side-bar through keyboard shortcuts.*



- bool **fullScreensActive** () const  
*Return true if managed window is currently in Full Screen Mode.*
- void **readFullScreenSettings** (const KConfigGroup &group)  
*Read full-screen settings from KDE config file.*
- void **registerPluginsActions** ()  
*Register all generic plugins action to this instance.*
- void **setConfigGroupName** (const QString &name)  
*Manage config group name used by window instance to get/set settings from config file.*
- void **setFullScreenOptions** (int options)  
*Set full-screen options to managed window.*
- void **unminimizeAndActivateWindow** ()

### Additional Inherited Members

### Public Types inherited from [Digikam::EditorWindow](#)

- enum **TransformType** { **RotateLeft** , **RotateRight** , **FlipHorizontal** , **FlipVertical** }

### Static Public Member Functions inherited from [Digikam::DXmlGuiWindow](#)

- static QAction \* **buildStdAction** (StdActionType type, const QObject \*const recvr, const char \*const slot, QObject \*const parent)
- static QString **configFullScreenHideSideBarsEntry** ()
- static QString **configFullScreenHideStatusBarEntry** ()
- static QString **configFullScreenHideThumbBarEntry** ()
- static QString **configFullScreenHideToolBarsEntry** ()  
*Shared with [FullScreenSettings](#).*
- static void **restoreWindowSize** (QWindow \*const win, const KConfigGroup &group)
- static void **saveWindowSize** (QWindow \*const win, KConfigGroup &group)
- static void **setGoodDefaultWindowSize** (QWindow \*const win)
- static void **setupIconTheme** ()  
*If we have some local breeze icon resource, prefer it.*

### Static Public Attributes inherited from [Digikam::EditorWindow](#)

- static const QString **CONFIG\_GROUP\_NAME**

### Protected Types inherited from [Digikam::EditorWindow](#)

- enum **SaveAskMode** {  
**AskIfNeeded** , **OverwriteWithoutAsking** , **AlwaysSaveAs** , **SaveVersionWithoutAsking** = Overwrite↔  
WithoutAsking ,  
**AlwaysNewVersion** = AlwaysSaveAs }

## Protected Slots inherited from [Digikam::EditorWindow](#)

- virtual bool **saveOrSaveAs** ()
- void **slotAboutToShowRedoMenu** ()
- void **slotAboutToShowUndoMenu** ()
- virtual void **slotAddedDroppedItems** (QDropEvent \*e)=0
- virtual void **slotBackward** ()=0
- virtual void **slotChanged** ()=0
- void **slotComponentsInfo** () override
- virtual void **slotContextMenu** ()=0
- virtual void **slotDeleteCurrentItem** ()=0
- virtual void **slotDiscardChanges** ()
- virtual void **slotFileOriginChanged** (const QString &filePath)
- virtual void **slotFileWithDefaultApplication** ()=0
- virtual void **slotFirst** ()=0
- virtual void **slotForward** ()=0
- virtual void **slotLast** ()=0
- virtual void **slotLoadingFinished** (const QString &filename, bool success)
- void **slotLoadingProgress** (const QString &filePath, float progress)
- virtual void **slotLoadingStarted** (const QString &filename)
- void **slotNameLabelCancelButtonPressed** ()
- virtual void **slotOpenOriginal** ()
- virtual void **slotOpenWith** (QAction \*action=nullptr)=0
- virtual void **slotPrepareToLoad** ()
- virtual void **slotRevert** ()=0
- void **slotSavingProgress** (const QString &filePath, float progress)
- virtual void **slotSavingStarted** (const QString &filename)
- void **slotSelected** (bool)
- virtual void **slotUpdateItemInfo** ()=0

## Protected Slots inherited from [Digikam::DXmlGuiWindow](#)

- bool **slotClose** ()

## Protected Member Functions inherited from [Digikam::EditorWindow](#)

- void **addServicesMenuForUrl** (const QUrl &url)
- void **applyColorManagementSettings** ()
- void **applyIOSettings** ()
- void **applyStandardSettings** ()
- bool **checkOverwrite** (const QUrl &url)
- bool **checkPermissions** (const QUrl &url)
- void **colorManage** ()
- [EditorStackView](#) \* **editorStackView** () const
- void **execSavingProgressDialog** ()
- [ExposureSettingsContainer](#) \* **exposureSettings** () const
- virtual bool **hasOriginalToRestore** ()
- bool **moveLocalFile** (const QString &src, const QString &dest)
- void **movingSaveFileFinished** (bool successful)
- void **openWith** (const QUrl &url, QAction \*action)
- bool **promptForOverWrite** ()
- bool **promptUserDelete** (const QUrl &url)
- bool **promptUserSave** (const QUrl &url, SaveAskMode mode=AskIfNeeded, bool allowCancel=true)

- void **readStandardSettings** ()
  - void **resetOrigin** ()
  - void **resetOriginSwitchFile** ()
  - virtual [DImageHistory](#) **resolvedImageHistory** (const [DImageHistory](#) &history)
  - [VersionFileOperation](#) **saveAsVersionFileOperation** (const [QUrl](#) &url, const [QUrl](#) &saveLocation, const [QString](#) &format)
  - [VersionFileOperation](#) **saveInFormatVersionFileOperation** (const [QUrl](#) &url, const [QString](#) &format)
  - void **saveStandardSettings** ()
  - [VersionFileOperation](#) **saveVersionFileOperation** (const [QUrl](#) &url, bool fork)
  - void **setupContextMenu** ()
  - void **setupSelectToolsAction** ()
  - void **setupStandardActions** ()
  - void **setupStandardConnections** ()
  - void **setupStatusBar** ()
  - [SidebarSplitter](#) \* **sidebarSplitter** () const
  - void **startingSave** (const [QUrl](#) &url)
  - bool **startingSaveAs** (const [QUrl](#) &url)
  - bool **startingSaveCurrentVersion** (const [QUrl](#) &url)
  - bool **startingSaveNewVersion** (const [QUrl](#) &url)
  - bool **startingSaveNewVersionAs** (const [QUrl](#) &url)
  - bool **startingSaveNewVersionInFormat** (const [QUrl](#) &url, const [QString](#) &format)
  - void **toggleNonDestructiveActions** ()
  - void **toggleStandardActions** (bool val)
  - void **toggleToolActions** ([EditorTool](#) \*tool=nullptr)
  - void **toggleZoomActions** (bool val)
- Method used by Editor Tools.*
- virtual [VersionManager](#) \* **versionManager** () const
  - bool **waitForSavingToComplete** ()

### Protected Member Functions inherited from [Digikam::DXmlGuiWindow](#)

- void **closeEvent** ([QCloseEvent](#) \*e) override
- void **editKeyboardShortcuts** ([KActionCollection](#) \*const extraac=nullptr, const [QString](#) &actitle=[QString](#)())  
*Call this method from your main window to show keyboard shortcut config dialog with an extra action collection to configure.*
- bool **eventFilter** ([QObject](#) \*obj, [QEvent](#) \*ev) override
- void **keyPressEvent** ([QKeyEvent](#) \*e) override
- [QAction](#) \* **showMenuBarAction** () const
- [QAction](#) \* **showStatusBarAction** () const

### Protected Attributes inherited from [Digikam::EditorWindow](#)

- bool **m\_actionEnabledState** = false
- [QAction](#) \* **m\_applyToolAction** = nullptr
- [QAction](#) \* **m\_backwardAction** = nullptr
- [QColor](#) **m\_bgColor**
- [Canvas](#) \* **m\_canvas** = nullptr
- [QAction](#) \* **m\_closeToolAction** = nullptr
- [QMenu](#) \* **m\_contextMenu** = nullptr
- [QAction](#) \* **m\_discardChangesAction** = nullptr
- bool **m\_editingOriginalImage** = true
- [QAction](#) \* **m\_exportAction** = nullptr

- QAction \* **m\_fileDeleteAction** = nullptr
- QAction \* **m\_firstAction** = nullptr
- QString **m\_formatForRAWVersioning**
- QString **m\_formatForSubversions**
- QAction \* **m\_forwardAction** = nullptr
- IOFileSettings \* **m\_IOFileSettings** = nullptr
- QAction \* **m\_lastAction** = nullptr
- StatusProgressBar \* **m\_nameLabel** = nullptr
- bool **m\_nonDestructive** = true
- QAction \* **m\_openVersionAction** = nullptr
- KToolBarPopupAction \* **m\_redoAction** = nullptr
- DAdjustableLabel \* **m\_resLabel** = nullptr
- QAction \* **m\_revertAction** = nullptr
- QAction \* **m\_saveAction** = nullptr
- QAction \* **m\_saveAsAction** = nullptr
- QAction \* **m\_saveCurrentVersionAction** = nullptr
- KToolBarPopupAction \* **m\_saveNewVersionAction** = nullptr
- QAction \* **m\_saveNewVersionAsAction** = nullptr
- QMenu \* **m\_saveNewVersionInFormatAction** = nullptr
- SavingContext **m\_savingContext**
- QPointer< QProgressDialog > **m\_savingProgressDialog** = nullptr
- QAction \* **m\_serviceAction** = nullptr
- QMenu \* **m\_servicesMenu** = nullptr
- bool **m\_setExifOrientationTag** = true
- QAction \* **m\_showBarAction** = nullptr
- SidebarSplitter \* **m\_splitter** = nullptr
- EditorStackView \* **m\_stackView** = nullptr
- QVector< TransformType > **m\_transformQue**
- KToolBarPopupAction \* **m\_undoAction** = nullptr

## Protected Attributes inherited from [Digikam::DXmlGuiWindow](#)

- [DLogoAction](#) \* **m\_animLogo** = nullptr

## 9.1436.1 Member Function Documentation

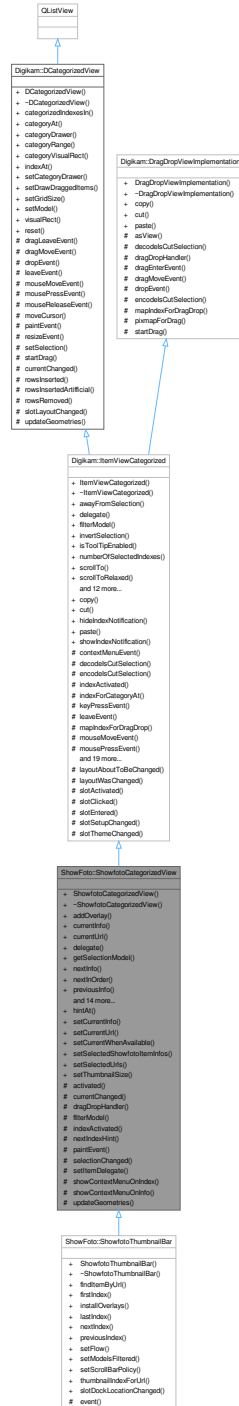
### 9.1436.1.1 infoface()

```
DInfoInterface * ShowFoto::Showfoto::infoIface (
    DPluginAction *const ac ) [override], [virtual]
```

Implements [Digikam::DXmlGuiWindow](#).

# 9.1437 ShowFoto::ShowfotoCategorizedView Class Reference

Inheritance diagram for ShowFoto::ShowfotoCategorizedView:



## Public Slots

- void **hintAt** (const [ShowfotoItemInfo](#) &info)  
*Does something to gain attention for info, but not changing current selection.*

- void **setCurrentInfo** (const [ShowfotoItemInfo](#) &info)  
*Set as current item the item identified by the [ShowfotoItemInfo](#).*
- void **setCurrentUrl** (const QUrl &url)  
*Set as current item the item identified by its file url.*
- void **setCurrentWhenAvailable** (qulonglong ShowfotoItemId)  
*Scroll the view to the given item when it becomes available.*
- void **setSelectedShowfotoItemInfos** (const QList< [ShowfotoItemInfo](#) > &infos)  
*Set selected items.*
- void **setSelectedUrls** (const QList< QUrl > &urlList)  
*Set selected items identified by their file urls.*
- void **setThumbnailSize** (int size)

### Public Slots inherited from [Digikam::ItemViewCategorized](#)

- void **copy** () override
- void **cut** () override
- void **hideIndexNotification** ()
- void **paste** () override
- void **showIndexNotification** (const QModelIndex &index, const QString &message)

### Public Slots inherited from [Digikam::DCategorizedView](#)

- void **reset** () override

### Signals

- void **currentChanged** (const [ShowfotoItemInfo](#) &info)
- void **deselected** (const QList< [ShowfotoItemInfo](#) > &nowDeselectedInfos)  
*Emitted when items are deselected.*
- void **modelChanged** ()  
*Emitted when a new model is set.*
- void **selected** (const QList< [ShowfotoItemInfo](#) > &newSelectedInfos)  
*Emitted when new items are selected.*
- void **showfotoItemInfoActivated** (const [ShowfotoItemInfo](#) &info)  
*Emitted when the given [ShowfotoItemInfo](#) is activated.*

### Signals inherited from [Digikam::ItemViewCategorized](#)

- void **clicked** (const QMouseEvent \*e, const QModelIndex &index)  
*For overlays: Like the respective parent class signals, but with additional info.*
- void **entered** (const QMouseEvent \*e, const QModelIndex &index)
- void **keyPressed** (QKeyEvent \*e)  
*Remember you may want to check if the event is accepted or ignored.*
- void **selectionChanged** ()  
*Emitted when any selection change occurs.*
- void **selectionCleared** ()  
*Emitted when the selection is completely cleared.*
- void **viewportClicked** (const QMouseEvent \*e)  
*While [clicked\(\)](#) is emitted with a valid index, this corresponds to clicking on empty space.*
- void **zoomInStep** ()
- void **zoomOutStep** ()

## Public Member Functions

- **ShowfotoCategorizedView** (QWidget \*const parent=nullptr)
- void **addOverlay** (ItemDelegateOverlay \*overlay, ShowfotoDelegate \*delegate=nullptr)
  - Add and remove an overlay.*
- **ShowfotoItemInfo currentInfo** () const
- QUrl **currentUrl** () const
- **ShowfotoDelegate \* delegate** () const
- QItemSelectionModel \* **getSelectionModel** () const
- **ShowfotoItemInfo nextInfo** (const ShowfotoItemInfo &info)
- **ShowfotoItemInfo nextInOrder** (const ShowfotoItemInfo &startingPoint, int nth)
  - Returns the n-th info after the given one.*
- **ShowfotoItemInfo previousInfo** (const ShowfotoItemInfo &info)
- void **removeOverlay** (ItemDelegateOverlay \*overlay)
- QList< ShowfotoItemInfo > **selectedShowfotoItemInfos** () const
- QList< ShowfotoItemInfo > **selectedShowfotoItemInfosCurrentFirst** () const
- QList< QUrl > **selectedUrls** () const
- void **setModels** (ShowfotoItemModel \*model, ShowfotoSortFilterModel \*filterModel)
- virtual void **setThumbnailSize** (const ThumbnailSize &size)
- **ShowfotoFilterModel \* showfotoFilterModel** () const
  - Returns any ShowfotoFilterModel in chain.*
- QList< ShowfotoItemInfo > **showfotoItemInfos** () const
- **ShowfotoItemModel \* showfotoItemModel** () const
- **ShowfotoSortFilterModel \* showfotoSortFilterModel** () const
- **ShowfotoThumbnailModel \* showfotoThumbnailModel** () const
  - Returns 0 if the ShowfotoItemModel is not an ShowfotoThumbnailModel.*
- **ThumbnailSize thumbnailSize** () const
- void **toIndex** (const QUrl &url)
  - Selects the index as current and scrolls to it.*
- QList< QUrl > **urls** () const

## Public Member Functions inherited from Digikam::ItemViewCategorized

- **ItemViewCategorized** (QWidget \*const parent=nullptr)
- void **awayFromSelection** ()
- **DItemDelegate \* delegate** () const
- void **invertSelection** ()
- bool **isToolTipEnabled** () const
- int **numberOfSelectedIndexes** () const
- void **scrollTo** (const QModelIndex &index, ScrollHint hint=EnsureVisible) override
- void **scrollToRelaxed** (const QModelIndex &index, ScrollHint hint=EnsureVisible)
  - Like scrollTo, but only scrolls if the index is not visible, regardless of hint.*
- void **setInitialSelectedItem** (bool enabled)
  - Ensure a initial selected item.*
- void **setScrollCurrentToCenter** (bool enabled)
  - Scroll automatically the current index to center of the view.*
- void **setScrollStepGranularity** (int factor)
  - Determine a step size for scrolling: The larger this number, the smaller and more precise is the scrolling.*
- void **setSelectedIndexes** (const QList< QModelIndex > &indexes)
- void **setSpacing** (int spacing)
  - Sets the spacing.*
- void **setToolTipEnabled** (bool enabled)

- void **setUsePointingHandCursor** (bool useCursor)  
*Set if the PointingHand Cursor should be shown over the activation area.*
- void **toFirstIndex** ()  
*Selects the index as current and scrolls to it.*
- void **toIndex** (const QModelIndex &index)
- void **toLastIndex** ()
- void **toNextIndex** ()
- void **toPreviousIndex** ()

## Public Member Functions inherited from [Digikam::DCategorizedView](#)

- **DCategorizedView** (QWidget \*const parent=nullptr)
- virtual QModelIndexList **categorizedIndexesIn** (const QRect &rect) const  
*This method will return all indexes whose visual rect intersects *rect*.*
- virtual QModelIndex **categoryAt** (const QPoint &point) const  
*This method will return the first index of the category in the region of which *point* is found.*
- **DCategoryDrawer** \* **categoryDrawer** () const
- virtual QItemSelectionRange **categoryRange** (const QModelIndex &index) const  
*This method returns the range of indexes contained in the category in which *index* is sorted.*
- virtual QRect **categoryVisualRect** (const QModelIndex &index) const  
*This method will return the visual rect of the header of the category in which *index* is sorted.*
- QModelIndex **indexAt** (const QPoint &point) const override
- void **setCategoryDrawer** (**DCategoryDrawer** \*categoryDrawer)
- void **setDrawDraggedItems** (bool drawDraggedItems)  
*Switch on drawing of dragged items.*
- void **setGridSize** (const QSize &size)
- void **setModel** (QAbstractItemModel \*model) override
- QRect **visualRect** (const QModelIndex &index) const override

## Public Member Functions inherited from [Digikam::DragDropViewImplementation](#)

- virtual void **copy** ()
- virtual void **cut** ()
- virtual void **paste** ()

## Protected Member Functions

- virtual void **activated** (const [ShowfotoItemInfo](#) &info, Qt::KeyboardModifiers modifiers)  
*Reimplement these in a subclass.*
- void **currentChanged** (const QModelIndex &index, const QModelIndex &previous) override
- [AbstractItemDragDropHandler](#) \* **dragDropHandler** () const override  
*You need to implement these three methods Returns the drag drop handler.*
- QSortFilterProxyModel \* **filterModel** () const override  
*reimplemented from parent class*
- void **indexActivated** (const QModelIndex &index, Qt::KeyboardModifiers modifiers) override
- QModelIndex **nextIndexHint** (const QModelIndex &indexToAnchor, const QItemSelectionRange &removed) const override  
*Assuming the given indexes would be removed (hypothetically!), return the index to be selected instead, starting from anchor.*
- void **paintEvent** (QPaintEvent \*e) override
- void **selectionChanged** (const QItemSelection &, const QItemSelection &) override
- void **setItemDelegate** ([ShowfotoDelegate](#) \*delegate)
- void **showContextMenuOnIndex** (QContextMenuEvent \*event, const QModelIndex &index) override  
*Reimplement these in a subclass.*
- virtual void **showContextMenuOnInfo** (QContextMenuEvent \*event, const [ShowfotoItemInfo](#) &info)
- void **updateGeometries** () override



## Protected Member Functions inherited from [Digikam::ItemViewCategorized](#)

- void **contextMenuEvent** (QContextMenuEvent \*event) override  
*reimplemented from parent class*
- bool **decodelsCutSelection** (const QMimeData \*mimeType)
- void **encodelsCutSelection** (QMimeData \*mimeType, bool isCutSelection)
- QModelIndex **indexForCategoryAt** (const QPoint &pos) const  
*Returns an index that is representative for the category at position pos.*
- void **keyPressEvent** (QKeyEvent \*event) override
- void **leaveEvent** (QEvent \*event) override
- QModelIndex **mapIndexForDragDrop** (const QModelIndex &index) const override  
*Note: pure virtual [dragDropHandler\(\)](#) still open from [DragDropViewImplementation](#).*
- void **mouseMoveEvent** (QMouseEvent \*event) override
- void **mousePressEvent** (QMouseEvent \*event) override
- void **mouseReleaseEvent** (QMouseEvent \*event) override
- QModelIndex **moveCursor** (CursorAction cursorAction, Qt::KeyboardModifiers modifiers) override
- QPixmap **pixmapForDrag** (const QList< QModelIndex > &indexes) const override  
*Creates a pixmap for dragging the given indexes.*
- void **reset** () override
- void **resizeEvent** (QResizeEvent \*e) override
- void **rowsAboutToBeRemoved** (const QModelIndex &parent, int start, int end) override
- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- void **rowsRemoved** (const QModelIndex &parent, int start, int end) override
- void **selectionChanged** (const QItemSelection &, const QItemSelection &) override
- void **setItemDelegate** (DItemDelegate \*delegate)
- void **setToolTip** (ItemViewToolTip \*tip)
- virtual void **showContextMenu** (QContextMenuEvent \*event)
- virtual bool **showToolTip** (const QModelIndex &index, QStyleOptionViewItem &option, QHelpEvent \*e=nullptr)  
*Provides default behavior, can reimplement in a subclass.*
- void **updateDelegateSizes** ()
- void **userInteraction** ()
- bool **viewportEvent** (QEvent \*event) override
- void **wheelEvent** (QWheelEvent \*event) override

## Protected Member Functions inherited from [Digikam::DCategorizedView](#)

- void **dragLeaveEvent** (QDragLeaveEvent \*event) override
- void **dragMoveEvent** (QDragMoveEvent \*event) override
- void **dropEvent** (QDropEvent \*event) override
- void **leaveEvent** (QEvent \*event) override
- void **mouseMoveEvent** (QMouseEvent \*event) override
- void **mousePressEvent** (QMouseEvent \*event) override
- void **mouseReleaseEvent** (QMouseEvent \*event) override
- QModelIndex **moveCursor** (CursorAction cursorAction, Qt::KeyboardModifiers modifiers) override
- void **paintEvent** (QPaintEvent \*event) override
- void **resizeEvent** (QResizeEvent \*event) override
- void **setSelection** (const QRect &rect, QItemSelectionModel::SelectionFlags flags) override
- void **startDrag** (Qt::DropActions supportedActions) override

## Protected Member Functions inherited from [Digikam::DragDropViewImplementation](#)

- virtual `QAbstractItemView * asView ()=0`  
*This one is implemented by `DECLARE_VIEW_DRAG_DROP_METHODS`.*
- bool **decodelsCutSelection** (const `QMimeData *mimeData`)
- void **dragEnterEvent** (`QDragEnterEvent *event`)  
*Implements the relevant `QAbstractItemView` methods for drag and drop.*
- void **dragMoveEvent** (`QDragMoveEvent *e`)
- void **dropEvent** (`QDropEvent *e`)
- void **encodelsCutSelection** (`QMimeData *mime`, bool `isCutSelection`)
- void **startDrag** (`Qt::DropActions supportedActions`)

## Additional Inherited Members

## Protected Slots inherited from [Digikam::ItemViewCategorized](#)

- void **layoutAboutToBeChanged** ()
- void **layoutWasChanged** ()
- void **slotActivated** (const `QModelIndex &index`)
- void **slotClicked** (const `QModelIndex &index`)
- void **slotEntered** (const `QModelIndex &index`)
- virtual void **slotSetupChanged** ()
- virtual void **slotThemeChanged** ()

## Protected Slots inherited from [Digikam::DCategorizedView](#)

- void **currentChanged** (const `QModelIndex &current`, const `QModelIndex &previous`) override
- void **rowsInserted** (const `QModelIndex &parent`, int `start`, int `end`) override
- virtual void **rowsInsertedArtificial** (const `QModelIndex &parent`, int `start`, int `end`)
- virtual void **slotLayoutChanged** ()
- void **updateGeometries** () override

## 9.1437.1 Member Function Documentation

### 9.1437.1.1 addOverlay()

```
void ShowFoto::ShowfotoCategorizedView::addOverlay (
    ItemDelegateOverlay * overlay,
    ShowfotoDelegate * delegate = nullptr )
```

It will as well be removed automatically when destroyed. Unless you pass a different delegate, the current delegate will be used.

### 9.1437.1.2 deselected

```
void ShowFoto::ShowfotoCategorizedView::deselected (
    const QList< ShowfotoItemInfo > & nowDeselectedInfos ) [signal]
```

There may be other selected infos left. This signal is not emitted when the model is reset; then only `selectionCleared` is emitted.

### 9.1437.1.3 dragDropHandler()

```
AbstractItemDragDropHandler * ShowFoto::ShowfotoCategorizedView::dragDropHandler ( ) const
[override], [protected], [virtual]
```

Implements [Digikam::DragDropViewImplementation](#).

### 9.1437.1.4 filterModel()

```
QSortFilterProxyModel * ShowFoto::ShowfotoCategorizedView::filterModel ( ) const [override],
[protected], [virtual]
```

Implements [Digikam::ItemViewCategorized](#).

### 9.1437.1.5 indexActivated()

```
void ShowFoto::ShowfotoCategorizedView::indexActivated (
    const QModelIndex & index,
    Qt::KeyboardModifiers modifiers ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ItemViewCategorized](#).

### 9.1437.1.6 nextIndexHint()

```
QModelIndex ShowFoto::ShowfotoCategorizedView::nextIndexHint (
    const QModelIndex & indexToAnchor,
    const QItemSelectionRange & removed ) const [override], [protected], [virtual]
```

The default implementation returns the next remaining sibling.

Reimplemented from [Digikam::ItemViewCategorized](#).

### 9.1437.1.7 nextInOrder()

```
ShowfotoItemInfo ShowFoto::ShowfotoCategorizedView::nextInOrder (
    const ShowfotoItemInfo & startingPoint,
    int nth )
```

Specifically, return the previous info for  $nth = -1$  and the next info for  $n = 1$ . Returns a null info if either startingPoint or the nth info are not contained in the model

### 9.1437.1.8 selected

```
void ShowFoto::ShowfotoCategorizedView::selected (
    const QList< ShowfotoItemInfo > & newSelectedInfos ) [signal]
```

The parameter includes only the newly selected infos, there may be other already selected infos.

### 9.1437.1.9 showContextMenuOnIndex()

```
void ShowFoto::ShowfotoCategorizedView::showContextMenuOnIndex (
    QContextMenuEvent * event,
    const QModelIndex & index ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ItemViewCategorized](#).

### 9.1437.1.10 showfotoFilterModel()

```
ShowfotoFilterModel * ShowFoto::ShowfotoCategorizedView::showfotoFilterModel ( ) const
```

May not be sourceModel()

### 9.1437.1.11 showfotoItemInfoActivated

```
void ShowFoto::ShowfotoCategorizedView::showfotoItemInfoActivated (
    const ShowfotoItemInfo & info ) [signal]
```

Info is never null.

## 9.1438 ShowFoto::ShowfotoCoordinatesOverlay Class Reference

Inheritance diagram for ShowFoto::ShowfotoCoordinatesOverlay:



### Public Member Functions

- **ShowfotoCoordinatesOverlay** (QObject \*const parent)
- [ShowfotoCoordinatesOverlayWidget](#) \* **buttonWidget** () const

## Public Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- [AbstractWidgetDelegateOverlay](#) (QObject \*const parent)  
*This class provides functionality for using a widget in an overlay.*

## Public Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- [ItemDelegateOverlay](#) (QObject \*const parent=nullptr)
- virtual bool [acceptsDelegate](#) (QAbstractItemDelegate \*) const
- QAbstractItemDelegate \* [delegate](#) () const
- virtual void [mouseMoved](#) (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)  
*Only these two methods are implemented as virtual methods.*
- virtual void [paint](#) (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index)
- void [setDelegate](#) (QAbstractItemDelegate \*delegate)
- void [setView](#) (QAbstractItemView \*view)
- QAbstractItemView \* [view](#) () const

## Protected Member Functions

- bool [checkIndex](#) (const QModelIndex &index) const override  
*Return true here if you want to show the overlay for the given index.*
- QWidget \* [createWidget](#) () override  
*Create your widget here.*
- void [setActive](#) (bool active) override  
*If active is true, this will call [createWidget\(\)](#), initialize the widget for use, and setup connections for the virtual slots.*
- void [slotEntered](#) (const QModelIndex &index) override  
*Default implementation shows the widget iff the index is valid and [checkIndex](#) returns true.*
- void [updatePosition](#) ()
- void [visualChange](#) () override  
*Called when any change from the delegate occurs - when the overlay is installed, when size hints, styles or fonts change.*

## Protected Member Functions inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- bool [checkIndexOnEnter](#) (const QModelIndex &index) const  
*Utility method called from [slotEntered](#).*
- bool [eventFilter](#) (QObject \*obj, QEvent \*event) override
- virtual void [hide](#) ()  
*Called when the widget shall be hidden (mouse cursor left index, viewport, uninstalled etc.).*
- virtual QString [notifyMultipleMessage](#) (const QModelIndex &, int number)
- QWidget \* [parentWidget](#) () const  
*Returns the widget to be used as parent for your widget created in [createWidget\(\)](#)*
- virtual void [viewportLeaveEvent](#) (QObject \*obj, QEvent \*event)  
*Called when a `QEvent::Leave` of the viewport is received.*
- virtual void [widgetEnterEvent](#) ()  
*Called when a `QEvent::Enter` resp.*
- void [widgetEnterNotifyMultiple](#) (const QModelIndex &index)  
*A sample implementation for above methods.*
- virtual void [widgetLeaveEvent](#) ()
- void [widgetLeaveNotifyMultiple](#) ()

## Protected Member Functions inherited from [Digikam::ItemDelegateOverlay](#)

- `QList< QModelIndex > affectedIndexes` (const QModelIndex &index) const
- `bool affectsMultiple` (const QModelIndex &index) const  
*For the context that an overlay can affect multiple items: Assuming the currently overlaid index is given.*
- `int numberOfAffectedIndexes` (const QModelIndex &index) const
- `bool viewHasMultiSelection` () const  
*Utility method.*

## Protected Attributes

- `QPersistentModelIndex m_index`

## Protected Attributes inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- `bool m_mouseButtonPressedOnWidget` = false
- `QWidget * m_widget` = nullptr

## Protected Attributes inherited from [Digikam::ItemDelegateOverlay](#)

- `QAbstractItemDelegate * m_delegate` = nullptr
- `QAbstractItemView * m_view` = nullptr

## Additional Inherited Members

## Signals inherited from [Digikam::ItemDelegateOverlay](#)

- `void hideNotification` ()
- `void requestNotification` (const QModelIndex &index, const QString &message)
- `void update` (const QModelIndex &index)

## Protected Slots inherited from [Digikam::AbstractWidgetDelegateOverlay](#)

- virtual `void slotLayoutChanged` ()
- virtual `void slotReset` ()  
*Default implementations of these three slots call `hide()`*
- virtual `void slotRowsRemoved` (const QModelIndex &parent, int start, int end)
- virtual `void slotViewportEntered` ()

## Protected Slots inherited from [Digikam::ItemDelegateOverlay](#)

### 9.1438.1 Member Function Documentation

#### 9.1438.1.1 `checkIndex()`

```
bool ShowFoto::ShowfotoCoordinatesOverlay::checkIndex (
    const QModelIndex & index ) const [override], [protected], [virtual]
```

The default implementation returns true.

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.1438.1.2 createWidget()

```
QWidget * ShowFoto::ShowfotoCoordinatesOverlay::createWidget ( ) [override], [protected], [virtual]
```

When creating the object, pass [parentWidget\(\)](#) as parent widget. Ownership of the object is passed. It will be deleted in [setActive\(false\)](#).

Implements [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.1438.1.3 setActive()

```
void ShowFoto::ShowfotoCoordinatesOverlay::setActive (
    bool active ) [override], [protected], [virtual]
```

If active is false, this will delete the widget and disconnect all signal from model and view to this object (!)

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

### 9.1438.1.4 slotEntered()

```
void ShowFoto::ShowfotoCoordinatesOverlay::slotEntered (
    const QModelIndex & index ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::AbstractWidgetDelegateOverlay](#).

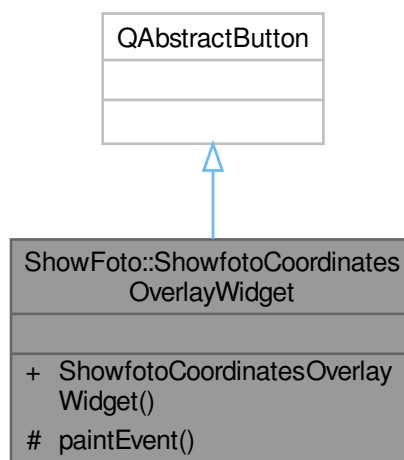
### 9.1438.1.5 visualChange()

```
void ShowFoto::ShowfotoCoordinatesOverlay::visualChange ( ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::ItemDelegateOverlay](#).

## 9.1439 ShowFoto::ShowfotoCoordinatesOverlayWidget Class Reference

Inheritance diagram for ShowFoto::ShowfotoCoordinatesOverlayWidget:





**Public Member Functions**

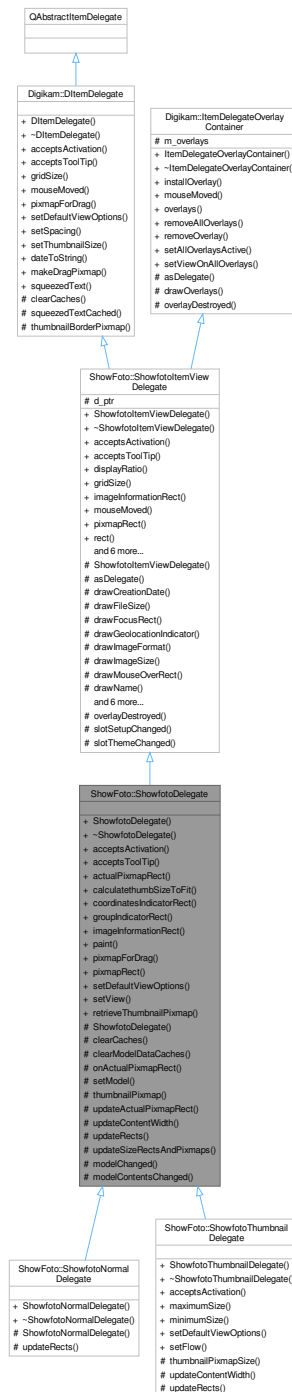
- **ShowfotoCoordinatesOverlayWidget** (QWidget \*const parent=nullptr)

**Protected Member Functions**

- void **paintEvent** (QPaintEvent \*) override

## 9.1440 ShowFoto::ShowfotoDelegate Class Reference

Inheritance diagram for ShowFoto::ShowfotoDelegate:



### Public Member Functions

- **ShowfotoDelegate** (QWidget \*const parent)
- bool **acceptsActivation** (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*activationRect=nullptr) const override

- bool [acceptsToolTip](#) (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*tooltipRect=nullptr) const override  
*These methods take four parameters: The position on viewport, the rect on viewport, the index, and optionally a parameter into which, if the return value is true, a rectangle can be written for which the return value will be true as well.*
- QRect [actualPixmapRect](#) (const QModelIndex &index) const
- int [calculatethumbSizeToFit](#) (int ws)
- QRect [coordinatesIndicatorRect](#) () const
- QRect [groupIndicatorRect](#) () const
- QRect [imageInformationRect](#) () const override  
*Returns the area where the image information is drawn, or null if empty / not supported.*
- void [paint](#) (QPainter \*painter, const QStyleOptionViewItem &option, const QModelIndex &index) const override
- QPixmap [pixmapForDrag](#) (const QStyleOptionViewItem &option, const QList< QModelIndex > &indexes) const override
- QRect [pixmapRect](#) () const override  
*Returns the area where the pixmap is drawn, or null if not supported.*
- void [setDefaultViewOptions](#) (const QStyleOptionViewItem &option) override  
*Style option with standard values to use for cached rendering.*
- void [setView](#) (ShowfotoThumbnailBar \*view)

### Public Member Functions inherited from [ShowFoto::ShowfotoItemViewDelegate](#)

- [ShowfotoItemViewDelegate](#) (QWidget \*const parent)
- bool [acceptsActivation](#) (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*activationRect=nullptr) const override
- bool [acceptsToolTip](#) (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*tooltipRect=nullptr) const override  
*These methods take four parameters: The position on viewport, the rect on viewport, the index, and optionally a parameter into which, if the return value is true, a rectangle can be written for which the return value will be true as well.*
- double [displayRatio](#) () const
- QSize [gridSize](#) () const override  
*Returns the gridsize to be set by the view.*
- void [mouseMoved](#) (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index) override
- QRect [rect](#) () const
- void [setDefaultViewOptions](#) (const QStyleOptionViewItem &option) override  
*Style option with standard values to use for cached rendering.*
- void [setSpacing](#) (int spacing) override
- void [setThumbnailSize](#) (const ThumbnailSize &thumbSize) override  
*reimplemented from DItemDelegate*
- QSize [sizeHint](#) (const QStyleOptionViewItem &option, const QModelIndex &index) const override
- int [spacing](#) () const
- ThumbnailSize [thumbnailSize](#) () const

### Public Member Functions inherited from [Digikam::DItemDelegate](#)

- [DItemDelegate](#) (QObject \*const parent=nullptr)

## Public Member Functions inherited from [Digikam::ItemDelegateOverlayContainer](#)

- [ItemDelegateOverlayContainer](#) ()=default  
*This is a sample implementation for delegate management methods, to be inherited by a delegate.*
- void **installOverlay** ([ItemDelegateOverlay](#) \*overlay)
- void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)
- QList< [ItemDelegateOverlay](#) \* > **overlays** () const
- void **removeAllOverlays** ()
- void **removeOverlay** ([ItemDelegateOverlay](#) \*overlay)
- void **setAllOverlaysActive** (bool active)
- void **setViewOnAllOverlays** (QAbstractItemView \*view)

## Static Public Member Functions

- static QPixmap **retrieveThumbnailPixmap** (const QModelIndex &index, int thumbnailSize)  
*Retrieve the thumbnail pixmap in given size for the [ShowfotoItemModel::ThumbnailRole](#) for the given index from the given index, which must adhere to [ShowfotoThumbnailModel](#) semantics.*

## Static Public Member Functions inherited from [Digikam::DItemDelegate](#)

- static QString **dateToString** (const QDateTime &datetime)
- static QPixmap **makeDragPixmap** (const QStyleOptionViewItem &option, const QList< QModelIndex > &indexes, double displayRatio, const QPixmap &suggestedPixmap=QPixmap())
- static QString **squeezedText** (const QFontMetrics &fm, int width, const QString &text)

## Protected Slots

- void **modelChanged** ()
- void **modelContentsChanged** ()

## Protected Slots inherited from [ShowFoto::ShowfotoItemViewDelegate](#)

- void **overlayDestroyed** (QObject \*o) override
- void **slotSetupChanged** ()
- void **slotThemeChanged** ()

## Protected Member Functions

- **ShowfotoDelegate** (ShowfotoDelegate::ShowfotoDelegatePrivate &dd, QWidget \*const parent)
- void **clearCaches** () override
- virtual void **clearModelDataCaches** ()  
*Reimplement to clear caches based on model indexes (hash on row number etc.) Change signals are listened to this is called whenever such properties become invalid.*
- bool **onActualPixmapRect** (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*actualRect) const
- void **setModel** (QAbstractItemModel \*model)
- virtual QPixmap **thumbnailPixmap** (const QModelIndex &index) const
- void **updateActualPixmapRect** (const QModelIndex &index, const QRect &rect)
- virtual void **updateContentWidth** ()  
*Reimplement this to set contentWidth.*
- virtual void **updateRects** ()=0  
*In a subclass, you need to implement this method to set up the rects for drawing.*
- void **updateSizeRectsAndPixmaps** () override

## Protected Member Functions inherited from [ShowFoto::ShowfotoItemViewDelegate](#)

- **ShowfotoItemViewDelegate** (ShowfotoItemViewDelegatePrivate &dd, QWidget \*const parent)
- QAbstractItemDelegate \* **asDelegate** () override
 

*Returns the delegate, typically, the derived class.*
- void **drawCreationDate** (QPainter \*p, const QRect &dateRect, const QDateTime &date) const
- void **drawFileSize** (QPainter \*p, const QRect &r, qlonglong bytes) const
- void **drawFocusRect** (QPainter \*p, const QStyleOptionViewItem &option, bool isSelected) const
- void **drawGeolocationIndicator** (QPainter \*p, const QRect &r) const
- void **drawImageFormat** (QPainter \*p, const QRect &dimsRect, const QString &mime) const
- void **drawImageSize** (QPainter \*p, const QRect &dimsRect, const QSize &dims) const
- void **drawMouseOverRect** (QPainter \*p, const QStyleOptionViewItem &option) const
- void **drawName** (QPainter \*p, const QRect &nameRect, const QString &name) const
- QRect **drawThumbnail** (QPainter \*p, const QRect &thumbRect, const QPixmap &background, const QPixmap &thumbnail) const
 

*Use the tool methods for painting in subclasses.*
- virtual void **invalidatePaintingCache** ()
 

*reimplement these in subclasses*
- void **prepareBackground** ()
- void **prepareFonts** ()
- void **prepareMetrics** (int maxWidth)

## Protected Member Functions inherited from [Digikam::DItemDelegate](#)

- QString **squeezedTextCached** (QPainter \*const p, int width, const QString &text) const
- QPixmap **thumbnailBorderPixmap** (const QSize &pixSize, bool isGrouped=false) const

## Protected Member Functions inherited from [Digikam::ItemDelegateOverlayContainer](#)

- virtual void **drawOverlays** (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index) const
- virtual void **overlayDestroyed** (QObject \*o)
 

*Declare as slot in the derived class calling this method.*

## Additional Inherited Members

## Signals inherited from [ShowFoto::ShowfotoItemViewDelegate](#)

- void **hideNotification** ()
- void **requestNotification** (const QModelIndex &index, const QString &message)

## Signals inherited from [Digikam::DItemDelegate](#)

- void **gridSizeChanged** (const QSize &newSize)
- void **visualChange** ()

## Protected Attributes inherited from [ShowFoto::ShowfotoItemViewDelegate](#)

- ShowfotoItemViewDelegatePrivate \*const **d\_ptr** = nullptr

## Protected Attributes inherited from [Digikam::ItemDelegateOverlayContainer](#)

- `QList< ItemDelegateOverlay * > m_overlays`

### 9.1440.1 Member Function Documentation

#### 9.1440.1.1 `acceptsActivation()`

```
bool ShowFoto::ShowfotoDelegate::acceptsActivation (
    const QPoint & pos,
    const QRect & visualRect,
    const QModelIndex & index,
    QRect * activationRect = nullptr ) const [override], [virtual]
```

Implements [Digikam::DItemDelegate](#).

#### 9.1440.1.2 `acceptsToolTip()`

```
bool ShowFoto::ShowfotoDelegate::acceptsToolTip (
    const QPoint & pos,
    const QRect & visualRect,
    const QModelIndex & index,
    QRect * tooltipRect = nullptr ) const [override], [virtual]
```

Implements [Digikam::DItemDelegate](#).

#### 9.1440.1.3 `clearCaches()`

```
void ShowFoto::ShowfotoDelegate::clearCaches ( ) [override], [protected], [virtual]
```

Reimplemented from [Digikam::DItemDelegate](#).

#### 9.1440.1.4 `imageInformationRect()`

```
QRect ShowFoto::ShowfotoDelegate::imageInformationRect ( ) const [override], [virtual]
```

The image information is textual or graphical information, but not the pixmap. The `ratingRect()` will e.g. typically be contained in this area.

Reimplemented from [ShowFoto::ShowfotoItemViewDelegate](#).

#### 9.1440.1.5 `pixmapForDrag()`

```
QPixmap ShowFoto::ShowfotoDelegate::pixmapForDrag (
    const QStyleOptionViewItem & option,
    const QList< QModelIndex > & indexes ) const [override], [virtual]
```

Implements [Digikam::DItemDelegate](#).

### 9.1440.1.6 pixmapRect()

```
QRect ShowFoto::ShowfotoDelegate::pixmapRect ( ) const [override], [virtual]
```

Reimplemented from [ShowFoto::ShowfotoItemViewDelegate](#).

### 9.1440.1.7 setDefaultViewOptions()

```
void ShowFoto::ShowfotoDelegate::setDefaultViewOptions (
    const QStyleOptionViewItem & option ) [override], [virtual]
```

option.rect shall be the viewport rectangle. Call on resize, font change.

Implements [Digikam::DItemDelegate](#).

Reimplemented in [ShowFoto::ShowfotoThumbnailDelegate](#).

### 9.1440.1.8 updateContentWidth()

```
void ShowFoto::ShowfotoDelegate::updateContentWidth ( ) [protected], [virtual]
```

This is the maximum width of all content rectangles, typically excluding margins on both sides.

Reimplemented in [ShowFoto::ShowfotoThumbnailDelegate](#).

### 9.1440.1.9 updateRects()

```
virtual void ShowFoto::ShowfotoDelegate::updateRects ( ) [protected], [pure virtual]
```

The paint() method operates depending on these rects.

Implemented in [ShowFoto::ShowfotoThumbnailDelegate](#), and [ShowFoto::ShowfotoNormalDelegate](#).

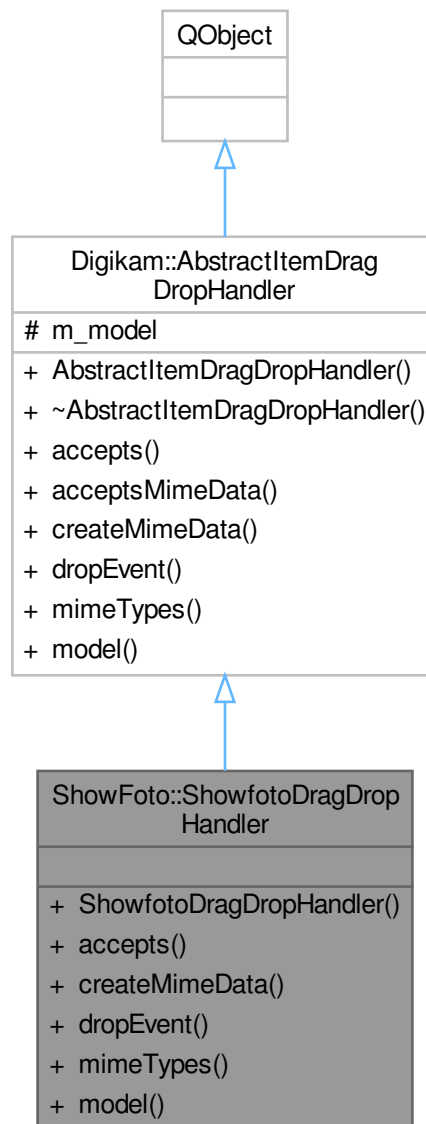
### 9.1440.1.10 updateSizeRectsAndPixmapes()

```
void ShowFoto::ShowfotoDelegate::updateSizeRectsAndPixmapes ( ) [override], [protected], [virtual]
```

Implements [ShowFoto::ShowfotoItemViewDelegate](#).

## 9.1441 ShowFoto::ShowfotoDragDropHandler Class Reference

Inheritance diagram for ShowFoto::ShowfotoDragDropHandler:



### Signals

- void **signalDroppedUrls** (const QList< QUrl > &droppedUrls, bool dropped, const QUrl &current)

### Public Member Functions

- **ShowfotoDragDropHandler** ([ShowfotoItemModel](#) \*const model)



- Qt::DropAction [accepts](#) (const QDropEvent \*e, const QModelIndex &dropIndex) override  
*Returns if the given mime data is accepted for drop on dropIndex.*
- QMimeData \* [createMimeData](#) (const QList< QModelIndex > &) override  
*Create a mime data object for starting a drag from the given Albums.*
- bool [dropEvent](#) (QAbstractItemView \*view, const QDropEvent \*e, const QModelIndex &droppedOn) override  
*Gives the view and the occurring drop event.*
- QStringList [mimeTypes](#) () const override  
*Returns the supported mime types.*
- [ShowfotoItemModel](#) \* **model** () const

## Public Member Functions inherited from [Digikam::AbstractItemDragDropHandler](#)

- [AbstractItemDragDropHandler](#) (QAbstractItemModel \*const model)
- virtual bool [acceptsMimeData](#) (const QMimeData \*data)  
*Returns if the given mime data can be handled.*
- QAbstractItemModel \* **model** () const

## Additional Inherited Members

## Protected Attributes inherited from [Digikam::AbstractItemDragDropHandler](#)

- QAbstractItemModel \* **m\_model** = nullptr

## 9.1441.1 Member Function Documentation

### 9.1441.1.1 [accepts\(\)](#)

```
Qt::DropAction ShowFoto::ShowfotoDragDropHandler::accepts (
    const QDropEvent * e,
    const QModelIndex & dropIndex ) [override], [virtual]
```

Returns the proposed action, or Qt::IgnoreAction if not accepted.

Reimplemented from [Digikam::AbstractItemDragDropHandler](#).

### 9.1441.1.2 [createMimeData\(\)](#)

```
QMimeData * ShowFoto::ShowfotoDragDropHandler::createMimeData (
    const QList< QModelIndex > & ) [override], [virtual]
```

Reimplemented from [Digikam::AbstractItemDragDropHandler](#).

### 9.1441.1.3 [dropEvent\(\)](#)

```
bool ShowFoto::ShowfotoDragDropHandler::dropEvent (
    QAbstractItemView * view,
    const QDropEvent * e,
    const QModelIndex & droppedOn ) [override], [virtual]
```

The index is the index where the drop was dropped on. It may be invalid (dropped on decoration, viewport) Returns true if the event is to be accepted.

Reimplemented from [Digikam::AbstractItemDragDropHandler](#).

#### 9.1441.1.4 mimeTypees()

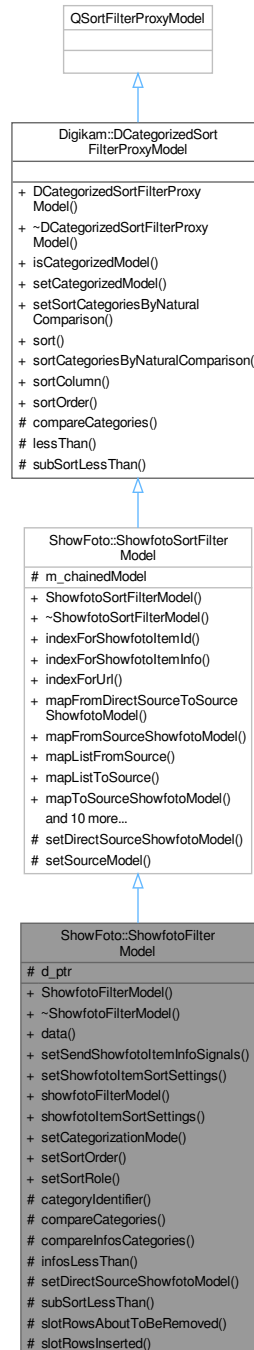
```
QStringList ShowFoto::ShowfotoDragDropHandler::mimeTypees ( ) const [override], [virtual]
```

Called by the default implementation of model's [mimeTypees\(\)](#).

Reimplemented from [Digikam::AbstractItemDragDropHandler](#).

## 9.1442 ShowFoto::ShowfotoFilterModel Class Reference

Inheritance diagram for ShowFoto::ShowfotoFilterModel:



### Public Types

- enum [ShowfotoFilterModelRoles](#) { [CategorizationModeRole](#) = ShowfotoItemModel::FilterModelRoles + 1 , [SortOrderRole](#) = ShowfotoItemModel::FilterModelRoles + 2 , [CategoryFormatRole](#) = ShowfotoItemModel::FilterModelRoles + 3 , [ShowfotoFilterModelPointerRole](#) = ShowfotoItemModel::FilterModelRoles + 50 }

## Public Types inherited from [Digikam::DCategorizedSortFilterProxyModel](#)

- enum [AdditionalRoles](#) { [CategoryDisplayRole](#) = 0x17CE990A , [CategorySortRole](#) = 0x27857E60 }

## Public Slots

- void **setCategorizationMode** ([ShowfotoItemSortSettings::CategorizationMode](#) mode)
- void **setSortOrder** ([ShowfotoItemSortSettings::SortOrder](#) order)
- void **setSortRole** ([ShowfotoItemSortSettings::SortRole](#) role)

## Signals

- void **showfotoItemInfosAboutToBeRemoved** (const QList< [ShowfotoItemInfo](#) > &infos)
- void **showfotoItemInfosAdded** (const QList< [ShowfotoItemInfo](#) > &infos)

*These signals need to be explicitly enabled with `setSendItemInfoSignals()`.*

## Public Member Functions

- ShowfotoFilterModel** (QObject \*const parent=nullptr)
- QVariant **data** (const QModelIndex &index, int role=Qt::DisplayRole) const override
- void **setSendShowfotoItemInfoSignals** (bool sendSignals)
 

*Enables sending `ShowfotoItemInfosAdded` and `ShowfotoItemInfosAboutToBeRemoved`.*
- void **setShowfotoItemSortSettings** (const [ShowfotoItemSortSettings](#) &sorter)
- [ShowfotoFilterModel](#) \* **showfotoFilterModel** () const override
 

*Returns this, any chained `ShowfotoFilterModel`, or 0.*
- [ShowfotoItemSortSettings](#) **showfotoItemSortSettings** () const

## Public Member Functions inherited from [ShowFoto::ShowfotoSortFilterModel](#)

- ShowfotoSortFilterModel** (QObject \*const parent=nullptr)
- QModelIndex **indexForShowfotoItemid** (qulonglong id) const
- QModelIndex **indexForShowfotoItemInfo** (const [ShowfotoItemInfo](#) &info) const
- QModelIndex **indexForUrl** (const QUrl &fileUrl) const
- QModelIndex **mapFromDirectSourceToSourceShowfotoModel** (const QModelIndex &sourceModelIndex) const
- QModelIndex **mapFromSourceShowfotoModel** (const QModelIndex &showfotoModelIndex) const
- QList< QModelIndex > **mapListFromSource** (const QList< QModelIndex > &sourceIndexes) const
- QList< QModelIndex > **mapListToSource** (const QList< QModelIndex > &indexes) const
- QModelIndex **mapToSourceShowfotoModel** (const QModelIndex &proxyIndex) const
 

*Convenience methods mapped to `ShowfotoItemModel`.*
- void **setSourceFilterModel** ([ShowfotoSortFilterModel](#) \*const sourceModel)
- void **setSourceShowfotoModel** ([ShowfotoItemModel](#) \*const sourceModel)
- qulonglong **showfotoItemid** (const QModelIndex &index) const
- QList< qulonglong > **showfotoItemids** (const QList< QModelIndex > &indexes) const
- [ShowfotoItemInfo](#) **showfotoItemInfo** (const QModelIndex &index) const
- QList< [ShowfotoItemInfo](#) > **showfotoItemInfos** (const QList< QModelIndex > &indexes) const
- QList< [ShowfotoItemInfo](#) > **showfotoItemInfosSorted** () const
 

*Returns a list of all showfoto infos, sorted according to this model.*
- [ShowfotoSortFilterModel](#) \* **sourceFilterModel** () const
- [ShowfotoItemModel](#) \* **sourceShowfotoModel** () const

## Public Member Functions inherited from [Digikam::DCategorizedSortFilterProxyModel](#)

- **DCategorizedSortFilterProxyModel** (QObject \*const parent=nullptr)
- bool **isCategorizedModel** () const
- void **setCategorizedModel** (bool categorizedModel)
 

*Enables or disables the categorization feature.*
- void **setSortCategoriesByNaturalComparison** (bool [sortCategoriesByNaturalComparison](#))
 

*Set if the sorting using CategorySortRole will use a natural comparison in the case that strings were returned.*
- void **sort** (int column, Qt::SortOrder order=Qt::AscendingOrder) override
 

*Overridden from QSortFilterProxyModel.*
- bool **sortCategoriesByNaturalComparison** () const
- int **sortColumn** () const
- Qt::SortOrder **sortOrder** () const

## Protected Slots

- void **slotRowsAboutToBeRemoved** (const QModelIndex &parent, int start, int end)
- void **slotRowsInserted** (const QModelIndex &parent, int start, int end)

## Protected Member Functions

- virtual QString **categoryIdentifier** (const [ShowfotoItemInfo](#) &info) const
 

*Returns a unique identifier for the category if info.*
- int **compareCategories** (const QModelIndex &left, const QModelIndex &right) const override
 

*This method compares the category of the left index with the category of the right index.*
- virtual int **compareInfosCategories** (const [ShowfotoItemInfo](#) &left, const [ShowfotoItemInfo](#) &right) const
 

*Reimplement to customize category sorting. Return negative if category of left < category right, Return 0 if left and right are in the same category, else return positive.*
- virtual bool **infosLessThan** (const [ShowfotoItemInfo](#) &left, const [ShowfotoItemInfo](#) &right) const
 

*Reimplement to customize sorting.*
- void **setDirectSourceShowfotoModel** ([ShowfotoItemModel](#) \*const sourceModel) override
 

*Reimplement if needed. Called only when model shall be set as (direct) sourceModel.*
- bool **subSortLessThan** (const QModelIndex &left, const QModelIndex &right) const override
 

*This method has a similar purpose as [lessThan\(\)](#) has on QSortFilterProxyModel.*

## Protected Member Functions inherited from [ShowFoto::ShowfotoSortFilterModel](#)

- void **setSourceModel** (QAbstractItemModel \*sourceModel) override

## Protected Member Functions inherited from [Digikam::DCategorizedSortFilterProxyModel](#)

- bool **lessThan** (const QModelIndex &left, const QModelIndex &right) const override
 

*Overridden from QSortFilterProxyModel.*

## Protected Attributes

- ShowfotoFilterModelPrivate \*const **d\_ptr** = nullptr

## Protected Attributes inherited from [ShowFoto::ShowfotoSortFilterModel](#)

- [ShowfotoSortFilterModel](#) \* `m_chainedModel` = nullptr

## 9.1442.1 Member Enumeration Documentation

### 9.1442.1.1 ShowfotoFilterModelRoles

enum [ShowFoto::ShowfotoFilterModel::ShowfotoFilterModelRoles](#)

## Enumerator

CategorizationModeRole	Returns the current categorization mode.
SortOrderRole	Returns the current sort order.
CategoryFormatRole	Returns the format of the index which is used for category.
ShowfotoFilterModelPointerRole	Returns true if the given showfoto item is a group leader, and the group is opened. TODO: GroupsOpenRole = ShowfotoItemModel::FilterModelRoles + 4

## 9.1442.2 Member Function Documentation

### 9.1442.2.1 categoryIdentifier()

```
QString ShowFoto::ShowfotoFilterModel::categoryIdentifier (
    const ShowfotoItemInfo & info ) const [protected], [virtual]
```

The string need not be for user display.

### 9.1442.2.2 compareCategories()

```
int ShowFoto::ShowfotoFilterModel::compareCategories (
    const QModelIndex & left,
    const QModelIndex & right ) const [override], [protected], [virtual]
```

Internally and if not reimplemented, this method will ask for `left` and `right` models for role `CategorySortRole`. In order to correctly sort categories, the `data()` method of the model should return a `qulonglong` (or numeric) value, or a `QString` object. `QString` objects will be sorted with `QString::localeAwareCompare` if `sortCategoriesByNaturalComparison()` is true.

#### Note

Please have present that: `QString(QChar(QChar::ObjectReplacementCharacter)) > QString(QChar(QChar::ReplacementCharacter)) > [ all possible strings ] > QString()`

This means that `QString()` will be sorted the first one, while `QString(QChar(QChar::ObjectReplacementCharacter))` and `QString(QChar(QChar::ReplacementCharacter))` will be sorted in last position.

#### Warning

Please note that `data()` method of the model should return always information of the same type. If you return a `QString` for an index, you should return always `QStrings` for all indexes for role `CategorySortRole` in order to correctly sort categories. You can't mix by returning a `QString` for one index, and a `qulonglong` for other.

#### Note

If you need a more complex layout, you will have to reimplement this method.

#### Returns

A negative value if the category of `left` should be placed before the category of `right`. 0 if `left` and `right` are on the same category, and a positive value if the category of `left` should be placed after the category of `right`.

Reimplemented from [Digikam::DCategorizedSortFilterProxyModel](#).

### 9.1442.2.3 infosLessThan()

```
bool ShowFoto::ShowfotoFilterModel::infosLessThan (
    const ShowfotoItemInfo & left,
    const ShowfotoItemInfo & right ) const [protected], [virtual]
```

Do not take categories into account here.

### 9.1442.2.4 setDirectSourceShowfotoModel()

```
void ShowFoto::ShowfotoFilterModel::setDirectSourceShowfotoModel (
    ShowfotoItemModel *const sourceModel ) [override], [protected], [virtual]
```

Reimplemented from [ShowFoto::ShowfotoSortFilterModel](#).

### 9.1442.2.5 showfotoFilterModel()

```
ShowfotoFilterModel * ShowFoto::ShowfotoFilterModel::showfotoFilterModel ( ) const [override],
[virtual]
```

Reimplemented from [ShowFoto::ShowfotoSortFilterModel](#).

### 9.1442.2.6 subSortLessThan()

```
bool ShowFoto::ShowfotoFilterModel::subSortLessThan (
    const QModelIndex & left,
    const QModelIndex & right ) const [override], [protected], [virtual]
```

It is used for sorting items that are in the same category.

#### Returns

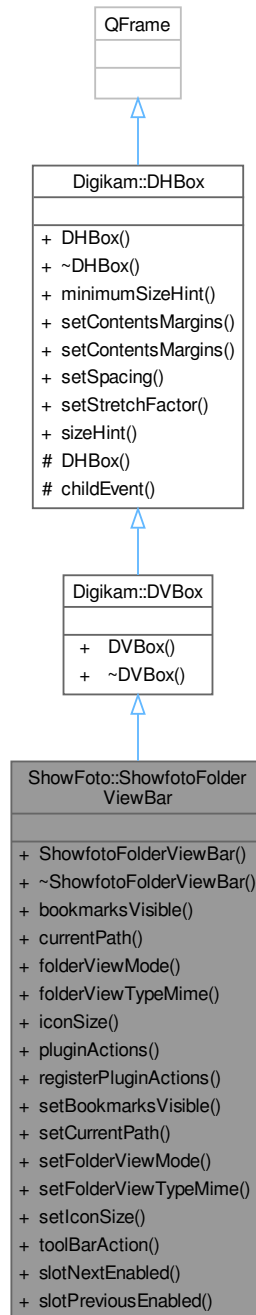
Returns true if the item `left` is less than the item `right` when sorting.

Reimplemented from [Digikam::DCategorizedSortFilterProxyModel](#).



## 9.1443 ShowFoto::ShowfotoFolderViewBar Class Reference

Inheritance diagram for ShowFoto::ShowfotoFolderViewBar:



### Public Types

- enum `FolderViewTypeMime` {  
`TYPE_MIME_JPEG = 0`, `TYPE_MIME_TIFF`, `TYPE_MIME_PNG`, `TYPE_MIME_PGF`,  
`TYPE_MIME_HEIF`, `TYPE_MIME_AVIF`, `TYPE_MIME_JXL`, `TYPE_MIME_WEBP`,  
`TYPE_MIME_DNG`, `TYPE_MIME_RAW`, `TYPE_MIME_NORAW`, `TYPE_MIME_ALL` }

## Public Slots

- void **slotNextEnabled** (bool)
- void **slotPreviousEnabled** (bool)

## Signals

- void **signalAppendContents** ()
- void **signalCustomPathChanged** (const QString &)
- void **signalGoHome** ()
- void **signalGoNext** ()
- void **signalGoPrevious** ()
- void **signalGoUp** ()
- void **signalIconSizeChanged** (int)
- void **signalLoadContents** ()
- void **signalPluginActionTriggered** (QAction \*)
- void **signalSetup** ()
- void **signalShowBookmarks** (bool)
- void **signalTypeMimesChanged** (const QString &)
- void **signalViewModeChanged** (int)

## Public Member Functions

- **ShowfotoFolderViewBar** ([ShowfotoFolderViewSideBar](#) \*const parent)
- bool **bookmarksVisible** () const
- QString **currentPath** () const
- int **folderViewMode** () const
- int **folderViewTypeMime** () const
- int **iconSize** () const
- QList< QAction \* > **pluginActions** () const
- void **registerPluginActions** (const QList< [DPluginAction](#) \* > &actions)
- void **setBookmarksVisible** (bool b)
- void **setCurrentPath** (const QString &path)
- void **setFolderViewMode** (int mode)
- void **setFolderViewTypeMime** (int mime)
- void **setIconSize** (int size)
- QAction \* **toolbarAction** (const QString &name) const

## Public Member Functions inherited from [Digikam::DVBox](#)

- **DVBox** (QWidget \*const parent=nullptr)

## Public Member Functions inherited from [Digikam::DHBox](#)

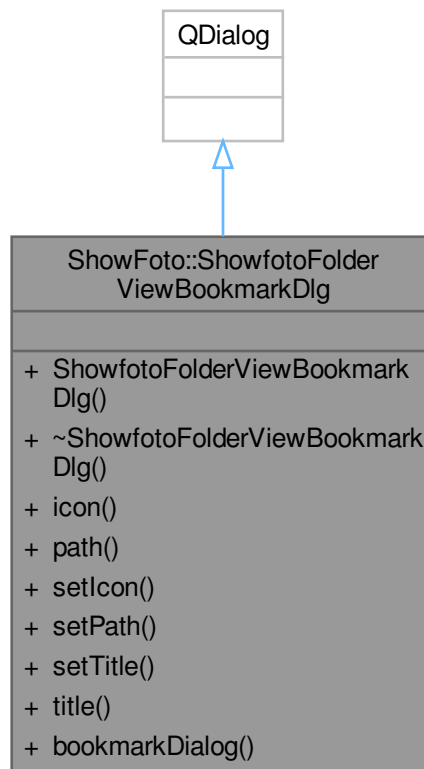
- **DHBox** (QWidget \*const parent=nullptr)
- QSize **minimumSizeHint** () const override
- void **setContentsMargins** (const QMargins &margins)
- void **setContentsMargins** (int left, int top, int right, int bottom)
- void **setSpacing** (int space)
- void **setStretchFactor** (QWidget \*const widget, int stretch)
- QSize **sizeHint** () const override

**Additional Inherited Members****Protected Member Functions inherited from [Digikam::DHBox](#)**

- **DHBox** (bool vertical, QWidget \*const parent)
- void **childEvent** (QChildEvent \*e) override

**9.1444 ShowFoto::ShowfotoFolderViewBookmarkDlg Class Reference**

Inheritance diagram for ShowFoto::ShowfotoFolderViewBookmarkDlg:

**Public Member Functions**

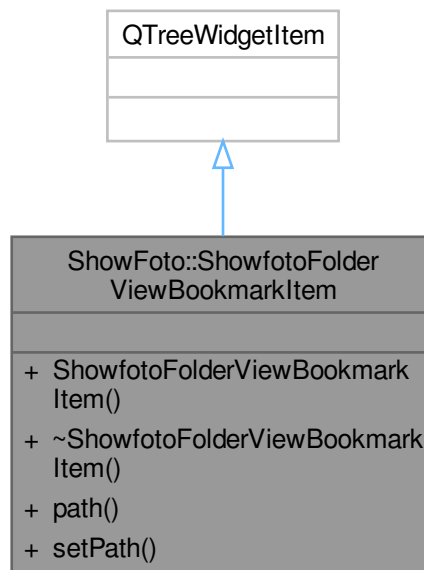
- **ShowfotoFolderViewBookmarkDlg** ([ShowfotoFolderViewBookmarkList](#) \*const parent, bool create=false)
- QString **icon** () const
- QString **path** () const
- void **setIcon** (const QString &icon)
- void **setPath** (const QString &path)
- void **setTitle** (const QString &title)
- QString **title** () const

### Static Public Member Functions

- static bool **bookmarkDialog** ([ShowfotoFolderViewBookmarkList](#) \*const parent, QString &title, QString &icon, QString &path, bool create=false)

## 9.1445 ShowFoto::ShowfotoFolderViewBookmarkItem Class Reference

Inheritance diagram for ShowFoto::ShowfotoFolderViewBookmarkItem:

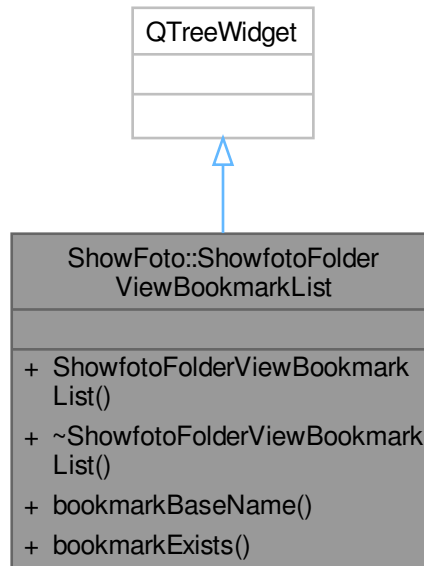


### Public Member Functions

- **ShowfotoFolderViewBookmarkItem** (`QTreeWidgetItem` \*const parent)
- QString **path** () const
- void **setPath** (const QString &)

## 9.1446 ShowFoto::ShowfotoFolderViewBookmarkList Class Reference

Inheritance diagram for ShowFoto::ShowfotoFolderViewBookmarkList:



### Signals

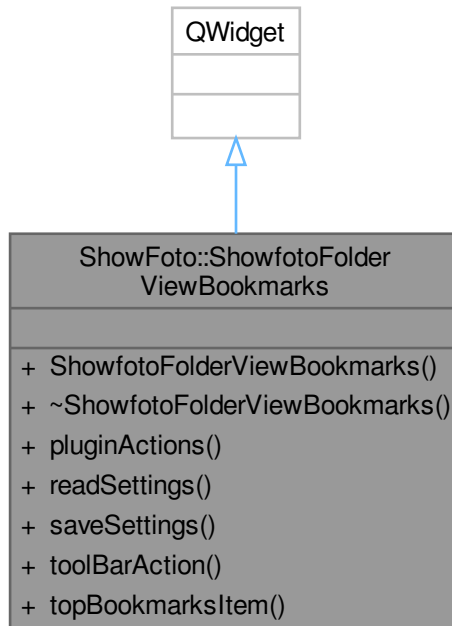
- void **signalAddBookmark** (const QString &path)
- void **signalLoadContents** (const QString &path)

### Public Member Functions

- **ShowfotoFolderViewBookmarkList** ([ShowfotoFolderViewBookmarks](#) \*const parent)
- QString **bookmarkBaseName** (const QString &path) const
- [ShowfotoFolderViewBookmarkItem](#) \* **bookmarkExists** (const QString &path) const

## 9.1447 ShowFoto::ShowfotoFolderViewBookmarks Class Reference

Inheritance diagram for ShowFoto::ShowfotoFolderViewBookmarks:



### Signals

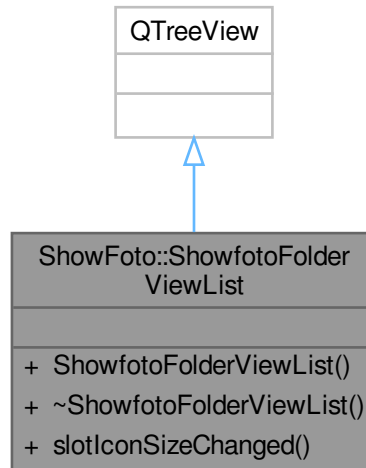
- void **signalLoadContents** ()

### Public Member Functions

- **ShowfotoFolderViewBookmarks** ([ShowfotoFolderViewSideBar](#) \*const sidebar)
- `QList< QAction * >` **pluginActions** () const
- void **readSettings** (const `KConfigGroup` &)
- void **saveSettings** (`KConfigGroup` &)
- `QAction *` **toolBarAction** (const `QString` &name) const
- `QTreeWidgetItem *` **topBookmarksItem** () const

## 9.1448 ShowFoto::ShowfotoFolderViewList Class Reference

Inheritance diagram for ShowFoto::ShowfotoFolderViewList:



### Public Types

- enum **FolderViewMode** { **ShortView** = 0 , **DetailedView** }
- enum **FolderViewRole** { **FileName** = 0 , **FileSize** , **FileType** , **FileDate** }

### Public Slots

- void **slotIconSizeChanged** (int)

### Signals

- void **signalAddBookmark** ()

### Public Member Functions

- **ShowfotoFolderViewList** ([ShowfotoFolderViewSideBar](#) \*const view, [ShowfotoFolderViewBar](#) \*const bar)

## 9.1448.1 Member Enumeration Documentation

### 9.1448.1.1 FolderViewRole

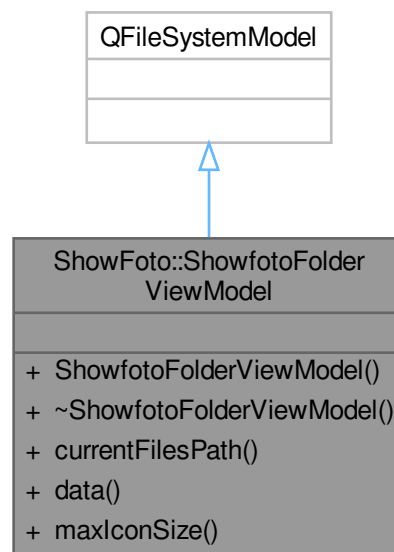
```
enum ShowFoto::ShowfotoFolderViewList::FolderViewRole
```

## Enumerator

FileDate	Modifier date.
----------	----------------

## 9.1449 ShowFoto::ShowfotoFolderViewModel Class Reference

Inheritance diagram for ShowFoto::ShowfotoFolderViewModel:



### Public Member Functions

- **ShowfotoFolderViewModel** ([ShowfotoFolderViewList](#) \*const view)
- `QStringList` **currentFilePath** () const  
*List all file paths from the current model root index selected in the view.*
- `QVariant` **data** (const `QModelIndex` &index, int role) const override

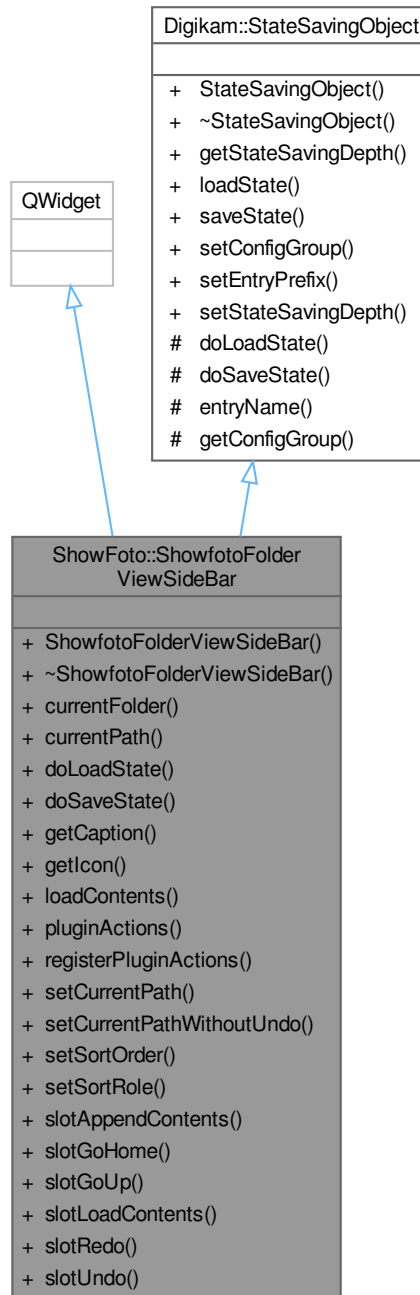
### Static Public Member Functions

- static int **maxIconSize** ()



## 9.1450 ShowFoto::ShowfotoFolderViewSideBar Class Reference

Inheritance diagram for ShowFoto::ShowfotoFolderViewSideBar:



### Public Slots

- void **slotAppendContents** ()
- void **slotGoHome** ()

- void **slotGoUp** ()
- void **slotLoadContents** ()
- void **slotRedo** ()
- void **slotUndo** ()

## Signals

- void **signalAddBookmark** ()
- void **signalAppendContentsFromFiles** (const QStringList &files, const QString &current)
- void **signalLoadContentsFromFiles** (const QStringList &files, const QString &current)
- void **signalLoadContentsFromPath** (const QString &path)
- void **signalSetup** ()

## Public Member Functions

- **ShowfotoFolderViewSideBar** ([Showfoto](#) \*const parent)
- QString **currentFolder** () const
- QString **currentPath** () const
- void **doLoadState** () override  
*Implement this hook method for state loading.*
- void **doSaveState** () override  
*Implement this hook method for state saving.*
- const QString **getCaption** ()
- const QIcon **getIcon** ()
- void **loadContents** (const QModelIndex &index, bool append=false)
- QList< QAction \* > **pluginActions** () const
- void **registerPluginActions** (const QList< [DPluginAction](#) \* > &actions)
- void **setCurrentPath** (const QString &newPathNative)
- void **setCurrentPathWithoutUndo** (const QString &newPath)
- void **setSortOrder** (int order)
- void **setSortRole** (int role)

## Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (QObject \*const host)  
*Constructor.*
- virtual ~[StateSavingObject](#) ()  
*Destructor.*
- [StateSavingDepth](#) **getStateSavingDepth** () const  
*Returns the depth used for state saving or loading.*
- void **loadState** ()  
*Invokes loading the class' state.*
- void **saveState** ()  
*Invokes saving the class' state.*
- virtual void **setConfigGroup** (const KConfigGroup &group)  
*Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void **setEntryPrefix** (const QString &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void **setStateSavingDepth** (const [StateSavingDepth](#) depth)  
*Sets the depth used for state saving or loading.*

## Additional Inherited Members

### Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { [INSTANCE](#) , [DIRECT\\_CHILDREN](#) , [RECURSIVE](#) }

*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

### Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- QString [entryName](#) (const QString &base) const  
*Always use this method to create config group entry names.*
- KConfigGroup [getConfigGroup](#) () const  
*Returns the config group that must be used for state saving and loading.*

## 9.1450.1 Member Function Documentation

### 9.1450.1.1 doLoadState()

```
void ShowFoto::ShowfotoFolderViewSideBar::doLoadState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

### 9.1450.1.2 doSaveState()

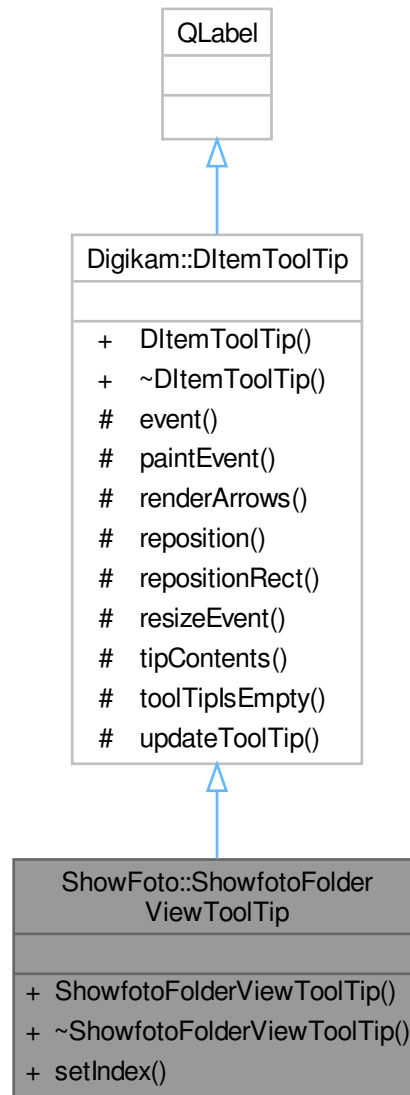
```
void ShowFoto::ShowfotoFolderViewSideBar::doSaveState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

## 9.1451 ShowFoto::ShowfotoFolderViewToolTip Class Reference

Inheritance diagram for ShowFoto::ShowfotoFolderViewToolTip:



### Public Member Functions

- **ShowfotoFolderViewToolTip** ([ShowfotoFolderViewList](#) \*const view)
- void **setIndex** (const QModelIndex &index)

### Public Member Functions inherited from [Digikam::DItemToolTip](#)

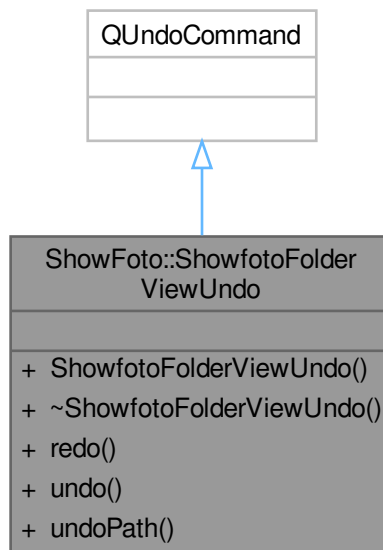
- **DItemToolTip** (QWidget \*const parent=nullptr)

**Additional Inherited Members****Protected Member Functions inherited from [Digikam::DItemToolTip](#)**

- bool **event** (QEvent \*) override
- void **paintEvent** (QPaintEvent \*) override
- void **renderArrows** ()
- void **reposition** ()
- void **resizeEvent** (QResizeEvent \*) override
- bool **toolTipsEmpty** () const
- void **updateToolTip** ()

**9.1452 ShowFoto::ShowfotoFolderViewUndo Class Reference**

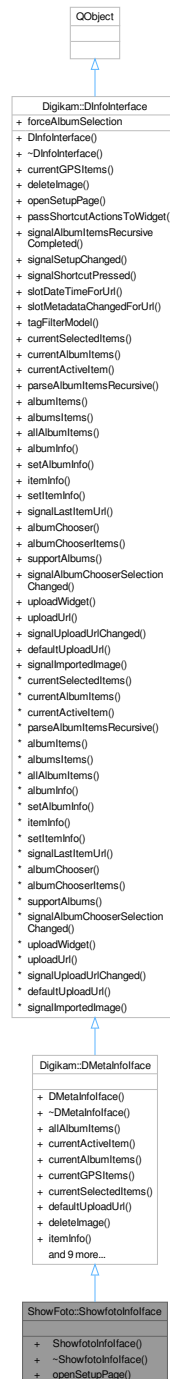
Inheritance diagram for ShowFoto::ShowfotoFolderViewUndo:

**Public Member Functions**

- **ShowfotoFolderViewUndo** ([ShowfotoFolderViewSideBar](#) \*const view, const QString &newPath)
- void **redo** ()
- void **undo** ()
- QString **undoPath** () const

## 9.1453 ShowFoto::ShowfotoInfolface Class Reference

Inheritance diagram for ShowFoto::ShowfotoInfolface:



### Public Member Functions

- **ShowfotoInfolface** (`QObject *const parent, const QList< QUrl > &lst, const QUrl &currentActive`)
- void **openSetupPage** (SetupPage page) override  
*Open configuration dialog page.*

## Public Member Functions inherited from Digikam::DMetalInterface

- **DMetalInterface** (QObject \*const parent, const QList< QUrl > &lst, const QUrl &currentActive)
- QList< QUrl > [allAlbumItems](#) () const override
- QUrl [currentActiveItem](#) () const override
- QList< QUrl > [currentAlbumItems](#) () const override
- QList< [GPSItemContainer](#) \* > [currentGPSItems](#) () const override
- QList< QUrl > [currentSelectedItems](#) () const override
- *Low level items and albums methods.*
- QUrl [defaultUploadUrl](#) () const override
- *Url to upload new items without to use album selector.*
- void [deleteImage](#) (const QUrl &url) override
- *Manipulate with item.*
- [DInfoMap](#) [itemInfo](#) (const QUrl &) const override
- void [parseAlbumItemsRecursive](#) () override
- void [setItemInfo](#) (const QUrl &, const [DInfoMap](#) &) override
- Q\_SIGNAL void [signalItemChanged](#) (const QUrl &url)
- Q\_SIGNAL void [signalRemoveImageFromAlbum](#) (const QUrl &)
- Q\_SLOT void [slotDateTimeForUrl](#) (const QUrl &url, const QDateTime &dt, bool updModDate) override
- *Slot to call when date time stamp from item is changed.*
- Q\_SLOT void [slotMetadataChangedForUrl](#) (const QUrl &url) override
- *Slot to call when something in metadata from item is changed.*
- bool [supportAlbums](#) () const override
- QUrl [uploadUrl](#) () const override
- QWidget \* [uploadWidget](#) (QWidget \*const parent) const override
- *Album selector view methods (to upload items from an external place).*

## Public Member Functions inherited from Digikam::DInfoInterface

- **DInfoInterface** (QObject \*const parent)
- virtual QMap< QString, QString > [passShortcutActionsToWidget](#) (QWidget \*const) const
- *Pass extra shortcut actions to widget and return prefixes of shortcuts.*
- Q\_SIGNAL void [signalAlbumItemsRecursiveCompleted](#) (const QList< QUrl > &imageList)
- Q\_SIGNAL void [signalSetupChanged](#) ()
- Q\_SIGNAL void [signalShortcutPressed](#) (const QString &shortcut, int val)
- virtual QAbstractItemModel \* [tagFilterModel](#) ()
- *Return an instance of tag filter model if host application support this feature, else null pointer.*
- virtual QList< QUrl > [albumItems](#) (int) const
- virtual QList< QUrl > [albumItems](#) (const [DAlbumIDs](#) &) const
- virtual [DInfoMap](#) [albumInfo](#) (int) const
- virtual void [setAlbumInfo](#) (int, const [DInfoMap](#) &) const
- Q\_SIGNAL void [signalLastItemUrl](#) (const QUrl &)
- virtual QWidget \* [albumChooser](#) (QWidget \*const parent) const
- *Albums chooser view methods (to use items from albums before to process).*
- virtual [DAlbumIDs](#) [albumChooserItems](#) () const
- Q\_SIGNAL void [signalAlbumChooserSelectionChanged](#) ()
- Q\_SIGNAL void [signalUploadUrlChanged](#) ()
- Q\_SIGNAL void [signalImportedImage](#) (const QUrl &)

## Additional Inherited Members

### Public Types inherited from [Digikam::DInfoInterface](#)

- typedef QList< int > **DAlbumIDs**  
*List of [Album](#) ids.*
- typedef QMap< QString, QVariant > **DInfoMap**  
*Map of properties name and value.*
- enum **SetupPage** { **ExifToolPage** = 0 , **ImageQualityPage** }

### Public Attributes inherited from [Digikam::DInfoInterface](#)

- bool **forceAlbumSelection** = false

## 9.1453.1 Member Function Documentation

### 9.1453.1.1 openSetupPage()

```
void ShowFoto::ShowfotoInfoIface::openSetupPage (
    SetupPage page ) [override], [virtual]
```

Reimplemented from [Digikam::DInfoInterface](#).

## 9.1454 ShowFoto::ShowfotoItemInfo Class Reference

### Public Member Functions

- bool **isNull** () const  
*Return true if all member in this container are null.*
- bool **operator!=** (const [ShowfotoItemInfo](#) &info) const
- bool **operator==** (const [ShowfotoItemInfo](#) &info) const  
*Compare for information equality and un-equality, not including variable values.*

### Static Public Member Functions

- static [ShowfotoItemInfo](#) **itemInfoFromFile** (const QFileInfo &inf)



## Public Attributes

- QDateTime **ctime**  
*camera date stamp*
- QDateTime **dtime**  
*creation time on disk*
- QString **folder**  
*Folder path to access to file.*
- int **height** = 0  
*Image height in pixels.*
- qlonglong **id** = -1  
*Unique image id.*
- QString **mime**  
*Type mime of file.*
- QString **name**  
*File name in file-system.*
- [PhotoInfoContainer](#) **photoInfo**
- qint64 **size** = -1  
*Static values.*
- QUrl **url**  
*file Url*
- int **width** = 0  
*Image width in pixels.*

## 9.1454.1 Member Data Documentation

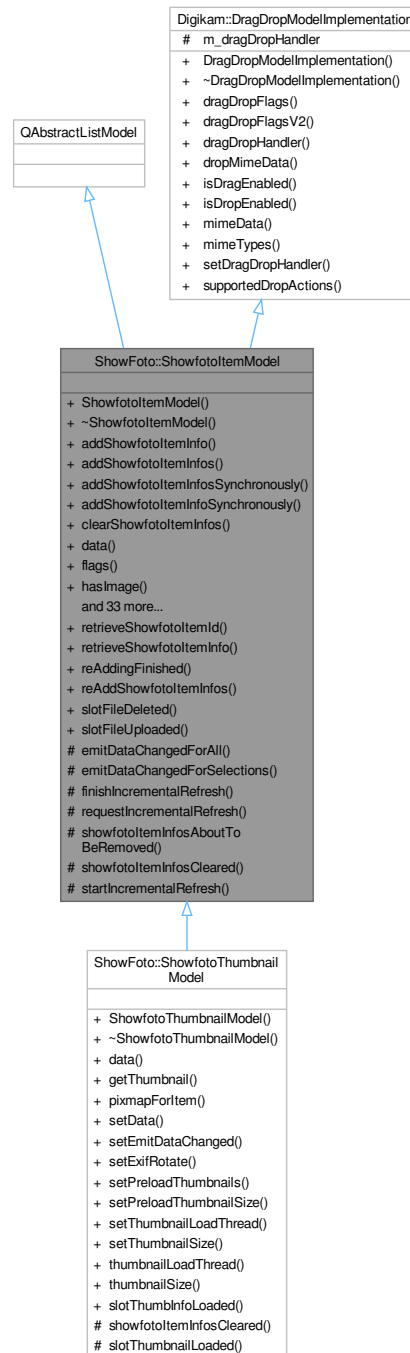
### 9.1454.1.1 size

```
qint64 ShowFoto::ShowfotoItemInfo::size = -1
```

file size in bytes.

## 9.1455 ShowFoto::ShowfotoItemModel Class Reference

Inheritance diagram for ShowFoto::ShowfotoItemModel:



### Public Types

- enum [ShowfotoItemModelRoles](#) {
  - [ShowfotoItemModelPointerRole](#) = Qt::UserRole , [ShowfotoItemModelInternalId](#) = Qt::UserRole + 1 ,
  - [ThumbnailRole](#) = Qt::UserRole + 2 , [ExtraDataRole](#) = Qt::UserRole + 3 ,
  - [ExtraDataDuplicateCount](#) = Qt::UserRole + 6 , [FilterModelRoles](#) = Qt::UserRole + 100 }

## Public Slots

- void **reAddingFinished** ()
- void **reAddShowfotoItemInfos** (const ShowfotoItemInfoList &infos)
- void **slotFileDeleted** (const QString &folder, const QString &file, bool status)
- void **slotFileUploaded** (const [ShowfotoItemInfo](#) &info)

## Signals

- void [allRefreshingFinished](#) ()  
*Signals that the model has finished currently with all scheduled refreshing, full or incremental, and all preprocessing.*
- void [itemInfosAboutToBeAdded](#) (const QList< [ShowfotoItemInfo](#) > &infos)  
*Informs that ItemInfos will be added to the model.*
- void [itemInfosAboutToBeRemoved](#) (const QList< [ShowfotoItemInfo](#) > &infos)  
*Informs that ShowfotoItemInfos will be removed from the model.*
- void [itemInfosAdded](#) (const QList< [ShowfotoItemInfo](#) > &infos)  
*Informs that ItemInfos have been added to the model.*
- void [itemInfosRemoved](#) (const QList< [ShowfotoItemInfo](#) > &infos)  
*Informs that ShowfotoItemInfos have been removed from the model.*
- void **preprocess** (const QList< [ShowfotoItemInfo](#) > &infos)  
*Connect to this signal only if you are the current preprocessor.*
- void **processAdded** (const QList< [ShowfotoItemInfo](#) > &infos)
- void [readyForIncrementalRefresh](#) ()  
*Signals that the model is right now ready to start an incremental refresh.*

## Public Member Functions

- **ShowfotoItemModel** (QObject \*const parent)
- void **addShowfotoItemInfo** (const [ShowfotoItemInfo](#) &info)
- void **addShowfotoItemInfos** (const QList< [ShowfotoItemInfo](#) > &infos)
- void **addShowfotoItemInfosSynchronously** (const QList< [ShowfotoItemInfo](#) > &infos)
- void [addShowfotoItemInfoSynchronously](#) (const [ShowfotoItemInfo](#) &info)  
*addShowfotoItemInfo() is asynchronous if a preprocessor is set.*
- void **clearShowfotoItemInfos** ()  
*Clears the ShowfotoItemInfos and resets the model.*
- QVariant **data** (const QModelIndex &index, int role) const override
- Qt::ItemFlags **flags** (const QModelIndex &index) const override
- bool **hasImage** (const [ShowfotoItemInfo](#) &info) const
- bool **hasImage** (qlonglong id) const
- QVariant **headerData** (int section, Qt::Orientation orientation, int role) const override
- QModelIndex **index** (int row, int column, const QModelIndex &parent) const override
- QList< QModelIndex > **indexesForShowfotoItemId** (qlonglong id) const
- QList< QModelIndex > **indexesForShowfotoItemInfo** (const [ShowfotoItemInfo](#) &info) const
- QList< QModelIndex > **indexesForUrl** (const QUrl &fileUrl) const
- QModelIndex **indexForShowfotoItemId** (qlonglong id) const
- QModelIndex **indexForShowfotoItemInfo** (const [ShowfotoItemInfo](#) &info) const  
*Return the index of a given [ShowfotoItemInfo](#), if it exists in the model.*
- QModelIndex [indexForUrl](#) (const QUrl &fileUrl) const  
*Returns the index or [ShowfotoItemInfo](#) object from the underlying data for the given file url.*
- bool **isEmpty** () const
- int **numberOfIndexesForShowfotoItemId** (qlonglong id) const

- int **numberOfIndexesForShowfotoItemInfo** (const [ShowfotoItemInfo](#) &info) const
- void **removeIndex** (const QModelIndex &index)
  - Remove the given infos or indexes directly from the model.*
- void **removeIndexes** (const QList< QModelIndex > &indexes)
- void **removeShowfotoItemInfo** (const [ShowfotoItemInfo](#) &info)
- void **removeShowfotoItemInfos** (const QList< [ShowfotoItemInfo](#) > &infos)
- int **rowCount** (const QModelIndex &parent) const override
  - QAbstractListModel implementations.*
- void **setKeepsFileUrlCache** (bool keepCache)
  - If a cache is kept, lookup by file path is fast, without a cache it is O(n).*
- DECLARE\_MODEL\_DRAG\_DROP\_METHODS void **setSendRemovalSignals** (bool send)
  - DragDrop methods.*
- void **setShowfotoItemInfos** (const QList< [ShowfotoItemInfo](#) > &infos)
  - Clears and adds infos.*
- qlonglong **showfotoItemId** (const QModelIndex &index) const
- qlonglong **showfotoItemId** (int row) const
- QList< qlonglong > **showfotoItemIds** () const
- QList< qlonglong > **showfotoItemIds** (const QList< QModelIndex > &indexes) const
- [ShowfotoItemInfo](#) **showfotoItemInfo** (const QModelIndex &index) const
  - Returns the [ShowfotoItemInfo](#) object, reference from the underlying data pointed to by the index.*
- [ShowfotoItemInfo](#) **showfotoItemInfo** (const QUrl &fileUrl) const
- [ShowfotoItemInfo](#) **showfotoItemInfo** (int row) const
  - Returns the [ShowfotoItemInfo](#) object, reference from the underlying data of the given row (parent is the invalid QModelIndex, column is 0).*
- [ShowfotoItemInfo](#) & **showfotoItemInfoRef** (const QModelIndex &index) const
- [ShowfotoItemInfo](#) & **showfotoItemInfoRef** (int row) const
- QList< [ShowfotoItemInfo](#) > **showfotoItemInfos** () const
- ShowfotoItemInfoList **showfotoItemInfos** (const QList< QModelIndex > &indexes) const
- QList< [ShowfotoItemInfo](#) > **showfotoItemInfos** (const QUrl &fileUrl) const
- QList< [ShowfotoItemInfo](#) > **uniqueShowfotoItemInfos** () const

## Public Member Functions inherited from [Digikam::DragDropModelImplementation](#)

- [DragDropModelImplementation](#) ()=default
  - A class providing a sample implementation for a QAbstractItemModel redirecting drag-and-drop support to a handler.*
- virtual Qt::ItemFlags **dragDropFlags** (const QModelIndex &index) const
  - Call from your flags() method, adding the relevant drag drop flags.*
- Qt::ItemFlags **dragDropFlagsV2** (const QModelIndex &index) const
  - This is an alternative approach to [dragDropFlags\(\)](#).*
- [AbstractItemDragDropHandler](#) \* **dragDropHandler** () const
- bool **dropMimeData** (const QMimeData \*, Qt::DropAction, int, int, const QModelIndex &)
- virtual bool **isDragEnabled** (const QModelIndex &index) const
- virtual bool **isDropEnabled** (const QModelIndex &index) const
- QMimeData \* **mimeData** (const QModelIndexList &indexes) const
- QStringList **mimeTypes** () const
- void **setDragDropHandler** ([AbstractItemDragDropHandler](#) \*handler)
  - Set a drag drop handler.*
- Qt::DropActions **supportedDropActions** () const
  - Implements the relevant QAbstractItemModel methods for drag and drop.*

## Static Public Member Functions

- static qlonglong **retrieveShowfotoItemId** (const QModelIndex &index)
- static [ShowfotoItemInfo](#) **retrieveShowfotoItemInfo** (const QModelIndex &index)

Retrieve the [ShowfotoItemInfo](#) object from the data() function of the given index. The index may be from a QSortFilterProxyModel as long as an [ShowfotoItemModel](#) is at the end.

## Protected Member Functions

- void **emitDataChangedForAll** ()
- void **emitDataChangedForSelections** (const QItemSelection &selection)
- void **finishIncrementalRefresh** ()
- void [requestIncrementalRefresh](#) ()

As soon as the model is ready to start an incremental refresh, the signal [readyForIncrementalRefresh\(\)](#) will be emitted.

- virtual void **showfotoItemInfosAboutToBeRemoved** (int, int)

Called before rowsAboutToBeRemoved.

- virtual void [showfotoItemInfosCleared](#) ()

Called when the internal storage is cleared.

- void [startIncrementalRefresh](#) ()

Starts an incremental refresh operation.

## Additional Inherited Members

## Protected Attributes inherited from [Digikam::DragDropModelImplementation](#)

- [AbstractItemDragDropHandler](#) \* **m\_dragDropHandler** = nullptr

## 9.1455.1 Member Enumeration Documentation

### 9.1455.1.1 ShowfotoItemModelRoles

```
enum ShowFoto::ShowfotoItemModel::ShowfotoItemModelRoles
```

#### Enumerator

ShowfotoItemModelPointerRole	An ShowfotoItemModel* pointer to this model.
ThumbnailRole	Returns a thumbnail pixmap. May be implemented by subclasses. Returns either a valid pixmap or a null QVariant.
ExtraDataRole	Return (optional) extraData field.
ExtraDataDuplicateCount	Returns the number of duplicate indexes for the same image id.

## 9.1455.2 Member Function Documentation

### 9.1455.2.1 addShowfotoItemInfoSynchronously()

```
void ShowFoto::ShowfotoItemModel::addShowfotoItemInfoSynchronously (
    const ShowfotoItemInfo & info )
```

This method first adds the info, synchronously. Only afterwards, the preprocessor will have the opportunity to process it. This method also bypasses any incremental updates.

#### 9.1455.2.2 allRefreshingFinished

```
void ShowFoto::ShowfotoItemModel::allRefreshingFinished ( ) [signal]
```

The model is in polished, clean situation right now.

#### 9.1455.2.3 indexForUrl()

```
QModelIndex ShowFoto::ShowfotoItemModel::indexForUrl (
    const QUrl & fileUrl ) const
```

In case of multiple occurrences of the same file, the simpler overrides returns any one found first, use the QList methods to retrieve all occurrences.

#### 9.1455.2.4 itemInfosAboutToBeAdded

```
void ShowFoto::ShowfotoItemModel::itemInfosAboutToBeAdded (
    const QList< ShowfotoItemInfo > & infos ) [signal]
```

This signal is sent before the model data is changed and views are informed.

#### 9.1455.2.5 itemInfosAboutToBeRemoved

```
void ShowFoto::ShowfotoItemModel::itemInfosAboutToBeRemoved (
    const QList< ShowfotoItemInfo > & infos ) [signal]
```

This signal is sent before the model data is changed and views are informed. Note: You need to explicitly enable sending of this signal. It is not sent in [clearShowfotoItemInfos\(\)](#).

#### 9.1455.2.6 itemInfosAdded

```
void ShowFoto::ShowfotoItemModel::itemInfosAdded (
    const QList< ShowfotoItemInfo > & infos ) [signal]
```

This signal is sent after the model data is changed and views are informed.

#### 9.1455.2.7 itemInfosRemoved

```
void ShowFoto::ShowfotoItemModel::itemInfosRemoved (
    const QList< ShowfotoItemInfo > & infos ) [signal]
```

This signal is sent after the model data is changed and views are informed. Note: You need to explicitly enable sending of this signal. It is not sent in [clearShowfotoItemInfos\(\)](#).

### 9.1455.2.8 readyForIncrementalRefresh

```
void ShowFoto::ShowfotoItemModel::readyForIncrementalRefresh ( ) [signal]
```

This is guaranteed only for the scope of emitting this signal.

### 9.1455.2.9 requestIncrementalRefresh()

```
void ShowFoto::ShowfotoItemModel::requestIncrementalRefresh ( ) [protected]
```

The signal will be emitted inline if the model is ready right now.

### 9.1455.2.10 setKeepsFileUrlCache()

```
void ShowFoto::ShowfotoItemModel::setKeepsFileUrlCache (
    bool keepCache )
```

Default is false.

### 9.1455.2.11 setSendRemovalSignals()

```
void ShowFoto::ShowfotoItemModel::setSendRemovalSignals (
    bool send )
```

Enable sending of itemInfosAboutToBeRemoved and itemsInfosRemoved signals. Default: false

### 9.1455.2.12 showfotoItemInfo() [1/2]

```
ShowfotoItemInfo ShowFoto::ShowfotoItemModel::showfotoItemInfo (
    const QModelIndex & index ) const
```

For [ShowfotoItemInfo](#) and [ShowfotoItemInfoRef](#) If the index is not valid they will return a null [ShowfotoItemInfo](#), and 0 respectively, [ShowfotoItemInfoRef](#) must not be called with an invalid index as it will crash.

### 9.1455.2.13 showfotoItemInfo() [2/2]

```
ShowfotoItemInfo ShowFoto::ShowfotoItemModel::showfotoItemInfo (
    int row ) const
```

Note that [ShowfotoItemInfoRef](#) must not be called with an invalid index as it will crash.

### 9.1455.2.14 showfotoItemInfosCleared()

```
virtual void ShowFoto::ShowfotoItemModel::showfotoItemInfosCleared ( ) [inline], [protected],
[virtual]
```

Reimplemented in [ShowFoto::ShowfotoThumbnailModel](#).

### 9.1455.2.15 startIncrementalRefresh()

```
void ShowFoto::ShowfotoItemModel::startIncrementalRefresh ( ) [protected]
```

You shall only call this method from a slot connected to [readyForIncrementalRefresh\(\)](#). To initiate an incremental refresh, call [requestIncrementalRefresh\(\)](#).

## 9.1456 ShowFoto::ShowfotoItemSortSettings Class Reference

### Public Types

- enum **CategorizationMode** { **NoCategories** , **CategoryByFolder** , **CategoryByFormat** }
- enum **SortOrder** { **AscendingOrder** = Qt::AscendingOrder , **DescendingOrder** = Qt::DescendingOrder , **DefaultOrder** }
- enum **SortRole** { **SortByCreationDate** , **SortByFileName** , **SortByFileSize** }

### Public Member Functions

- int **compare** (const [ShowfotoItemInfo](#) &left, const [ShowfotoItemInfo](#) &right) const  
*Compares the showfotoItemInfos left and right.*
- int **compare** (const [ShowfotoItemInfo](#) &left, const [ShowfotoItemInfo](#) &right, SortRole sortRole) const
- int **compareCategories** (const [ShowfotoItemInfo](#) &left, const [ShowfotoItemInfo](#) &right) const  
*Compares the categories of left and right ShowfotoItemInfos.*
- bool **isCategorized** () const
- bool **lessThan** (const QVariant &left, const QVariant &right) const  
*Returns true if left QVariant is less than right.*
- bool **lessThan** (const [ShowfotoItemInfo](#) &left, const [ShowfotoItemInfo](#) &right) const  
*Returns true if left is less than right.*
- bool **operator==** (const [ShowfotoItemSortSettings](#) &other) const
- void **setCategorizationMode** (CategorizationMode mode)  
*— Categories -----*
- void **setCategorizationSortOrder** (SortOrder order)
- void **setSortOrder** (SortOrder order)
- void **setSortRole** (SortRole role)  
*— Showfoto Items Sorting -----*

### Static Public Member Functions

- template<typename T >  
static int **compareByOrder** (const T &a, const T &b, Qt::SortOrder sortOrder)
- static int **compareByOrder** (int compareResult, Qt::SortOrder sortOrder)  
*Takes a typical result from a compare method (0 is equal, -1 is less than, 1 is greater than) and applies the given sort order to it.*
- template<typename T >  
static int **compareValue** (const T &a, const T &b)  
*Returns the usual compare result of -1, 0, or 1 for lessThan, equals and greaterThan.*
- static Qt::SortOrder **defaultSortOrderForCategorizationMode** (CategorizationMode mode)
- static Qt::SortOrder **defaultSortOrderForSortRole** (SortRole role)
- template<typename T >  
static bool **lessThanByOrder** (const T &a, const T &b, Qt::SortOrder sortOrder)  
*Returns a < b if sortOrder is Ascending, or b < a if order is descending.*
- static int **naturalCompare** (const QString &a, const QString &b, Qt::SortOrder sortOrder, Qt::CaseSensitivity caseSensitive=Qt::CaseSensitive)  
*Compares the two string by natural comparison and adheres to given sort order.*



## Public Attributes

- Qt::CaseSensitivity **categoryizationCaseSensitivity** = Qt::CaseSensitive
- CategorizationMode **categoryizationMode** = NoCategories
- [SortOrder](#) **categoryizationSortOrder** = [DefaultOrder](#)
- Qt::SortOrder **currentCategoryizationSortOrder** = Qt::AscendingOrder  
*Only Ascending or Descending, never be DefaultOrder.*
- Qt::SortOrder **currentSortOrder** = Qt::AscendingOrder
- Qt::CaseSensitivity **sortCaseSensitivity** = Qt::CaseSensitive
- [SortOrder](#) **sortOrder** = [DefaultOrder](#)
- SortRole **sortRole** = SortByFileName

## 9.1456.1 Member Enumeration Documentation

### 9.1456.1.1 SortOrder

```
enum ShowFoto::ShowfotoItemSortSettings::SortOrder
```

#### Enumerator

DefaultOrder	sort order depends on the chosen sort role
--------------	--

## 9.1456.2 Member Function Documentation

### 9.1456.2.1 compare()

```
int ShowFoto::ShowfotoItemSortSettings::compare (
    const ShowfotoItemInfo & left,
    const ShowfotoItemInfo & right ) const
```

Return -1 if left is less than right, 1 if left is greater than right, and 0 if left equals right comparing the current sort role's value. Adheres to set sort role and sort order.

### 9.1456.2.2 compareCategories()

```
int ShowFoto::ShowfotoItemSortSettings::compareCategories (
    const ShowfotoItemInfo & left,
    const ShowfotoItemInfo & right ) const
```

It returns -1 if the left [ShowfotoItemInfo](#) is less than right, and 0 if both fall in the same category, and 1 if the left [ShowfotoItemInfo](#) is greater than right. Adheres to set categorization mode and current category sort order.

### 9.1456.2.3 lessThan() [1/2]

```
bool ShowFoto::ShowfotoItemSortSettings::lessThan (
    const QVariant & left,
    const QVariant & right ) const
```

Adheres to current sort role and sort order.

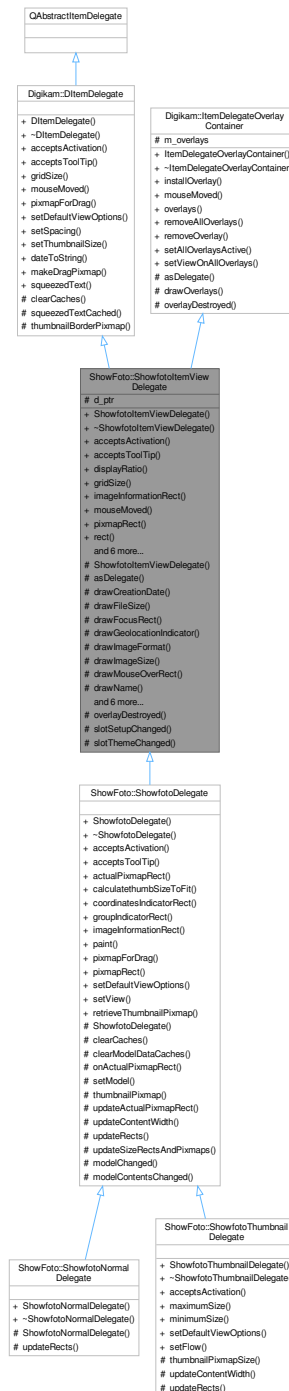
**9.1456.2.4 lessThan()** [2/2]

```
bool ShowFoto::ShowfotoItemSortSettings::lessThan (  
    const ShowfotoItemInfo & left,  
    const ShowfotoItemInfo & right ) const
```

Adheres to current sort role and sort order.

## 9.1457 ShowFoto::ShowfotoItemViewDelegate Class Reference

Inheritance diagram for ShowFoto::ShowfotoItemViewDelegate:



### Signals

- void **hideNotification** ()
- void **requestNotification** (const QModelIndex &index, const QString &message)

## Signals inherited from [Digikam::DItemDelegate](#)

- void **gridSizeChanged** (const QSize &newSize)
- void **visualChange** ()

## Public Member Functions

- **ShowfotoItemViewDelegate** (QWidget \*const parent)
- bool **acceptsActivation** (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*activationRect=nullptr) const override
- bool **acceptsToolTip** (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*tooltipRect=nullptr) const override
 

*These methods take four parameters: The position on viewport, the rect on viewport, the index, and optionally a parameter into which, if the return value is true, a rectangle can be written for which the return value will be true as well.*
- double **displayRatio** () const
- QSize **gridSize** () const override
 

*Returns the gridsize to be set by the view.*
- virtual QRect **imageInformationRect** () const
 

*Returns the area where the image information is drawn, or null if empty / not supported.*
- void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index) override
- virtual QRect **pixmapRect** () const
 

*Returns the area where the pixmap is drawn, or null if not supported.*
- QRect **rect** () const
- void **setDefaultViewOptions** (const QStyleOptionViewItem &option) override
 

*Style option with standard values to use for cached rendering.*
- void **setSpacing** (int spacing) override
- void **setThumbnailSize** (const ThumbnailSize &thumbSize) override
 

*reimplemented from DItemDelegate*
- QSize **sizeHint** (const QStyleOptionViewItem &option, const QModelIndex &index) const override
- int **spacing** () const
- ThumbnailSize **thumbnailSize** () const

## Public Member Functions inherited from [Digikam::DItemDelegate](#)

- **DItemDelegate** (QObject \*const parent=nullptr)
- virtual QPixmap **pixmapForDrag** (const QStyleOptionViewItem &option, const QList< QModelIndex > &indexes) const =0

## Public Member Functions inherited from [Digikam::ItemDelegateOverlayContainer](#)

- **ItemDelegateOverlayContainer** ()=default
 

*This is a sample implementation for delegate management methods, to be inherited by a delegate.*
- void **installOverlay** (**ItemDelegateOverlay** \*overlay)
- void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)
- QList< **ItemDelegateOverlay** \* > **overlays** () const
- void **removeAllOverlays** ()
- void **removeOverlay** (**ItemDelegateOverlay** \*overlay)
- void **setAllOverlaysActive** (bool active)
- void **setViewOnAllOverlays** (QAbstractItemView \*view)

### Protected Slots

- void **overlayDestroyed** (QObject \*o) override
- void **slotSetupChanged** ()
- void **slotThemeChanged** ()

### Protected Member Functions

- **ShowfotoItemViewDelegate** (ShowfotoItemViewDelegatePrivate &dd, QWidget \*const parent)
- QAbstractItemDelegate \* **asDelegate** () override  
*Returns the delegate, typically, the derived class.*
- void **drawCreationDate** (QPainter \*p, const QRect &dateRect, const QDateTime &date) const
- void **drawFileSize** (QPainter \*p, const QRect &r, qlonglong bytes) const
- void **drawFocusRect** (QPainter \*p, const QStyleOptionViewItem &option, bool isSelected) const
- void **drawGeolocationIndicator** (QPainter \*p, const QRect &r) const
- void **drawImageFormat** (QPainter \*p, const QRect &dimsRect, const QString &mime) const
- void **drawImageSize** (QPainter \*p, const QRect &dimsRect, const QSize &dims) const
- void **drawMouseOverRect** (QPainter \*p, const QStyleOptionViewItem &option) const
- void **drawName** (QPainter \*p, const QRect &nameRect, const QString &name) const
- QRect **drawThumbnail** (QPainter \*p, const QRect &thumbRect, const QPixmap &background, const QPixmap &thumbnail) const  
*Use the tool methods for painting in subclasses.*
- virtual void **invalidatePaintingCache** ()  
*reimplement these in subclasses*
- void **prepareBackground** ()
- void **prepareFonts** ()
- void **prepareMetrics** (int maxWidth)
- virtual void **updateSizeRectsAndPixmaps** ()=0

### Protected Member Functions inherited from [Digikam::DItemDelegate](#)

- virtual void **clearCaches** ()
- QString **squeezedTextCached** (QPainter \*const p, int width, const QString &text) const
- QPixmap **thumbnailBorderPixmap** (const QSize &pixSize, bool isGrouped=false) const

### Protected Member Functions inherited from [Digikam::ItemDelegateOverlayContainer](#)

- virtual void **drawOverlays** (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index) const
- virtual void **overlayDestroyed** (QObject \*o)  
*Declare as slot in the derived class calling this method.*

### Protected Attributes

- ShowfotoItemViewDelegatePrivate \*const **d\_ptr** = nullptr

### Protected Attributes inherited from [Digikam::ItemDelegateOverlayContainer](#)

- QList< [ItemDelegateOverlay](#) \* > **m\_overlays**

## Additional Inherited Members

### Static Public Member Functions inherited from [Digikam::DItemDelegate](#)

- static QString **dateToString** (const QDateTime &datetime)
- static QPixmap **makeDragPixmap** (const QStyleOptionViewItem &option, const QList< QModelIndex > &indexes, double displayRatio, const QPixmap &suggestedPixmap=QPixmap())
- static QString **squeezedText** (const QFontMetrics &fm, int width, const QString &text)

## 9.1457.1 Member Function Documentation

### 9.1457.1.1 **acceptsActivation()**

```
bool ShowFoto::ShowfotoItemViewDelegate::acceptsActivation (
    const QPoint & pos,
    const QRect & visualRect,
    const QModelIndex & index,
    QRect * activationRect = nullptr ) const [override], [virtual]
```

Implements [Digikam::DItemDelegate](#).

### 9.1457.1.2 **acceptsToolTip()**

```
bool ShowFoto::ShowfotoItemViewDelegate::acceptsToolTip (
    const QPoint & pos,
    const QRect & visualRect,
    const QModelIndex & index,
    QRect * tooltipRect = nullptr ) const [override], [virtual]
```

Implements [Digikam::DItemDelegate](#).

### 9.1457.1.3 **asDelegate()**

```
QAbstractItemDelegate * ShowFoto::ShowfotoItemViewDelegate::asDelegate ( ) [override], [protected], [virtual]
```

Implements [Digikam::ItemDelegateOverlayContainer](#).

### 9.1457.1.4 **gridSize()**

```
QSize ShowFoto::ShowfotoItemViewDelegate::gridSize ( ) const [override], [virtual]
```

It's sizeHint plus spacing.

Implements [Digikam::DItemDelegate](#).

### 9.1457.1.5 imageInformationRect()

```
QRect ShowFoto::ShowfotoItemViewDelegate::imageInformationRect ( ) const [virtual]
```

The image information is textual or graphical information, but not the pixmap. The `ratingRect()` will e.g. typically be contained in this area.

Reimplemented in [ShowFoto::ShowfotoDelegate](#).

### 9.1457.1.6 mouseMoved()

```
void ShowFoto::ShowfotoItemViewDelegate::mouseMoved (
    QMouseEvent * e,
    const QRect & visualRect,
    const QModelIndex & index ) [override], [virtual]
```

#### Note

to be called by `ItemViewCategorized` only

Implements [Digikam::DItemDelegate](#).

### 9.1457.1.7 pixmapRect()

```
QRect ShowFoto::ShowfotoItemViewDelegate::pixmapRect ( ) const [virtual]
```

Reimplemented in [ShowFoto::ShowfotoDelegate](#).

### 9.1457.1.8 setDefaultViewOptions()

```
void ShowFoto::ShowfotoItemViewDelegate::setDefaultViewOptions (
    const QStyleOptionViewItem & option ) [override], [virtual]
```

`option.rect` shall be the viewport rectangle. Call on resize, font change.

Implements [Digikam::DItemDelegate](#).

Reimplemented in [ShowFoto::ShowfotoThumbnailDelegate](#).

### 9.1457.1.9 setSpacing()

```
void ShowFoto::ShowfotoItemViewDelegate::setSpacing (
    int spacing ) [override], [virtual]
```

Implements [Digikam::DItemDelegate](#).

### 9.1457.1.10 setThumbnailSize()

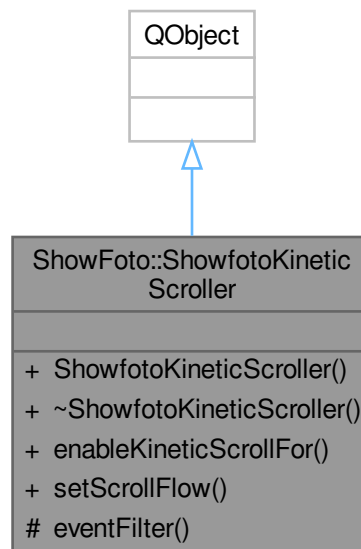
```
void ShowFoto::ShowfotoItemViewDelegate::setThumbnailSize (
    const ThumbnailSize & thumbSize ) [override], [virtual]
```

Implements [Digikam::DItemDelegate](#).

## 9.1458 ShowFoto::ShowfotoKineticScroller Class Reference

Vertical kinetic scroller implementation without overshoot and bouncing.

Inheritance diagram for ShowFoto::ShowfotoKineticScroller:



### Public Member Functions

- **ShowfotoKineticScroller** (`QObject *const parent=nullptr`)
- void **enableKineticScrollFor** (`QAbstractScrollArea *const scrollArea`)  
*NOTE: enabled for one widget only, new calls remove previous association.*
- void **setScrollFlow** (`QListView::Flow flow`)

### Protected Member Functions

- bool **eventFilter** (`QObject *object, QEvent *event`) override  
*intercepts mouse events to make the scrolling work*

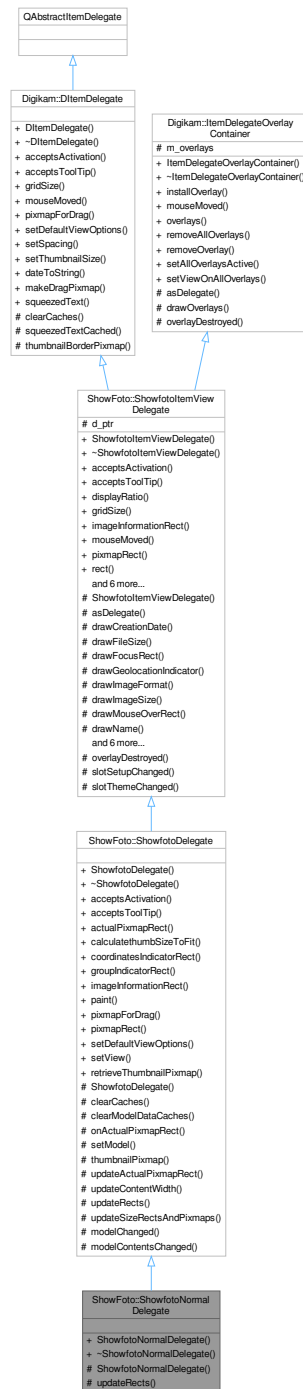


## 9.1458.1 Detailed Description

A temporary solution to get kinetic-like scrolling on Symbian.

## 9.1459 ShowFoto::ShowfotoNormalDelegate Class Reference

Inheritance diagram for ShowFoto::ShowfotoNormalDelegate:



## Public Member Functions

- **ShowfotoNormalDelegate** ([ShowfotoThumbnailBar](#) \*const bar, QObject \*const parent=nullptr)

## Public Member Functions inherited from [ShowFoto::ShowfotoDelegate](#)

- **ShowfotoDelegate** (QWidget \*const parent)
- bool [acceptsActivation](#) (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*activationRect=nullptr) const override
- bool [acceptsToolTip](#) (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*tooltipRect=nullptr) const override
 

*These methods take four parameters: The position on viewport, the rect on viewport, the index, and optionally a parameter into which, if the return value is true, a rectangle can be written for which the return value will be true as well.*
- QRect **actualPixmapRect** (const QModelIndex &index) const
- int **calculatethumbSizeToFit** (int ws)
- QRect **coordinatesIndicatorRect** () const
- QRect **groupIndicatorRect** () const
- QRect [imageInformationRect](#) () const override
 

*Returns the area where the image information is drawn, or null if empty / not supported.*
- void **paint** (QPainter \*painter, const QStyleOptionViewItem &option, const QModelIndex &index) const override
- QPixmap [pixmapForDrag](#) (const QStyleOptionViewItem &option, const QList< QModelIndex > &indexes) const override
- QRect [pixmapRect](#) () const override
 

*Returns the area where the pixmap is drawn, or null if not supported.*
- void [setDefaultViewOptions](#) (const QStyleOptionViewItem &option) override
 

*Style option with standard values to use for cached rendering.*
- void **setView** ([ShowfotoThumbnailBar](#) \*view)

## Public Member Functions inherited from [ShowFoto::ShowfotoItemViewDelegate](#)

- **ShowfotoItemViewDelegate** (QWidget \*const parent)
- bool [acceptsActivation](#) (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*activationRect=nullptr) const override
- bool [acceptsToolTip](#) (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*tooltipRect=nullptr) const override
 

*These methods take four parameters: The position on viewport, the rect on viewport, the index, and optionally a parameter into which, if the return value is true, a rectangle can be written for which the return value will be true as well.*
- double **displayRatio** () const
- QSize [gridSize](#) () const override
 

*Returns the gridsize to be set by the view.*
- void [mouseMoved](#) (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index) override
- QRect **rect** () const
- void [setDefaultViewOptions](#) (const QStyleOptionViewItem &option) override
 

*Style option with standard values to use for cached rendering.*
- void [setSpacing](#) (int spacing) override
- void [setThumbnailSize](#) (const ThumbnailSize &thumbSize) override
 

*reimplemented from DItemDelegate*
- QSize **sizeHint** (const QStyleOptionViewItem &option, const QModelIndex &index) const override
- int **spacing** () const
- ThumbnailSize **thumbnailSize** () const

## Public Member Functions inherited from [Digikam::DItemDelegate](#)

- [DItemDelegate](#) (QObject \*const parent=nullptr)

## Public Member Functions inherited from [Digikam::ItemDelegateOverlayContainer](#)

- [ItemDelegateOverlayContainer](#) ()=default  
*This is a sample implementation for delegate management methods, to be inherited by a delegate.*
- void **installOverlay** ([ItemDelegateOverlay](#) \*overlay)
- void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)
- QList< [ItemDelegateOverlay](#) \* > **overlays** () const
- void **removeAllOverlays** ()
- void **removeOverlay** ([ItemDelegateOverlay](#) \*overlay)
- void **setAllOverlaysActive** (bool active)
- void **setViewOnAllOverlays** (QAbstractItemView \*view)

## Protected Member Functions

- **ShowfotoNormalDelegate** (ShowfotoNormalDelegatePrivate &dd, [ShowfotoThumbnailBar](#) \*const bar, QObject \*const parent=nullptr)
- void [updateRects](#) () override  
*In a subclass, you need to implement this method to set up the rects for drawing.*

## Protected Member Functions inherited from [ShowFoto::ShowfotoDelegate](#)

- **ShowfotoDelegate** (ShowfotoDelegate::ShowfotoDelegatePrivate &dd, QWidget \*const parent)
- void [clearCaches](#) () override
- virtual void **clearModelDataCaches** ()  
*Reimplement to clear caches based on model indexes (hash on row number etc.) Change signals are listened to this is called whenever such properties become invalid.*
- bool **onActualPixmapRect** (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*actualRect) const
- void **setModel** (QAbstractItemModel \*model)
- virtual QPixmap **thumbnailPixmap** (const QModelIndex &index) const
- void **updateActualPixmapRect** (const QModelIndex &index, const QRect &rect)
- virtual void [updateContentWidth](#) ()  
*Reimplement this to set contentWidth.*
- void [updateSizeRectsAndPixmaps](#) () override

## Protected Member Functions inherited from [ShowFoto::ShowfotoItemViewDelegate](#)

- **ShowfotoItemViewDelegate** (ShowfotoItemViewDelegatePrivate &dd, QWidget \*const parent)
- QAbstractItemDelegate \* [asDelegate](#) () override  
*Returns the delegate, typically, the derived class.*
- void **drawCreationDate** (QPainter \*p, const QRect &dateRect, const QDateTime &date) const
- void **drawFileSize** (QPainter \*p, const QRect &r, qlonglong bytes) const
- void **drawFocusRect** (QPainter \*p, const QStyleOptionViewItem &option, bool isSelected) const
- void **drawGeolocationIndicator** (QPainter \*p, const QRect &r) const
- void **drawImageFormat** (QPainter \*p, const QRect &dimsRect, const QString &mime) const
- void **drawImageSize** (QPainter \*p, const QRect &dimsRect, const QSize &dims) const

- void **drawMouseOverRect** (QPainter \*p, const QStyleOptionViewItem &option) const
- void **drawName** (QPainter \*p, const QRect &nameRect, const QString &name) const
- QRect **drawThumbnail** (QPainter \*p, const QRect &thumbRect, const QPixmap &background, const QPixmap &thumbnail) const

*Use the tool methods for painting in subclasses.*

- virtual void **invalidatePaintingCache** ()  
*reimplement these in subclasses*
- void **prepareBackground** ()
- void **prepareFonts** ()
- void **prepareMetrics** (int maxWidth)

### Protected Member Functions inherited from [Digikam::DItemDelegate](#)

- QString **squeezedTextCached** (QPainter \*const p, int width, const QString &text) const
- QPixmap **thumbnailBorderPixmap** (const QSize &pixSize, bool isGrouped=false) const

### Protected Member Functions inherited from [Digikam::ItemDelegateOverlayContainer](#)

- virtual void **drawOverlays** (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index) const
- virtual void **overlayDestroyed** (QObject \*o)

*Declare as slot in the derived class calling this method.*

### Additional Inherited Members

### Signals inherited from [ShowFoto::ShowfotoItemViewDelegate](#)

- void **hideNotification** ()
- void **requestNotification** (const QModelIndex &index, const QString &message)

### Signals inherited from [Digikam::DItemDelegate](#)

- void **gridSizeChanged** (const QSize &newSize)
- void **visualChange** ()

### Static Public Member Functions inherited from [ShowFoto::ShowfotoDelegate](#)

- static QPixmap **retrieveThumbnailPixmap** (const QModelIndex &index, int thumbnailSize)

*Retrieve the thumbnail pixmap in given size for the [ShowfotoItemModel::ThumbnailRole](#) for the given index from the given index, which must adhere to [ShowfotoThumbnailModel](#) semantics.*

### Static Public Member Functions inherited from [Digikam::DItemDelegate](#)

- static QString **dateToString** (const QDateTime &datetime)
- static QPixmap **makeDragPixmap** (const QStyleOptionViewItem &option, const QList< QModelIndex > &indexes, double displayRatio, const QPixmap &suggestedPixmap=QPixmap())
- static QString **squeezedText** (const QFontMetrics &fm, int width, const QString &text)

### Protected Slots inherited from [ShowFoto::ShowfotoDelegate](#)

- void `modelChanged` ()
- void `modelContentsChanged` ()

### Protected Slots inherited from [ShowFoto::ShowfotoItemViewDelegate](#)

- void `overlayDestroyed` (QObject \*o) override
- void `slotSetupChanged` ()
- void `slotThemeChanged` ()

### Protected Attributes inherited from [ShowFoto::ShowfotoItemViewDelegate](#)

- ShowfotoItemViewDelegatePrivate \*const `d_ptr` = nullptr

### Protected Attributes inherited from [Digikam::ItemDelegateOverlayContainer](#)

- QList< [ItemDelegateOverlay](#) \* > `m_overlays`

## 9.1459.1 Member Function Documentation

### 9.1459.1.1 `updateRects()`

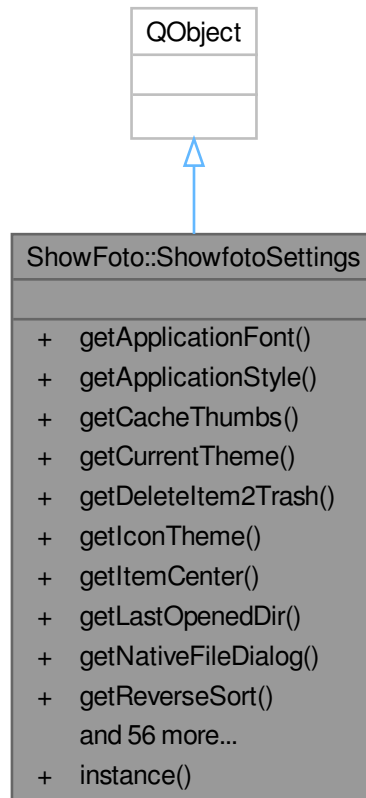
```
void ShowFoto::ShowfotoNormalDelegate::updateRects ( ) [override], [protected], [virtual]
```

The `paint()` method operates depending on these `rects`.

Implements [ShowFoto::ShowfotoDelegate](#).

## 9.1460 ShowFoto::ShowfotoSettings Class Reference

Inheritance diagram for ShowFoto::ShowfotoSettings:



### Public Member Functions

- QFont **getApplicationFont** () const
- QString **getApplicationStyle** () const
- bool **getCacheThumbs** () const
- QString **getCurrentTheme** () const
- bool **getDeleteItem2Trash** () const
- QString **getIconTheme** () const
- bool **getItemCenter** () const
- QString **getLastOpenedDir** () const
- bool **getNativeFileDialog** () const
- bool **getReverseSort** () const
- int **getRightSideBarStyle** () const
- bool **getShowCoordinates** () const
- bool **getShowFileDate** () const
- bool **getShowFileDim** () const
- bool **getShowFileName** () const
- bool **getShowFileSize** () const

- bool **getShowFileType** () const
- bool **getShowFormatOverThumbnail** () const
- bool **getShowPhotoDate** () const
- bool **getShowPhotoExpo** () const
- bool **getShowPhotoFlash** () const
- bool **getShowPhotoFocal** () const
- bool **getShowPhotoLens** () const
- bool **getShowPhotoMake** () const
- bool **getShowPhotoMode** () const
- bool **getShowPhotoWB** () const
- bool **getShowSplash** () const
- bool **getShowToolTip** () const
- int **getSortRole** () const
- QFont **getToolTipFont** () const
- int **getUpdateType** () const
- bool **getUpdateWithDebug** () const
- void **readSettings** ()
- void **setApplicationFont** (const QFont &fnt)
- void **setApplicationStyle** (const QString &style)
- void **setCacheThumbs** (bool item)
- void **setCurrentTheme** (const QString &theme)
- void **setDeleteItem2Trash** (bool D2t)
- void **setIconTheme** (const QString &theme)
- void **setItemCenter** (bool item)
- void **setLastOpenedDir** (const QString &dir)
- void **setNativeFileDialog** (bool item)
- void **setReverseSort** (bool reverse)
- void **setRightSideBarStyle** (int style)
- void **setShowCoordinates** (bool show)
- void **setShowFileDate** (bool show)
- void **setShowFileDim** (bool show)
- void **setShowFileName** (bool show)
- void **setShowFileSize** (bool show)
- void **setShowFileType** (bool show)
- void **setShowFormatOverThumbnail** (bool show)
- void **setShowPhotoDate** (bool show)
- void **setShowPhotoExpo** (bool show)
- void **setShowPhotoFlash** (bool show)
- void **setShowPhotoFocal** (bool show)
- void **setShowPhotoLens** (bool show)
- void **setShowPhotoMake** (bool show)
- void **setShowPhotoMode** (bool show)
- void **setShowPhotoWB** (bool show)
- void **setShowSplash** (bool show)
- void **setShowToolTip** (bool show)
- void **setSortRole** (int order)
- void **setToolTipFont** (const QFont &font)
- void **setUpdateType** (int type)
- void **setUpdateWithDebug** (bool dbg)
- void **syncConfig** ()

#### Static Public Member Functions

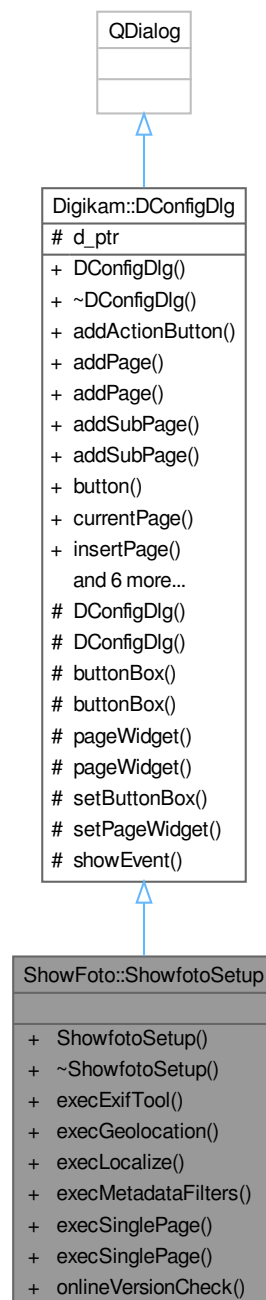
- static [ShowfotoSettings](#) \* **instance** ()

## Friends

- class **ShowfotoSettingsCreator**

## 9.1461 ShowFoto::ShowfotoSetup Class Reference

Inheritance diagram for ShowFoto::ShowfotoSetup:





## Public Types

- enum **Page** {  
**LastPageUsed** = -1 , **EditorPage** = 0 , **MetadataPage** , **ToolTipPage** ,  
**RawPage** , **IOFilesPage** , **ICCPage** , **GeolocationPage** ,  
**PluginsPage** , **MiscellaneousPage** , **SetupPageEnumLast** }

## Public Types inherited from [Digikam::DConfigDlg](#)

- enum **FaceType** {  
**Auto** = DConfigDlgView::Auto , **Plain** = DConfigDlgView::Plain , **List** = DConfigDlgView::List , **Tree** =  
DConfigDlgView::Tree ,  
**Tabbed** = DConfigDlgView::Tabbed }

## Public Member Functions

- **ShowfotoSetup** (QWidget \*const parent=nullptr, Page page=LastPageUsed)

## Public Member Functions inherited from [Digikam::DConfigDlg](#)

- **DConfigDlg** (QWidget \*const parent=nullptr, Qt::WindowFlags flags=Qt::WindowFlags())  
*Creates a new page dialog.*
- **~DConfigDlg** () override  
*Destroys the page dialog.*
- void **addActionButton** (QAbstractButton \*const button)  
*Set an action button.*
- void **addPage** (DConfigDlgWdgItem \*const item)  
*Adds a new top level page to the dialog.*
- DConfigDlgWdgItem \* **addPage** (QWidget \*const widget, const QString &name)  
*Adds a new top level page to the dialog.*
- void **addSubPage** (DConfigDlgWdgItem \*const parent, DConfigDlgWdgItem \*const item)  
*Inserts a new sub page in the dialog.*
- DConfigDlgWdgItem \* **addSubPage** (DConfigDlgWdgItem \*const parent, QWidget \*const widget, const  
QString &name)  
*Inserts a new sub page in the dialog.*
- QPushButton \* **button** (QDialogButtonBox::StandardButton which) const  
*Returns the QPushButton corresponding to the standard button which, or 0 if the standard button doesn't exist in this  
dialog.*
- DConfigDlgWdgItem \* **currentPage** () const  
*Returns the.*
- void **insertPage** (DConfigDlgWdgItem \*const before, DConfigDlgWdgItem \*const item)  
*Inserts a new page in the dialog.*
- DConfigDlgWdgItem \* **insertPage** (DConfigDlgWdgItem \*const before, QWidget \*const widget, const  
QString &name)  
*Inserts a new page in the dialog.*
- void **removePage** (DConfigDlgWdgItem \*const item)  
*Removes the page associated with the given.*
- void **setConfigGroup** (const QString &group)  
*Sets the config group name for restore or save dialog window size.*
- void **setCurrentPage** (DConfigDlgWdgItem \*const item)  
*Sets the page which is associated with the given.*
- void **setFaceType** (FaceType faceType)  
*Sets the face type of the dialog.*
- void **setStandardButtons** (QDialogButtonBox::StandardButtons buttons)  
*Sets the collection of standard buttons displayed by this dialog.*

### Static Public Member Functions

- static bool **execExifTool** (QWidget \*const parent)
- static bool **execGeolocation** (QWidget \*const parent, int tab)
- static bool **execLocalize** (QWidget \*const parent)
- static bool **execMetadataFilters** (QWidget \*const parent, int tab)
- static bool **execSinglePage** (Page page)
  - *Show a setup dialog.*
- static bool **execSinglePage** (QWidget \*const parent, Page page)
- static void **onlineVersionCheck** ()

### Additional Inherited Members

#### Signals inherited from [Digikam::DConfigDlg](#)

- void **currentPageChanged** (DConfigDlgWdgItem \*current, DConfigDlgWdgItem \*before)
  - *This signal is emitted whenever the current page has changed.*
- void **pageRemoved** (DConfigDlgWdgItem \*page)
  - *This signal is emitted whenever a page has been removed.*

#### Protected Member Functions inherited from [Digikam::DConfigDlg](#)

- **DConfigDlg** (DConfigDlgPrivate &dd, DConfigDlgWdg \*const widget, QWidget \*const parent, Qt::Window↔Flags flags=Qt::WindowFlags())
- **DConfigDlg** (DConfigDlgWdg \*const widget, QWidget \*const parent, Qt::WindowFlags flags=Qt::Window↔Flags())
  - *This constructor can be used by subclasses to provide a custom page widget.*
- QDialogButtonBox \* **buttonBox** ()
  - *Returns the button box of the dialog or 0 if no button box is set.*
- const QDialogButtonBox \* **buttonBox** () const
  - *Returns the button box of the dialog or 0 if no button box is set.*
- DConfigDlgWdg \* **pageWidget** ()
  - *Returns the page widget of the dialog or 0 if no page widget is set.*
- const DConfigDlgWdg \* **pageWidget** () const
  - *Returns the page widget of the dialog or 0 if no page widget is set.*
- void **setButtonBox** (QDialogButtonBox \*const box)
  - *Set the button box of the dialog.*
- void **setPageWidget** (DConfigDlgWdg \*const widget)
  - *Set the page widget of the dialog.*
- void **showEvent** (QShowEvent \*) override

#### Protected Attributes inherited from [Digikam::DConfigDlg](#)

- DConfigDlgPrivate \*const **d\_ptr** = nullptr

## 9.1461.1 Member Function Documentation

### 9.1461.1.1 execSinglePage()

```
bool ShowFoto::ShowfotoSetup::execSinglePage (
    Page page ) [static]
```

Only the specified page will be available.

## 9.1462 ShowFoto::ShowfotoSetupMetadata Class Reference

Inheritance diagram for ShowFoto::ShowfotoSetupMetadata:



### Public Types

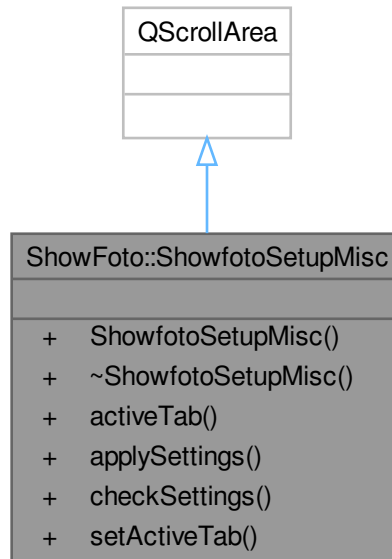
- enum `MetadataTab` {  
**Behavior** = 0 , **ExifViewer** , **MakernotesViewer** , **IptcViewer** ,  
**XmpViewer** , **ExifTool** }

### Public Member Functions

- `ShowfotoSetupMetadata` (`QWidget *const parent=nullptr`)
- `MetadataTab activeTab` () const
- void `applySettings` ()
- void `setActiveTab` (`MetadataTab tab`)

## 9.1463 ShowFoto::ShowfotoSetupMisc Class Reference

Inheritance diagram for ShowFoto::ShowfotoSetupMisc:



### Public Types

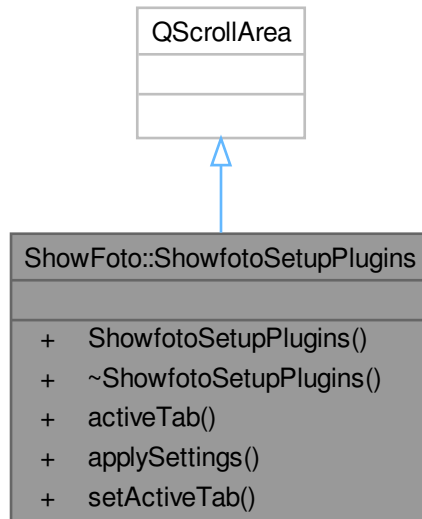
- enum **MiscTab** { **Behaviour** = 0 , **Appearance** , **SpellCheck** , **Localize** , **System** }
- enum **SortOrder** { **SortByDate** = 0 , **SortByName** , **SortByFileSize** }

### Public Member Functions

- **ShowfotoSetupMisc** (QWidget \*const parent=nullptr)
- MiscTab **activeTab** () const
- void **applySettings** ()
- bool **checkSettings** ()
- void **setActiveTab** (MiscTab tab)

## 9.1464 ShowFoto::ShowfotoSetupPlugins Class Reference

Inheritance diagram for ShowFoto::ShowfotoSetupPlugins:



### Public Types

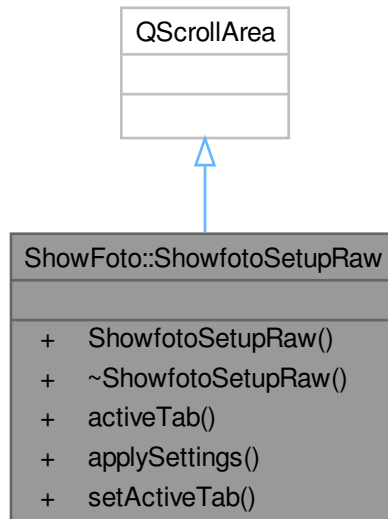
- enum **PluginTab** { **Generic** = 0 , **Editor** , **Loaders** }

### Public Member Functions

- **ShowfotoSetupPlugins** (QWidget \*const parent=nullptr)
- PluginTab **activeTab** () const
- void **applySettings** ()
- void **setActiveTab** (PluginTab tab)

## 9.1465 ShowFoto::ShowfotoSetupRaw Class Reference

Inheritance diagram for ShowFoto::ShowfotoSetupRaw:



### Public Types

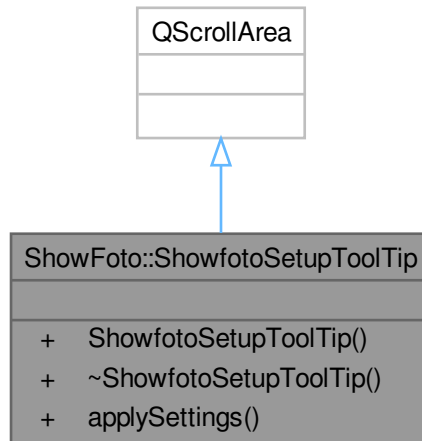
- enum `RAWTab` { `RAWBehavior = 0` , `RAWDefaultSettings` }

### Public Member Functions

- `ShowfotoSetupRaw` (`QWidget *const parent=nullptr`)
- `RAWTab activeTab` () const
- void `applySettings` ()
- void `setActiveTab` (`RAWTab tab`)

## 9.1466 ShowFoto::ShowfotoSetupToolTip Class Reference

Inheritance diagram for ShowFoto::ShowfotoSetupToolTip:

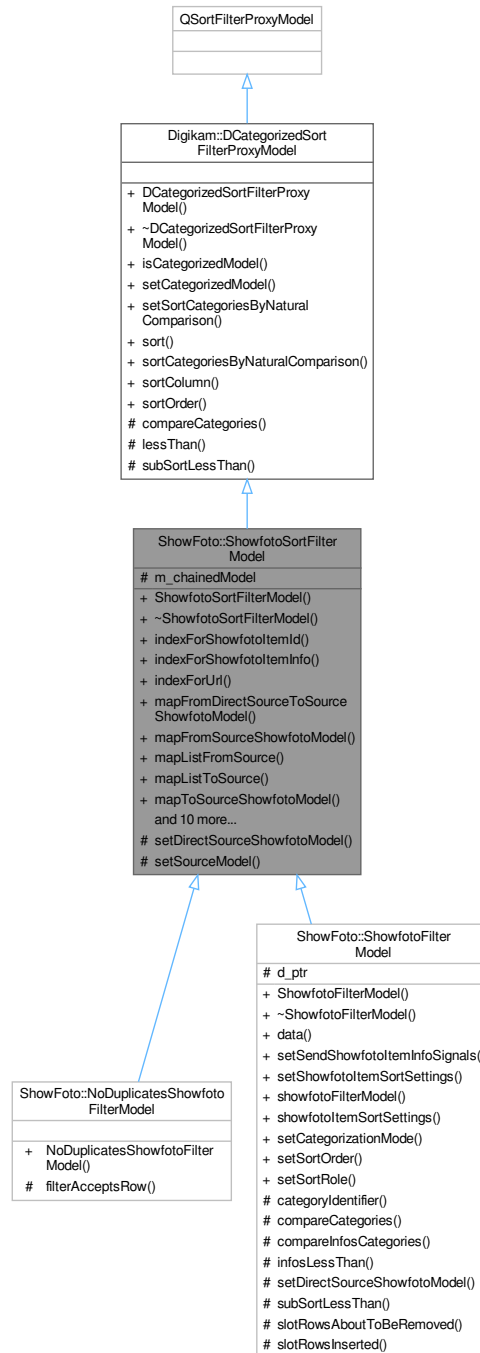


### Public Member Functions

- **ShowfotoSetupToolTip** (QWidget \*const parent=nullptr)
- void **applySettings** ()

## 9.1467 ShowFoto::ShowfotoSortFilterModel Class Reference

Inheritance diagram for ShowFoto::ShowfotoSortFilterModel:



### Public Member Functions

- **ShowfotoSortFilterModel** (QObject \*const parent=nullptr)
- QModelIndex **indexForShowfotoItemId** (qulonglong id) const



- QModelIndex **indexForShowfotoItemInfo** (const [ShowfotoItemInfo](#) &info) const
- QModelIndex **indexForUrl** (const QUrl &fileUrl) const
- QModelIndex **mapFromDirectSourceToSourceShowfotoModel** (const QModelIndex &sourceModelIndex) const
- QModelIndex **mapFromSourceShowfotoModel** (const QModelIndex &showfotoModelIndex) const
- QList< QModelIndex > **mapListFromSource** (const QList< QModelIndex > &sourceIndexes) const
- QList< QModelIndex > **mapListToSource** (const QList< QModelIndex > &indexes) const
- QModelIndex **mapToSourceShowfotoModel** (const QModelIndex &proxyIndex) const  
*Convenience methods mapped to [ShowfotoItemModel](#).*
- void **setSourceFilterModel** ([ShowfotoSortFilterModel](#) \*const sourceModel)
- void **setSourceShowfotoModel** ([ShowfotoItemModel](#) \*const sourceModel)
- virtual [ShowfotoFilterModel](#) \* **showfotoFilterModel** () const  
*Returns this, any chained [ShowfotoFilterModel](#), or 0.*
- qlonglong **showfotoItemId** (const QModelIndex &index) const
- QList< qlonglong > **showfotoItemIds** (const QList< QModelIndex > &indexes) const
- [ShowfotoItemInfo](#) **showfotoItemInfo** (const QModelIndex &index) const
- QList< [ShowfotoItemInfo](#) > **showfotoItemInfos** (const QList< QModelIndex > &indexes) const
- QList< [ShowfotoItemInfo](#) > **showfotoItemInfosSorted** () const  
*Returns a list of all showfoto infos, sorted according to this model.*
- [ShowfotoSortFilterModel](#) \* **sourceFilterModel** () const
- [ShowfotoItemModel](#) \* **sourceShowfotoModel** () const

## Public Member Functions inherited from [Digikam::DCategorizedSortFilterProxyModel](#)

- [DCategorizedSortFilterProxyModel](#) (QObject \*const parent=nullptr)
- bool **isCategorizedModel** () const
- void **setCategorizedModel** (bool categorizedModel)  
*Enables or disables the categorization feature.*
- void **setSortCategoriesByNaturalComparison** (bool [sortCategoriesByNaturalComparison](#))  
*Set if the sorting using [CategorySortRole](#) will use a natural comparison in the case that strings were returned.*
- void **sort** (int column, Qt::SortOrder order=Qt::AscendingOrder) override  
*Overridden from [QSortFilterProxyModel](#).*
- bool **sortCategoriesByNaturalComparison** () const
- int **sortColumn** () const
- Qt::SortOrder **sortOrder** () const

## Protected Member Functions

- virtual void **setDirectSourceShowfotoModel** ([ShowfotoItemModel](#) \*const sourceModel)  
*Reimplement if needed. Called only when model shall be set as (direct) sourceModel.*
- void **setSourceModel** (QAbstractItemModel \*sourceModel) override

## Protected Member Functions inherited from [Digikam::DCategorizedSortFilterProxyModel](#)

- virtual int **compareCategories** (const QModelIndex &left, const QModelIndex &right) const  
*This method compares the category of the *left* index with the category of the *right* index.*
- bool **lessThan** (const QModelIndex &left, const QModelIndex &right) const override  
*Overridden from [QSortFilterProxyModel](#).*
- virtual bool **subSortLessThan** (const QModelIndex &left, const QModelIndex &right) const  
*This method has a similar purpose as [lessThan\(\)](#) has on [QSortFilterProxyModel](#).*

## Protected Attributes

- [ShowfotoSortFilterModel](#) \* `m_chainedModel` = nullptr

## Additional Inherited Members

## Public Types inherited from [Digikam::DCategorizedSortFilterProxyModel](#)

- enum [AdditionalRoles](#) { `CategoryDisplayRole` = 0x17CE990A , `CategorySortRole` = 0x27857E60 }

## 9.1467.1 Member Function Documentation

### 9.1467.1.1 `mapToSourceShowfotoModel()`

```
QModelIndex ShowFoto::ShowfotoSortFilterModel::mapToSourceShowfotoModel (
    const QModelIndex & proxyIndex ) const
```

Mentioned indexes returned come from the source [Showfoto](#) image model.

### 9.1467.1.2 `setDirectSourceShowfotoModel()`

```
void ShowFoto::ShowfotoSortFilterModel::setDirectSourceShowfotoModel (
    ShowfotoItemModel *const sourceModel ) [protected], [virtual]
```

Reimplemented in [ShowFoto::ShowfotoFilterModel](#).

### 9.1467.1.3 `showfotoFilterModel()`

```
ShowfotoFilterModel * ShowFoto::ShowfotoSortFilterModel::showfotoFilterModel ( ) const [virtual]
```

Reimplemented in [ShowFoto::ShowfotoFilterModel](#).

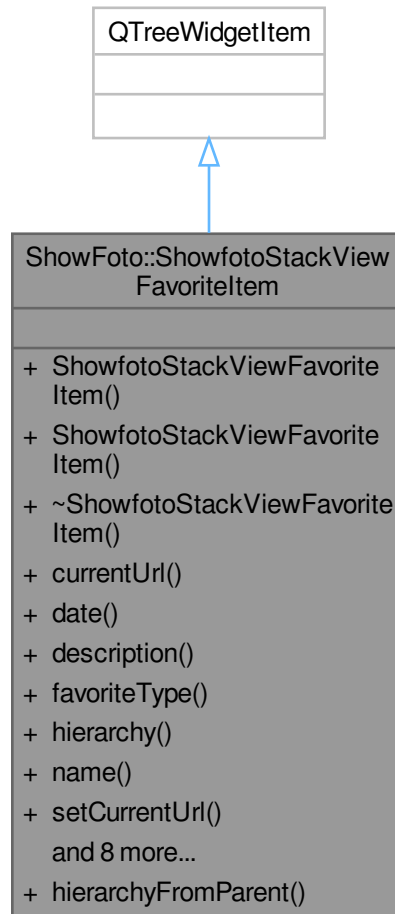
### 9.1467.1.4 `showfotoItemInfosSorted()`

```
QList< ShowfotoItemInfo > ShowFoto::ShowfotoSortFilterModel::showfotoItemInfosSorted ( ) const
```

If you do not need a sorted list, use [ShowfotoItemModel](#)'s `showfotoItemInfo()` method.

## 9.1468 ShowFoto::ShowfotoStackViewFavoriteItem Class Reference

Inheritance diagram for ShowFoto::ShowfotoStackViewFavoriteItem:



### Public Types

- enum `FavoriteType` { `FavoriteRoot` = -1 , `FavoriteFolder` , `FavoriteItem` }

### Public Member Functions

- `ShowfotoStackViewFavoriteItem` (`QTreeWidgetItem *const parent`)
- `ShowfotoStackViewFavoriteItem` (`QTreeWidgetItem *const parent`, `int favType`)
- `QUrl currentUrl () const`
- `QDate date () const`
- `QString description () const`
- `int favoriteType () const`
- `QString hierarchy () const`

- QString **name** () const
  - void **setCurrentUrl** (const QUrl &url)
  - void **setDate** (const QDate &date)
  - void **setDescription** (const QString &desc)
  - void **setFavoriteType** (int favoriteType)
  - void **setHierarchy** (const QString &desc)
  - void **setName** (const QString &name)
  - void **setUrls** (const QList< QUrl > &)
  - QList< QUrl > **urls** () const
  - QStringList **urlsToPaths** () const
- Helper method to get a list local paths from image urls included in favorite item.*

### Static Public Member Functions

- static QString **hierarchyFromParent** (const QString &name, [ShowfotoStackViewFavoriteItem](#) \*const pItem)
- Helper static method to get hierarchy path from item.*

## 9.1468.1 Member Enumeration Documentation

### 9.1468.1.1 FavoriteType

enum [ShowFoto::ShowfotoStackViewFavoriteItem::FavoriteType](#)

#### Enumerator

FavoriteRoot	Favorite is root item from hierarchy.
FavoriteFolder	Favorite is a simple folder in hierarchy.
FavoriteItem	Favorite is a hierarchy item including all properties.

## 9.1468.2 Member Function Documentation

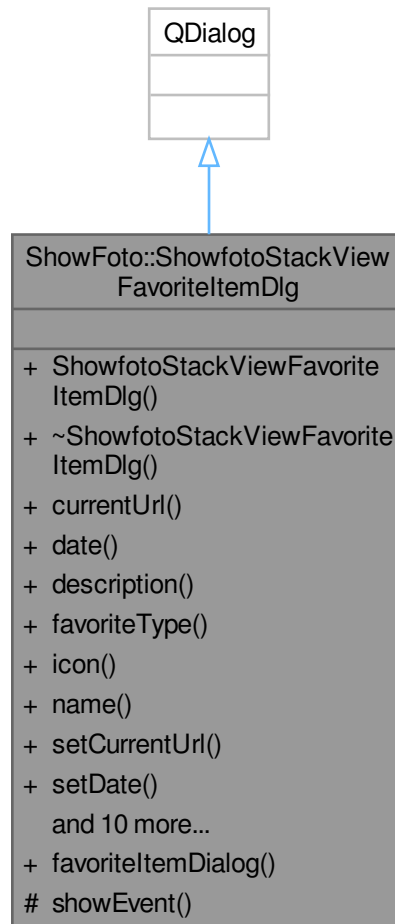
### 9.1468.2.1 hierarchyFromParent()

```
QString ShowFoto::ShowfotoStackViewFavoriteItem::hierarchyFromParent (
    const QString & name,
    ShowfotoStackViewFavoriteItem *const pItem ) [static]
```

'name' is the title and 'pitem' the parent instance.

## 9.1469 ShowFoto::ShowfotoStackViewFavoriteItemDlg Class Reference

Inheritance diagram for ShowFoto::ShowfotoStackViewFavoriteItemDlg:



### Public Member Functions

- **ShowfotoStackViewFavoriteItemDlg** ([ShowfotoStackViewFavoriteList](#) \*const list, bool create=false)
- `QUrl` **currentUrl** () const
- `QDate` **date** () const
- `QString` **description** () const
- `int` **favoriteType** () const
- `QString` **icon** () const
- `QString` **name** () const
- `void` **setCurrentUrl** (const `QUrl` &url)
- `void` **setDate** (const `QDate` &name)
- `void` **setDescription** (const `QString` &desc)
- `void` **setFavoriteType** (int favoriteType)
- `void` **setIcon** (const `QString` &icon)

- void **setIconSize** (int size)
- void **setName** (const QString &name)
- void **setParentItem** ([ShowfotoStackViewFavoriteItem](#) \*const pItem)
- void **setSortOrder** (int order)
- void **setSortRole** (int role)
- void **setUrls** (const QList< QUrl > &urls)
- QList< QUrl > **urls** () const

### Static Public Member Functions

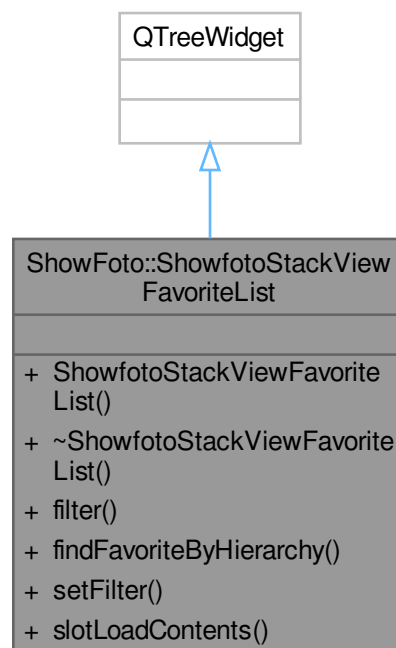
- static bool **favoriteItemDialog** ([ShowfotoStackViewFavoriteList](#) \*const list, QString &name, int &favoriteType, QString &desc, QDate &date, QString &icon, QList< QUrl > &urls, QUrl &current, int iconSize, int sortOrder, int sortRole, [ShowfotoStackViewFavoriteItem](#) \*const pItem, bool create=false)

### Protected Member Functions

- void **showEvent** (QShowEvent \*) override

## 9.1470 ShowFoto::ShowfotoStackViewFavoriteList Class Reference

Inheritance diagram for ShowFoto::ShowfotoStackViewFavoriteList:



## Public Slots

- void **slotLoadContents** ()

## Signals

- void **signalAddFavorite** ()
- void **signalAddFavorite** (const QList< QUrl > &, const QUrl &current)
- void **signalLoadContentsFromFiles** (const QStringList &files, const QString &current)
- void **signalSearchResult** (int)

*Signal emitted when filtering is done through slotSetFilter().*

## Public Member Functions

- **ShowfotoStackViewFavoriteList** (**ShowfotoStackViewFavorites** \*const parent)
  - QString **filter** () const
- Return the current string used to filter the favorites list.*
- **ShowfotoStackViewFavoriteItem** \* **findFavoriteByHierarchy** (const QString &hierarchy)
  - void **setFilter** (const QString &filter, Qt::CaseSensitivity cs)

*Set the string used to filter the favorites list.*

## 9.1470.1 Member Function Documentation

### 9.1470.1.1 setFilter()

```
void ShowFoto::ShowfotoStackViewFavoriteList::setFilter (
    const QString & filter,
    Qt::CaseSensitivity cs )
```

**signalSearchResult()** is emitted when all is done.

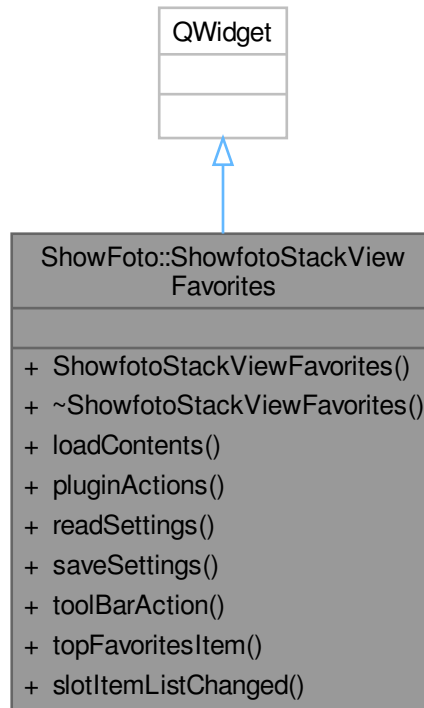
### 9.1470.1.2 signalSearchResult

```
void ShowFoto::ShowfotoStackViewFavoriteList::signalSearchResult (
    int ) [signal]
```

Number of favorites found is sent when item relevant of filtering match the query.

## 9.1471 ShowFoto::ShowfotoStackViewFavorites Class Reference

Inheritance diagram for ShowFoto::ShowfotoStackViewFavorites:



### Public Slots

- void **slotItemListChanged** (int nbitems)

### Signals

- void **signalLoadContents** ()
- void **signalLoadContentsFromFiles** (const QStringList &files, const QString &current)

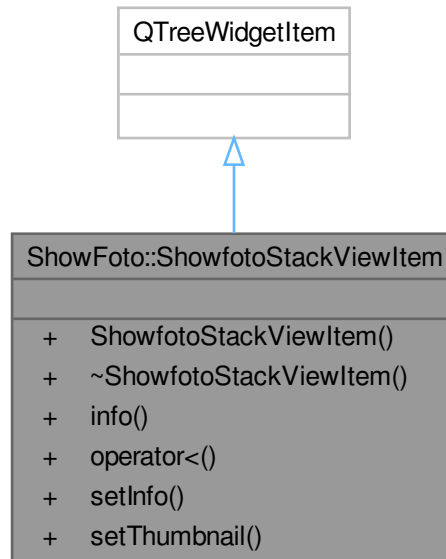
### Public Member Functions

- **ShowfotoStackViewFavorites** ([ShowfotoStackViewSideBar](#) \*const sidebar)
- void **loadContents** ()
- QList< QAction \* > **pluginActions** () const
- bool **readSettings** ()
- bool **saveSettings** ()
- QAction \* **toolBarAction** (const QString &name) const
- QTreeWidgetItem \* **topFavoritesItem** () const



## 9.1472 ShowFoto::ShowfotoStackViewItem Class Reference

Inheritance diagram for ShowFoto::ShowfotoStackViewItem:

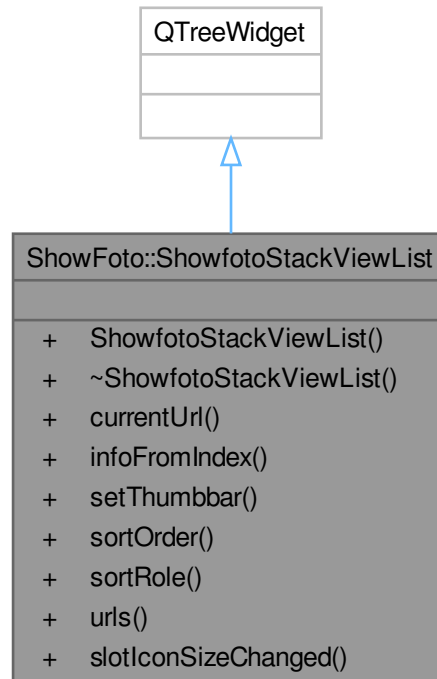


### Public Member Functions

- `ShowfotoStackViewItem` ([ShowfotoStackViewList](#) \*const parent)
- [ShowfotoItemInfo](#) `info` () const
- bool `operator<` (const `QTreeWidgetItem` &other) const override
- void `setInfo` (const [ShowfotoItemInfo](#) &)
- void `setThumbnail` (const `QPixmap` &)

## 9.1473 ShowFoto::ShowfotoStackViewList Class Reference

Inheritance diagram for ShowFoto::ShowfotoStackViewList:



### Public Types

- enum `StackViewRole` { `FileName` = 0 , `FileSize` , `FileType` , `FileDate` }
- enum `ThumbnailSize` { `SizeSmall` = 32 , `SizeMedium` = 48 , `SizeLarge` = 64 , `SizeHuge` = 96 }

### Public Slots

- void `slotIconSizeChanged` (int)

### Signals

- void `signalAddFavorite` ()
- void `signalClearItemsList` ()
- void `signalItemListItemChanged` (int nbitems)
- void `signalRemoveItemInfos` (const QList< `ShowfotoItemInfo` > &infos)
- void `signalShowfotoItemInfoActivated` (const `ShowfotoItemInfo` &info)

## Public Member Functions

- **ShowfotoStackViewList** ([ShowfotoStackViewSideBar](#) \*const view)
- `QUrl` **currentUrl** () const
- [ShowfotoItemInfo](#) **infoFromIndex** (const QModelIndex &index) const
- void **setThumbbar** ([ShowfotoThumbnailBar](#) \*const thumbbar)
- int **sortOrder** () const
- int **sortRole** () const
- `QList< QUrl >` **urls** ()

## 9.1473.1 Member Enumeration Documentation

### 9.1473.1.1 StackViewRole

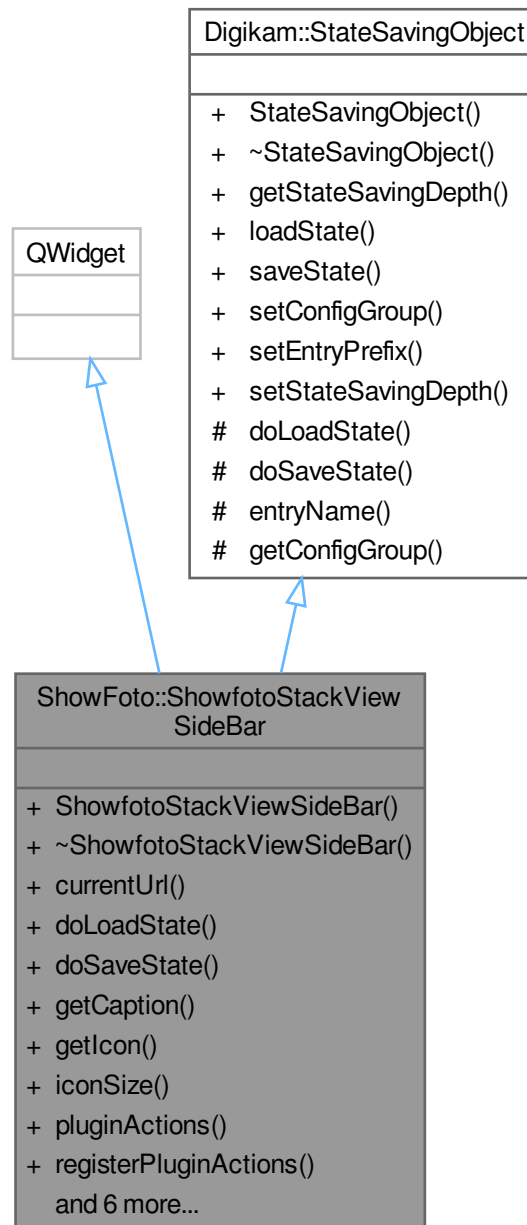
enum `ShowFoto::ShowfotoStackViewList::StackViewRole`

#### Enumerator

FileDate	Metadata date if exists, else Modifier date.
----------	--

## 9.1474 ShowFoto::ShowfotoStackViewSideBar Class Reference

Inheritance diagram for ShowFoto::ShowfotoStackViewSideBar:



### Signals

- void **signalAddFavorite** ()
- void **signalClearItemsList** ()
- void **signalLoadContentsFromFiles** (const QStringList &files, const QString &current)
- void **signalRemoveItemInfos** (const QList< [ShowfotoItemInfo](#) > &infos)
- void **signalShowfotoItemInfoActivated** (const [ShowfotoItemInfo](#) &info)

## Public Member Functions

- **ShowfotoStackViewSideBar** ([Showfoto](#) \*const parent)
- `QUrl` **currentUrl** () const
- void **doLoadState** () override  
*Implement this hook method for state loading.*
- void **doSaveState** () override  
*Implement this hook method for state saving.*
- const `QString` **getCaption** ()
- const `QIcon` **getIcon** ()
- int **iconSize** () const
- `QList< QAction * >` **pluginActions** () const
- void **registerPluginActions** (const `QList< DPluginAction * >` &actions)
- void **setSortOrder** (int order)
- void **setSortRole** (int role)
- void **setThumbbar** ([ShowfotoThumbnailBar](#) \*const thumbbar)
- int **sortOrder** () const
- int **sortRole** () const
- `QList< QUrl >` **urls** () const

## Public Member Functions inherited from [Digikam::StateSavingObject](#)

- [StateSavingObject](#) (`QObject` \*const host)  
*Constructor.*
- virtual `~StateSavingObject` ()  
*Destructor.*
- [StateSavingDepth](#) **getStateSavingDepth** () const  
*Returns the depth used for state saving or loading.*
- void **loadState** ()  
*Invokes loading the class' state.*
- void **saveState** ()  
*Invokes saving the class' state.*
- virtual void **setConfigGroup** (const `KConfigGroup` &group)  
*Sets a dedicated config group that will be used to store and reload the state from.*
- virtual void **setEntryPrefix** (const `QString` &prefix)  
*Define a prefix that will be used for every entry in the config group.*
- void **setStateSavingDepth** (const [StateSavingDepth](#) depth)  
*Sets the depth used for state saving or loading.*

## Additional Inherited Members

## Public Types inherited from [Digikam::StateSavingObject](#)

- enum [StateSavingDepth](#) { `INSTANCE` , `DIRECT_CHILDREN` , `RECURSIVE` }  
*This enum defines the "depth" of the [StateSavingObject::loadState\(\)](#) and [StateSavingObject::saveState\(\)](#) methods.*

## Protected Member Functions inherited from [Digikam::StateSavingObject](#)

- `QString` **entryName** (const `QString` &base) const  
*Always use this method to create config group entry names.*
- `KConfigGroup` **getConfigGroup** () const  
*Returns the config group that must be used for state saving and loading.*

## 9.1474.1 Member Function Documentation

### 9.1474.1.1 doLoadState()

```
void ShowFoto::ShowfotoStackViewSideBar::doLoadState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

### 9.1474.1.2 doSaveState()

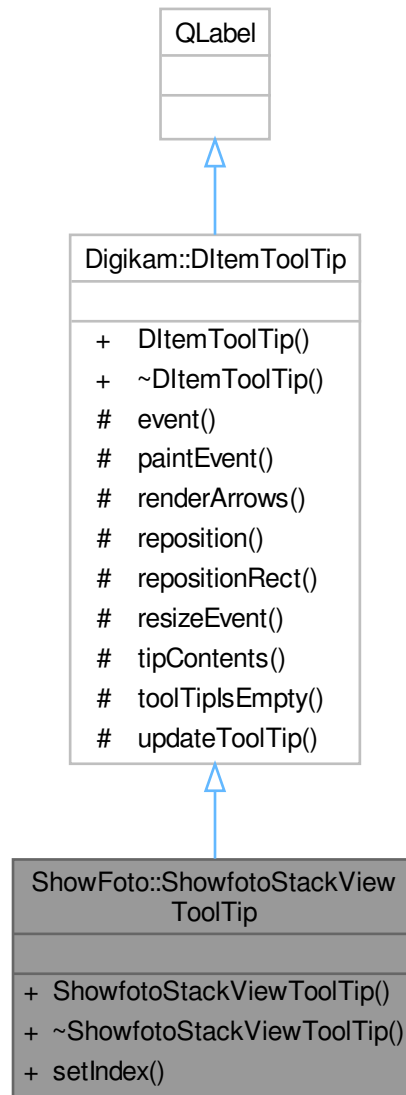
```
void ShowFoto::ShowfotoStackViewSideBar::doSaveState ( ) [override], [virtual]
```

Use [getConfigGroup\(\)](#) and [entryName\(\)](#) for the implementation.

Implements [Digikam::StateSavingObject](#).

## 9.1475 ShowFoto::ShowfotoStackViewToolTip Class Reference

Inheritance diagram for ShowFoto::ShowfotoStackViewToolTip:



### Public Member Functions

- **ShowfotoStackViewToolTip** ([ShowfotoStackViewList](#) \*const view)
- void **setIndex** (const QModelIndex &index)

### Public Member Functions inherited from [Digikam::DItemToolTip](#)

- **DItemToolTip** (QWidget \*const parent=nullptr)

### Additional Inherited Members

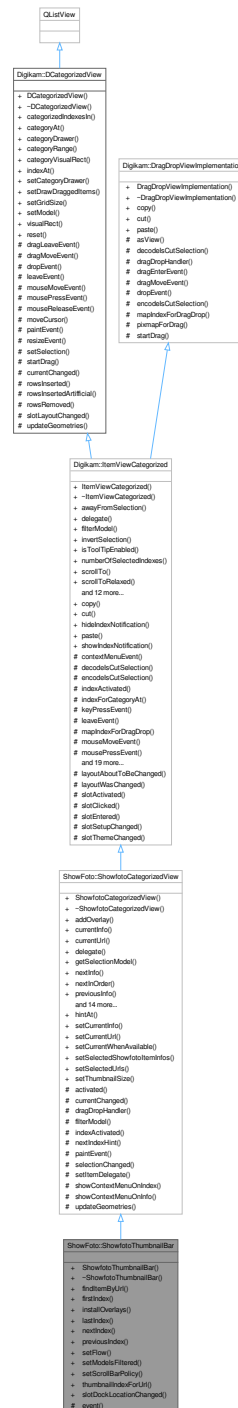
### Protected Member Functions inherited from [Digikam::DItemToolTip](#)

- bool **event** (QEvent \*) override
- void **paintEvent** (QPaintEvent \*) override
- void **renderArrows** ()
- void **reposition** ()
- void **resizeEvent** (QResizeEvent \*) override
- bool **toolTipsEmpty** () const
- void **updateToolTip** ()



## 9.1476 ShowFoto::ShowfotoThumbnailBar Class Reference

Inheritance diagram for ShowFoto::ShowfotoThumbnailBar:



### Public Slots

- void **slotDockLocationChanged** (Qt::DockWidgetArea area)

## Public Slots inherited from [ShowFoto::ShowfotoCategorizedView](#)

- void **hintAt** (const [ShowfotoItemInfo](#) &info)  
*Does something to gain attention for info, but not changing current selection.*
- void **setCurrentInfo** (const [ShowfotoItemInfo](#) &info)  
*Set as current item the item identified by the [ShowfotoItemInfo](#).*
- void **setCurrentUrl** (const QUrl &url)  
*Set as current item the item identified by its file url.*
- void **setCurrentWhenAvailable** (qulonglong ShowfotoItemId)  
*Scroll the view to the given item when it becomes available.*
- void **setSelectedShowfotoItemInfos** (const QList< [ShowfotoItemInfo](#) > &infos)  
*Set selected items.*
- void **setSelectedUrls** (const QList< QUrl > &urlList)  
*Set selected items identified by their file urls.*
- void **setThumbnailSize** (int size)

## Public Slots inherited from [Digikam::ItemViewCategorized](#)

- void **copy** () override
- void **cut** () override
- void **hideIndexNotification** ()
- void **paste** () override
- void **showIndexNotification** (const QModelIndex &index, const QString &message)

## Public Slots inherited from [Digikam::DCategorizedView](#)

- void **reset** () override

## Public Member Functions

- **ShowfotoThumbnailBar** (QWidget \*const parent=nullptr)
- [ShowfotoItemInfo](#) **findItemByUrl** (const QUrl &url)
- QModelIndex **firstIndex** () const
- void **installOverlays** ()
- QModelIndex **lastIndex** () const
- QModelIndex **nextIndex** (const QModelIndex &index) const
- QModelIndex **previousIndex** (const QModelIndex &index) const
- void **setFlow** (QListView::Flow newFlow)
- void **setModelsFiltered** ([ShowfotoItemModel](#) \*model, [ShowfotoSortFilterModel](#) \*filterModel)  
*This installs a duplicate filter model, if the [ShwofotoItemModel](#) may contain duplicates.*
- void **setScrollBarPolicy** (Qt::ScrollBarPolicy policy)  
*Sets the policy always for the one scroll bar which is relevant, depending on orientation.*
- int **thumbnailIndexForUrl** (const QUrl &url) const

## Public Member Functions inherited from ShowFoto::ShowfotoCategorizedView

- **ShowfotoCategorizedView** (QWidget \*const parent=nullptr)
- void **addOverlay** (ItemDelegateOverlay \*overlay, ShowfotoDelegate \*delegate=nullptr)
  - Add and remove an overlay.*
- **ShowfotoItemInfo currentInfo** () const
- QUrl **currentUrl** () const
- ShowfotoDelegate \* **delegate** () const
- QItemSelectionModel \* **getSelectionModel** () const
- ShowfotoItemInfo **nextInfo** (const ShowfotoItemInfo &info)
- ShowfotoItemInfo **nextInOrder** (const ShowfotoItemInfo &startingPoint, int nth)
  - Returns the n-th info after the given one.*
- ShowfotoItemInfo **previousInfo** (const ShowfotoItemInfo &info)
- void **removeOverlay** (ItemDelegateOverlay \*overlay)
- QList< ShowfotoItemInfo > **selectedShowfotoItemInfos** () const
- QList< ShowfotoItemInfo > **selectedShowfotoItemInfosCurrentFirst** () const
- QList< QUrl > **selectedUrls** () const
- void **setModels** (ShowfotoItemModel \*model, ShowfotoSortFilterModel \*filterModel)
- virtual void **setThumbnailSize** (const ThumbnailSize &size)
- ShowfotoFilterModel \* **showfotoFilterModel** () const
  - Returns any ShowfotoFilterModel in chain.*
- QList< ShowfotoItemInfo > **showfotoItemInfos** () const
- ShowfotoItemModel \* **showfotoItemModel** () const
- ShowfotoSortFilterModel \* **showfotoSortFilterModel** () const
- ShowfotoThumbnailModel \* **showfotoThumbnailModel** () const
  - Returns 0 if the ShowfotoItemModel is not an ShowfotoThumbnailModel.*
- ThumbnailSize **thumbnailSize** () const
- void **toIndex** (const QUrl &url)
  - Selects the index as current and scrolls to it.*
- QList< QUrl > **urls** () const

## Public Member Functions inherited from Digikam::ItemViewCategorized

- **ItemViewCategorized** (QWidget \*const parent=nullptr)
- void **awayFromSelection** ()
- DItemDelegate \* **delegate** () const
- void **invertSelection** ()
- bool **isToolTipEnabled** () const
- int **numberOfSelectedIndexes** () const
- void **scrollTo** (const QModelIndex &index, ScrollHint hint=EnsureVisible) override
- void **scrollToRelaxed** (const QModelIndex &index, ScrollHint hint=EnsureVisible)
  - Like scrollTo, but only scrolls if the index is not visible, regardless of hint.*
- void **setInitialSelectedItem** (bool enabled)
  - Ensure a initial selected item.*
- void **setScrollCurrentToCenter** (bool enabled)
  - Scroll automatically the current index to center of the view.*
- void **setScrollStepGranularity** (int factor)
  - Determine a step size for scrolling: The larger this number, the smaller and more precise is the scrolling.*
- void **setSelectedIndexes** (const QList< QModelIndex > &indexes)
- void **setSpacing** (int spacing)
  - Sets the spacing.*
- void **setToolTipEnabled** (bool enabled)

- void **setUsePointingHandCursor** (bool useCursor)  
*Set if the PointingHand Cursor should be shown over the activation area.*
- void **toFirstIndex** ()  
*Selects the index as current and scrolls to it.*
- void **toIndex** (const QModelIndex &index)
- void **toLastIndex** ()
- void **toNextIndex** ()
- void **toPreviousIndex** ()

### Public Member Functions inherited from [Digikam::DCategorizedView](#)

- **DCategorizedView** (QWidget \*const parent=nullptr)
- virtual QModelIndexList **categorizedIndexesIn** (const QRect &rect) const  
*This method will return all indexes whose visual rect intersects *rect*.*
- virtual QModelIndex **categoryAt** (const QPoint &point) const  
*This method will return the first index of the category in the region of which *point* is found.*
- **DCategoryDrawer** \* **categoryDrawer** () const
- virtual QItemSelectionRange **categoryRange** (const QModelIndex &index) const  
*This method returns the range of indexes contained in the category in which *index* is sorted.*
- virtual QRect **categoryVisualRect** (const QModelIndex &index) const  
*This method will return the visual rect of the header of the category in which *index* is sorted.*
- QModelIndex **indexAt** (const QPoint &point) const override
- void **setCategoryDrawer** (**DCategoryDrawer** \*categoryDrawer)
- void **setDrawDraggedItems** (bool drawDraggedItems)  
*Switch on drawing of dragged items.*
- void **setGridSize** (const QSize &size)
- void **setModel** (QAbstractItemModel \*model) override
- QRect **visualRect** (const QModelIndex &index) const override

### Public Member Functions inherited from [Digikam::DragDropViewImplementation](#)

- virtual void **copy** ()
- virtual void **cut** ()
- virtual void **paste** ()

### Protected Member Functions

- bool **event** (QEvent \*) override

## Protected Member Functions inherited from ShowFoto::ShowfotoCategorizedView

- virtual void **activated** (const ShowfotoItemInfo &info, Qt::KeyboardModifiers modifiers)
  - Reimplement these in a subclass.*
- void **currentChanged** (const QModelIndex &index, const QModelIndex &previous) override
- AbstractItemDragDropHandler \* **dragDropHandler** () const override
  - You need to implement these three methods Returns the drag drop handler.*
- QSortFilterProxyModel \* **filterModel** () const override
  - reimplemented from parent class*
- void **indexActivated** (const QModelIndex &index, Qt::KeyboardModifiers modifiers) override
- QModelIndex **nextIndexHint** (const QModelIndex &indexToAnchor, const QItemSelectionRange &removed) const override
  - Assuming the given indexes would be removed (hypothetically!), return the index to be selected instead, starting from anchor.*
- void **paintEvent** (QPaintEvent \*e) override
- void **selectionChanged** (const QItemSelection &, const QItemSelection &) override
- void **setItemDelegate** (ShowfotoDelegate \*delegate)
- void **showContextMenuOnIndex** (QContextMenuEvent \*event, const QModelIndex &index) override
  - Reimplement these in a subclass.*
- virtual void **showContextMenuOnInfo** (QContextMenuEvent \*event, const ShowfotoItemInfo &info)
- void **updateGeometries** () override

## Protected Member Functions inherited from Digikam::ItemViewCategorized

- void **contextMenuEvent** (QContextMenuEvent \*event) override
  - reimplemented from parent class*
- bool **decodelsCutSelection** (const QMimeData \*mimeData)
- void **encodelsCutSelection** (QMimeData \*mime, bool isCutSelection)
- QModelIndex **indexForCategoryAt** (const QPoint &pos) const
  - Returns an index that is representative for the category at position pos.*
- void **keyPressEvent** (QKeyEvent \*event) override
- void **leaveEvent** (QEvent \*event) override
- QModelIndex **mapIndexForDragDrop** (const QModelIndex &index) const override
  - Note: pure virtual dragDropHandler() still open from DragDropViewImplementation.*
- void **mouseMoveEvent** (QMouseEvent \*event) override
- void **mousePressEvent** (QMouseEvent \*event) override
- void **mouseReleaseEvent** (QMouseEvent \*event) override
- QModelIndex **moveCursor** (CursorAction cursorAction, Qt::KeyboardModifiers modifiers) override
- QPixmap **pixmapForDrag** (const QList< QModelIndex > &indexes) const override
  - Creates a pixmap for dragging the given indexes.*
- void **reset** () override
- void **resizeEvent** (QResizeEvent \*e) override
- void **rowsAboutToBeRemoved** (const QModelIndex &parent, int start, int end) override
- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- void **rowsRemoved** (const QModelIndex &parent, int start, int end) override
- void **selectionChanged** (const QItemSelection &, const QItemSelection &) override
- void **setItemDelegate** (DItemDelegate \*delegate)
- void **setToolTip** (ItemViewToolTip \*tip)
- virtual void **showContextMenu** (QContextMenuEvent \*event)
- virtual bool **showToolTip** (const QModelIndex &index, QStyleOptionViewItem &option, QHelpEvent \*e=nullptr)
  - Provides default behavior, can reimplement in a subclass.*
- void **updateDelegateSizes** ()
- void **userInteraction** ()
- bool **viewportEvent** (QEvent \*event) override
- void **wheelEvent** (QWheelEvent \*event) override

## Protected Member Functions inherited from [Digikam::DCategorizedView](#)

- void **dragLeaveEvent** (QDragLeaveEvent \*event) override
- void **dragMoveEvent** (QDragMoveEvent \*event) override
- void **dropEvent** (QDropEvent \*event) override
- void **leaveEvent** (QEvent \*event) override
- void **mouseMoveEvent** (QMouseEvent \*event) override
- void **mousePressEvent** (QMouseEvent \*event) override
- void **mouseReleaseEvent** (QMouseEvent \*event) override
- QModelIndex **moveCursor** (CursorAction cursorAction, Qt::KeyboardModifiers modifiers) override
- void **paintEvent** (QPaintEvent \*event) override
- void **resizeEvent** (QResizeEvent \*event) override
- void **setSelection** (const QRect &rect, QItemSelectionModel::SelectionFlags flags) override
- void **startDrag** (Qt::DropActions supportedActions) override

## Protected Member Functions inherited from [Digikam::DragDropViewImplementation](#)

- virtual QAbstractItemView \* **asView** ()=0  
*This one is implemented by DECLARE\_VIEW\_DRAG\_DROP\_METHODS.*
- bool **decodelsCutSelection** (const QMimeData \*mimeData)
- void **dragEnterEvent** (QDragEnterEvent \*event)  
*Implements the relevant QAbstractItemView methods for drag and drop.*
- void **dragMoveEvent** (QDragMoveEvent \*e)
- void **dropEvent** (QDropEvent \*e)
- void **encodelsCutSelection** (QMimeData \*mime, bool isCutSelection)
- void **startDrag** (Qt::DropActions supportedActions)

## Additional Inherited Members

## Signals inherited from [ShowFoto::ShowfotoCategorizedView](#)

- void **currentChanged** (const [ShowfotoItemInfo](#) &info)
- void **deselected** (const QList< [ShowfotoItemInfo](#) > &nowDeselectedInfos)  
*Emitted when items are deselected.*
- void **modelChanged** ()  
*Emitted when a new model is set.*
- void **selected** (const QList< [ShowfotoItemInfo](#) > &newSelectedInfos)  
*Emitted when new items are selected.*
- void **showfotoItemInfoActivated** (const [ShowfotoItemInfo](#) &info)  
*Emitted when the given [ShowfotoItemInfo](#) is activated.*

## Signals inherited from [Digikam::ItemViewCategorized](#)

- void **clicked** (const QMouseEvent \*e, const QModelIndex &index)  
*For overlays: Like the respective parent class signals, but with additional info.*
- void **entered** (const QMouseEvent \*e, const QModelIndex &index)
- void **keyPressed** (QKeyEvent \*e)  
*Remember you may want to check if the event is accepted or ignored.*
- void **selectionChanged** ()  
*Emitted when any selection change occurs.*
- void **selectionCleared** ()  
*Emitted when the selection is completely cleared.*
- void **viewportClicked** (const QMouseEvent \*e)  
*While [clicked\(\)](#) is emitted with a valid index, this corresponds to clicking on empty space.*
- void **zoomInStep** ()
- void **zoomOutStep** ()

## Protected Slots inherited from [Digikam::ItemViewCategorized](#)

- void **layoutAboutToBeChanged** ()
- void **layoutWasChanged** ()
- void **slotActivated** (const QModelIndex &index)
- void **slotClicked** (const QModelIndex &index)
- void **slotEntered** (const QModelIndex &index)
- virtual void **slotSetupChanged** ()
- virtual void **slotThemeChanged** ()

## Protected Slots inherited from [Digikam::DCategorizedView](#)

- void **currentChanged** (const QModelIndex &current, const QModelIndex &previous) override
- void **rowsInserted** (const QModelIndex &parent, int start, int end) override
- virtual void **rowsInsertedArtificial** (const QModelIndex &parent, int start, int end)
- virtual void **slotLayoutChanged** ()
- void **updateGeometries** () override

## 9.1476.1 Member Function Documentation

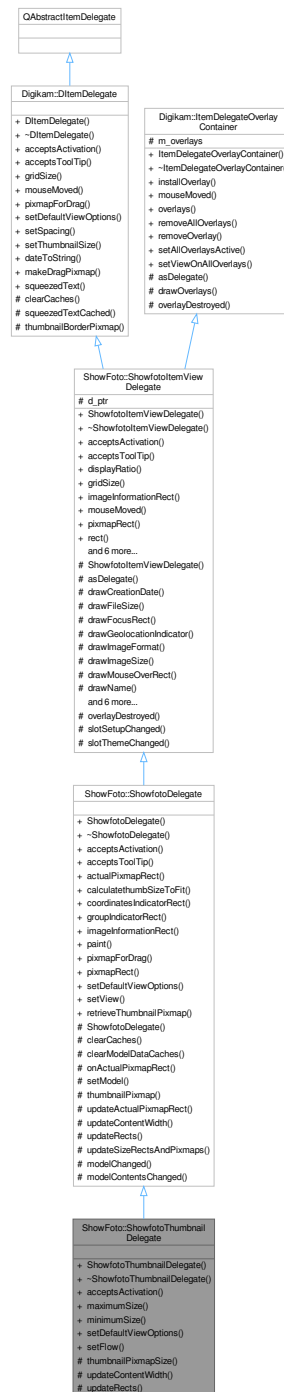
### 9.1476.1.1 setModelsFiltered()

```
void ShowFoto::ShowfotoThumbnailBar::setModelsFiltered (
    ShowfotoItemModel * model,
    ShowfotoSortFilterModel * filterModel )
```

Otherwise, just use setModels().

## 9.1477 ShowFoto::ShowfotoThumbnailDelegate Class Reference

Inheritance diagram for ShowFoto::ShowfotoThumbnailDelegate:



### Public Member Functions

- `ShowfotoThumbnailDelegate` (`ShowfotoThumbnailBar` \*const parent)
- bool `acceptsActivation` (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*activationRect) const override



- int **maximumSize** () const  
*Returns the minimum or maximum viewport size in the limiting dimension, width or height, depending on current flow.*
- int **minimumSize** () const
- void **setDefaultViewOptions** (const QStyleOptionViewItem &option) override  
*Style option with standard values to use for cached rendering.*
- void **setFlow** (QListView::Flow flow)

## Public Member Functions inherited from ShowFoto::ShowfotoDelegate

- **ShowfotoDelegate** (QWidget \*const parent)
- bool **acceptsToolTip** (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*tooltipRect=nullptr) const override  
*These methods take four parameters: The position on viewport, the rect on viewport, the index, and optionally a parameter into which, if the return value is true, a rectangle can be written for which the return value will be true as well.*
- QRect **actualPixmapRect** (const QModelIndex &index) const
- int **calculatethumbSizeToFit** (int ws)
- QRect **coordinatesIndicatorRect** () const
- QRect **groupIndicatorRect** () const
- QRect **imageInformationRect** () const override  
*Returns the area where the image information is drawn, or null if empty / not supported.*
- void **paint** (QPainter \*painter, const QStyleOptionViewItem &option, const QModelIndex &index) const override
- QPixmap **pixmapForDrag** (const QStyleOptionViewItem &option, const QList< QModelIndex > &indexes) const override
- QRect **pixmapRect** () const override  
*Returns the area where the pixmap is drawn, or null if not supported.*
- void **setView** (ShowfotoThumbnailBar \*view)

## Public Member Functions inherited from ShowFoto::ShowfotoItemViewDelegate

- **ShowfotoItemViewDelegate** (QWidget \*const parent)
- bool **acceptsToolTip** (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*tooltipRect=nullptr) const override  
*These methods take four parameters: The position on viewport, the rect on viewport, the index, and optionally a parameter into which, if the return value is true, a rectangle can be written for which the return value will be true as well.*
- double **displayRatio** () const
- QSize **gridSize** () const override  
*Returns the gridsize to be set by the view.*
- void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index) override
- QRect **rect** () const
- void **setSpacing** (int spacing) override
- void **setThumbnailSize** (const ThumbnailSize &thumbSize) override  
*reimplemented from DItemDelegate*
- QSize **sizeHint** (const QStyleOptionViewItem &option, const QModelIndex &index) const override
- int **spacing** () const
- ThumbnailSize **thumbnailSize** () const

## Public Member Functions inherited from Digikam::DItemDelegate

- **DItemDelegate** (QObject \*const parent=nullptr)

## Public Member Functions inherited from [Digikam::ItemDelegateOverlayContainer](#)

- [ItemDelegateOverlayContainer](#) ()=default  
*This is a sample implementation for delegate management methods, to be inherited by a delegate.*
- void **installOverlay** ([ItemDelegateOverlay](#) \*overlay)
- void **mouseMoved** (QMouseEvent \*e, const QRect &visualRect, const QModelIndex &index)
- QList< [ItemDelegateOverlay](#) \* > **overlays** () const
- void **removeAllOverlays** ()
- void **removeOverlay** ([ItemDelegateOverlay](#) \*overlay)
- void **setAllOverlaysActive** (bool active)
- void **setViewOnAllOverlays** (QAbstractItemView \*view)

## Protected Member Functions

- int **thumbnailPixmapSize** (bool withHighlight, int size)
- void **updateContentWidth** () override  
*Reimplement this to set contentWidth.*
- void **updateRects** () override  
*In a subclass, you need to implement this method to set up the rects for drawing.*

## Protected Member Functions inherited from [ShowFoto::ShowfotoDelegate](#)

- **ShowfotoDelegate** (ShowfotoDelegate::ShowfotoDelegatePrivate &dd, QWidget \*const parent)
- void **clearCaches** () override
- virtual void **clearModelDataCaches** ()  
*Reimplement to clear caches based on model indexes (hash on row number etc.) Change signals are listened to this is called whenever such properties become invalid.*
- bool **onActualPixmapRect** (const QPoint &pos, const QRect &visualRect, const QModelIndex &index, QRect \*actualRect) const
- void **setModel** (QAbstractItemModel \*model)
- virtual QPixmap **thumbnailPixmap** (const QModelIndex &index) const
- void **updateActualPixmapRect** (const QModelIndex &index, const QRect &rect)
- void **updateSizeRectsAndPxmmaps** () override

## Protected Member Functions inherited from [ShowFoto::ShowfotoItemViewDelegate](#)

- **ShowfotoItemViewDelegate** (ShowfotoItemViewDelegatePrivate &dd, QWidget \*const parent)
- QAbstractItemDelegate \* **asDelegate** () override  
*Returns the delegate, typically, the derived class.*
- void **drawCreationDate** (QPainter \*p, const QRect &dateRect, const QDateTime &date) const
- void **drawFileSize** (QPainter \*p, const QRect &r, qlonglong bytes) const
- void **drawFocusRect** (QPainter \*p, const QStyleOptionViewItem &option, bool isSelected) const
- void **drawGeolocationIndicator** (QPainter \*p, const QRect &r) const
- void **drawImageFormat** (QPainter \*p, const QRect &dimsRect, const QString &mime) const
- void **drawImageSize** (QPainter \*p, const QRect &dimsRect, const QSize &dims) const
- void **drawMouseOverRect** (QPainter \*p, const QStyleOptionViewItem &option) const
- void **drawName** (QPainter \*p, const QRect &nameRect, const QString &name) const
- QRect **drawThumbnail** (QPainter \*p, const QRect &thumbRect, const QPixmap &background, const QPixmap &thumbnail) const  
*Use the tool methods for painting in subclasses.*
- virtual void **invalidatePaintingCache** ()  
*reimplement these in subclasses*
- void **prepareBackground** ()
- void **prepareFonts** ()
- void **prepareMetrics** (int maxWidth)

### Protected Member Functions inherited from [Digikam::DItemDelegate](#)

- QString **squeezedTextCached** (QPainter \*const p, int width, const QString &text) const
- QPixmap **thumbnailBorderPixmap** (const QSize &pixSize, bool isGrouped=false) const

### Protected Member Functions inherited from [Digikam::ItemDelegateOverlayContainer](#)

- virtual void **drawOverlays** (QPainter \*p, const QStyleOptionViewItem &option, const QModelIndex &index) const
- virtual void **overlayDestroyed** (QObject \*o)

*Declare as slot in the derived class calling this method.*

### Additional Inherited Members

### Signals inherited from [ShowFoto::ShowfotoItemViewDelegate](#)

- void **hideNotification** ()
- void **requestNotification** (const QModelIndex &index, const QString &message)

### Signals inherited from [Digikam::DItemDelegate](#)

- void **gridSizeChanged** (const QSize &newSize)
- void **visualChange** ()

### Static Public Member Functions inherited from [ShowFoto::ShowfotoDelegate](#)

- static QPixmap **retrieveThumbnailPixmap** (const QModelIndex &index, int thumbnailSize)  
*Retrieve the thumbnail pixmap in given size for the [ShowfotoItemModel::ThumbnailRole](#) for the given index from the given index, which must adhere to [ShowfotoThumbnailModel](#) semantics.*

### Static Public Member Functions inherited from [Digikam::DItemDelegate](#)

- static QString **dateToString** (const QDateTime &datetime)
- static QPixmap **makeDragPixmap** (const QStyleOptionViewItem &option, const QList< QModelIndex > &indexes, double displayRatio, const QPixmap &suggestedPixmap=QPixmap())
- static QString **squeezedText** (const QFontMetrics &fm, int width, const QString &text)

### Protected Slots inherited from [ShowFoto::ShowfotoDelegate](#)

- void **modelChanged** ()
- void **modelContentsChanged** ()

### Protected Slots inherited from [ShowFoto::ShowfotoItemViewDelegate](#)

- void **overlayDestroyed** (QObject \*o) override
- void **slotSetupChanged** ()
- void **slotThemeChanged** ()

## Protected Attributes inherited from [ShowFoto::ShowfotoItemViewDelegate](#)

- ShowfotoItemViewDelegatePrivate \*const **d\_ptr** = nullptr

## Protected Attributes inherited from [Digikam::ItemDelegateOverlayContainer](#)

- QList< [ItemDelegateOverlay](#) \* > **m\_overlays**

### 9.1477.1 Member Function Documentation

#### 9.1477.1.1 `acceptsActivation()`

```
bool ShowFoto::ShowfotoThumbnailDelegate::acceptsActivation (
    const QPoint & pos,
    const QRect & visualRect,
    const QModelIndex & index,
    QRect * activationRect ) const [override], [virtual]
```

Reimplemented from [ShowFoto::ShowfotoDelegate](#).

#### 9.1477.1.2 `setDefaultViewOptions()`

```
void ShowFoto::ShowfotoThumbnailDelegate::setDefaultViewOptions (
    const QStyleOptionViewItem & option ) [override], [virtual]
```

`option.rect` shall be the viewport rectangle. Call on resize, font change.

Reimplemented from [ShowFoto::ShowfotoDelegate](#).

#### 9.1477.1.3 `updateContentWidth()`

```
void ShowFoto::ShowfotoThumbnailDelegate::updateContentWidth ( ) [override], [protected],
[virtual]
```

This is the maximum width of all content rectangles, typically excluding margins on both sides.

Reimplemented from [ShowFoto::ShowfotoDelegate](#).

#### 9.1477.1.4 `updateRects()`

```
void ShowFoto::ShowfotoThumbnailDelegate::updateRects ( ) [override], [protected], [virtual]
```

The `paint()` method operates depending on these `rects`.

Implements [ShowFoto::ShowfotoDelegate](#).

## 9.1478 ShowFoto::ShowfotoThumbnailModel Class Reference

Inheritance diagram for ShowFoto::ShowfotoThumbnailModel:



### Public Slots

- void **slotThumbInfoLoaded** (const [ShowfotoItemInfo](#) &info, const QImage &thumbnailImage)

## Public Slots inherited from [ShowFoto::ShowfotoItemModel](#)

- void **reAddingFinished** ()
- void **reAddShowfotoItemInfos** (const ShowfotoItemInfoList &infos)
- void **slotFileDeleted** (const QString &folder, const QString &file, bool status)
- void **slotFileUploaded** (const [ShowfotoItemInfo](#) &info)

## Signals

- void **signalItemThumbnail** (const [ShowfotoItemInfo](#) &info, const QPixmap &pix)
- void **signalThumbInfo** (const [ShowfotoItemInfo](#) &info, const QImage &thumbnailImage) const
- void **thumbnailAvailable** (const QModelIndex &index, int requestedSize)
- void **thumbnailFailed** (const QModelIndex &index, int requestedSize)

## Signals inherited from [ShowFoto::ShowfotoItemModel](#)

- void **allRefreshingFinished** ()  
*Signals that the model has finished currently with all scheduled refreshing, full or incremental, and all preprocessing.*
- void **itemInfosAboutToBeAdded** (const QList< [ShowfotoItemInfo](#) > &infos)  
*Informs that ItemInfos will be added to the model.*
- void **itemInfosAboutToBeRemoved** (const QList< [ShowfotoItemInfo](#) > &infos)  
*Informs that ShowfotoItemInfos will be removed from the model.*
- void **itemInfosAdded** (const QList< [ShowfotoItemInfo](#) > &infos)  
*Informs that ItemInfos have been added to the model.*
- void **itemInfosRemoved** (const QList< [ShowfotoItemInfo](#) > &infos)  
*Informs that ShowfotoItemInfos have been removed from the model.*
- void **preprocess** (const QList< [ShowfotoItemInfo](#) > &infos)  
*Connect to this signal only if you are the current preprocessor.*
- void **processAdded** (const QList< [ShowfotoItemInfo](#) > &infos)
- void **readyForIncrementalRefresh** ()  
*Signals that the model is right now ready to start an incremental refresh.*

## Public Member Functions

- [ShowfotoThumbnailModel](#) (QWidget \*const parent)  
*An ItemModel that supports thumbnail loading.*
- QVariant **data** (const QModelIndex &index, int role=Qt::DisplayRole) const override  
*Handles the ThumbnailRole.*
- bool **getThumbnail** (const [ShowfotoItemInfo](#) &itemInfo, QImage &thumbnail) const
- bool **ixmapForItem** (const QString &url, QPixmap &pix) const
- bool **setData** (const QModelIndex &index, const QVariant &value, int role=Qt::DisplayRole) override  
*You can override the current thumbnail size by giving an integer value for ThumbnailRole.*
- void **setEmitDataChanged** (bool emitSignal)  
*Enable emitting dataChanged() when a thumbnail becomes available.*
- void **setExifRotate** (bool rotate)
- void **setPreloadThumbnails** (bool preload)  
*Enable preloading of thumbnails: If preloading is enabled, for every entry in the model a thumbnail generation is started.*
- void **setPreloadThumbnailSize** (const [ThumbnailSize](#) &thumbSize)  
*If you want to fix a size for preloading, do it here.*
- void **setThumbnailLoadThread** ([ThumbnailLoadThread](#) \*thread)  
*Enable thumbnail loading and set the thread that shall be used.*
- void **setThumbnailSize** (const [ThumbnailSize](#) &thumbSize)  
*Set the thumbnail size to use.*
- [ThumbnailLoadThread](#) \* **thumbnailLoadThread** () const
- [ThumbnailSize](#) **thumbnailSize** () const

## Public Member Functions inherited from ShowFoto::ShowfotoItemModel

- **ShowfotoItemModel** (QObject \*const parent)
- void **addShowfotoItemInfo** (const ShowfotoItemInfo &info)
- void **addShowfotoItemInfos** (const QList< ShowfotoItemInfo > &infos)
- void **addShowfotoItemInfosSynchronously** (const QList< ShowfotoItemInfo > &infos)
- void **addShowfotoItemInfoSynchronously** (const ShowfotoItemInfo &info)
  - addShowfotoItemInfo() is asynchronous if a preprocessor is set.*
- void **clearShowfotoItemInfos** ()
  - Clears the ShowfotoItemInfos and resets the model.*
- QVariant **data** (const QModelIndex &index, int role) const override
- Qt::ItemFlags **flags** (const QModelIndex &index) const override
- bool **hasImage** (const ShowfotoItemInfo &info) const
- bool **hasImage** (qulonglong id) const
- QVariant **headerData** (int section, Qt::Orientation orientation, int role) const override
- QModelIndex **index** (int row, int column, const QModelIndex &parent) const override
- QList< QModelIndex > **indexesForShowfotoItemId** (qulonglong id) const
- QList< QModelIndex > **indexesForShowfotoItemInfo** (const ShowfotoItemInfo &info) const
- QList< QModelIndex > **indexesForUrl** (const QUrl &fileUrl) const
- QModelIndex **indexForShowfotoItemId** (qulonglong id) const
- QModelIndex **indexForShowfotoItemInfo** (const ShowfotoItemInfo &info) const
  - Return the index of a given ShowfotoItemInfo, if it exists in the model.*
- QModelIndex **indexForUrl** (const QUrl &fileUrl) const
  - Returns the index or ShowfotoItemInfo object from the underlying data for the given file url.*
- bool **isEmpty** () const
- int **numberOfIndexesForShowfotoItemId** (qulonglong id) const
- int **numberOfIndexesForShowfotoItemInfo** (const ShowfotoItemInfo &info) const
- void **removeIndex** (const QModelIndex &index)
  - Remove the given infos or indexes directly from the model.*
- void **removeIndexes** (const QList< QModelIndex > &indexes)
- void **removeShowfotoItemInfo** (const ShowfotoItemInfo &info)
- void **removeShowfotoItemInfos** (const QList< ShowfotoItemInfo > &infos)
- int **rowCount** (const QModelIndex &parent) const override
  - QAbstractListModel implementations.*
- void **setKeepsFileUrlCache** (bool keepCache)
  - If a cache is kept, lookup by file path is fast, without a cache it is O(n).*
- DECLARE\_MODEL\_DRAG\_DROP\_METHODS void **setSendRemovalSignals** (bool send)
  - DragDrop methods.*
- void **setShowfotoItemInfos** (const QList< ShowfotoItemInfo > &infos)
  - Clears and adds infos.*
- qulonglong **showfotoItemId** (const QModelIndex &index) const
- qulonglong **showfotoItemid** (int row) const
- QList< qulonglong > **showfotoItemIds** () const
- QList< qulonglong > **showfotoItemIds** (const QList< QModelIndex > &indexes) const
- ShowfotoItemInfo **showfotoItemInfo** (const QModelIndex &index) const
  - Returns the ShowfotoItemInfo object, reference from the underlying data pointed to by the index.*
- ShowfotoItemInfo **showfotoItemInfo** (const QUrl &fileUrl) const
- ShowfotoItemInfo **showfotoItemInfo** (int row) const
  - Returns the ShowfotoItemInfo object, reference from the underlying data of the given row (parent is the invalid QModelIndex, column is 0).*
- ShowfotoItemInfo & **showfotoItemInfoRef** (const QModelIndex &index) const
- ShowfotoItemInfo & **showfotoItemInfoRef** (int row) const
- QList< ShowfotoItemInfo > **showfotoItemInfos** () const
- ShowfotoItemInfoList **showfotoItemInfos** (const QList< QModelIndex > &indexes) const
- QList< ShowfotoItemInfo > **showfotoItemInfos** (const QUrl &fileUrl) const
- QList< ShowfotoItemInfo > **uniqueShowfotoItemInfos** () const

## Public Member Functions inherited from [Digikam::DragDropModelImplementation](#)

- [DragDropModelImplementation](#) ()=default  
*A class providing a sample implementation for a QAbstractItemModel redirecting drag-and-drop support to a handler.*
- virtual Qt::ItemFlags [dragDropFlags](#) (const QModelIndex &index) const  
*Call from your flags() method, adding the relevant drag drop flags.*
- Qt::ItemFlags [dragDropFlagsV2](#) (const QModelIndex &index) const  
*This is an alternative approach to [dragDropFlags\(\)](#).*
- [AbstractItemDragDropHandler](#) \* **dragDropHandler** () const
- bool **dropMimeData** (const QMimeData \*, Qt::DropAction, int, int, const QModelIndex &)
- virtual bool **isDragEnabled** (const QModelIndex &index) const
- virtual bool **isDropEnabled** (const QModelIndex &index) const
- QMimeData \* **mimeData** (const QModelIndexList &indexes) const
- QStringList **mimeTypes** () const
- void **setDragDropHandler** ([AbstractItemDragDropHandler](#) \*handler)  
*Set a drag drop handler.*
- Qt::DropActions [supportedDropActions](#) () const  
*Implements the relevant QAbstractItemModel methods for drag and drop.*

## Protected Slots

- void **slotThumbnailLoaded** (const [LoadingDescription](#) &loadingDescription, const QPixmap &thumb)

## Protected Member Functions

- void [showfotoItemInfosCleared](#) () override  
*Called when the internal storage is cleared.*

## Protected Member Functions inherited from [ShowFoto::ShowfotoItemModel](#)

- void **emitDataChangedForAll** ()
- void **emitDataChangedForSelections** (const QListItemSelection &selection)
- void **finishIncrementalRefresh** ()
- void [requestIncrementalRefresh](#) ()  
*As soon as the model is ready to start an incremental refresh, the signal [readyForIncrementalRefresh\(\)](#) will be emitted.*
- virtual void **showfotoItemInfosAboutToBeRemoved** (int, int)  
*Called before rowsAboutToBeRemoved.*
- void [startIncrementalRefresh](#) ()  
*Starts an incremental refresh operation.*

## Additional Inherited Members

## Public Types inherited from [ShowFoto::ShowfotoItemModel](#)

- enum [ShowfotoItemModelRoles](#) {  
[ShowfotoItemModelPointerRole](#) = Qt::UserRole , **ShowfotoItemModelInternalId** = Qt::UserRole + 1 ,  
[ThumbnailRole](#) = Qt::UserRole + 2 , [ExtraDataRole](#) = Qt::UserRole + 3 ,  
[ExtraDataDuplicateCount](#) = Qt::UserRole + 6 , **FilterModelRoles** = Qt::UserRole + 100 }



## Static Public Member Functions inherited from ShowFoto::ShowfotoItemModel

- static qlonglong **retrieveShowfotoItemId** (const QModelIndex &index)
- static [ShowfotoItemInfo](#) **retrieveShowfotoItemInfo** (const QModelIndex &index)

Retrieve the [ShowfotoItemInfo](#) object from the `data()` function of the given index. The index may be from a `QSortFilterProxyModel` as long as an [ShowfotoItemModel](#) is at the end.

## Protected Attributes inherited from Digikam::DragDropModelImplementation

- [AbstractItemDragDropHandler](#) \* `m_dragDropHandler` = nullptr

## 9.1478.1 Constructor & Destructor Documentation

### 9.1478.1.1 ShowfotoThumbnailModel()

```
ShowFoto::ShowfotoThumbnailModel::ShowfotoThumbnailModel (
    QWidget *const parent ) [explicit]
```

You need to set a `ThumbnailLoadThread` to enable thumbnail loading. Adjust the thumbnail size to your needs. Note that `setKeepsFilePatindexesForPathCache` is enabled per default.

## 9.1478.2 Member Function Documentation

### 9.1478.2.1 data()

```
QVariant ShowFoto::ShowfotoThumbnailModel::data (
    const QModelIndex & index,
    int role = Qt::DisplayRole ) const [override]
```

If the pixmap is available, returns it in the `QVariant`. If it still needs to be loaded, returns a null `QVariant` and emits `thumbnailAvailable()` as soon as it is available.

### 9.1478.2.2 setData()

```
bool ShowFoto::ShowfotoThumbnailModel::setData (
    const QModelIndex & index,
    const QVariant & value,
    int role = Qt::DisplayRole ) [override]
```

Set a null `QVariant` to use the thumbnail size set by [setThumbnailSize\(\)](#) again. The index given here is ignored for this purpose.

### 9.1478.2.3 setEmitDataChanged()

```
void ShowFoto::ShowfotoThumbnailModel::setEmitDataChanged (
    bool emitSignal )
```

The `thumbnailAvailable()` signal will be emitted in any case. Default is true.

#### 9.1478.2.4 setPreloadThumbnails()

```
void ShowFoto::ShowfotoThumbnailModel::setPreloadThumbnails (
    bool preload )
```

Default: false.

#### 9.1478.2.5 setThumbnailLoadThread()

```
void ShowFoto::ShowfotoThumbnailModel::setThumbnailLoadThread (
    ThumbnailLoadThread * thread )
```

The thumbnail size of this thread will be adjusted.

#### 9.1478.2.6 showfotoItemInfosCleared()

```
void ShowFoto::ShowfotoThumbnailModel::showfotoItemInfosCleared ( ) [override], [protected],
[virtual]
```

Reimplemented from [ShowFoto::ShowfotoItemModel](#).

# Index

- ~ActionJob
  - Digikam::ActionJob, [218](#)
- ~Album
  - Digikam::Album, [257](#)
- ~BackendGoogleMaps
  - Digikam::BackendGoogleMaps, [436](#)
- ~BackendMarble
  - Digikam::BackendMarble, [444](#)
- ~ExifToolProcess
  - Digikam::ExifToolProcess, [1391](#)
- ~MapWidget
  - Digikam::MapWidget, [2430](#)
- ~VersionsTreeView
  - Digikam::VersionsTreeView, [3367](#)
- abortInitialization
  - Digikam::ScanController, [2810](#)
- aboutToDeactivate
  - Digikam::RecognitionWorker, [2746](#)
  - Digikam::TrainerWorker, [3317](#)
  - Digikam::WorkerObject, [3397](#)
- AboutToEditMetadata
  - Digikam::ItemMetadataAdjustmentHint, [2159](#)
- aboutToQuitLoop
  - Digikam::WorkerObject, [3397](#)
- aboutToSetInfo
  - Digikam::FaceGroup, [1416](#)
- aboutToShowContextMenu
  - Digikam::DDateTable, [903](#)
- absoluteToRelative
  - Digikam::TagRegion, [3180](#)
- AbstractAlbumModel
  - Digikam::AbstractAlbumModel, [151](#)
- AbstractAlbumTreeView
  - Digikam::AbstractAlbumTreeView, [158](#)
- AbstractAlbumTreeViewSelectComboBox
  - Digikam::AbstractAlbumTreeViewSelectComboBox, [167](#)
- AbstractCheckableAlbumModel
  - Digikam::AbstractCheckableAlbumModel, [174](#)
- AbstractCheckableAlbumTreeView
  - Digikam::AbstractCheckableAlbumTreeView, [182](#)
- AbstractWidgetDelegateOverlay
  - Digikam::AbstractWidgetDelegateOverlay, [208](#)
- acceptedCharacters
  - Digikam::DPlainTextEdit, [1189](#)
  - Digikam::DTextEdit, [1288](#)
- accepts
  - Digikam::AbstractItemDragDropHandler, [195](#)
  - Digikam::AlbumDragDropHandler, [266](#)
  - Digikam::AlbumModelDragDropHandler, [311](#)
  - Digikam::ImportDragDropHandler, [1877](#)
  - Digikam::ItemDragDropHandler, [2051](#)
  - Digikam::MapDragDropHandler, [2422](#)
  - Digikam::TagDragDropHandler, [3127](#)
  - ShowFoto::ShowfotoDragDropHandler, [3447](#)
- acceptsActivation
  - Digikam::ImportDelegate, [1869](#)
  - Digikam::ImportThumbnailDelegate, [1955](#)
  - Digikam::ItemDelegate, [2039](#)
  - Digikam::ItemThumbnailDelegate, [2246](#)
  - Digikam::ItemViewDelegate, [2266](#)
  - Digikam::ItemViewImportDelegate, [2274](#)
  - ShowFoto::ShowfotoDelegate, [3444](#)
  - ShowFoto::ShowfotoItemViewDelegate, [3484](#)
  - ShowFoto::ShowfotoThumbnailDelegate, [3530](#)
- acceptsMimeType
  - Digikam::AbstractItemDragDropHandler, [195](#)
  - Digikam::AlbumModelDragDropHandler, [311](#)
- acceptsMouseClicked
  - Digikam::ImportPreviewView, [1921](#)
  - Digikam::ItemPreviewView, [2177](#)
- acceptsToolTip
  - Digikam::DItemDelegate, [1055](#)
  - Digikam::ImportDelegate, [1869](#)
  - Digikam::ItemDelegate, [2039](#)
  - Digikam::ItemViewDelegate, [2266](#)
  - Digikam::ItemViewImportDelegate, [2274](#)
  - ShowFoto::ShowfotoDelegate, [3444](#)
  - ShowFoto::ShowfotoItemViewDelegate, [3484](#)
- accessCol
  - Digikam, [141](#)
- AccessMode
  - Digikam::LoadSaveThread, [2379](#)
- accessMode
  - Digikam::SharedLoadingTask, [2981](#)
- AccessModeRead
  - Digikam::LoadSaveThread, [2379](#)
- AccessModeReadWrite
  - Digikam::LoadSaveThread, [2379](#)
- accessRow
  - Digikam, [141](#)
- Action
  - Digikam::ExifToolProcess, [1390](#)
- action
  - Digikam::DImageHistory::Entry, [979](#)
  - Digikam::Token, [3295](#)
- ActionCategory
  - Digikam::DPluginAction, [1197](#)

- actionForIndex
  - Digikam::ActionItemModel, [216](#)
- ActionItemModel
  - Digikam::ActionItemModel, [216](#)
- ActionJobCollection
  - Digikam, [134](#)
- actionRequested
  - Digikam::DCategoryDrawer, [828](#)
- ActionType
  - Digikam::DPluginAction, [1197](#)
- activated
  - Digikam::DigikamItemView, [975](#)
  - Digikam::ImportCategorizedView, [1849](#)
  - Digikam::ImportIconView, [1893](#)
  - Digikam::ItemCategorizedView, [2016](#)
- ActiveIconText
  - Digikam::DMultiTabBar, [1104](#)
- activeNextTab
  - Digikam::Sidebar, [3008](#)
- activePreviousTab
  - Digikam::Sidebar, [3008](#)
- add
  - Digikam::FaceTagsEditor, [1493](#)
  - Digikam::KDTreeBase, [2296](#)
- addAction
  - Digikam::ContextMenuHelper, [630](#), [631](#)
  - Digikam::DNotificationWidget, [1162](#)
  - Digikam::ImportContextMenuHelper, [1856](#), [1857](#)
- addActionNewAlbum
  - Digikam::ContextMenuHelper, [631](#)
- addActionNewTag
  - Digikam::ContextMenuHelper, [631](#)
- addActions
  - Digikam::AbstractAlbumTreeView::ContextMenuElement, [163](#)
- addActionsToConfigurationMenu
  - Digikam::BackendGoogleMaps, [436](#)
  - Digikam::BackendMarble, [444](#)
- addAlbum
  - Digikam::CoreDB, [649](#)
- addAlbumCheckUncheckActions
  - Digikam::ContextMenuHelper, [631](#)
- addAlbumRoot
  - Digikam::CoreDB, [649](#)
- addAlbums
  - Digikam::AlbumHistory, [279](#)
- addAsReferredImage
  - Digikam::DImg, [987](#)
- addAssignTagsMenu
  - Digikam::ContextMenuHelper, [632](#)
  - Digikam::ImportContextMenuHelper, [1857](#)
- addCamItemInfoSynchronously
  - Digikam::ImportItemModel, [1899](#)
- addCheckUncheckContextMenuActions
  - Digikam::AbstractAlbumTreeViewSelectComboBox, [167](#)
- addColumnAt
  - Digikam::TableViewModel, [3111](#)
- addComment
  - Digikam::ItemComments, [2023](#)
- addCurrent
  - Digikam::AltLangStrEdit, [366](#)
- addCurrentUniqueImageId
  - Digikam::DImg, [987](#)
- addCustomContextMenuActions
  - Digikam::AbstractAlbumTreeView, [158](#)
  - Digikam::AlbumSelectTreeView, [345](#)
  - Digikam::EditableSearchTreeView, [1330](#)
  - Digikam::NormalSearchTreeView, [2578](#)
  - Digikam::TagCheckView, [3123](#)
  - Digikam::TagFilterView, [3137](#)
  - Digikam::TagFolderView, [3144](#)
- addDataInTree
  - Digikam::RGTagModel, [2783](#)
- Added
  - Digikam::CollectionImageChangeset, [585](#)
- AddEntryToExisting
  - Digikam::ItemCopyright, [2032](#)
- addExternalTags
  - Digikam::RGTagModel, [2783](#)
- addGotoMenu
  - Digikam::ContextMenuHelper, [632](#)
- addGroupMenu
  - Digikam::ContextMenuHelper, [633](#)
  - Digikam::ImportContextMenuHelper, [1858](#)
- addGroupToLayout
  - Digikam::AbstractSearchGroupContainer, [202](#)
  - Digikam::SearchGroup, [2892](#)
  - Digikam::SearchView, [2929](#)
- addHeadline
  - Digikam::ItemComments, [2023](#)
- addHistory
  - Digikam::ItemHistoryGraph, [2098](#)
- addHook
  - Digikam::ItemQueryPostHooks, [2199](#)
- addId
  - Digikam::DbKeysCollection, [807](#)
- addIdentity
  - Digikam::FacialRecognitionWrapper, [1504](#)
  - Digikam::IdentityProvider, [1791](#)
- addImageMetadata
  - Digikam::CoreDB, [649](#)
- addImageTagProperty
  - Digikam::CoreDB, [650](#)
- addIQSAction
  - Digikam::ContextMenuHelper, [633](#)
- addItem
  - Digikam::CoreDB, [650](#)
  - Digikam::DExpanderBox, [933](#)
  - Digikam::ItemVisibilityController, [2283](#)
  - Digikam::SinglePhotoPreviewLayout, [3030](#)
  - Digikam::TagMngrListModel, [3151](#)
- addItemInfo
  - Digikam::ItemModel, [2165](#)
- addItemInformation
  - Digikam::CoreDB, [650](#)

- addItemInfoSynchronously
  - Digikam::ItemModel, 2165
- addItemPosition
  - Digikam::CoreDB, 651
- addItemTag
  - Digikam::CoreDB, 651
- AdditionalRoles
  - Digikam::DCategorizedSortFilterProxyModel, 820
- addLabelsAction
  - Digikam::ContextMenuHelper, 633
  - Digikam::ImportContextMenuHelper, 1858
- addListener
  - Digikam::SharedLoadingTask, 2981
- addLoadingProcess
  - Digikam::LoadingCache, 2358
- addLocation
  - Digikam::CollectionManager, 592
- addMoreWorkers
  - Digikam::FacePipelineDetect, 1433
  - Digikam::FacePipelineDetectRecognize, 1437
  - Digikam::FacePipelineEdit, 1441
  - Digikam::FacePipelineRecognize, 1455
  - Digikam::FacePipelineReset, 1459
  - Digikam::FacePipelineRetrain, 1463
- addNewData
  - Digikam::RGTagModel, 2783
- addNewTag
  - Digikam::RGTagModel, 2784
- addNormalTag
  - Digikam::FaceTagsEditor, 1493
  - Digikam::FaceUtils, 1501
- addObject
  - Digikam::VisibilityController, 3382
- addOpenAndNavigateActions
  - Digikam::ContextMenuHelper, 633
- addOverlay
  - Digikam::ImportCategorizedView, 1849
  - ShowFoto::ShowfotoCategorizedView, 3432
- addPage
  - Digikam::DConfigDlg, 848
  - Digikam::DConfigDlgWdg, 874
  - Digikam::DConfigDlgWdgModel, 884
- addProfile
  - Digikam::lccProfilesMenuAction, 1771
- addProfileSqueezed
  - Digikam::lccProfilesComboBox, 1769
- addProgressItem
  - Digikam::ProgressManager, 2672
- addProperty
  - Digikam::ItemTagPair, 2233
  - Digikam::TagProperties, 3173
- addRelations
  - Digikam::ItemHistoryGraph, 2098
- addRemoveAllTags
  - Digikam::ContextMenuHelper, 634
- addRemoveTagsMenu
  - Digikam::ContextMenuHelper, 634
  - Digikam::ImportContextMenuHelper, 1858
- addRotateMenu
  - Digikam::ImportContextMenuHelper, 1859
- addScannedHistory
  - Digikam::ItemHistoryGraph, 2098
- addSearch
  - Digikam::CoreDB, 652
- addServicesMenu
  - Digikam::ContextMenuHelper, 634
  - Digikam::ImportContextMenuHelper, 1859
- addShowfotoItemInfoSynchronously
  - ShowFoto::ShowfotoItemModel, 3475
- addSpacerTag
  - Digikam::RGTagModel, 2784
- addSqueezedItem
  - Digikam::SqueezedComboBox, 3038
- addStandardActionCopy
  - Digikam::ContextMenuHelper, 635
- addStandardActionCut
  - Digikam::ContextMenuHelper, 635
- addStandardActionItemDelete
  - Digikam::ContextMenuHelper, 635
- addStandardActionLightTable
  - Digikam::ContextMenuHelper, 635
- addStandardActionPaste
  - Digikam::ContextMenuHelper, 635
- addStandardActionThumbnail
  - Digikam::ContextMenuHelper, 636
- addSubMenu
  - Digikam::ContextMenuHelper, 636
  - Digikam::ImportContextMenuHelper, 1859
- addSubPage
  - Digikam::DConfigDlg, 849
  - Digikam::DConfigDlgWdg, 874, 875
  - Digikam::DConfigDlgWdgModel, 885
- addTag
  - Digikam::CoreDB, 652
- addTagPaths
  - Digikam::ItemInfo, 2117
- addTagProperty
  - Digikam::CoreDB, 652
- addTitle
  - Digikam::ItemComments, 2024
- addToDownloadHistory
  - Digikam::CoreDB, 653
- addToken
  - Digikam::Rule, 2795
- addToXmpTagStringBag
  - Digikam::DMetadata, 1090
  - Digikam::MetaEngine, 2485
- addUngroupedModel
  - Digikam::MapWidget, 2430
- addVideoMetadata
  - Digikam::CoreDB, 653
- addWidget
  - Digikam::DConfigDlgMngr, 855
- AdjacencyFlags
  - Digikam::Graph< VertexProperties, EdgeProperties >, 1686

- adjustBoundariesToGroupedMarkers
  - Digikam::MapWidget, [2430](#)
- adjustedEnvironmentForApplImage
  - Digikam, [136](#)
- AdjustmentStatus
  - Digikam::ItemMetadataAdjustmentHint, [2159](#)
- adjustToOrientation
  - Digikam::TagRegion, [3180](#)
- advance
  - Digikam::ProgressItem, [2665](#)
- AdvancedMetadataTab
  - Digikam::AdvancedMetadataTab, [239](#)
- affectsMultiple
  - Digikam::ItemDelegateOverlay, [2043](#)
- album
  - Digikam::CollectionManager, [592](#)
  - Digikam::CoreDbUrl, [693](#)
- albumAt
  - Digikam::ItemCategorizedView, [2016](#)
- albumChooser
  - Digikam::DBInfoface, [795](#)
  - Digikam::DInfoInterface, [1032](#)
- albumChooserItems
  - Digikam::DBInfoface, [795](#)
- albumCleared
  - Digikam::AbstractAlbumModel, [151](#)
  - Digikam::AbstractCheckableAlbumModel, [174](#)
  - Digikam::AbstractCountingAlbumModel, [187](#)
- albumCount
  - Digikam::AbstractCountingAlbumModel, [187](#)
- albumData
  - Digikam::AbstractAlbumModel, [151](#)
  - Digikam::AbstractCheckableAlbumModel, [174](#)
  - Digikam::AbstractCountingAlbumModel, [187](#)
  - Digikam::AlbumModel, [309](#)
  - Digikam::SearchModel, [2901](#)
  - Digikam::TagModel, [3166](#)
- AlbumDataRole
  - Digikam::AbstractAlbumModel, [150](#)
- albumForId
  - Digikam::AbstractCountingAlbumModel, [187](#)
  - Digikam::AlbumModel, [309](#)
  - Digikam::DateAlbumModel, [753](#)
  - Digikam::SearchModel, [2901](#)
  - Digikam::TagModel, [3166](#)
- albumForSelectedItems
  - Digikam::AlbumLabelsSearchHandler, [281](#)
- AlbumGlobalIdRole
  - Digikam::AbstractAlbumModel, [150](#)
- albumId
  - Digikam::ItemInfo, [2117](#)
- AlbumIdRole
  - Digikam::AbstractAlbumModel, [150](#)
- albumInfo
  - Digikam::DBInfoface, [795](#)
- albumItems
  - Digikam::DBInfoface, [795](#)
- AlbumModificationHelper
  - Digikam::AlbumModificationHelper, [313](#)
- albumName
  - Digikam::AbstractCountingAlbumModel, [187](#)
  - Digikam::DateAlbumModel, [753](#)
- AlbumPointerRole
  - Digikam::AbstractAlbumModel, [150](#)
- albumRoot
  - Digikam::CollectionManager, [592](#)
  - Digikam::CoreDbUrl, [693](#)
- albumRootLabel
  - Digikam::CollectionManager, [592](#)
- albumRootPath
  - Digikam::CollectionLocation, [588](#)
  - Digikam::CollectionManager, [592](#)
- AlbumSelectors
  - Digikam::AlbumSelectors, [337](#)
- AlbumSelectTreeView
  - Digikam::AlbumSelectTreeView, [345](#)
- albumsItems
  - Digikam::DBInfoface, [795](#)
- albumsListing
  - Digikam::AlbumsDBJobsThread, [324](#)
- AlbumSortRole
  - Digikam::AbstractAlbumModel, [150](#)
- AlbumTitleRole
  - Digikam::AbstractAlbumModel, [150](#)
- albumTitles
  - Digikam::AlbumManager, [287](#)
- AlbumTypeRole
  - Digikam::AbstractAlbumModel, [150](#)
- alignFace
  - Digikam::DNNOpenFaceExtractor, [1140](#)
  - Digikam::DNNSFaceExtractor, [1145](#)
- All
  - Digikam::QueueListView, [2689](#)
- allAlbumItems
  - Digikam::DBInfoface, [796](#)
  - Digikam::DMetaInfoface, [1099](#)
- allAlbumsCleared
  - Digikam::AbstractAlbumModel, [151](#)
  - Digikam::AbstractCheckableAlbumModel, [175](#)
  - Digikam::AbstractCountingAlbumModel, [188](#)
- allIDAlbums
  - Digikam::AlbumManager, [287](#)
- AllIconsText
  - Digikam::DMultiTabBar, [1104](#)
- allIdentities
  - Digikam::FacialRecognitionWrapper, [1504](#)
- AllItems
  - Digikam::AutoTagsScanSettings, [425](#)
  - Digikam::ImageQualitySorter, [1820](#)
- allNeedGroupResolving
  - Digikam::ItemIconView, [2112](#)
- allowLift
  - Digikam::CoreDbOperationGroup, [689](#)
  - Digikam::FaceDbOperationGroup, [1411](#)
- allPALbums
  - Digikam::AlbumManager, [287](#)

- allRefreshingFinished
  - Digikam::ImportItemModel, [1899](#)
  - Digikam::ItemModel, [2165](#)
  - ShowFoto::ShowfotoItemModel, [3476](#)
- allSAlbums
  - Digikam::AlbumManager, [287](#)
- allTAlbums
  - Digikam::AlbumManager, [288](#)
- allUrls
  - Digikam::ItemIconView, [2112](#)
- AlreadyScannedHandling
  - Digikam::FaceScanSettings, [1479](#)
- AltLangMap
  - Digikam::MetaEngine, [2484](#)
- AltLangStrEdit
  - Digikam::AltLangStrEdit, [366](#)
- AlwaysShowInclusiveCounts
  - Digikam::AbstractAlbumTreeView, [158](#)
- ambientAcceleration
  - Digikam::DRawInfo, [1270](#)
- ambientElevationAngle
  - Digikam::DRawInfo, [1270](#)
- ambientHumidity
  - Digikam::DRawInfo, [1270](#)
- ambientPressure
  - Digikam::DRawInfo, [1270](#)
- ambientTemperature
  - Digikam::DRawInfo, [1270](#)
- ambientWaterDepth
  - Digikam::DRawInfo, [1270](#)
- animatedShowTemporized
  - Digikam::DNotificationWidget, [1163](#)
- AnimatedVisibility
  - Digikam::AnimatedVisibility, [371](#)
- append
  - Digikam::DTrashItemModel, [1296](#)
- appendButton
  - Digikam::DMultiTabBar, [1105](#)
- appendControlButtonsWidget
  - Digikam::DItemsList, [1061](#)
- AppendDecorationRole
  - Digikam::SetupCollectionModel, [2960](#)
- appendJobs
  - Digikam::ActionThreadBase, [225](#)
- appendPluginToBlackList
  - Digikam::DPluginLoader, [1225](#)
- appendPluginToWhiteList
  - Digikam::DPluginLoader, [1226](#)
- appendTab
  - Digikam::DMultiTabBar, [1105](#)
  - Digikam::Sidebar, [3008](#)
- appliedFilterActions
  - Digikam::FilterActionFilter, [1561](#)
- apply
  - Digikam::BatchTool, [461](#)
  - Digikam::IccTransform, [1782](#)
  - Digikam::ItemComments, [2024](#)
  - Digikam::ItemPosition, [2170](#)
- APPLY\_CHANGES
  - Digikam::ExifToolProcess, [1390](#)
- APPLY\_CHANGES\_EXV
  - Digikam::ExifToolProcess, [1390](#)
- APPLY\_METADATA\_FILE
  - Digikam::ExifToolProcess, [1390](#)
- applyCacheToBackend
  - Digikam::MapWidget, [2431](#)
- applyCacheToWidget
  - Digikam::BackendMarble, [444](#)
- applyChanges
  - Digikam::ExifToolParser, [1384](#)
  - Digikam::MetaEngine, [2485](#)
- applyFilter
  - Digikam::BatchTool, [461](#)
- applyMetadata
  - Digikam::FileActionMngrDatabaseWorker, [1519](#)
- applyMetadataFile
  - Digikam::ExifToolParser, [1385](#)
- applySettings
  - Digikam::AlbumFolderViewSideBarWidget, [276](#)
  - Digikam::DateFolderViewSideBarWidget, [759](#)
  - Digikam::FuzzySearchSideBarWidget, [1603](#)
  - Digikam::GPSSearchSideBarWidget, [1675](#)
  - Digikam::ImportItemPropertiesSideBarImport, [1905](#)
  - Digikam::LabelsSideBarWidget, [2305](#)
  - Digikam::PeopleSideBarWidget, [2628](#)
  - Digikam::SearchSideBarWidget, [2909](#)
  - Digikam::SidebarWidget, [3013](#)
  - Digikam::TagViewSideBarWidget, [3222](#)
  - Digikam::TimelineSideBarWidget, [3289](#)
- applyTagIdentityMapping
  - Digikam::FaceTags, [1489](#)
- ApplyTransform
  - Digikam::LoadingDescription, [2363](#)
- areaCoordinates
  - Digikam::CoreDbUrl, [693](#)
- arrowDirection
  - Digikam::DSelector, [1276](#)
- asDBDateTime
  - Digikam::BdEngineBackend, [475](#)
- asDelegate
  - Digikam::ItemDelegateOverlayContainer, [2046](#)
  - Digikam::ItemViewDelegate, [2266](#)
  - Digikam::ItemViewImportDelegate, [2274](#)
  - Digikam::VersionsDelegate, [3364](#)
  - ShowFoto::ShowfotoItemViewDelegate, [3484](#)
- askGroupingOperateOnAll
  - Digikam::ApplicationSettings, [383](#)
- aspectRatio
  - Digikam::ItemInfo, [2118](#)
- asQObject
  - Digikam::ParallelAdapter< A >, [2617](#)
  - Digikam::ParallelWorkers, [2622](#)
- asQtCaseSensitivity
  - Digikam::CollectionLocation, [588](#)
- assignColorLabel



- Digikam::FileActionMngrDatabaseWorker, [1519](#)
- assignDate
  - Digikam::DDateEdit, [890](#)
- assigned
  - Digikam::AssignNameWidget, [398](#)
- assignPickLabel
  - Digikam::FileActionMngrDatabaseWorker, [1519](#)
- assignRating
  - Digikam::FileActionMngrDatabaseWorker, [1519](#)
- assignTags
  - Digikam::FileActionMngrDatabaseWorker, [1520](#)
- atEnd
  - Digikam::EmptyImageListProvider, [1367](#)
  - Digikam::QListImageListProvider, [2686](#)
- autoDelete
  - Digikam::DNotificationPopup, [1153](#)
- autoExifTransform
  - Digikam::JPEGUtils::JpegRotator, [2287](#)
- autoHideTimeout
  - Digikam::DConfigDlgTitle, [862](#)
- AutotagsAssignment
  - Digikam::AutotagsAssignment, [422](#)
- autoWBAdjustmentFromColor
  - Digikam::WBFilter, [3388](#)
- availablePairs
  - Digikam::ItemTagPair, [2233](#)
- AVI
  - Digikam::VidSlideSettings, [3375](#)
- azimuth
  - Digikam::GeodeticCalculator, [1611](#)
- Backend
  - Digikam::MetaEngine, [2484](#)
- BackendGeonamesRG
  - Digikam::BackendGeonamesRG, [429](#)
- BackendGeonamesUSRG
  - Digikam::BackendGeonamesUSRG, [432](#)
- backendHumanName
  - Digikam::BackendGoogleMaps, [436](#)
  - Digikam::BackendMarble, [444](#)
  - Digikam::LookupAltitudeGeonames, [2397](#)
- backendName
  - Digikam::BackendGeonamesRG, [429](#)
  - Digikam::BackendGeonamesUSRG, [432](#)
  - Digikam::BackendGoogleMaps, [437](#)
  - Digikam::BackendMarble, [444](#)
  - Digikam::BackendOsmRG, [451](#)
  - Digikam::LookupAltitudeGeonames, [2397](#)
  - Digikam::MetaEngine, [2485](#)
  - Digikam::RGBBackend, [2778](#)
- BackendOsmRG
  - Digikam::BackendOsmRG, [451](#)
- backgroundColor
  - Digikam::DFontProperties, [945](#)
- backup
  - Digikam::Sidebar, [3008](#)
- Balloon
  - Digikam::DNotificationPopup, [1153](#)
- BarMode
  - Digikam::DZoomBar, [1323](#)
- baselineExposure
  - Digikam::DRawInfo, [1270](#)
- baseName
  - Digikam::DefaultVersionNamingScheme, [918](#)
  - Digikam::VersionNamingScheme, [3360](#)
- BaseTool
  - Digikam::BatchTool, [460](#)
- BasicDImgFilterGenerator
  - Digikam::BasicDImgFilterGenerator< T >, [456](#)
- BatchTool
  - Digikam::BatchTool, [461](#)
- BatchToolGroup
  - Digikam::BatchTool, [460](#)
- bcg
  - Digikam::DRawDecoding, [1267](#)
- BdEngineBackend
  - Digikam::BdEngineBackend, [475](#)
- beginFileMetadataWrite
  - Digikam::ScanController, [2810](#)
- BehaviorEnum
  - Digikam::ICCSettingsContainer, [1781](#)
- bestMatchesForImageWithThreshold
  - Digikam::HaarFace, [1719](#)
- bestRepresentativeIndexFromList
  - Digikam::AbstractMarkerTiler, [197](#)
  - Digikam::GeoModelHelper, [1626](#)
  - Digikam::GPSGeoFaceModelHelper, [1647](#)
  - Digikam::GPSMarkerTiler, [1667](#)
  - Digikam::ItemGPSModelHelper, [2095](#)
  - Digikam::ItemMarkerTiler, [2155](#)
  - Digikam::MapViewModelHelper, [2424](#)
- bindAlbum
  - Digikam::AlbumModificationHelper, [313](#)
- bindMultipleTags
  - Digikam::TagModificationHelper, [3169](#)
- bindTag
  - Digikam::TagModificationHelper, [3169](#)
- bitBlendImage
  - Digikam::DImg, [987](#)
- bitBlendImageOnColor
  - Digikam::DImg, [988](#)
- bitBlitImage
  - Digikam::DImg, [988](#)
- BlackWhiteConversionType
  - Digikam::BWSepiaContainer, [516](#)
- blendZero
  - Digikam::DColor, [834](#)
- blockedEventTypes
  - Digikam::DWItemDelegate, [1308](#)
- BLUERAY
  - Digikam::VidSlideSettings, [3376](#)
- bottomBarPixmap
  - Digikam::SearchView, [2929](#)
- boundAlbum
  - Digikam::AlbumModificationHelper, [313](#)
- boundingRect
  - Digikam::DImgChildItem, [1000](#)



- boundMultipleTags
  - Digikam::TagModificationHelper, [3169](#)
- boundTag
  - Digikam::TagModificationHelper, [3170](#)
- Boxed
  - Digikam::DNotificationPopup, [1153](#)
- branchFromIndex
  - Digikam::RGTagModel, [2784](#)
- buildCollectionTrashCounters
  - Digikam::IOJobsManager, [1986](#)
- bundleProperties
  - Digikam::OnlineVersionChecker, [2596](#)
- BWGeneric
  - Digikam::BWSepiaContainer, [516](#)
- BWIlfordSFX200
  - Digikam::BWSepiaContainer, [516](#)
- BWKodakHIE
  - Digikam::BWSepiaContainer, [516](#)
- BWNoFilter
  - Digikam::BWSepiaContainer, [516](#)
- BWNoTone
  - Digikam::BWSepiaContainer, [516](#)
- cacheByName
  - Digikam::ItemInfoCache, [2125](#)
- cacheKey
  - Digikam::SharedLoadingTask, [2981](#)
- calcHaar
  - Digikam::Haar::Calculator, [1715](#)
- callRGBackend
  - Digikam::BackendGeonamesRG, [429](#)
  - Digikam::BackendGeonamesUSRG, [432](#)
  - Digikam::BackendOsmRG, [451](#)
  - Digikam::RGBackend, [2778](#)
- cameraAbout
  - Digikam::GPCamera, [1631](#)
  - Digikam::UMSCamera, [3339](#)
- cameraDriverType
  - Digikam::GPCamera, [1631](#)
  - Digikam::UMSCamera, [3339](#)
- cameraManual
  - Digikam::GPCamera, [1631](#)
  - Digikam::UMSCamera, [3339](#)
- cameraMD5ID
  - Digikam::GPCamera, [1632](#)
  - Digikam::UMSCamera, [3339](#)
- cameraSummary
  - Digikam::GPCamera, [1632](#)
  - Digikam::UMSCamera, [3339](#)
- camItemInfo
  - Digikam::ImportItemModel, [1899](#)
- camItemInfoActivated
  - Digikam::ImportCategorizedView, [1850](#)
- camItemInfosAdded
  - Digikam::ImportFilterModel, [1884](#)
- camItemInfosSorted
  - Digikam::ImportSortFilterModel, [1941](#)
- canBeCanceled
  - Digikam::ProgressItem, [2665](#)
- canBeWrittenToMetadata
  - Digikam::TagsCache, [3187](#)
- cancel
  - Digikam::BatchTool, [461](#)
  - Digikam::GPCamera, [1632](#)
  - Digikam::LookupAltitudeGeonames, [2397](#)
  - Digikam::MLPipelineFoundation, [2526](#)
  - Digikam::ThumbnailImageCatcher, [3252](#)
  - Digikam::UMSCamera, [3340](#)
- cancelAllAndSuspendCollectionScan
  - Digikam::ScanController, [2810](#)
- cancelCompleteScan
  - Digikam::ScanController, [2810](#)
- cancelFilter
  - Digikam::DImgThreadedFilter, [1023](#)
  - Digikam::GreycstorageFilter, [1705](#)
- cancelRequests
  - Digikam::BackendGeonamesRG, [430](#)
  - Digikam::BackendGeonamesUSRG, [433](#)
  - Digikam::BackendOsmRG, [452](#)
- canRead
  - Digikam::DPluginDImg, [1215](#)
- canWrite
  - Digikam::DPluginDImg, [1215](#)
- capture
  - Digikam::DKCamera, [1068](#)
  - Digikam::GPCamera, [1632](#)
  - Digikam::UMSCamera, [3340](#)
- CaseInsensitive
  - Digikam::CollectionLocation, [587](#)
- CaseSensitive
  - Digikam::CollectionLocation, [587](#)
- CaseSensitivity
  - Digikam::CollectionLocation, [587](#)
- caseSensitivity
  - Digikam::CollectionLocation, [588](#)
- categories
  - Digikam::DPlugin, [1192](#)
  - Digikam::DPluginBqm, [1201](#)
  - Digikam::DPluginDImg, [1215](#)
  - Digikam::DPluginEditor, [1220](#)
  - Digikam::DPluginGeneric, [1223](#)
  - Digikam::DPluginRawImport, [1230](#)
- CategorizationMode
  - Digikam::ItemSortSettings, [2230](#)
- CategorizationModeRole
  - Digikam::ImportFilterModel, [1884](#)
  - Digikam::ItemFilterModel, [2065](#)
  - ShowFoto::ShowfotoFilterModel, [3453](#)
- categorize
  - Digikam::ItemHistoryGraph, [2098](#)
- categorizedIndexesIn
  - Digikam::DCategorizedView, [825](#)
- Category
  - Digikam::FilterAction, [1554](#)
- CategoryAlbumIdRole
  - Digikam::ItemFilterModel, [2065](#)
- categoryAt

- Digikam::DCategorizedView, 826
- CategoryButtonDisplayRole
  - Digikam::SetupCollectionModel, 2960
- CategoryDateRole
  - Digikam::ImportFilterModel, 1884
  - Digikam::ItemFilterModel, 2065
- CategoryDisplayRole
  - Digikam::DCategorizedSortFilterProxyModel, 820
- CategoryFaceRole
  - Digikam::ItemFilterModel, 2065
- CategoryFormatRole
  - Digikam::ImportFilterModel, 1884
  - Digikam::ItemFilterModel, 2065
  - ShowFoto::ShowfotoFilterModel, 3453
- categoryHeight
  - Digikam::DCategoryDrawer, 828
  - Digikam::ImportCategoryDrawer, 1853
  - Digikam::ItemCategoryDrawer, 2019
- categoryIdentifier
  - Digikam::ImportFilterModel, 1884
  - Digikam::ItemFilterModel, 2065
  - ShowFoto::ShowfotoFilterModel, 3453
- categoryRange
  - Digikam::DCategorizedView, 826
- CategorySortRole
  - Digikam::DCategorizedSortFilterProxyModel, 820
- categoryVisualRect
  - Digikam::DCategorizedView, 826
- centerOn
  - Digikam::BackendGoogleMaps, 437
  - Digikam::BackendMarble, 444
  - Digikam::MapBackend, 2420
- CHANGE\_TIMESTAMPS
  - Digikam::ExifToolProcess, 1390
- changeAlbumFromHistory
  - Digikam::AlbumFolderViewSideBarWidget, 276
  - Digikam::DateFolderViewSideBarWidget, 759
  - Digikam::FuzzySearchSideBarWidget, 1603
  - Digikam::GPSSearchSideBarWidget, 1675
  - Digikam::LabelsSideBarWidget, 2305
  - Digikam::PeopleSideBarWidget, 2628
  - Digikam::SearchSideBarWidget, 2909
  - Digikam::SidebarWidget, 3013
  - Digikam::TagViewSideBarWidget, 3222
  - Digikam::TimelineSideBarWidget, 3289
- changeComment
  - Digikam::ItemComments, 2024
- changeDatabase
  - Digikam::AlbumManager, 288
- changedFlags
  - Digikam::DisjointMetadata, 1044
- changeImageComment
  - Digikam::CoreDB, 653
- changeImageMetadata
  - Digikam::CoreDB, 653
- changeItemInformation
  - Digikam::CoreDB, 654
- changeItemPosition
  - Digikam::CoreDB, 654
- changeRegion
  - Digikam::FaceTagsEditor, 1493
- changeTag
  - Digikam::FaceTagsEditor, 1493
- changeThumbSize
  - Digikam::DTrashItemModel, 1296
- changeTimestamps
  - Digikam::ExifToolParser, 1385
- ChangeType
  - Digikam::ItemChangeHint, 2020
- changeVideoMetadata
  - Digikam::CoreDB, 654
- channelToBinary
  - Digikam::ImageCurves, 1794
- checkAzimuth
  - Digikam::GeodeticCalculator, 1611
- checkDatabaseSettings
  - Digikam::DatabaseSettingsWidget, 740
- checkedKeys
  - Digikam::ChoiceSearchModel, 579
- checkedTagsChanged
  - Digikam::TagCheckView, 3124
- checkHardWiredLocations
  - Digikam::CollectionManager, 593
- checkIndex
  - Digikam::AbstractWidgetDelegateOverlay, 208
  - Digikam::ActionVersionsOverlay, 229
  - Digikam::AssignNameOverlay, 393
  - Digikam::FaceRejectionOverlay, 1474
  - Digikam::GroupIndicatorOverlay, 1711
  - Digikam::ImportCoordinatesOverlay, 1863
  - Digikam::ImportDownloadOverlay, 1874
  - Digikam::ImportLockOverlay, 1911
  - Digikam::ImportRotateOverlay, 1931
  - Digikam::ItemCoordinatesOverlay, 2028
  - Digikam::ItemFullScreenOverlay, 2085
  - Digikam::ItemRotateOverlay, 2208
  - Digikam::ShowHideVersionsOverlay, 3003
  - ShowFoto::ShowfotoCoordinatesOverlay, 3437
- checkLatitude
  - Digikam::GeodeticCalculator, 1612
- checkLocation
  - Digikam::CollectionManager, 593
- checkLongitude
  - Digikam::GeodeticCalculator, 1612
- checkOrSetWALMode
  - Digikam::BdEngineBackend, 475
- checkOrthodromicDistance
  - Digikam::GeodeticCalculator, 1612
- checkPosition
  - Digikam::ItemQueryPostHooks, 2199
- checkReadyForUse
  - Digikam::CoreDbAccess, 680
  - Digikam::DbEngineAccess, 779
  - Digikam::SimilarityDbAccess, 3021
- checkStateChanged
  - Digikam::AbstractCheckableAlbumModel, 175

- checkToCancelWaitingData
  - Digikam::DRawDecoder, [1254](#)
- childAlbumIds
  - Digikam::Album, [257](#)
- childAlbums
  - Digikam::Album, [257](#)
- childAtRow
  - Digikam::Album, [257](#)
- childCount
  - Digikam::Album, [257](#)
- ChildMatch
  - Digikam::AlbumFilterModel, [270](#)
- ChildToParent
  - Digikam, [136](#)
- ChoiceSearchComboBox
  - Digikam::ChoiceSearchComboBox, [577](#)
- chooserMode
  - Digikam::DColorValueSelector, [841](#)
  - Digikam::DHueSaturationSelector, [956](#)
- Classifier
  - Digikam::MLPipelineFoundation, [2526](#)
  - Digikam::OpenCVDNNFaceRecognizer, [2600](#)
- classifier
  - Digikam::FacePipelineDetect, [1433](#)
  - Digikam::FacePipelineDetectRecognize, [1437](#)
  - Digikam::FacePipelineEdit, [1441](#)
  - Digikam::FacePipelineRecognize, [1455](#)
  - Digikam::FacePipelineReset, [1459](#)
  - Digikam::FacePipelineRetrain, [1463](#)
- cleanCache
  - Digikam::LoadingCacheInterface, [2361](#)
- CleanScan
  - Digikam::CollectionScanner, [600](#)
- cleanThumbnailCache
  - Digikam::LoadingCacheInterface, [2361](#)
- cleanUp
  - Digikam::DPlugin, [1192](#)
  - Digikam::DPluginLoader, [1226](#)
- cleanUpDatabase
  - Digikam::CoreDbAccess, [680](#)
- cleanupFilter
  - Digikam::DImgThreadedFilter, [1023](#)
- cleanupTags
  - Digikam::MetadataHub, [2441](#)
- clear
  - Digikam::TrackManager, [3312](#)
- ClearAll
  - Digikam::FaceScanSettings, [1479](#)
- clearAllActions
  - Digikam::DNotificationWidget, [1163](#)
- clearCaches
  - Digikam::ImportDelegate, [1870](#)
  - Digikam::ItemDelegate, [2040](#)
  - ShowFoto::ShowfotoDelegate, [3444](#)
- clearDNNTraining
  - Digikam::FaceDb, [1404](#)
- clearImageSimilarity
  - Digikam::SimilarityDb, [3015](#)
- clearReferredImages
  - Digikam::DImageHistory, [978](#)
- clicked
  - Digikam::ItemViewCategorized, [2260](#)
- climbTreeAndGetSpacers
  - Digikam::RGTagModel, [2784](#)
- clone
  - Digikam::BatchTool, [461](#)
- close
  - Digikam::BdEngineBackend, [475](#)
  - Digikam::DPopupFrame, [1237](#)
  - Digikam::IccProfile, [1764](#)
  - Digikam::IccTransform, [1782](#)
  - Digikam::PanIconFrame, [2612](#)
- closestItem
  - Digikam::FaceGroup, [1416](#)
- CODE\_OF\_CONDUCT, [25](#)
- CollectionImageChangeset
  - Digikam::CollectionImageChangeset, [585](#)
- collectionName
  - Digikam::DbKeysCollection, [807](#)
- color
  - Digikam::DFontProperties, [945](#)
- colorLabel
  - Digikam::DisjointMetadata, [1044](#)
- colorLabelForTag
  - Digikam::TagsCache, [3187](#)
- colorLabelFromTags
  - Digikam::TagsCache, [3187](#)
- colorLabelInterval
  - Digikam::DisjointMetadata, [1044](#)
- ColorManagementSettings
  - Digikam::LoadingDescription, [2363](#)
- colorRectPixmap
  - Digikam::LabelsTreeView, [2309](#)
- ColorTool
  - Digikam::BatchTool, [460](#)
- colorValue
  - Digikam::DColorValueSelector, [841](#)
  - Digikam::DHueSaturationSelector, [956](#)
- columnAffectedByChangeset
  - Digikam::TableViewColumn, [3072](#)
  - Digikam::TableViewColumns::ColumnDigikamProperties, [3083](#)
- columnHeader
  - Digikam::AbstractAlbumModel, [152](#)
  - Digikam::AbstractSpecificAlbumModel, [206](#)
- command
  - Digikam::ExifToolProcess, [1391](#)
- comment
  - Digikam::DConfigDlgTitle, [862](#)
  - Digikam::ItemInfo, [2118](#)
- commentForLanguage
  - Digikam::ItemComments, [2024](#)
- comments
  - Digikam::DisjointMetadata, [1045](#)
- commit
  - Digikam::ItemScanner, [2216](#)

- COMMIT POLICY, [27](#)
- compare
  - Digikam::CamItemSortSettings, [543](#)
  - Digikam::ItemSortSettings, [2231](#)
  - Digikam::TableViewColumn, [3072](#)
  - Digikam::TableViewColumns::ColumnAudioVideoProperties, [3079](#)
  - Digikam::TableViewColumns::ColumnDigikamProperties, [3083](#)
  - Digikam::TableViewColumns::ColumnFileProperties, [3088](#)
  - Digikam::TableViewColumns::ColumnGeoProperties, [3093](#)
  - Digikam::TableViewColumns::ColumnItemProperties, [3097](#)
  - Digikam::TableViewColumns::ColumnPhotoProperties, [3102](#)
  - ShowFoto::ShowfotoItemSortSettings, [3479](#)
- compareCategories
  - Digikam::CamItemSortSettings, [544](#)
  - Digikam::DCategorizedSortFilterProxyModel, [821](#)
  - Digikam::ImportFilterModel, [1884](#)
  - Digikam::ItemFilterModel, [2065](#)
  - Digikam::ItemSortSettings, [2231](#)
  - ShowFoto::ShowfotoFilterModel, [3453](#)
  - ShowFoto::ShowfotoItemSortSettings, [3479](#)
- compareInfosCategories
  - Digikam::ItemAlbumFilterModel, [2001](#)
  - Digikam::ItemFilterModel, [2066](#)
- CompleteCollectionScan
  - Digikam::NewItemFinder, [2558](#)
- completeCollectionScan
  - Digikam::ScanController, [2810](#)
- completed
  - Digikam::SharedLoadingTask, [2981](#)
- completelyApplied
  - Digikam::FilterActionFilter, [1561](#)
- completeScan
  - Digikam::CollectionScanner, [600](#)
- ComplexFilter
  - Digikam::FilterAction, [1555](#)
- compose
  - Digikam::DColorComposer, [836](#)
- CompositingOperation
  - Digikam::DColorComposer, [836](#)
- computeDirection
  - Digikam::GeodeticCalculator, [1612](#)
- confirm
  - Digikam::FacePipeline, [1424](#)
- Confirmed
  - Digikam::FacePipelineFaceTagsIface, [1447](#)
- confirmFaces
  - Digikam::DigikamItemView, [975](#)
- confirmName
  - Digikam::FaceTagsEditor, [1493](#)
- connectAndSchedule
  - Digikam::WorkerObject, [3397](#)
- connectFinishAndErrorSignals
  - Digikam::DBJobsThread, [805](#)
- ConnectionError
  - Digikam::BdEngineBackend, [475](#)
- connectionError
  - Digikam::DbEngineErrorHandler, [783](#)
- connectionErrorHandling
  - Digikam::BdEngineBackend, [476](#)
- consultUserForError
  - Digikam::DbEngineErrorHandler, [784](#)
- contactInfo
  - Digikam::ItemCopyright, [2032](#)
- contains
  - Digikam::SqueezedComboBox, [3039](#)
- containsItem
  - Digikam::ListItem, [2352](#)
- contentsRect
  - Digikam::DPointSelect, [1233](#)
  - Digikam::DSelector, [1276](#)
- contextMenuEvent
  - Digikam::TagFolderView, [3144](#)
  - Digikam::TagMngrTreeView, [3159](#)
- ContextMenuHelper
  - Digikam::ContextMenuHelper, [630](#)
- contextMenuIcon
  - Digikam::AbstractAlbumTreeView, [159](#)
- contextMenuTitle
  - Digikam::AbstractAlbumTreeView, [159](#)
  - Digikam::EditableSearchTreeView, [1330](#)
  - Digikam::TagFolderView, [3144](#)
- continueQuery
  - Digikam::LoadingTask, [2369](#)
  - Digikam::SavingTask, [2806](#)
- convertDegreeAngleToDouble
  - Digikam::MetaEngine, [2485](#)
- convertDepth
  - Digikam::DImg, [988](#)
- ConvertError
  - Digikam::DNGWriter, [1115](#)
- ConvertForDisplay
  - Digikam::LoadingDescription, [2363](#)
- ConvertForOutput
  - Digikam::LoadingDescription, [2363](#)
- convertFromGPSCoordinateString
  - Digikam::MetaEngine, [2485](#), [2486](#)
- convertToGPSCoordinateString
  - Digikam::MetaEngine, [2486](#)
- ConvertTool
  - Digikam::BatchTool, [460](#)
- convertToRational
  - Digikam::MetaEngine, [2486](#)
- convertToRationalSmallDenominator
  - Digikam::MetaEngine, [2486](#)
- convertToUserPresentableNumbers
  - Digikam::MetaEngine, [2487](#)
- convertZoomToBackendZoom
  - Digikam::MapWidget, [2431](#)
- coordinatesToClipboard
  - Digikam, [136](#)

- Copied
  - Digikam::CollectionImageChangeset, [585](#)
- copiedFrom
  - Digikam::ItemScanner, [2216](#)
- copy
  - Digikam::DIO, [1038](#)
- COPY\_ALL
  - Digikam::ExifToolProcess, [1390](#)
- COPY\_EXIF
  - Digikam::ExifToolProcess, [1390](#)
- COPY\_ICC
  - Digikam::ExifToolProcess, [1390](#)
- COPY\_IPTC
  - Digikam::ExifToolProcess, [1390](#)
- COPY\_MAKERNOTES
  - Digikam::ExifToolProcess, [1390](#)
- COPY\_NONE
  - Digikam::ExifToolProcess, [1390](#)
- COPY\_TAGS
  - Digikam::ExifToolProcess, [1390](#)
- COPY\_XMP
  - Digikam::ExifToolProcess, [1390](#)
- copyAlbumProperties
  - Digikam::CoreDB, [654](#)
- copyAttributes
  - Digikam::FileActionMngrDatabaseWorker, [1520](#)
- copyItem
  - Digikam::CoreDB, [654](#)
  - Digikam::ItemInfo, [2118](#)
- copyMetaData
  - Digikam::DImg, [988](#)
- copyOrMove
  - Digikam::IOJobsThread, [1990](#)
- copyrightNotice
  - Digikam::ItemCopyright, [2032](#)
- copySearch
  - Digikam::NormalSearchTreeView, [2578](#)
- copyTags
  - Digikam::ExifToolParser, [1385](#)
- CopyTagsSource
  - Digikam::ExifToolProcess, [1390](#)
- CoreDbAccess
  - Digikam::CoreDbAccess, [680](#)
- CoreDbAccessUnlock
  - Digikam::CoreDbAccessUnlock, [681](#)
- CoreDbNameFilter
  - Digikam::CoreDbNameFilter, [688](#)
- CoreDbWatchAdaptor, [147](#)
- count
  - Digikam::DPlugin, [1192](#)
  - Digikam::DPluginBqm, [1201](#)
  - Digikam::DPluginDImg, [1215](#)
  - Digikam::DPluginEditor, [1220](#)
  - Digikam::DPluginGeneric, [1223](#)
  - Digikam::DPluginRawImport, [1230](#)
- CR\_basis
  - Digikam, [141](#)
- CREATE\_NEW\_GROUPS
  - Digikam::ExifToolProcess, [1391](#)
- CREATE\_NEW\_TAGS
  - Digikam::ExifToolProcess, [1391](#)
- createAnimation
  - Digikam::ItemVisibilityController, [2284](#)
- createButton
  - Digikam::ActionVersionsOverlay, [229](#)
  - Digikam::FaceRejectionOverlay, [1474](#)
  - Digikam::HoverButtonDelegateOverlay, [1747](#)
  - Digikam::ImportRotateOverlay, [1931](#)
  - Digikam::ItemFullScreenOverlay, [2085](#)
  - Digikam::ItemRotateOverlay, [2208](#)
  - Digikam::ItemSelectionOverlay, [2222](#)
  - Digikam::ShowHideVersionsOverlay, [3003](#)
- CreateDefaultDelegate
  - Digikam::AbstractAlbumTreeView, [158](#)
- CreateDefaultFilterModel
  - Digikam::AbstractAlbumTreeView, [158](#)
- CreateDefaultModel
  - Digikam::AbstractAlbumTreeView, [158](#)
- createEllipsoid
  - Digikam::Ellipsoid, [1355](#)
- createExifUserStringFromValue
  - Digikam::MetaEngine, [2487](#)
- createField
  - Digikam::SearchField, [2826](#)
- createFilter
  - Digikam::BasicDImgFilterGenerator< T >, [456](#)
  - Digikam::DImgFilterGenerator, [1003](#)
  - Digikam::DImgFilterManager, [1005](#)
- createFilterModel
  - Digikam::ActionItemModel, [216](#)
- createFlattenedSphere
  - Digikam::Ellipsoid, [1355](#)
- createFullScreenAction
  - Digikam::DXmlGuiWindow, [1316](#)
- createFuzzySearchFromDropped
  - Digikam::SearchModificationHelper, [2903](#)
- createFuzzySearchFromImage
  - Digikam::SearchModificationHelper, [2904](#)
- createFuzzySearchFromSketch
  - Digikam::SearchModificationHelper, [2904](#)
- createHintContainer
  - Digikam::CollectionScanner, [600](#)
- createImageUniqueld
  - Digikam::DImg, [989](#)
- createItemWidgets
  - Digikam::DWItemDelegate, [1309](#)
  - Digikam::SetupCollectionDelegate, [2956](#)
- createMimeData
  - Digikam::AbstractItemDragDropHandler, [195](#)
  - Digikam::AlbumDragDropHandler, [266](#)
  - Digikam::AlbumModelDragDropHandler, [311](#)
  - Digikam::GPSItemListDragDropHandler, [1658](#)
  - Digikam::ImportDragDropHandler, [1877](#)
  - Digikam::ItemDragDropHandler, [2051](#)
  - Digikam::MapDragDropHandler, [2422](#)
  - Digikam::TagDragDropHandler, [3127](#)

- ShowFoto::ShowfotoDragDropHandler, [3447](#)
- CreateNewImageHistoryUUID
  - Digikam::DImg, [986](#)
- CreateNewMetadataPreview
  - Digikam::DImg, [986](#)
- createNode
  - Digikam::KDNodeBase, [2290](#)
  - Digikam::KDNodeOpenFace, [2292](#)
  - Digikam::KDNodeSFace, [2294](#)
  - Digikam::KDTreeBase, [2296](#)
- createPALbum
  - Digikam::AlbumManager, [288](#), [289](#)
- createProgressItem
  - Digikam::ProgressManager, [2672–2674](#)
- createSAlbum
  - Digikam::AlbumManager, [289](#)
- createSearchGroup
  - Digikam::AbstractSearchGroupContainer, [202](#)
  - Digikam::SearchGroup, [2892](#)
  - Digikam::SearchView, [2929](#)
- createTag
  - Digikam::TagsCache, [3187](#)
- createTAlbum
  - Digikam::AlbumManager, [290](#)
  - Digikam::TagEditDlg, [3129](#)
- createView
  - Digikam::DConfigDlgView, [869](#)
- createWidget
  - Digikam::AbstractWidgetDelegateOverlay, [208](#)
  - Digikam::AssignNameOverlay, [393](#)
  - Digikam::GroupIndicatorOverlay, [1711](#)
  - Digikam::HoverButtonDelegateOverlay, [1747](#)
  - Digikam::ImportCoordinatesOverlay, [1863](#)
  - Digikam::ImportDownloadOverlay, [1874](#)
  - Digikam::ImportLockOverlay, [1911](#)
  - Digikam::ImportRatingOverlay, [1925](#)
  - Digikam::ItemCoordinatesOverlay, [2028](#)
  - Digikam::ItemRatingOverlay, [2203](#)
  - Digikam::TagsLineEditOverlay, [3201](#)
  - ShowFoto::ShowfotoCoordinatesOverlay, [3437](#)
- creationDateFromFilesystem
  - Digikam::ItemScanner, [2216](#)
- CreationDateRole
  - Digikam::ItemModel, [2164](#)
- creator
  - Digikam::ItemCopyright, [2032](#)
- creatorJobTitle
  - Digikam::ItemCopyright, [2033](#)
- Crop
  - Digikam::DImgBuiltinFilter, [996](#)
- Current
  - Digikam::HistoryImageId, [1735](#)
- currentActiveItem
  - Digikam::DMetaInfoface, [1099](#)
- currentAlbumItems
  - Digikam::DBInfoface, [796](#)
  - Digikam::DMetaInfoface, [1099](#)
- currentAlbums
  - Digikam::AlbumManager, [290](#)
- currentCamItemInfo
  - Digikam::MapViewWidget, [2436](#)
- currentGPSItems
  - Digikam::DBInfoface, [796](#)
  - Digikam::DMetaInfoface, [1099](#)
- currentItemInfo
  - Digikam::MapViewWidget, [2436](#)
- currentPage
  - Digikam::DConfigDlg, [850](#)
  - Digikam::DConfigDlgWdg, [875](#)
- currentPageChanged
  - Digikam::DConfigDlg, [850](#)
  - Digikam::DConfigDlgView, [869](#)
  - Digikam::DConfigDlgWdg, [876](#)
- currentPALbum
  - Digikam::AlbumManager, [290](#)
- currentSeed
  - Digikam::RandomNumberGenerator, [2706](#)
- currentSelectedItems
  - Digikam::DBInfoface, [796](#)
  - Digikam::DInfoInterface, [1032](#)
  - Digikam::DMetaInfoface, [1099](#)
- currentTaggingAction
  - Digikam::AddTagsComboBox, [236](#)
- currentTAlbums
  - Digikam::AlbumManager, [291](#)
- CURVE\_FREE
  - Digikam::ImageCurves, [1794](#)
- CURVE\_SMOOTH
  - Digikam::ImageCurves, [1794](#)
- CurvesContainer
  - Digikam::CurvesContainer, [701](#)
- curvesType
  - Digikam::CurvesContainer, [701](#)
- CurveType
  - Digikam::ImageCurves, [1793](#)
- customizedFullScreenMode
  - Digikam::DXmlGuiWindow, [1316](#)
- CustomSettings
  - Digikam::ImageQualityConfSelector, [1814](#)
- CustomTool
  - Digikam::BatchTool, [460](#)
- CVD1
  - Digikam::VidSlideSettings, [3375](#)
- CVD2
  - Digikam::VidSlideSettings, [3375](#)
- data
  - Digikam::IccProfile, [1764](#)
  - Digikam::ImportThumbnailModel, [1961](#)
  - Digikam::ItemFilterModel, [2066](#)
  - Digikam::ItemThumbnailModel, [2253](#)
  - Digikam::TableViewColumn, [3072](#)
  - Digikam::TableViewColumns::ColumnAudioVideoProperties, [3079](#)
  - Digikam::TableViewColumns::ColumnDigikamProperties, [3083](#)



- Digikam::TableViewColumns::ColumnFileProperties, [3088](#)
- Digikam::TableViewColumns::ColumnGeoProperties, [3093](#)
- Digikam::TableViewColumns::ColumnItemProperties, [3097](#)
- Digikam::TableViewColumns::ColumnPhotoProperties, [3102](#)
- Digikam::TableViewColumns::ColumnThumbnail, [3106](#)
- ShowFoto::ShowfotoThumbnailModel, [3535](#)
- databaseChanged
  - Digikam::CoreDbWatch, [698](#)
- databaseInitialization
  - Digikam::ScanController, [2810](#)
- databaseInitialScanDone
  - Digikam::CollectionScanner, [600](#)
- DatabaseServerErrorEnum
  - Digikam::DatabaseServerError, [737](#)
- databaseUrl
  - Digikam::Album, [257](#)
  - Digikam::DAAlbum, [722](#)
  - Digikam::PALbum, [2610](#)
  - Digikam::SAlbum, [2802](#)
  - Digikam::TAlbum, [3226](#)
- databaseUuid
  - Digikam::CoreDB, [655](#)
- dataSize
  - Digikam::MetaEnginePreviews, [2507](#)
- DATE
  - Digikam::Album, [256](#)
- date
  - Digikam::DDateEdit, [891](#)
  - Digikam::DDatePicker, [895](#)
  - Digikam::DDateTable, [903](#)
- DateAlbumModel
  - Digikam::DateAlbumModel, [752](#)
- dateChanged
  - Digikam::DDateEdit, [891](#)
  - Digikam::DDatePicker, [895](#)
  - Digikam::DDateTable, [903](#)
- dateEntered
  - Digikam::DDatePicker, [895](#)
- dateFromPos
  - Digikam::DDateTable, [903](#)
- datePicker
  - Digikam::DDatePickerPopup, [899](#)
- dateSelected
  - Digikam::DDatePicker, [896](#)
- datesListing
  - Digikam::DatesDBJobsThread, [768](#)
- dateTable
  - Digikam::DDatePicker, [896](#)
- dateTime
  - Digikam::DDateTimeEdit, [906](#)
  - Digikam::DisjointMetadata, [1045](#)
  - Digikam::ItemInfo, [2118](#)
- dateTimeChanged
  - Digikam::DDateTimeEdit, [906](#)
- dateTimeInterval
- Digikam::DisjointMetadata, [1045](#)
- DB
  - Digikam::OpenCVDNNFaceRecognizer, [2601](#)
- DbKeysCollection
  - Digikam::DbKeysCollection, [807](#)
- dcbIterations
  - Digikam::DRawDecoderSettings, [1261](#)
- DColor
  - Digikam::DColor, [834](#)
- DConfigDlg
  - Digikam::DConfigDlg, [848](#)
- DConfigDlgMgr
  - Digikam::DConfigDlgMgr, [855](#)
- DConfigDlgTitle
  - Digikam::DConfigDlgTitle, [862](#)
- DConfigDlgWdg
  - Digikam::DConfigDlgWdg, [873](#)
- DConfigDlgWdgItem
  - Digikam::DConfigDlgWdgItem, [880](#)
- DConfigDlgWdgModel
  - Digikam::DConfigDlgWdgModel, [884](#)
- DDatePicker
  - Digikam::DDatePicker, [895](#)
- DDatePickerPopup
  - Digikam::DDatePickerPopup, [899](#)
- DDateTimeEdit
  - Digikam::DDateTimeEdit, [906](#)
- deactivate
  - Digikam::WorkerObject, [3397](#)
- DeactivatingMode
  - Digikam::WorkerObject, [3396](#)
- decodeHalfRAWImage
  - Digikam::DRawDecoder, [1254](#)
- decodeRAWImage
  - Digikam::DRawDecoder, [1254](#)
- DecodingQuality
  - Digikam::DRawDecoderSettings, [1260](#)
- DecorateTool
  - Digikam::BatchTool, [460](#)
- decorationRoleData
  - Digikam::AbstractAlbumModel, [152](#)
  - Digikam::AlbumModel, [309](#)
  - Digikam::DateAlbumModel, [753](#)
  - Digikam::TagModel, [3166](#)
- defaultComment
  - Digikam::ItemComments, [2024](#)
- defaultFieldOperator
  - Digikam::SearchXmlReader, [2938](#)
- defaultLocation
  - Digikam::DNotificationPopup, [1153](#)
- DefaultOrder
  - Digikam::CamItemSortSettings, [543](#)
  - Digikam::ItemSortSettings, [2231](#)
  - ShowFoto::ShowfotoItemSortSettings, [3479](#)
- defaultParameters
  - Digikam::DbEngineParameters, [787](#)

- defaultSearchPaths
  - Digikam::IccProfile, [1764](#)
- defaultThread
  - Digikam::ThumbnailLoadThread, [3265](#)
- defaultUploadUrl
  - Digikam::DBInfolface, [796](#)
  - Digikam::DInfoInterface, [1032](#)
  - Digikam::DMetalInfolface, [1099](#)
- deleteAlbum
  - Digikam::CoreDB, [655](#)
- deleteAlbumRoot
  - Digikam::CoreDB, [655](#)
- deleteAllSpacersOrNewTags
  - Digikam::RGTagModel, [2785](#)
- Deleted
  - Digikam::CollectionImageChangeset, [585](#)
- DeleteDecorationRole
  - Digikam::SetupCollectionModel, [2960](#)
- deleteDirRecursively
  - Digikam::DTrash, [1293](#)
- deleteFiles
  - Digikam::IOJobsThread, [1990](#)
- deleteFilterInstance
  - Digikam::EditorToolThreaded, [1344](#)
- deleteImage
  - Digikam::DBInfolface, [796](#)
  - Digikam::DInfoInterface, [1032](#)
  - Digikam::DMetalInfolface, [1100](#)
  - Digikam::DTrash, [1294](#)
- deleteItem
  - Digikam::CoreDB, [655](#), [656](#)
  - Digikam::GPCamera, [1632](#)
  - Digikam::UMSCamera, [3340](#)
- deleteObsoleteItem
  - Digikam::CoreDB, [656](#)
- deleteRemovedItems
  - Digikam::CoreDB, [656](#)
- deleteSAlbum
  - Digikam::AlbumManager, [291](#)
- deleteSearch
  - Digikam::CoreDB, [656](#)
- deleteTag
  - Digikam::CoreDB, [657](#)
  - Digikam::RGTagModel, [2785](#)
- deleteTAlbum
  - Digikam::AlbumManager, [291](#)
- deleteThumbnail
  - Digikam::ThumbnailLoadThread, [3265](#)
- deleteThumbnailsFromDisk
  - Digikam::ThumbnailCreator, [3248](#)
- depth\_first\_search\_sorted
  - Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch, [1690](#)
- description
  - Digikam::FilterAction, [1555](#)
  - Digikam::IccProfile, [1764](#)
  - Digikam::Token, [3295](#)
- deselected
  - Digikam::ImportCategorizedView, [1850](#)
  - ShowFoto::ShowfotoCategorizedView, [3432](#)
- destinationGeographicPoint
  - Digikam::GeodeticCalculator, [1612](#)
- detach
  - Digikam::DImg, [989](#)
- detAdjustmentByClockPhotoUrl
  - Digikam::TimeAdjustSettings, [3286](#)
- detect
  - Digikam::AestheticDetector, [253](#)
  - Digikam::BlurDetector, [486](#)
  - Digikam::CompressionDetector, [622](#)
  - Digikam::ExposureDetector, [1399](#)
  - Digikam::NoiseDetector, [2565](#)
- detectAccuracy
  - Digikam::FaceScanSettings, [1480](#)
- DetectAndRecognize
  - Digikam::FaceScanSettings, [1479](#)
- detectedFormat
  - Digikam::DImg, [989](#)
- detectFaces
  - Digikam::DNNFaceDetectorSSD, [1122](#)
  - Digikam::DNNFaceDetectorYOLO, [1124](#)
  - Digikam::DNNFaceDetectorYuNet, [1126](#)
  - Digikam::FaceDetector, [1413](#)
  - Digikam::OpenCVDNNFaceDetector, [2599](#)
- detectLanguage
  - Digikam::DOnlineTranslator, [1172](#)
- detectLanguageAlt
  - Digikam::MetaEngine, [2487](#)
- DetectorModel
  - Digikam::AutoTagsScanSettings, [425](#)
- DetectorNNModel
  - Digikam, [134](#)
- DFontProperties
  - Digikam::DFontProperties, [944](#)
- Digikam, [95](#)
  - accessCol, [141](#)
  - accessRow, [141](#)
  - ActionJobCollection, [134](#)
  - adjustedEnvironmentForApplmage, [136](#)
  - ChildToParent, [136](#)
  - coordinatesToClipboard, [136](#)
  - CR\_basis, [141](#)
  - DetectorNNModel, [134](#)
  - DItemsListIsLessThanHandler, [134](#)
  - DNNDetectorSSD, [134](#)
  - DNNDetectorYOLOv3, [134](#)
  - DNNDetectorYuNet, [134](#)
  - DNotificationWrapper, [136](#)
  - ExifHumanList, [141](#)
  - FACE\_TEMPLATE, [142](#)
  - faceenum2size, [142](#)
  - fastNumberToString, [137](#)
  - FS\_ALBUMGUI, [134](#)
  - FS\_EDITOR, [134](#)
  - FS\_IMPORTUI, [134](#)
  - FS\_LIGHTTABLE, [134](#)



- FS\_NONE, 134
- FS\_SIDEBARS, 134
- FS\_STATUSBAR, 134
- FS\_THUMBBAR, 134
- FS\_TOOLBARS, 134
- FullImageHistogram, 135
- FullScreenOptions, 134
- GeoGroupStateEnum, 135
- GeofaceHelperParseLatLonString, 137
- GeofaceMinMarkerGroupingRadius, 142
- HistogramRenderingType, 135
- HistogramScale, 135
- HS\_None, 135
- HudSide, 135
- image2Mat, 137
- image2Mat\_shared, 138
- ImageSelectionHistogram, 135
- IptcHumanList, 142
- LinScaleHistogram, 135
- LogScaleHistogram, 135
- MeaningOfDirection, 135
- namespaceTitleDefinitions, 143
- openOnlineDocumentation, 138
- OperationType, 136
- operator<<, 138
- ParentToChild, 136
- QPointSquareDistance, 139
- RESNET50, 136
- s\_inlineTranslateString, 139
- s\_rawFileExtensionsdWithDesc, 139
- s\_rawFileExtensionsVersion, 140
- s\_setXmpTagStringFromEntry, 140
- setExifXmpTagDataVariant, 140
- spectral\_chromaticity, 143
- supportedImageMimeTypes, 140
- UnspecifiedOps, 136
- videoStrip16, 143
- videoStrip4, 144
- videoStrip8, 144
- XmpHumanList, 144
- YOLOV5NANO, 136
- YOLOV5XLARGE, 136
- YoloVersions, 136
- digikam project API reference., 1
- Digikam::AbstractAlbumModel, 148
  - AbstractAlbumModel, 151
  - albumCleared, 151
  - albumData, 151
  - AlbumDataRole, 150
  - AlbumGlobalIdRole, 150
  - AlbumIdRole, 150
  - AlbumPointerRole, 150
  - AlbumSortRole, 150
  - AlbumTitleRole, 150
  - AlbumTypeRole, 150
  - allAlbumsCleared, 151
  - columnHeader, 152
  - decorationRoleData, 152
  - filterAlbum, 152
  - fontRoleData, 152
  - IgnoreRootAlbum, 151
  - IncludeRootAlbum, 151
  - retrieveAlbum, 152
  - rootAlbumAvailable, 152
  - RootAlbumBehavior, 151
  - rootAlbumIndex, 152
  - setEnableDrag, 153
  - sortRoleData, 153
- Digikam::AbstractAlbumTreeView, 153
  - AbstractAlbumTreeView, 158
  - addCustomContextMenuActions, 158
  - AlwaysShowInclusiveCounts, 158
  - contextMenuIcon, 159
  - contextMenuTitle, 159
  - CreateDefaultDelegate, 158
  - CreateDefaultFilterModel, 158
  - CreateDefaultModel, 158
  - doLoadState, 159
  - doSaveState, 159
  - expandEverything, 159
  - expandMatches, 160
  - Flag, 158
  - handleCustomContextMenuAction, 160
  - indexVisuallyAt, 160
  - pixmapForDrag, 160
  - selectedAlbumsChanged, 160
  - setAlbumManagerCurrentAlbum, 161
  - setContextMenuIcon, 161
  - setCurrentAlbums, 161
  - setEnableContextMenu, 161
  - setSelectAlbumOnClick, 161
  - setSelectOnContextMenu, 162
  - showContextMenuAt, 162
  - ShowCountAccordingToSettings, 158
- Digikam::AbstractAlbumTreeView::ContextMenuElement, 162
  - addActions, 163
- Digikam::AbstractAlbumTreeViewSelectComboBox, 164
  - AbstractAlbumTreeViewSelectComboBox, 167
  - addCheckUncheckContextMenuActions, 167
  - installView, 167
  - sendViewportEventToView, 167
  - setTreeView, 168
- Digikam::AbstractCheckableAlbumModel, 169
  - AbstractCheckableAlbumModel, 174
  - albumCleared, 174
  - albumData, 174
  - allAlbumsCleared, 175
  - checkStateChanged, 175
  - setData, 175
  - setRootCheckable, 175
  - setTristate, 175
- Digikam::AbstractCheckableAlbumTreeView, 177
  - AbstractCheckableAlbumTreeView, 182
  - doLoadState, 182
  - doSaveState, 182

- isRestoreCheckState, 182
- middleButtonPressed, 182
- setRestoreCheckState, 182
- Digikam::AbstractCountingAlbumModel, 183
  - albumCleared, 187
  - albumCount, 187
  - albumData, 187
  - albumForId, 187
  - albumName, 187
  - allAlbumsCleared, 188
  - excludeChildrenCount, 188
  - includeChildrenCount, 188
  - setCountHash, 188
- Digikam::AbstractCountingAlbumTreeView, 189
- Digikam::AbstractDetector, 193
- Digikam::AbstractItemDragDropHandler, 194
  - accepts, 195
  - acceptsMimeData, 195
  - createMimeData, 195
  - dropEvent, 195
  - mimeType, 195
- Digikam::AbstractMarkerTiler, 196
  - bestRepresentativeIndexFromList, 197
  - getTile, 197
  - getTileGroupState, 198
  - getTileRepresentativeMarker, 198
  - indicesEqual, 198
  - onIndicesClicked, 198
  - pixmapFromRepresentativeIndex, 198
  - prepareTiles, 198
  - setActive, 199
  - tilerFlags, 199
- Digikam::AbstractMarkerTiler::ClickInfo, 199
- Digikam::AbstractMarkerTiler::NonEmptyIterator, 199
- Digikam::AbstractMarkerTiler::Tile, 200
- Digikam::AbstractSearchGroupContainer, 201
  - addGroupToLayout, 202
  - createSearchGroup, 202
- Digikam::AbstractSpecificAlbumModel, 203
  - columnHeader, 206
- Digikam::AbstractWidgetDelegateOverlay, 206
  - AbstractWidgetDelegateOverlay, 208
  - checkIndex, 208
  - createWidget, 208
  - hide, 209
  - setActive, 209
  - slotEntered, 209
  - slotReset, 209
  - viewportLeaveEvent, 209
  - widgetEnterEvent, 210
- Digikam::ActionCategorizedView, 211
- Digikam::ActionData, 213
- Digikam::ActionItemModel, 214
  - actionForIndex, 216
  - ActionItemModel, 216
  - createFilterModel, 216
  - hover, 216
  - itemForAction, 216
- MenuCategoryFlag, 215
- ParentMenuCategory, 216
- SortCategoriesAlphabetically, 216
- SortCategoriesByInsertionOrder, 216
- ToplevelMenuCategory, 216
- Digikam::ActionJob, 217
  - ~ActionJob, 218
- Digikam::ActionSortFilterProxyModel, 218
- Digikam::ActionTask, 220
- Digikam::ActionThread, 222
- Digikam::ActionThreadBase, 224
  - appendJobs, 225
  - setDefaultMaximumNumberOfThreads, 225
- Digikam::ActionVersionsOverlay, 226
  - checkIndex, 229
  - createButton, 229
  - setActive, 229
  - updateButton, 230
- Digikam::AddBookmarkDialog, 230
- Digikam::AddBookmarkProxyModel, 231
- Digikam::AddTagsComboBox, 232
  - currentTaggingAction, 236
  - setAlbumModels, 236
  - taggingActionSelected, 236
- Digikam::AddTagsLineEdit, 237
  - setFilterModel, 238
  - setParentTag, 238
  - taggingActionSelected, 238
- Digikam::AdvancedMetadataTab, 239
  - AdvancedMetadataTab, 239
- Digikam::AdvancedRenameDialog, 240
- Digikam::AdvancedRenameInput, 241
- Digikam::AdvancedRenameLineEdit, 242
- Digikam::AdvancedRenameListItem, 243
- Digikam::AdvancedRenameManager, 244
- Digikam::AdvancedRenameProcessDialog, 246
- Digikam::AdvancedRenameWidget, 248
  - parse, 249
  - setControlWidgets, 249
  - setLayoutStyle, 250
  - setParser, 250
  - setParseString, 250
- Digikam::AdvancedSettings, 251
- Digikam::AestheticDetector, 252
  - detect, 253
- Digikam::Akonadiface, 253
- Digikam::Album, 254
  - ~Album, 257
  - childAlbumIds, 257
  - childAlbums, 257
  - childAtRow, 257
  - childCount, 257
  - databaseUrl, 257
  - DATE, 256
  - extraData, 257
  - FACE, 256
  - firstChild, 258
  - globalID, 258

- id, [259](#)
- isAncestorOf, [259](#)
- isRoot, [259](#)
- isTrashAlbum, [259](#)
- isUsedByLabelsTree, [259](#)
- lastChild, [260](#)
- next, [260](#)
- parent, [260](#)
- PHYSICAL, [256](#)
- prev, [260](#)
- removeExtraData, [260](#)
- rowFromAlbum, [261](#)
- SEARCH, [256](#)
- setExtraData, [261](#)
- setUsedByLabelsTree, [261](#)
- TAG, [256](#)
- title, [262](#)
- Type, [256](#)
- type, [262](#)
- Digikam::AlbumChangeset, [262](#)
- Digikam::AlbumCopyMoveHint, [263](#)
- Digikam::AlbumCustomizer, [264](#)
- Digikam::AlbumDragDropHandler, [265](#)
  - accepts, [266](#)
  - createMimeData, [266](#)
  - dropEvent, [266](#)
  - mimeTypes, [266](#)
- Digikam::AlbumFilterModel, [268](#)
  - ChildMatch, [270](#)
  - DirectMatch, [270](#)
  - FilterBehavior, [270](#)
  - FullFiltering, [270](#)
  - hasSearchResult, [271](#)
  - isFiltering, [271](#)
  - lessThan, [271](#)
  - matches, [271](#)
  - MatchResult, [270](#)
  - matchResult, [271](#), [272](#)
  - NoMatch, [270](#)
  - ParentMatch, [270](#)
  - searchTextSettings, [272](#)
  - searchTextSettingsAboutToChange, [272](#)
  - searchTextSettingsChanged, [272](#)
  - setFilterBehavior, [272](#)
  - setSearchTextSettings, [272](#)
  - setSourceAlbumModel, [273](#)
  - setSourceFilterModel, [273](#)
  - setSourceModel, [273](#)
  - SimpleFiltering, [270](#)
  - SpecialMatch, [270](#)
  - StrictFiltering, [270](#)
- Digikam::AlbumFolderViewSideBarWidget, [274](#)
  - applySettings, [276](#)
  - changeAlbumFromHistory, [276](#)
  - doLoadState, [276](#)
  - doSaveState, [277](#)
  - getCaption, [277](#)
  - getIcon, [277](#)
    - setActive, [277](#)
- Digikam::AlbumHistory, [278](#)
  - addAlbums, [279](#)
- Digikam::AlbumInfo, [280](#)
- Digikam::AlbumIterator, [280](#)
- Digikam::AlbumLabelsSearchHandler, [281](#)
  - albumForSelectedItems, [281](#)
  - generatedName, [281](#)
  - imagesUrls, [282](#)
  - isRestoringSelectionFromHistory, [282](#)
  - restoreSelectionFromHistory, [282](#)
- Digikam::AlbumManager, [282](#)
  - albumTitles, [287](#)
  - allDAAlbums, [287](#)
  - allPAAlbums, [287](#)
  - allSAAlbums, [287](#)
  - allTAAlbums, [288](#)
  - changeDatabase, [288](#)
  - createPAAlbum, [288](#), [289](#)
  - createSAAlbum, [289](#)
  - createTAAlbum, [290](#)
  - currentAlbums, [290](#)
  - currentPAAlbum, [290](#)
  - currentTAAlbums, [291](#)
  - deleteSAAlbum, [291](#)
  - deleteTAAlbum, [291](#)
  - findAlbum, [292](#)
  - findDAAlbum, [292](#)
  - findOrCreateTAAlbums, [293](#)
  - findPAAlbum, [293](#)
  - findSAAlbum, [294](#)
  - findSAAlbumsBySearchType, [294](#)
  - findTAAlbum, [295](#)
  - getDAAlbumsCount, [295](#)
  - getFaceCount, [295](#)
  - getItemFromAlbum, [295](#)
  - getPAAlbumsCount, [296](#)
  - getRecentlyAssignedTags, [296](#)
  - getTAAlbumsCount, [296](#)
  - getUnconfirmedFaceCount, [296](#)
  - isMovingAlbum, [296](#)
  - mergeTAAlbum, [297](#)
  - moveTAAlbum, [297](#)
  - refresh, [297](#)
  - renamePAAlbum, [298](#)
  - renameTAAlbum, [298](#)
  - setCurrentAlbums, [299](#)
  - setDatabase, [299](#)
  - signalAlbumAboutToBeMoved, [299](#)
  - signalAlbumHasBeenDeleted, [299](#)
  - signalAlbumMoved, [299](#)
  - signalShowOnlyAvailableAlbumsChanged, [299](#)
  - startScan, [299](#)
  - tagNames, [300](#)
  - tagPaths, [300](#), [301](#)
  - updatePAAlbumIcon, [301](#)
  - updateSAAlbum, [301](#)
  - updateTAAlbumIcon, [302](#)

- Digikam::AlbumModel, 303
  - albumData, 309
  - albumForId, 309
  - decorationRoleData, 309
- Digikam::AlbumModelDragDropHandler, 310
  - accepts, 311
  - acceptsMimeData, 311
  - createMimeData, 311
  - dropEvent, 311
  - mimeTypes, 311
- Digikam::AlbumModificationHelper, 312
  - AlbumModificationHelper, 313
  - bindAlbum, 313
  - boundAlbum, 313
  - slotAlbumDelete, 314
  - slotAlbumEdit, 314
  - slotAlbumNew, 314
  - slotAlbumRename, 314
- Digikam::AlbumParser, 315
- Digikam::AlbumPointer< T >, 317
- Digikam::AlbumPointerList< T >, 318
- Digikam::AlbumPropsEdit, 319
- Digikam::AlbumRootChangeset, 320
- Digikam::AlbumRootInfo, 320
- Digikam::AlbumsDBJobInfo, 321
- Digikam::AlbumsDBJobsThread, 322
  - albumsListing, 324
- Digikam::AlbumSelectComboBox, 325
  - installView, 328
  - model, 328
  - setCheckable, 328
  - setCloseOnActivate, 328
  - setDefaultAlbumModel, 328
  - setNoSelectionText, 328
  - setShowCheckStateSummary, 328
  - updateText, 329
- Digikam::AlbumSelectDialog, 329
- Digikam::AlbumSelectionTreeView, 330
  - signalFindDuplicates, 335
- Digikam::AlbumSelectors, 336
  - AlbumSelectors, 337
  - loadState, 338
  - saveState, 338
  - setAlbumSelected, 338
  - setTagSelected, 338
- Digikam::AlbumSelectTabs, 339
- Digikam::AlbumSelectTreeView, 339
  - addCustomContextMenuActions, 345
  - AlbumSelectTreeView, 345
  - handleCustomContextMenuAction, 345
- Digikam::AlbumSelectWidget, 346
- Digikam::AlbumShortInfo, 347
- Digikam::AlbumSimplified, 347
- Digikam::AlbumsJob, 348
- Digikam::AlbumThumbnailLoader, 350
  - getAlbumThumbnail, 352
  - getAlbumThumbnailDirectly, 352
  - getStandardTagIcon, 352
  - getTagThumbnail, 352
  - getTagThumbnailDirectly, 352
  - instance, 353
  - RelativeSize, 352
  - setThumbnailSize, 353
  - signalFailed, 353
  - signalReloadThumbnails, 353
  - signalThumbnail, 353
- Digikam::AlbumTreeView, 354
- Digikam::AlbumTreeViewSelectComboBox, 359
- Digikam::AlbumWatch, 362
- Digikam::AltLangStrEdit, 364
  - addCurrent, 366
  - AltLangStrEdit, 366
  - setLinesVisible, 366
  - setTitle, 366
  - setTitleWidget, 366
  - slotEnabledInternalWidgets, 366
  - titleWidget, 367
- Digikam::AnimatedClearButton, 368
  - setShallBeShown, 369
  - stayVisibleWhenAnimatedOut, 369
- Digikam::AnimatedVisibility, 370
  - AnimatedVisibility, 371
- Digikam::AntiVignettingContainer, 371
- Digikam::AntiVignettingFilter, 372
  - filterAction, 376
  - filterIdentifier, 376
  - readParameters, 376
- Digikam::AntiVignettingSettings, 376
- Digikam::ApplicationSettings, 377
  - askGroupingOperateOnAll, 383
  - getGroupingOperateOnAll, 383
  - getStringComparisonType, 384
  - Natural, 383
  - Normal, 383
  - operationTypeExplanation, 384
  - operationTypeTitle, 384
  - readMsgBoxShouldBeShown, 384
  - saveMsgBoxShouldBeShown, 385
  - setGroupingOperateOnAll, 385
  - setStringComparisonType, 385
  - StringComparisonType, 383
- Digikam::AssignedBatchTools, 386
- Digikam::AssignedListView, 387
- Digikam::AssignedListViewItem, 388
- Digikam::AssignNameOverlay, 390
  - checkIndex, 393
  - createWidget, 393
  - setActive, 393
  - setFocusOnWidget, 394
  - showOnIndex, 394
  - updateFace, 394
  - viewportLeaveEvent, 394
  - visualChange, 394
  - widgetEnterEvent, 394
  - widgetLeaveEvent, 395
- Digikam::AssignNameWidget, 396

- assigned, 398
- rejected, 398
- selected, 398
- setMode, 398
- setUserData, 399
- Digikam::AssignNameWidgetStates, 400
- Digikam::AudPlayerWdg, 403
- Digikam::AutoCrop, 404
  - startAnalyse, 408
- Digikam::AutoExpoFilter, 409
  - filterAction, 413
  - filterIdentifier, 413
  - readParameters, 414
- Digikam::AutoLevelsFilter, 415
  - filterAction, 419
  - filterIdentifier, 419
  - readParameters, 419
- Digikam::AutoTagsAssign, 419
  - generateTagsList, 419
- Digikam::AutotagsAssignment, 420
  - AutotagsAssignment, 422
  - setUseMultiCoreCPU, 423
- Digikam::AutotagsAssignmentTask, 423
- Digikam::AutoTagsScanSettings, 425
  - AllItems, 425
  - DetectorModel, 425
  - NonAssignedItems, 425
  - ScanMode, 425
  - YOLOV5NANO, 425
  - YOLOV5XLARGE, 425
- Digikam::AutoTagsScanWidget, 426
  - doLoadState, 427
  - doSaveState, 427
- Digikam::BackendGeonamesRG, 428
  - BackendGeonamesRG, 429
  - backendName, 429
  - callRGBBackend, 429
  - cancelRequests, 430
  - getErrorMessage, 430
  - makeQMapFromXML, 430
- Digikam::BackendGeonamesUSRG, 430
  - BackendGeonamesUSRG, 432
  - backendName, 432
  - callRGBBackend, 432
  - cancelRequests, 433
  - getErrorMessage, 433
  - makeQMapFromXML, 433
- Digikam::BackendGoogleMaps, 434
  - ~BackendGoogleMaps, 436
  - addActionsToConfigurationMenu, 436
  - backendHumanName, 436
  - backendName, 437
  - centerOn, 437
  - geoCoordinates, 437
  - getCenter, 437
  - getMarkerModelLevel, 437
  - getNormalizedBounds, 437
  - getZoom, 437
- isReady, 438
- mapSize, 438
- mapWidget, 438
- mapWidgetDocked, 438
- mouseModeChanged, 438
- readSettingsFromGroup, 438
- regionSelectionChanged, 438
- releaseWidget, 439
- reload, 439
- saveSettingsToGroup, 439
- screenCoordinates, 439
- setActive, 439
- setCenter, 439
- setMarkerPixmap, 439
- setZoom, 440
- updateActionAvailability, 440
- updateClusters, 440
- updateMarkers, 440
- zoomIn, 440
- zoomOut, 440
- Digikam::BackendMarble, 441
  - ~BackendMarble, 444
  - addActionsToConfigurationMenu, 444
  - applyCacheToWidget, 444
  - backendHumanName, 444
  - backendName, 444
  - centerOn, 444
  - eventFilter, 444
  - geoCoordinates, 444
  - GeoPainter\_drawPixmapAtCoordinates, 445
  - getCenter, 445
  - getMarkerModelLevel, 445
  - getNormalizedBounds, 445
  - getProjection, 445
  - getZoom, 445
  - isReady, 446
  - mapSize, 446
  - mapWidget, 446
  - mapWidgetDocked, 446
  - marbleCustomPaint, 446
  - mouseModeChanged, 446
  - readSettingsFromGroup, 446
  - regionSelectionChanged, 446
  - releaseWidget, 447
  - reload, 447
  - saveSettingsToGroup, 447
  - screenCoordinates, 447
  - setActive, 447
  - setCenter, 447
  - setZoom, 447
  - slotScheduleUpdate, 448
  - updateActionAvailability, 448
  - updateClusters, 448
  - updateMarkers, 448
  - zoomIn, 448
  - zoomOut, 448
- Digikam::BackendMarbleLayer, 449
- Digikam::BackendOsmRG, 449

- backendName, [451](#)
- BackendOsmRG, [451](#)
- callRGBBackend, [451](#)
- cancelRequests, [452](#)
- getErrorMessage, [452](#)
- makeQMapFromXML, [452](#)
- Digikam::BalooInfo, [452](#)
- Digikam::BalooWrap, [452](#)
  - getSemanticInfo, [454](#)
  - setSemanticInfo, [454](#)
- Digikam::BasicDImgFilterGenerator< T >, [455](#)
  - BasicDImgFilterGenerator, [456](#)
  - createFilter, [456](#)
  - displayableName, [456](#)
  - supportedFilters, [456](#)
  - supportedVersions, [456](#)
- Digikam::BatchTool, [457](#)
  - apply, [461](#)
  - applyFilter, [461](#)
  - BaseTool, [460](#)
  - BatchTool, [461](#)
  - BatchToolGroup, [460](#)
  - cancel, [461](#)
  - clone, [461](#)
  - ColorTool, [460](#)
  - ConvertTool, [460](#)
  - CustomTool, [460](#)
  - DecorateTool, [460](#)
  - EnhanceTool, [460](#)
  - FiltersTool, [460](#)
  - isCancelled, [461](#)
  - MetadataTool, [460](#)
  - outputSuffix, [461](#)
  - registerSettingsWidget, [462](#)
  - savefromDImg, [462](#)
  - setOutputUrlFromInputUrl, [462](#)
  - setSettings, [462](#)
  - settingsWidget, [462](#)
  - signalAssignSettings2Widget, [462](#)
  - slotAssignSettings2Widget, [462](#)
  - toolGroup, [463](#)
  - toolOperations, [463](#)
  - toolVersion, [463](#)
  - TransformTool, [460](#)
- Digikam::BatchToolSet, [463](#)
  - operator==, [464](#)
- Digikam::BatchToolsFactory, [464](#)
- Digikam::BCGContainer, [465](#)
- Digikam::BCGFilter, [466](#)
  - filterAction, [470](#)
  - filterIdentifier, [470](#)
  - readParameters, [470](#)
- Digikam::BCGSettings, [470](#)
- Digikam::BdEngineBackend, [471](#)
  - asDBDateTime, [475](#)
  - BdEngineBackend, [475](#)
  - checkOrSetWALMode, [475](#)
  - close, [475](#)
  - ConnectionError, [475](#)
  - connectionErrorHandling, [476](#)
  - execDBAction, [476](#)
  - execDBActionQuery, [476](#)
  - execDirectSql, [476](#)
  - execDirectSqlWithResult, [476](#)
  - execQuery, [477](#)
  - execSql, [477](#)
  - execUpsertDBAction, [477](#)
  - handleQueryResult, [477](#)
  - isInTransaction, [478](#)
  - lastError, [478](#)
  - lastSQLError, [478](#)
  - maximumBoundValues, [478](#)
  - NoErrors, [475](#)
  - Open, [475](#)
  - open, [478](#)
  - OpenSchemaChecked, [475](#)
  - queryErrorHandling, [478](#)
  - QueryStateEnum, [474](#)
  - readToList, [479](#)
  - setDbEngineErrorHandler, [479](#)
  - setForeignKeyChecks, [479](#)
  - SQLException, [475](#)
  - Status, [475](#)
  - Unavailable, [475](#)
- Digikam::BdEngineBackend::QueryState, [479](#)
- Digikam::BlackFrameListView, [480](#)
- Digikam::BlackFrameListViewItem, [481](#)
- Digikam::BlackFrameParser, [482](#)
- Digikam::BlackFrameToolTip, [483](#)
  - repositionRect, [484](#)
  - tipContents, [484](#)
- Digikam::BlurDetector, [485](#)
  - detect, [486](#)
- Digikam::BlurFilter, [487](#)
  - filterAction, [491](#)
  - filterIdentifier, [491](#)
  - readParameters, [491](#)
- Digikam::BlurFXFilter, [492](#)
  - filterAction, [496](#)
  - filterIdentifier, [496](#)
  - readParameters, [496](#)
- Digikam::BookmarkNode, [497](#)
- Digikam::BookmarksDialog, [498](#)
- Digikam::BookmarksManager, [499](#)
- Digikam::BookmarksMenu, [500](#)
  - prePopulated, [502](#)
- Digikam::BookmarksModel, [503](#)
- Digikam::BorderContainer, [504](#)
- Digikam::BorderFilter, [505](#)
  - filterAction, [509](#)
  - filterIdentifier, [509](#)
  - readParameters, [509](#)
- Digikam::BorderSettings, [509](#)
- Digikam::BqmInfolface, [511](#)
- Digikam::BuildTrashCountersJob, [514](#)
- Digikam::BWSepiaContainer, [515](#)



- BlackWhiteConversionType, [516](#)
- BWGeneric, [516](#)
- BWIlfordSFX200, [516](#)
- BWKodakHIE, [516](#)
- BWNoFilter, [516](#)
- BWNoTone, [516](#)
- Digikam::BWSepiaFilter, [517](#)
  - filterAction, [521](#)
  - filterIdentifier, [521](#)
  - readParameters, [521](#)
- Digikam::BWSepiaSettings, [521](#)
- Digikam::CameraAutoDetectThread, [522](#)
- Digikam::CameraController, [524](#)
- Digikam::CameraFolderDialog, [526](#)
- Digikam::CameraFolderItem, [527](#)
- Digikam::CameraFolderView, [528](#)
- Digikam::CameraHistoryUpdater, [529](#)
- Digikam::CameraInfoDialog, [530](#)
- Digikam::CameraItem, [531](#)
- Digikam::CameraItemList, [532](#)
- Digikam::CameraList, [533](#)
- Digikam::CameraMessageBox, [534](#)
  - warningContinueCancelList, [534](#)
- Digikam::CameraNameHelper, [534](#)
- Digikam::CameraNameOption, [535](#)
  - parseOperation, [537](#)
- Digikam::CameraSelection, [538](#)
- Digikam::CameraThumbsCtrl, [539](#)
  - getThumbInfo, [539](#)
- Digikam::CameraType, [540](#)
- Digikam::CamItemInfo, [540](#)
  - downloaded, [542](#)
  - DownloadedNo, [541](#)
  - DownloadedYes, [541](#)
  - DownloadFailed, [541](#)
  - DownloadStarted, [541](#)
  - DownloadStatus, [541](#)
  - DownloadUnknown, [541](#)
  - NewPicture, [541](#)
  - size, [542](#)
- Digikam::CamItemSortSettings, [542](#)
  - compare, [543](#)
  - compareCategories, [544](#)
  - DefaultOrder, [543](#)
  - lessThan, [544](#)
  - SortOrder, [543](#)
- Digikam::Canvas, [545](#)
- Digikam::CaptionEdit, [549](#)
- Digikam::CaptionsMap, [550](#)
  - setAuthorsList, [551](#)
- Digikam::CaptionValues, [552](#)
- Digikam::CaptureDlg, [552](#)
- Digikam::CaptureWidget, [553](#)
- Digikam::CaseModifier, [554](#)
  - parseOperation, [556](#)
- Digikam::CategorizedItemModel, [557](#)
  - ExtraRoles, [558](#)
  - ItemOrderRole, [558](#)
- Digikam::CBContainer, [558](#)
- Digikam::CBFilter, [559](#)
  - filterAction, [563](#)
  - filterIdentifier, [563](#)
  - readParameters, [563](#)
- Digikam::CBSettings, [563](#)
- Digikam::ChangeBookmarkCommand, [564](#)
- Digikam::ChangeFaceRecognitionModelDlg, [565](#)
- Digikam::CharcoalFilter, [566](#)
  - filterAction, [570](#)
  - filterIdentifier, [570](#)
  - readParameters, [570](#)
- Digikam::CheckableAlbumFilterModel, [570](#)
  - isFiltering, [574](#)
  - matches, [574](#)
- Digikam::ChoiceSearchComboBox, [575](#)
  - ChoiceSearchComboBox, [577](#)
  - installView, [577](#)
  - setSearchModel, [577](#)
- Digikam::ChoiceSearchModel, [578](#)
  - checkedKeys, [579](#)
  - setChecked, [579](#)
  - setChoice, [580](#)
- Digikam::ChoiceSearchModel::Entry, [580](#)
  - operator==, [580](#)
- Digikam::CIETongueWidget, [581](#)
- Digikam::ClickDragReleaseItem, [582](#)
  - started, [583](#)
- Digikam::ClockPhotoDialog, [583](#)
  - setImage, [584](#)
- Digikam::CMat, [584](#)
- Digikam::CollectionImageChangeset, [584](#)
  - Added, [585](#)
  - CollectionImageChangeset, [585](#)
  - Copied, [585](#)
  - Deleted, [585](#)
  - ids, [586](#)
  - Moved, [585](#)
  - Operation, [585](#)
  - operator<<, [586](#)
  - Removed, [585](#)
  - RemovedAll, [585](#)
  - RemovedDeleted, [585](#)
- Digikam::CollectionLocation, [586](#)
  - albumRootPath, [588](#)
  - asQtCaseSensitivity, [588](#)
  - CaseInsensitive, [587](#)
  - CaseSensitive, [587](#)
  - CaseSensitivity, [587](#)
  - caseSensitivity, [588](#)
  - LocationAvailable, [587](#)
  - LocationDeleted, [587](#)
  - LocationHidden, [587](#)
  - LocationNull, [587](#)
  - LocationUnavailable, [587](#)
  - Network, [588](#)
  - Status, [587](#)
  - status, [588](#)

- Type, [587](#)
- type, [588](#)
- Undefined, [588](#)
- UnknownCaseSensitivity, [587](#)
- VolumeHardWired, [588](#)
- VolumeRemovable, [588](#)
- Digikam::CollectionManager, [589](#)
  - addLocation, [592](#)
  - album, [592](#)
  - albumRoot, [592](#)
  - albumRootLabel, [592](#)
  - albumRootPath, [592](#)
  - checkHardWiredLocations, [593](#)
  - checkLocation, [593](#)
  - isAlbumRoot, [593](#)
  - LocationAllRight, [592](#)
  - LocationCheckResult, [591](#)
  - locationForAlbumRoot, [593](#)
  - locationForUrl, [593](#)
  - LocationHasProblems, [592](#)
  - LocationInvalidCheck, [592](#)
  - LocationNotAllowed, [592](#)
  - locationStatusChanged, [594](#)
  - migrateToVolume, [594](#)
  - oneAlbumRoot, [594](#)
  - refresh, [594](#)
  - removeLocation, [594](#)
  - setWatchDisabled, [594](#)
- Digikam::CollectionPage, [595](#)
- Digikam::CollectionScanner, [597](#)
  - CleanScan, [600](#)
  - completeScan, [600](#)
  - createHintContainer, [600](#)
  - databaseInitialScanDone, [600](#)
  - FileScanMode, [599](#)
  - finishCompleteScan, [600](#)
  - finishedScanningAlbumRoot, [600](#)
  - ModifiedScan, [600](#)
  - NormalScan, [600](#)
  - partialScan, [600](#)
  - Rescan, [600](#)
  - scanFile, [601](#)
  - scannedFiles, [601](#)
  - setNeedFileCount, [601](#)
  - setPerformFastScan, [601](#)
  - setSignalsEnabled, [601](#)
  - totalFilesToScan, [602](#)
- Digikam::CollectionScannerHintContainer, [602](#)
- Digikam::CollectionScannerObserver, [603](#)
- Digikam::ColorCorrectionDlg, [604](#)
- Digikam::ColorFXContainer, [604](#)
- Digikam::ColorFXFilter, [605](#)
  - filterAction, [609](#)
  - filterIdentifier, [609](#)
  - readParameters, [609](#)
- Digikam::ColorFXSettings, [609](#)
- Digikam::ColorGradientWidget, [610](#)
- Digikam::ColorLabelFilter, [611](#)
- Digikam::ColorLabelMenuAction, [613](#)
- Digikam::ColorLabelSelector, [614](#)
- Digikam::ColorLabelWidget, [615](#)
  - setButtonsExclusive, [616](#)
  - setColorLabels, [616](#)
- Digikam::ComboBoxDelegate, [617](#)
  - startEditing, [618](#)
- Digikam::CommentInfo, [618](#)
- Digikam::CommonKeys, [619](#)
  - getDbValue, [620](#)
- Digikam::CompressionDetector, [621](#)
  - detect, [622](#)
- Digikam::ContentAwareContainer, [622](#)
- Digikam::ContentAwareFilter, [623](#)
  - filterAction, [627](#)
  - filterIdentifier, [627](#)
  - readParameters, [627](#)
- Digikam::ContextMenuHelper, [627](#)
  - addAction, [630](#), [631](#)
  - addActionNewAlbum, [631](#)
  - addActionNewTag, [631](#)
  - addAlbumCheckUncheckActions, [631](#)
  - addAssignTagsMenu, [632](#)
  - addGotoMenu, [632](#)
  - addGroupMenu, [633](#)
  - addIQSAction, [633](#)
  - addLabelsAction, [633](#)
  - addOpenAndNavigateActions, [633](#)
  - addRemoveAllTags, [634](#)
  - addRemoveTagsMenu, [634](#)
  - addServicesMenu, [634](#)
  - addStandardActionCopy, [635](#)
  - addStandardActionCut, [635](#)
  - addStandardActionItemDelete, [635](#)
  - addStandardActionLightTable, [635](#)
  - addStandardActionPaste, [635](#)
  - addStandardActionThumbnail, [636](#)
  - addSubMenu, [636](#)
  - ContextMenuHelper, [630](#)
  - exec, [636](#)
  - setAlbumModel, [636](#)
  - setItemFilterModel, [637](#)
- Digikam::CoordinatesOverlayWidget, [637](#)
- Digikam::CopyOrMoveJob, [638](#)
- Digikam::CopyrightInfo, [639](#)
- Digikam::CoreDB, [640](#)
  - addAlbum, [649](#)
  - addAlbumRoot, [649](#)
  - addImageMetadata, [649](#)
  - addImageTagProperty, [650](#)
  - addItem, [650](#)
  - addItemInformation, [650](#)
  - addItemPosition, [651](#)
  - addItemTag, [651](#)
  - addSearch, [652](#)
  - addTag, [652](#)
  - addTagProperty, [652](#)
  - addToDownloadHistory, [653](#)



- addVideoMetadata, [653](#)
- changeImageComment, [653](#)
- changeImageMetadata, [653](#)
- changeItemInformation, [654](#)
- changeItemPosition, [654](#)
- changeVideoMetadata, [654](#)
- copyAlbumProperties, [654](#)
- copyItem, [654](#)
- databaseUuid, [655](#)
- deleteAlbum, [655](#)
- deleteAlbumRoot, [655](#)
- deleteItem, [655](#), [656](#)
- deleteObsoleteItem, [656](#)
- deleteRemovedItems, [656](#)
- deleteSearch, [656](#)
- deleteTag, [657](#)
- findImageId, [657](#)
- findInDownloadHistory, [657](#)
- getAlbumAndSubalbumsForPath, [657](#)
- getAlbumAverageDate, [658](#)
- getAlbumForPath, [658](#)
- getAlbumHighestDate, [658](#)
- getAlbumLowestDate, [659](#)
- getAlbumModificationDate, [659](#)
- getAlbumModificationMap, [659](#)
- getAlbumRelativePath, [659](#)
- getAlbumRootId, [660](#)
- getAlbumRoots, [660](#)
- getAlbumsOnAlbumRoot, [660](#)
- getAllItemsWithAlbum, [660](#)
- getDatabaseEncoding, [660](#)
- getDirtyOrMissingFacelImageUrls, [661](#)
- getFilterSettings, [661](#)
- getIdentialFiles, [661](#)
- getImageId, [661](#)
- getImageIds, [661](#), [662](#)
- getImageMetadata, [663](#)
- getImageFields, [663](#)
- getImageRelatedFrom, [663](#)
- getImageRelatingTo, [663](#)
- getImageTagProperties, [663](#)
- getItemAlbum, [663](#)
- getItemCommonTagIDs, [664](#)
- getItemCopyright, [664](#)
- getItemFromAlbum, [664](#)
- getItemIDsAndURLsInAlbum, [664](#)
- getItemIDsInAlbum, [665](#)
- getItemIDsInTag, [665](#)
- getItemInformation, [665](#)
- getItemName, [666](#)
- getItemNamesInAlbum, [666](#)
- getItemPosition, [666](#)
- getItemTagIDs, [666](#)
- getItemTagIDs, [666](#)
- getItemTagNames, [667](#)
- getItemURLsInAlbum, [667](#)
- getItemURLsInTag, [667](#)
- getNumberOfAllItemsAndAlbums, [668](#)
- getNumberOfItemsInAlbum, [668](#)
- getOneRelatedImageEach, [668](#)
- getRecentlyAssignedTags, [668](#)
- getRelationCloud, [669](#)
- getSetting, [669](#)
- getTagsWithProperty, [669](#)
- getUniqueHashVersion, [669](#)
- getUserFilterSettings, [669](#)
- getVideoMetadata, [669](#)
- hasTags, [670](#)
- makeStaleAlbum, [670](#)
- migrateAlbumRoot, [670](#)
- moveItem, [670](#)
- removeImageRelation, [671](#)
- removeImageTagProperties, [671](#)
- removeItemAllTags, [671](#)
- removeItemCopyrightProperties, [671](#)
- removeItems, [672](#)
- removeItemsFromAlbum, [672](#)
- removeItemsPermanently, [672](#)
- removeItemTag, [673](#)
- removeTagProperties, [673](#)
- renameItem, [673](#)
- scanAlbums, [673](#)
- scanSearches, [673](#)
- scanTags, [673](#)
- setAlbumCaption, [674](#)
- setAlbumCategory, [674](#)
- setAlbumDate, [674](#)
- setAlbumIcon, [674](#)
- setAlbumModificationDate, [675](#)
- setAlbumRootLabel, [675](#)
- setAlbumRootPath, [675](#)
- setFilterSettings, [675](#)
- setImageComment, [675](#)
- setItemAlbum, [676](#)
- setItemStatus, [676](#)
- setSetting, [676](#)
- setTagIcon, [677](#)
- setTagName, [677](#)
- setTagParentID, [677](#)
- setUserFilterSettings, [677](#)
- updateItem, [678](#)
- updateSearch, [678](#)
- Digikam::CoreDbAccess, [678](#)
- checkReadyForUse, [680](#)
- cleanUpDatabase, [680](#)
- CoreDbAccess, [680](#)
- setLastError, [680](#)
- setParameters, [680](#)
- Digikam::CoreDbAccessUnlock, [681](#)
- CoreDbAccessUnlock, [681](#)
- Digikam::CoreDbBackend, [682](#)
- initSchema, [686](#)
- Digikam::CoreDbCopyManager, [686](#)
- Digikam::CoreDbDownloadHistory, [687](#)
- status, [687](#)
- Digikam::CoreDbNameFilter, [688](#)

- CoreDbNameFilter, [688](#)
- Digikam::CoreDbOperationGroup, [688](#)
  - allowLift, [689](#)
  - lift, [689](#)
- Digikam::CoreDbPrivilegesChecker, [689](#)
- Digikam::CoreDbSchemaUpdater, [689](#)
- Digikam::CoreDbTransaction, [689](#)
- Digikam::CoreDbUrl, [691](#)
  - album, [693](#)
  - albumRoot, [693](#)
  - areaCoordinates, [693](#)
  - fromAlbumAndName, [693](#)
  - fromDateForMonth, [694](#)
  - fromDateForYear, [694](#)
  - fromDateRange, [694](#)
  - fromFileUrl, [694](#)
  - fromTagIds, [694](#)
  - isAlbumUrl, [695](#)
  - name, [695](#)
  - parameters, [695](#)
  - searchId, [695](#)
  - startDate, [695](#)
  - tagId, [695](#)
- Digikam::CoreDbWatch, [696](#)
  - databaseChanged, [698](#)
  - imageChange, [698](#)
- Digikam::CountrySelector, [698](#)
- Digikam::CurvesBox, [699](#)
- Digikam::CurvesContainer, [700](#)
  - CurvesContainer, [701](#)
  - curvesType, [701](#)
  - isEmpty, [701](#)
  - isStoredLosslessly, [701](#)
- Digikam::CurvesFilter, [702](#)
  - filterAction, [706](#)
  - filterIdentifier, [706](#)
  - readParameters, [706](#)
- Digikam::CurvesSettings, [707](#)
- Digikam::CurvesWidget, [709](#)
  - restoreCurve, [710](#)
  - saveCurve, [711](#)
  - updateData, [711](#)
- Digikam::CustomStepsDoubleSpinBox, [712](#)
  - setSuggestedValues, [713](#)
- Digikam::CustomStepsIntSpinBox, [713](#)
  - setSuggestedValues, [714](#)
- Digikam::DAboutData, [715](#)
- Digikam::DAbstractSliderSpinBox, [716](#)
  - setBlockUpdateSignalOnDrag, [717](#)
  - setInternalValue, [717](#)
- Digikam::DActiveLabel, [718](#)
- Digikam::DAdjustableLabel, [719](#)
- Digikam::DAlbum, [720](#)
  - databaseUrl, [722](#)
- Digikam::DAlbumDrag, [722](#)
- Digikam::DAlbumInfo, [723](#)
- Digikam::DArrowClickLabel, [724](#)
- Digikam::DatabaseCopyThread, [725](#)
- Digikam::DatabaseFields::DatabaseFieldsEnumIterator<
  - FieldName >, [725](#)
- Digikam::DatabaseFields::DatabaseFieldsEnumIteratorSetOnly<
  - FieldName >, [726](#)
- Digikam::DatabaseFields::FieldMetaInfo<
  - FieldName >, [726](#)
- Digikam::DatabaseFields::Hash< T >, [726](#)
- Digikam::DatabaseFields::Set, [728](#)
- Digikam::DatabaseLoadSaveFileInfoProvider, [729](#)
  - dimensionsHint, [729](#)
  - orientationHint, [729](#)
- Digikam::DatabaseMigrationDialog, [730](#)
- Digikam::DatabaseOption, [731](#)
  - parseOperation, [733](#)
- Digikam::DatabaseOptionDialog, [734](#)
- Digikam::DatabasePage, [735](#)
- Digikam::DatabaseServer, [736](#)
- Digikam::DatabaseServerError, [737](#)
  - DatabaseServerErrorEnum, [737](#)
  - NoErrors, [737](#)
  - NotSupported, [737](#)
  - StartError, [737](#)
- Digikam::DatabaseServerStarter, [738](#)
  - instance, [738](#)
- Digikam::DatabaseSettingsWidget, [739](#)
  - checkDatabaseSettings, [740](#)
- Digikam::DatabaseTask, [740](#)
- Digikam::DatabaseWorkerInterface, [742](#)
- Digikam::DatabaseWriter, [745](#)
- Digikam::DateAlbumModel, [747](#)
  - albumForId, [753](#)
  - albumName, [753](#)
  - DateAlbumModel, [752](#)
  - decorationRoleData, [753](#)
  - monthIndexForDate, [753](#)
  - sortRoleData, [753](#)
- Digikam::DateFolderView, [754](#)
  - doLoadState, [756](#)
  - doSaveState, [756](#)
  - setConfigGroup, [756](#)
- Digikam::DateFolderViewSideBarWidget, [757](#)
  - applySettings, [759](#)
  - changeAlbumFromHistory, [759](#)
  - doLoadState, [759](#)
  - doSaveState, [759](#)
  - getCaption, [759](#)
  - getIcon, [760](#)
  - setActive, [760](#)
- Digikam::DateFormat, [760](#)
- Digikam::DateOption, [761](#)
  - parseOperation, [763](#)
- Digikam::DateOptionDialog, [764](#)
- Digikam::DatesDBJobInfo, [765](#)
- Digikam::DatesDBJobsThread, [766](#)
  - datesListing, [768](#)
- Digikam::DatesJob, [769](#)
- Digikam::DateTreeView, [771](#)
- Digikam::DbCleaner, [776](#)

- setUseMultiCoreCPU, 778
- Digikam::DbEngineAccess, 779
  - checkReadyForUse, 779
- Digikam::DbEngineAction, 779
- Digikam::DbEngineActionElement, 779
- Digikam::DbEngineActionType, 779
  - isValue, 780
  - setValue, 780
- Digikam::DbEngineConfig, 780
- Digikam::DbEngineConfigSettings, 780
- Digikam::DbEngineConfigSettingsLoader, 781
- Digikam::DbEngineConnectionChecker, 781
- Digikam::DbEngineErrorAnswer, 782
- Digikam::DbEngineErrorHandler, 783
  - connectionError, 783
  - consultUserForError, 784
- Digikam::DbEngineGuiErrorHandler, 784
- Digikam::DbEngineLocking, 785
- Digikam::DbEngineParameters, 785
  - defaultParameters, 787
  - getCoreDatabaseNameOrDir, 787
  - readFromConfig, 788
  - SQLiteDatabaseType, 788
- Digikam::DbEngineSqlQuery, 788
- Digikam::DbHeaderListItem, 789
- Digikam::DBinaryIface, 790
- Digikam::DBinarySearch, 792
- Digikam::DBInIface, 793
  - albumChooser, 795
  - albumChooserItems, 795
  - albumInfo, 795
  - albumItems, 795
  - albumItems, 795
  - allAlbumItems, 796
  - currentAlbumItems, 796
  - currentGPSItems, 796
  - currentSelectedItem, 796
  - defaultUploadUrl, 796
  - deleteImage, 796
  - itemInfo, 796
  - openSetupPage, 797
  - parseAlbumItemsRecursive, 797
  - passShortcutActionsToWidget, 797
  - setItemInfo, 797
  - supportAlbums, 797
  - tagFilterModel, 797
  - uploadUrl, 797
  - uploadWidget, 798
- Digikam::DBJob, 798
- Digikam::DBJobInfo, 800
- Digikam::DBJobsManager, 801
  - instance, 802
  - startAlbumsJobThread, 802
  - startDatesJobThread, 802
  - startGPSJobThread, 802
  - startSearchesJobThread, 803
  - startTagsJobThread, 803
- Digikam::DBJobsThread, 804
  - connectFinishAndErrorSignals, 805
  - error, 805
  - errorsList, 806
  - hasErrors, 806
- Digikam::DbKeysCollection, 806
  - addId, 807
  - collectionName, 807
  - DbKeysCollection, 807
  - getDbValue, 808
  - getValue, 808
  - ids, 808
- Digikam::DbKeySelector, 809
- Digikam::DbKeySelectorItem, 810
- Digikam::DbKeySelectorView, 811
- Digikam::DbShrinkDialog, 812
- Digikam::DBStatDlg, 813
- Digikam::DBusSignalListenerThread, 814
- Digikam::DBusyDlg, 815
- Digikam::DBusyThread, 816
- Digikam::DCameraDragObject, 817
- Digikam::DCameraItemListDrag, 818
- Digikam::DCategorizedSortFilterProxyModel, 819
  - AdditionalRoles, 820
  - CategoryDisplayRole, 820
  - CategorySortRole, 820
  - compareCategories, 821
  - isCategorizedModel, 821
  - lessThan, 821
  - setCategorizedModel, 822
  - setSortCategoriesByNaturalComparison, 822
  - sort, 822
  - sortCategoriesByNaturalComparison, 822
  - sortColumn, 823
  - sortOrder, 823
  - subSortLessThan, 823
- Digikam::DCategorizedView, 824
  - categorizedIndexesIn, 825
  - categoryAt, 826
  - categoryRange, 826
  - categoryVisualRect, 826
  - setDrawDraggedItems, 826
- Digikam::DCategoryDrawer, 827
  - actionRequested, 828
  - categoryHeight, 828
  - drawCategory, 829
  - leftMargin, 829
  - mouseButtonDoubleClicked, 829
  - mouseButtonPressed, 830
  - mouseButtonReleased, 830
  - mouseLeft, 830
  - mouseMoved, 831
  - rightMargin, 831
  - view, 831
- Digikam::DClickLabel, 832
- Digikam::DColor, 833
  - blendZero, 834
  - DColor, 834
  - getHSL, 834

- getYCbCr, [834](#)
- premultiply, [834](#)
- setColor, [834](#)
- setHSL, [835](#)
- setPixel, [835](#)
- setYCbCr, [835](#)
- Digikam::DColorComposer, [835](#)
  - compose, [836](#)
  - CompositingOperation, [836](#)
  - getComposer, [837](#)
- Digikam::DColorSelector, [837](#)
- Digikam::DColorValueSelector, [839](#)
  - chooserMode, [841](#)
  - colorValue, [841](#)
  - drawContents, [841](#)
  - hue, [841](#)
  - saturation, [842](#)
  - setChooserMode, [842](#)
  - setColorValue, [842](#)
  - setHue, [842](#)
  - setSaturation, [843](#)
- Digikam::DComboBox, [843](#)
- Digikam::DConfigDlg, [844](#)
  - addPage, [848](#)
  - addSubPage, [849](#)
  - currentPage, [850](#)
  - currentPageChanged, [850](#)
  - DConfigDlg, [848](#)
  - FaceType, [847](#)
  - insertPage, [850](#), [851](#)
  - pageRemoved, [851](#)
  - removePage, [851](#)
  - setButtonBox, [852](#)
  - setCurrentPage, [852](#)
  - setPageWidget, [852](#)
- Digikam::DConfigDlgMngr, [852](#)
  - addWidget, [855](#)
  - DConfigDlgMngr, [855](#)
  - getCustomProperty, [855](#)
  - getCustomPropertyChangedSignal, [856](#)
  - init, [856](#)
  - parseChildren, [856](#)
  - settingsChanged, [856](#)
  - updateSettings, [857](#)
  - updateWidgets, [857](#)
  - updateWidgetsDefault, [857](#)
  - widgetModified, [857](#)
- Digikam::DConfigDlgModel, [858](#)
  - HeaderRole, [859](#)
  - Role, [859](#)
  - WidgetRole, [859](#)
- Digikam::DConfigDlgTitle, [859](#)
  - autoHideTimeout, [862](#)
  - comment, [862](#)
  - DConfigDlgTitle, [862](#)
  - ErrorMessage, [862](#)
  - ImageAlignment, [861](#)
  - ImageLeft, [862](#)
  - ImageRight, [862](#)
  - InfoMessage, [862](#)
  - MessageType, [862](#)
  - pixmap, [862](#)
  - PlainTextMessage, [862](#)
  - setAutoHideTimeout, [863](#)
  - setBuddy, [863](#)
  - setComment, [863](#)
  - setPixmap, [864](#), [865](#)
  - setText, [865](#)
  - setWidget, [866](#)
  - text, [866](#)
  - WarningMessage, [862](#)
- Digikam::DConfigDlgView, [866](#)
  - createView, [869](#)
  - currentPageChanged, [869](#)
  - FaceType, [869](#)
  - setCurrentPage, [869](#)
  - setItemDelegate, [870](#)
  - setModel, [870](#)
  - showPageHeader, [870](#)
  - viewPosition, [870](#)
- Digikam::DConfigDlgWdg, [871](#)
  - addPage, [874](#)
  - addSubPage, [874](#), [875](#)
  - currentPage, [875](#)
  - currentPageChanged, [876](#)
  - DConfigDlgWdg, [873](#)
  - insertPage, [876](#)
  - pageRemoved, [877](#)
  - pageToggled, [877](#)
  - removePage, [877](#)
  - setCurrentPage, [877](#)
- Digikam::DConfigDlgWdgItem, [878](#)
  - DConfigDlgWdgItem, [880](#)
  - enabled, [881](#)
  - setCheckable, [880](#)
  - setHeader, [881](#)
  - setIcon, [881](#)
  - toggled, [881](#)
- Digikam::DConfigDlgWdgModel, [881](#)
  - addPage, [884](#)
  - addSubPage, [885](#)
  - DConfigDlgWdgModel, [884](#)
  - index, [886](#)
  - insertPage, [886](#)
  - item, [887](#)
  - removePage, [887](#)
  - toggled, [887](#)
- Digikam::DCursorTracker, [888](#)
- Digikam::DDateEdit, [889](#)
  - assignDate, [890](#)
  - date, [891](#)
  - dateChanged, [891](#)
  - isReadOnly, [891](#)
  - setDate, [891](#)
  - setReadOnly, [891](#)
  - setupKeywords, [892](#)

- Digikam::DDatePicker, 892
  - date, 895
  - dateChanged, 895
  - dateEntered, 895
  - dateSelected, 896
  - dateTable, 896
  - DDatePicker, 895
  - hasCloseButton, 896
  - setCloseButton, 896
  - setDate, 896
  - sizeHint, 897
- Digikam::DDatePickerPopup, 897
  - datePicker, 899
  - DDatePickerPopup, 899
  - items, 900
- Digikam::DDateTable, 900
  - aboutToShowContextMenu, 903
  - date, 903
  - dateChanged, 903
  - dateFromPos, 903
  - posFromDate, 903
  - setPopupMenuEnabled, 904
  - sizeHint, 904
- Digikam::DDateTimeEdit, 904
  - dateTime, 906
  - dateTimeChanged, 906
  - DDateTimeEdit, 906
- Digikam::DDoubleNumInput, 907
- Digikam::DDoubleSliderSpinBox, 909
  - setInternalValue, 911
  - valueString, 911
- Digikam::DefaultRenameParser, 912
- Digikam::DefaultValueDialog, 913
- Digikam::DefaultValueModifier, 914
  - parseOperation, 916
- Digikam::DefaultVersionNamingScheme, 917
  - baseName, 918
  - directory, 918
  - incrementedCounter, 918
  - initialCounter, 918
  - intermediateDirectory, 919
  - intermediateFileName, 919
  - versionFileName, 919
- Digikam::DeleteDialog, 920
- Digikam::DeleteItem, 921
- Digikam::DeleteItemList, 922
- Digikam::DeleteJob, 923
- Digikam::DeleteWidget, 925
- Digikam::DeltaTime, 925
- Digikam::DetByClockPhotoButton, 926
- Digikam::DetectionBenchmark, 927
  - result, 929
- Digikam::DetectionWorker, 930
- Digikam::DExpanderBox, 932
  - addItem, 933
  - insertItem, 933
- Digikam::DExpanderBoxExclusive, 935
- Digikam::DFileDialog, 937
- Digikam::DFileOperations, 938
  - findExecutable, 939
  - localFileRename, 939
  - removeAndCopyFile, 939
  - setModificationTime, 939
- Digikam::DFileSelector, 939
- Digikam::DFontProperties, 942
  - backgroundColor, 945
  - color, 945
  - DFontProperties, 944
  - DisplayFlag, 944
  - enableColumn, 945
  - font, 946
  - FontColumn, 944
  - FontDiff, 944
  - fontDiffFlags, 946
  - FontListCriteria, 944
  - getFontList, 946
  - makeColumnVisible, 946
  - sampleText, 947
  - setFont, 947
  - setSampleBoxVisible, 947
  - setSampleText, 947
  - setSizesRelative, 948
  - sizeRelative, 948
- Digikam::DFontSelect, 949
- Digikam::DGradientSlider, 951
- Digikam::DHBox, 952
- Digikam::DHistoryView, 953
- Digikam::DHueSaturationSelector, 954
  - chooserMode, 956
  - colorValue, 956
  - drawContents, 956
  - hue, 957
  - saturation, 957
  - setChooserMode, 957
  - setColorValue, 957
  - setHue, 957
  - setSaturation, 958
- Digikam::DigikamApp, 959
  - infoface, 962
- Digikam::DigikamItemDelegate, 963
  - updateRects, 967
- Digikam::DigikamItemView, 968
  - activated, 975
  - confirmFaces, 975
  - hasHiddenGroupedImages, 975
  - rejectFaces, 975
  - removeFaces, 976
  - setThumbnailSize, 976
  - showContextMenu, 976
  - showContextMenuOnInfo, 976
  - slotSetupChanged, 976
- Digikam::DImageHistory, 976
  - clearReferredImages, 978
  - entries, 978
  - hasActions, 978
  - operator<<, 978

- purgePathFromReferredImages, 978
  - toXml, 978
- Digikam::DImageHistory::Entry, 979
  - action, 979
- Digikam::DImg, 979
  - addAsReferredImage, 987
  - addCurrentUniqueImageld, 987
  - bitBlendImage, 987
  - bitBlendImageOnColor, 988
  - bitBlitImage, 988
  - convertDepth, 988
  - copyMetaData, 988
  - createImageUniqueld, 989
  - CreateNewImageHistoryUUID, 986
  - CreateNewMetadataPreview, 986
  - detach, 989
  - detectedFormat, 989
  - DImg, 986, 987
  - fileOriginData, 989
  - fill, 990
  - FORMAT, 984
  - format, 990
  - getPixelColor, 990
  - getUniqueHash, 990
  - getUniqueHashVersion, 990
  - hasTransparentPixels, 991
  - imageSavedAs, 991
  - isReadOnly, 991
  - lastSavedFilePath, 991
  - loadItemInfo, 991
  - operator==, 992
  - originalBitDepth, 992
  - originalColorModel, 992
  - originalFilePath, 992
  - PrepareMetadataFlag, 984
  - prepareMetadataToSave, 992
  - pureColorMask, 992
  - putImageData, 993
  - QIMAGE, 984
  - rawDecodingSettings, 993
  - removeAlphaChannel, 993
  - RemoveOldMetadataPreviews, 986
  - ResetExifOrientationTag, 986
  - rotateAndFlip, 993
  - savedFormat, 993
  - setHistoryBranchAfter, 994
  - smoothScale, 994
  - smoothScaleClipped, 994
  - striplImageData, 994
  - transform, 994
  - wasExifRotated, 995
- Digikam::DImgBuiltinFilter, 995
  - Crop, 996
  - DImgBuiltinFilter, 996
  - filterAction, 997
  - Resize, 996
  - reverseFilter, 997
  - Type, 996
- Digikam::DImgChildItem, 998
  - boundingRect, 1000
  - DImgChildItem, 1000
  - originalRect, 1000
  - positionChanged, 1000
  - positionOnImageChanged, 1000
  - rect, 1000
  - relativeRect, 1001
  - setOriginalPos, 1001
  - setPos, 1001
  - setRelativePos, 1001
- Digikam::DImgFilterGenerator, 1002
  - createFilter, 1003
  - displayableName, 1003
  - isSupported, 1003
  - supportedFilters, 1003
  - supportedVersions, 1003
- Digikam::DImgFilterManager, 1004
  - createFilter, 1005
  - displayableName, 1005
  - filterIcon, 1005
  - isSupported, 1006
  - supportedFilters, 1006
  - supportedVersions, 1006
- Digikam::DImgLoader, 1006
  - LoadAll, 1008
  - LoadFlag, 1008
  - LoadICCData, 1008
  - LoadImageData, 1008
  - LoadImageHistory, 1008
  - LoadItemInfo, 1008
  - LoadMetadata, 1008
  - LoadPreview, 1008
  - LoadUniqueHash, 1008
- Digikam::DImgLoaderObserver, 1009
  - granularity, 1010
  - progressInfo, 1010
- Digikam::DImgLoaderSettings, 1010
- Digikam::DImgPreviewItem, 1012
  - userLoadingHint, 1014
- Digikam::DImgThreadedAnalyser, 1015
  - DImgThreadedAnalyser, 1018
  - startAnalyse, 1019
- Digikam::DImgThreadedFilter, 1019
  - cancelFilter, 1023
  - cleanupFilter, 1023
  - DImgThreadedFilter, 1022
  - filterAction, 1023
  - filterIdentifier, 1023
  - filterImage, 1024
  - finished, 1024
  - initFilter, 1024
  - initSlave, 1024
  - m\_master, 1026
  - m\_slave, 1026
  - modulateProgress, 1024
  - multithreadedSteps, 1025
  - parametersSuccessfullyRead, 1025



- run, 1025
- setFilterVersion, 1025
- setSlave, 1025
- setupFilter, 1025
- Digikam::DImgThreadedFilter::DefaultFilterAction< Filter >, 1026
- Digikam::DInfoInterface, 1030
  - albumChooser, 1032
  - currentSelectedItems, 1032
  - defaultUploadUrl, 1032
  - deleteImage, 1032
  - openSetupPage, 1032
  - passShortcutActionsToWidget, 1032
  - slotDateTimeForUrl, 1032
  - slotMetadataChangedForUrl, 1033
  - tagFilterModel, 1033
  - uploadWidget, 1033
- Digikam::DIntNumInput, 1034
- Digikam::DIntRangeBox, 1035
  - maxValue, 1036
  - minValue, 1036
  - setEnabled, 1036
  - setInterval, 1036
  - setRange, 1036
  - setSuffix, 1037
- Digikam::DIO, 1037
  - copy, 1038
- Digikam::DirectoryNameOption, 1039
  - parseOperation, 1041
- Digikam::DisjointMetadata, 1042
  - changedFlags, 1044
  - colorLabel, 1044
  - colorLabelInterval, 1044
  - comments, 1045
  - dateTime, 1045
  - dateTimeInterval, 1045
  - FullWrite, 1044
  - FullWriteIfChanged, 1044
  - keywords, 1045
  - metadataTemplate, 1045
  - PartialWrite, 1044
  - pickLabel, 1045
  - pickLabelInterval, 1046
  - rating, 1046
  - ratingInterval, 1046
  - replaceColorLabel, 1046
  - tags, 1046
  - titles, 1046
  - write, 1047
  - WriteMode, 1044
- Digikam::DisjointMetadataDataFields, 1047
  - MetadataAvailable, 1048
  - MetadataDisjoint, 1048
  - MetadataInvalid, 1048
  - Status, 1048
- Digikam::DistortionFXFilter, 1049
  - filterAction, 1053
  - filterIdentifier, 1053
  - readParameters, 1053
- Digikam::DItemDelegate, 1054
  - acceptsToolTip, 1055
  - gridSize, 1055
  - mouseMoved, 1055
  - setDefaultViewOptions, 1056
  - setThumbnailSize, 1056
- Digikam::DItemDrag, 1056
- Digikam::DItemInfo, 1057
- Digikam::DItemsList, 1059
  - appendControlButtonsWidget, 1061
  - setControlButtonsPlacement, 1061
  - setIsLessThanHandler, 1061
- Digikam::DItemsListView, 1062
- Digikam::DItemsListViewItem, 1063
  - updateItemWidgets, 1064
- Digikam::DItemToolTip, 1065
  - tipContents, 1065
- Digikam::DKCamera, 1066
  - capture, 1068
  - getFreeSpace, 1068
  - getItemsInfoList, 1068
  - getPreview, 1068
- Digikam::DLabelExpander, 1069
- Digikam::DLineWidget, 1070
- Digikam::DLogoAction, 1071
- Digikam::DMessageBox, 1071
  - readMsgBoxShouldBeShown, 1072
  - saveMsgBoxShouldBeShown, 1072
  - showContinueCancel, 1073
  - showContinueCancelList, 1073
  - showContinueCancelWidget, 1073
  - showYesNo, 1073
  - showYesNoList, 1073
  - showYesNoWidget, 1073
- Digikam::DMetadata, 1075
  - addToXmpTagStringBag, 1090
  - getCopyrightInformation, 1090
  - getIccProfile, 1090
  - getItemFacesMap, 1091
  - getLensDescription, 1091
  - getMetadataField, 1091
  - getXmpKeywords, 1092
  - getXmpSubCategories, 1092
  - getXmpSubjects, 1092
  - load, 1092
  - mSecTimeStamp, 1092
  - possibleValuesForEnumField, 1092
  - removeFromXmpTagStringBag, 1093
  - removeXmpKeywords, 1093
  - removeXmpSubCategories, 1093
  - removeXmpSubjects, 1093
  - setItemFacesMap, 1093
  - setXmpKeywords, 1094
  - setXmpSubCategories, 1094
  - setXmpSubjects, 1094
  - valueToString, 1094
  - VIDEOCOLORMODEL, 1090

- Digikam::DMetadataSettings, 1095
  - instance, 1096
- Digikam::DMetadataSettingsContainer, 1096
- Digikam::DMetaInfolface, 1097
  - allAlbumItems, 1099
  - currentActiveItem, 1099
  - currentAlbumItems, 1099
  - currentGPSItems, 1099
  - currentSelectedItems, 1099
  - defaultUploadUrl, 1099
  - deleteImage, 1100
  - itemInfo, 1100
  - parseAlbumItemsRecursive, 1100
  - setItemInfo, 1100
  - slotDateTimeForUrl, 1100
  - slotMetadataChangedForUrl, 1100
  - supportAlbums, 1100
  - uploadUrl, 1101
  - uploadWidget, 1101
- Digikam::DModelFactory, 1101
- Digikam::DMultiTabBar, 1102
  - ActiveIconText, 1104
  - AllIconsText, 1104
  - appendButton, 1105
  - appendTab, 1105
  - position, 1105
  - setPosition, 1105
  - setTab, 1106
  - tabStyle, 1106
  - TextStyle, 1104
- Digikam::DMultiTabBarButton, 1107
  - signalClicked, 1108
- Digikam::DMultiTabBarFrame, 1109
- Digikam::DMultiTabBarTab, 1110
  - setPosition, 1112
  - setState, 1112
  - setStyle, 1112
- Digikam::DNGConvertSettings, 1113
- Digikam::DNGSettings, 1114
- Digikam::DNGWriter, 1115
  - ConvertError, 1115
  - DNG\_SDK\_INTERNAL\_ERROR, 1115
  - FILE\_NOT\_SUPPORTED, 1115
  - FULL\_SIZE, 1116
  - JPEGPreview, 1115
  - MEDIUM, 1116
  - NONE, 1116
  - PROCESS\_CANCELED, 1115
  - PROCESS\_COMPLETE, 1115
  - PROCESS\_CONTINUE, 1115
  - PROCESS\_FAILED, 1115
- Digikam::DNGWriterHost, 1116
- Digikam::DNNBaseDetectorModel, 1117
  - uiConfidenceThreshold, 1118
- Digikam::DNNFaceDetectorBase, 1119
  - selectBbox, 1120
- Digikam::DNNFaceDetectorSSD, 1121
  - detectFaces, 1122
- Digikam::DNNFaceDetectorYOLO, 1123
  - detectFaces, 1124
- Digikam::DNNFaceDetectorYuNet, 1125
  - detectFaces, 1126
  - setFaceDetectionSize, 1126
- Digikam::DNNFaceExtractorBase, 1127
  - getThreshold, 1128
  - loadModels, 1128
- Digikam::DNNModelBase, 1129
  - getThreshold, 1130
- Digikam::DNNModelConfig, 1130
- Digikam::DNNModelInfoContainer, 1131
- Digikam::DNNModelManager, 1133
  - getModel, 1133
  - instance, 1133
- Digikam::DNNModelNet, 1134
- Digikam::DNNModelSFace, 1135
- Digikam::DNNModelYuNet, 1137
- Digikam::DNNOpenFaceExtractor, 1139
  - alignFace, 1140
  - getFaceEmbedding, 1140
  - getThreshold, 1140
  - loadModels, 1141
- Digikam::DNNResnetDetector, 1142
  - loadModels, 1143
- Digikam::DNNSFaceExtractor, 1144
  - alignFace, 1145
  - getFaceEmbedding, 1145
  - getThreshold, 1145
  - loadModels, 1146
- Digikam::DNNYoloDetector, 1147
  - loadModels, 1149
- Digikam::DNotificationPopup, 1149
  - autoDelete, 1153
  - Balloon, 1153
  - Boxed, 1153
  - defaultLocation, 1153
  - message, 1154–1157
  - moveNear, 1157
  - PopupStyle, 1153
  - positionSelf, 1158
  - setAnchor, 1158
  - setAutoDelete, 1158
  - setPopupStyle, 1158
  - setTimeout, 1158
  - standardView, 1159
- Digikam::DNotificationWidget, 1159
  - addAction, 1162
  - animatedShowTemporized, 1163
  - clearAllActions, 1163
  - heightForWidth, 1163
  - hideAnimationFinished, 1163
  - icon, 1163
  - isCloseButtonVisible, 1164
  - isHideAnimationRunning, 1164
  - isShowAnimationRunning, 1164
  - linkActivated, 1164
  - linkHovered, 1165



- MessageType, 1162
- messageType, 1165
- removeAction, 1165
- setCloseButtonVisible, 1165
- setMessageType, 1166
- setText, 1166
- setWordWrap, 1166
- showAnimationFinished, 1166
- text, 1167
- wordWrap, 1167
- Digikam::DOnlineTranslator, 1167
  - detectLanguage, 1172
  - DOnlineTranslator, 1171
  - error, 1172
  - errorString, 1172
  - isRunning, 1172
  - isSourceTranscriptionEnabled, 1172
  - isSourceTranslitEnabled, 1173
  - isSupportTranslation, 1173
  - isTranslationOptionsEnabled, 1173
  - isTranslationTranslitEnabled, 1173
  - language, 1174
  - languageCode, 1174
  - languageName, 1175
  - NetworkError, 1171
  - NoError, 1171
  - ParametersError, 1171
  - ParsingError, 1171
  - ServiceError, 1171
  - setEngineApiKey, 1175
  - setEngineUrl, 1175
  - setSourceTranscriptionEnabled, 1175
  - setSourceTranslitEnabled, 1176
  - setTranslationOptionsEnabled, 1176
  - setTranslationTranslitEnabled, 1176
  - signalFinished, 1176
  - source, 1176
  - sourceLanguage, 1177
  - sourceLanguageName, 1177
  - sourceTranscription, 1177
  - sourceTranslit, 1177
  - toJson, 1177
  - translate, 1177
  - translation, 1178
  - TranslationError, 1171
  - translationLanguage, 1178
  - translationLanguageName, 1178
  - translationOptions, 1178
  - translationTranslit, 1178
- Digikam::DOnlineTranslatorOption, 1179
  - toJson, 1180
- Digikam::DOnlineTts, 1180
  - DOnlineTts, 1182
  - Emotion, 1181
  - emotion, 1182
  - emotionCode, 1182
  - error, 1183
  - errorString, 1183
  - generateUrls, 1183
  - media, 1184
  - NoError, 1182
  - TtsError, 1181
  - UnsupportedEmotion, 1182
  - UnsupportedEngine, 1182
  - UnsupportedLanguage, 1182
  - UnsupportedVoice, 1182
  - Voice, 1182
  - voice, 1184
  - voiceCode, 1184
- Digikam::DownloadInfo, 1185
- Digikam::DownloadSettings, 1185
- Digikam::DPixelsAliasFilter, 1186
  - pixelAntiAliasing, 1186
  - pixelAntiAliasing16, 1186
- Digikam::DPlainTextEdit, 1187
  - acceptedCharacters, 1189
  - DPlainTextEdit, 1188
  - ignoredCharacters, 1189
  - isClearButtonEnabled, 1189
  - returnPressed, 1189
  - setCurrentLanguage, 1189
  - setLinesVisible, 1189
  - setMaxLength, 1189
  - spellCheckSettings, 1190
  - text, 1190
- Digikam::DPlugin, 1190
  - categories, 1192
  - cleanUp, 1192
  - count, 1192
  - extraAboutData, 1192
  - extraAboutDataRowTitles, 1192
  - extraAboutDataTitle, 1192
  - handbookChapter, 1192
  - handbookReference, 1193
  - handbookSection, 1193
  - hasVisibilityProperty, 1193
  - icon, 1193
  - ifaceId, 1193
  - iid, 1193
  - libraryFileName, 1194
  - name, 1194
  - setLibraryFileName, 1194
  - setShouldLoaded, 1194
  - setVisible, 1194
  - shouldLoaded, 1194
  - version, 1194
- Digikam::DPluginAboutDlg, 1195
- Digikam::DPluginAction, 1196
  - ActionCategory, 1197
  - ActionType, 1197
  - Editor, 1197
  - EditorColors, 1197
  - EditorDecorate, 1197
  - EditorEnhance, 1197
  - EditorFile, 1197
  - EditorFilters, 1197

- EditorTransform, [1197](#)
- Generic, [1197](#)
- GenericExport, [1197](#)
- GenericImport, [1197](#)
- GenericMetadata, [1197](#)
- GenericTool, [1197](#)
- GenericView, [1197](#)
- InvalidType, [1197](#)
- toString, [1198](#)
- Digikam::DPluginAuthor, [1198](#)
- toString, [1198](#)
- Digikam::DPluginBqm, [1199](#)
- categories, [1201](#)
- count, [1201](#)
- hasVisibilityProperty, [1201](#)
- ifacelid, [1201](#)
- setVisible, [1202](#)
- Digikam::DPluginConfView, [1202](#)
- setFilter, [1203](#)
- signalSearchResult, [1203](#)
- Digikam::DPluginConfViewBqm, [1204](#)
- loadPlugins, [1205](#)
- Digikam::DPluginConfViewDImg, [1206](#)
- loadPlugins, [1207](#)
- Digikam::DPluginConfViewEditor, [1208](#)
- loadPlugins, [1209](#)
- Digikam::DPluginConfViewGeneric, [1210](#)
- loadPlugins, [1211](#)
- Digikam::DPluginDialog, [1212](#)
- Digikam::DPluginDImg, [1213](#)
- canRead, [1215](#)
- canWrite, [1215](#)
- categories, [1215](#)
- count, [1215](#)
- exportWidget, [1216](#)
- extraAboutData, [1216](#)
- extraAboutDataRowTitles, [1216](#)
- extraAboutDataTitle, [1216](#)
- hasVisibilityProperty, [1216](#)
- ifacelid, [1216](#)
- loaderName, [1216](#)
- setVisible, [1217](#)
- typeMimes, [1217](#)
- Digikam::DPluginEditor, [1218](#)
- categories, [1220](#)
- count, [1220](#)
- ifacelid, [1220](#)
- setVisible, [1220](#)
- Digikam::DPluginGeneric, [1221](#)
- categories, [1223](#)
- count, [1223](#)
- ifacelid, [1223](#)
- setVisible, [1223](#)
- Digikam::DPluginLoader, [1224](#)
- appendPluginToBlackList, [1225](#)
- appendPluginToWhiteList, [1226](#)
- cleanUp, [1226](#)
- exportWidget, [1226](#)
- init, [1226](#)
- instance, [1226](#)
- pluginAction, [1226](#)
- pluginActions, [1227](#)
- pluginsActions, [1227](#)
- Digikam::DPluginRawImport, [1228](#)
- categories, [1230](#)
- count, [1230](#)
- ifacelid, [1230](#)
- setVisible, [1230](#)
- Digikam::DPluginSetup, [1231](#)
- Digikam::DPointSelect, [1232](#)
- contentsRect, [1233](#)
- drawContents, [1233](#)
- setMarkerColor, [1234](#)
- setValues, [1234](#)
- setXValue, [1234](#)
- setYValue, [1234](#)
- valueChanged, [1235](#)
- xValue, [1235](#)
- yValue, [1235](#)
- Digikam::DPopupFrame, [1236](#)
- close, [1237](#)
- DPopupFrame, [1237](#)
- resizeEvent, [1237](#)
- setMainWidget, [1238](#)
- Digikam::DPreviewImage, [1239](#)
- setSelectionArea, [1240](#)
- slotSetHighlightArea, [1241](#)
- slotSetHighlightShown, [1241](#)
- slotSetSelection, [1241](#)
- Digikam::DPreviewManager, [1242](#)
- setSelectionArea, [1243](#)
- Digikam::DProgressDlg, [1244](#)
- Digikam::DProgressWdg, [1245](#)
- progressScheduled, [1246](#)
- Digikam::DragDropModelImplementation, [1247](#)
- dragDropFlags, [1248](#)
- dragDropFlagsV2, [1248](#)
- DragDropModelImplementation, [1248](#)
- supportedDropActions, [1248](#)
- Digikam::DragDropViewImplementation, [1249](#)
- dragDropHandler, [1250](#)
- mapIndexForDragDrop, [1250](#)
- pixmapForDrag, [1250](#)
- Digikam::DragHandle, [1251](#)
- Digikam::DRawDecoder, [1252](#)
- checkToCancelWaitingData, [1254](#)
- decodeHalfRAWImage, [1254](#)
- decodeRAWImage, [1254](#)
- extractRAWData, [1255](#)
- librawUseGomp, [1255](#)
- loadEmbeddedPreview, [1255](#), [1256](#)
- loadFullImage, [1256](#)
- loadHalfPreview, [1256](#)
- loadRawPreview, [1256](#), [1257](#)
- m\_cancel, [1258](#)
- m\_decoderSettings, [1258](#)

- rawFileIdentify, [1257](#)
- rawFilesVersion, [1257](#)
- setWaitingDataProgress, [1257](#)
- Digikam::DRawDecoderSettings, [1258](#)
  - dcblIterations, [1261](#)
  - DecodingQuality, [1260](#)
  - DontStretchPixels, [1261](#)
  - expoCorrectionHighlight, [1261](#)
  - expoCorrectionShift, [1261](#)
  - halfSizeColorImage, [1261](#)
  - InputColorSpace, [1260](#)
  - inputColorSpace, [1261](#)
  - NoiseReduction, [1260](#)
  - NRThreshold, [1262](#)
  - OutputColorSpace, [1260](#)
  - outputColorSpace, [1262](#)
  - RAWQuality, [1262](#)
  - unclipColors, [1262](#)
  - WhiteBalance, [1261](#)
  - whiteBalance, [1262](#)
- Digikam::DRawDecoderWidget, [1263](#)
  - DRawDecoderWidget, [1265](#)
  - readSettings, [1265](#)
  - writeSettings, [1265](#)
- Digikam::DRawDecoding, [1266](#)
  - bcg, [1267](#)
  - DRawDecoding, [1266](#)
- Digikam::DRawInfo, [1267](#)
  - ambientAcceleration, [1270](#)
  - ambientElevationAngle, [1270](#)
  - ambientHumidity, [1270](#)
  - ambientPressure, [1270](#)
  - ambientTemperature, [1270](#)
  - ambientWaterDepth, [1270](#)
  - baselineExposure, [1270](#)
  - DNGVersion, [1271](#)
  - DRawInfo, [1270](#)
  - exposureIndex, [1271](#)
  - exposureProgram, [1271](#)
  - flashUsed, [1271](#)
  - meteringMode, [1271](#)
  - pixelAspectRatio, [1271](#)
- Digikam::DSaveSettingsWidget, [1272](#)
- Digikam::DSelectionItem, [1273](#)
- Digikam::DSelector, [1274](#)
  - arrowDirection, [1276](#)
  - contentsRect, [1276](#)
  - drawContents, [1277](#)
  - indent, [1277](#)
  - setIndent, [1277](#)
- Digikam::DServiceInfo, [1277](#)
- Digikam::DServiceMenu, [1278](#)
- Digikam::DSliderSpinBox, [1279](#)
  - setInternalValue, [1281](#)
  - valueString, [1281](#)
- Digikam::DSplashScreen, [1282](#)
- Digikam::DSqueezedClickLabel, [1283](#)
- Digikam::DTagListDrag, [1284](#)
- Digikam::DTextBrowser, [1285](#)
- Digikam::DTextEdit, [1286](#)
  - acceptedCharacters, [1288](#)
  - DTextEdit, [1288](#)
  - ignoredCharacters, [1288](#)
  - isClearButtonEnabled, [1288](#)
  - returnPressed, [1288](#)
  - setCurrentLanguage, [1288](#)
  - setLinesVisible, [1288](#)
  - setMaxLength, [1289](#)
  - spellCheckSettings, [1289](#)
  - text, [1289](#)
- Digikam::DTextLabelName, [1290](#)
- Digikam::DTextLabelValue, [1291](#)
- Digikam::DTextList, [1292](#)
- Digikam::DToolTipStyleSheet, [1292](#)
- Digikam::DTrash, [1293](#)
  - deleteDirRecursively, [1293](#)
  - deleteImage, [1294](#)
  - extractJsonForItem, [1294](#)
- Digikam::DTrashItemInfo, [1294](#)
- Digikam::DTrashItemModel, [1295](#)
  - append, [1296](#)
  - changeThumbSize, [1296](#)
  - isEmpty, [1297](#)
  - loadItemsForCollection, [1297](#)
  - pixmapForItem, [1297](#)
  - refreshThumbnails, [1297](#)
  - removeItems, [1298](#)
- Digikam::DTrashItemsListingJob, [1299](#)
- Digikam::DuplicatesFinder, [1301](#)
- Digikam::DuplicatesProgressObserver, [1304](#)
  - imageProcessed, [1304](#)
  - isCanceled, [1304](#)
- Digikam::DVBox, [1305](#)
- Digikam::DWItemDelegate, [1306](#)
  - blockedEventTypes, [1308](#)
  - createItemWidgets, [1309](#)
  - DWItemDelegate, [1308](#)
  - focusedIndex, [1309](#)
  - itemView, [1309](#)
  - setBlockedEventTypes, [1309](#)
  - updateItemWidgets, [1310](#)
- Digikam::DWItemDelegatePool, [1310](#)
  - DWItemDelegatePool, [1311](#)
  - findWidgets, [1311](#)
- Digikam::DWItemDelegatePoolPrivate, [1311](#)
- Digikam::DWizardDlg, [1312](#)
- Digikam::DWizardPage, [1312](#)
- Digikam::DWorkingPixmap, [1313](#)
- Digikam::DXmlGuiWindow, [1314](#)
  - createFullScreenAction, [1316](#)
  - customizedFullScreenMode, [1316](#)
  - editKeyboardShortcuts, [1316](#)
  - infolface, [1316](#)
  - registerPluginsActions, [1317](#)
  - showSideBars, [1317](#)
  - showThumbBar, [1317](#)

- thumbbarVisibility, [1317](#)
- Digikam::DynamicLayout, [1318](#)
- Digikam::DynamicThread, [1319](#)
  - DynamicThread, [1320](#)
  - run, [1320](#)
  - setPriority, [1320](#)
  - shutDown, [1320](#)
  - start, [1321](#)
  - threadMutex, [1321](#)
  - wait, [1321](#)
- Digikam::DZoomBar, [1322](#)
  - BarMode, [1323](#)
  - NoPreviewZoomCtrl, [1324](#)
  - PreviewZoomCtrl, [1324](#)
  - ThumbsSizeCtrl, [1324](#)
- Digikam::EditableSearchTreeView, [1324](#)
  - addCustomContextMenuActions, [1330](#)
  - contextMenuTitle, [1330](#)
  - EditableSearchTreeView, [1330](#)
  - handleCustomContextMenuAction, [1330](#)
- Digikam::EditorCore, [1331](#)
- Digikam::EditorStackView, [1334](#)
- Digikam::EditorTool, [1336](#)
- Digikam::EditorTooliface, [1338](#)
- Digikam::EditorToolSettings, [1339](#)
- Digikam::EditorToolThreaded, [1341](#)
  - deleteFilterInstance, [1344](#)
  - setProgressMessage, [1344](#)
- Digikam::EditorWindow, [1345](#)
  - m\_transformQueue, [1351](#)
  - registerExtraPluginsActions, [1350](#)
  - saveDestinationUrl, [1350](#)
  - toggleZoomActions, [1351](#)
- Digikam::EffectMgr, [1351](#)
  - EffectType, [1351](#)
  - None, [1352](#)
- Digikam::EffectPreview, [1352](#)
- Digikam::Ellipsoid, [1352](#)
  - createEllipsoid, [1355](#)
  - createFlattenedSphere, [1355](#)
  - eccentricity, [1355](#)
  - Ellipsoid, [1354](#)
  - inverseFlattening, [1355](#)
  - isIvfDefinitive, [1355](#)
  - isSphere, [1356](#)
  - m\_inverseFlattening, [1357](#)
  - m\_ivfDefinitive, [1357](#)
  - m\_semiMajorAxis, [1358](#)
  - m\_semiMinorAxis, [1358](#)
  - orthodromicDistance, [1356](#)
  - radiusOfCurvature, [1356](#)
  - semiMajorAxis, [1357](#)
  - semiMinorAxis, [1357](#)
  - SPHERE, [1357](#)
  - WGS84, [1357](#)
- Digikam::EmbossFilter, [1359](#)
  - filterAction, [1363](#)
  - filterIdentifier, [1363](#)
  - readParameters, [1363](#)
- Digikam::EmptyDTrashItemsJob, [1364](#)
- Digikam::EmptyImageListProvider, [1366](#)
  - atEnd, [1367](#)
  - image, [1367](#)
  - images, [1367](#)
  - proceed, [1367](#)
  - setImages, [1367](#)
  - setUnpairedImages, [1367](#)
  - size, [1367](#)
- Digikam::EqualizeFilter, [1368](#)
  - filterAction, [1372](#)
  - filterIdentifier, [1372](#)
  - readParameters, [1372](#)
- Digikam::ExifMetaEngineMergeHelper, [1372](#)
- Digikam::ExifToolBinary, [1374](#)
- Digikam::ExifToolConfPanel, [1376](#)
- Digikam::ExifToolErrorView, [1377](#)
- Digikam::ExifToolListView, [1378](#)
  - setGroupList, [1379](#)
- Digikam::ExifToolListViewGroup, [1379](#)
- Digikam::ExifToolListViewItem, [1380](#)
- Digikam::ExifToolLoadingView, [1381](#)
- Digikam::ExifToolParser, [1382](#)
  - applyChanges, [1384](#)
  - applyMetadataFile, [1385](#)
  - changeTimestamps, [1385](#)
  - copyTags, [1385](#)
  - ExifToolData, [1384](#)
  - load, [1386](#)
  - loadChunk, [1386](#)
  - readableFormats, [1386](#)
  - tagsDatabase, [1386](#)
  - tagsDbToOrderedMap, [1386](#)
  - translateTags, [1386](#)
  - translationsList, [1387](#)
  - version, [1387](#)
  - writableFormats, [1387](#)
- Digikam::ExifToolProcess, [1388](#)
  - ~ExifToolProcess, [1391](#)
  - Action, [1390](#)
  - APPLY\_CHANGES, [1390](#)
  - APPLY\_CHANGES\_EXV, [1390](#)
  - APPLY\_METADATA\_FILE, [1390](#)
  - CHANGE\_TIMESTAMP, [1390](#)
  - command, [1391](#)
  - COPY\_ALL, [1390](#)
  - COPY\_EXIF, [1390](#)
  - COPY\_ICC, [1390](#)
  - COPY\_IPTC, [1390](#)
  - COPY\_MAKERNOTES, [1390](#)
  - COPY\_NONE, [1390](#)
  - COPY\_TAGS, [1390](#)
  - COPY\_XMP, [1390](#)
  - CopyTagsSource, [1390](#)
  - CREATE\_NEW\_GROUPS, [1391](#)
  - CREATE\_NEW\_TAGS, [1391](#)
  - initExifTool, [1391](#)

- LOAD\_CHUNKS, [1390](#)
- LOAD\_METADATA, [1390](#)
- NO\_ACTION, [1390](#)
- READ\_FORMATS, [1390](#)
- RESTORE\_PREVIEW, [1390](#)
- setExifToolProgram, [1391](#)
- shutDownExifTool, [1391](#)
- TAGS\_DATABASE, [1390](#)
- TRANS\_ALL\_EXIF, [1390](#)
- TRANS\_ALL\_IPTC, [1390](#)
- TRANS\_ALL\_XMP, [1390](#)
- TRANS\_TAGS, [1390](#)
- TranslateTagsOps, [1390](#)
- TRANSLATIONS\_LIST, [1390](#)
- VERSION\_STRING, [1390](#)
- waitForExifToolResult, [1391](#)
- WRITE\_EXISTING\_TAGS, [1391](#)
- WRITE\_FORMATS, [1390](#)
- WritingTagsMode, [1391](#)
- Digikam::ExifToolProcess::Result, [1392](#)
- Digikam::ExifToolThread, [1392](#)
- Digikam::ExifToolWidget, [1393](#)
- Digikam::ExifWidget, [1395](#)
  - getMetadataTitle, [1397](#)
  - getTagDescription, [1397](#)
  - getTagTitle, [1397](#)
  - loadFromURL, [1397](#)
- Digikam::ExposureDetector, [1398](#)
  - detect, [1399](#)
- Digikam::ExposureSettingsContainer, [1399](#)
  - exposureIndicatorMode, [1399](#)
- Digikam::FaceClassifier, [1400](#)
  - loadTrainingData, [1401](#)
  - predict, [1401](#)
  - retrain, [1401](#)
- Digikam::FaceClassifierBase, [1402](#)
- Digikam::FaceDb, [1403](#)
  - clearDNNTraining, [1404](#)
  - insertFaceVector, [1404](#)
  - removeFaceVector, [1404](#)
  - trainData, [1405](#)
- Digikam::FaceDbAccess, [1405](#)
  - FaceDbAccess, [1405](#)
  - setLastError, [1406](#)
- Digikam::FaceDbAccessUnlock, [1406](#)
  - FaceDbAccessUnlock, [1406](#)
- Digikam::FaceDbBackend, [1407](#)
  - initSchema, [1411](#)
- Digikam::FaceDbOperationGroup, [1411](#)
  - allowLift, [1411](#)
  - lift, [1411](#)
- Digikam::FaceDbSchemaUpdater, [1412](#)
- Digikam::FaceDetector, [1412](#)
  - detectFaces, [1413](#)
  - FaceDetector, [1412](#)
  - recommendedImageSize, [1413](#)
  - setParameter, [1413](#)
- Digikam::FaceGroup, [1414](#)
  - aboutToSetInfo, [1416](#)
  - closestItem, [1416](#)
  - setAutoSuggest, [1416](#)
- Digikam::FacelItem, [1417](#)
- Digikam::FacelItemRetriever, [1420](#)
- Digikam::FacePipeline, [1421](#)
  - confirm, [1424](#)
  - editRegion, [1424](#)
  - editTag, [1424](#)
  - FilterMode, [1423](#)
  - NormalWrite, [1424](#)
  - OverwriteAllFaces, [1424](#)
  - OverwriteUnconfirmed, [1424](#)
  - plugDatabaseFilter, [1424](#)
  - process, [1425](#)
  - ReadConfirmedFaces, [1423](#)
  - ReadFacesForTraining, [1423](#)
  - ReadUnconfirmedFaces, [1423](#)
  - ScanAll, [1423](#)
  - setPriority, [1425](#)
  - SkipAlreadyScanned, [1423](#)
  - WriteMode, [1423](#)
- Digikam::FacePipelineBase, [1426](#)
  - enqueue, [1429](#)
  - FilterMode, [1428](#)
  - NormalWrite, [1429](#)
  - OverwriteAllFaces, [1429](#)
  - OverwriteUnconfirmed, [1429](#)
  - ScanAll, [1428](#)
  - ScanNew, [1428](#)
  - TrainAll, [1428](#)
  - TrainNew, [1428](#)
  - TrainRemove, [1428](#)
  - TrainReset, [1428](#)
  - WriteMode, [1429](#)
- Digikam::FacePipelineDetect, [1430](#)
  - addMoreWorkers, [1433](#)
  - classifier, [1433](#)
  - extractor, [1433](#)
  - finder, [1433](#)
  - loader, [1433](#)
  - start, [1433](#)
  - trainer, [1433](#)
  - writer, [1433](#)
- Digikam::FacePipelineDetectRecognize, [1434](#)
  - addMoreWorkers, [1437](#)
  - classifier, [1437](#)
  - extractor, [1437](#)
  - finder, [1437](#)
  - loader, [1437](#)
  - start, [1437](#)
  - trainer, [1437](#)
  - writer, [1437](#)
- Digikam::FacePipelineEdit, [1438](#)
  - addMoreWorkers, [1441](#)
  - classifier, [1441](#)
  - extractor, [1441](#)
  - finder, [1441](#)

- loader, [1442](#)
- start, [1442](#)
- trainer, [1442](#)
- writer, [1442](#)
- Digikam::FacePipelineExtendedPackage, [1443](#)
- Digikam::FacePipelineFaceTagsIface, [1445](#)
  - Confirmed, [1447](#)
  - ForRecognition, [1447](#)
  - GivenAsArgument, [1447](#)
  - Role, [1447](#)
- Digikam::FacePipelineFaceTagsIfaceList, [1448](#)
- Digikam::FacePipelinePackage, [1449](#)
- Digikam::FacePipelinePackageBase, [1450](#)
- Digikam::FacePipelineRecognize, [1452](#)
  - addMoreWorkers, [1455](#)
  - classifier, [1455](#)
  - extractor, [1455](#)
  - finder, [1455](#)
  - loader, [1455](#)
  - start, [1455](#)
  - trainer, [1455](#)
  - writer, [1455](#)
- Digikam::FacePipelineReset, [1456](#)
  - addMoreWorkers, [1459](#)
  - classifier, [1459](#)
  - extractor, [1459](#)
  - finder, [1459](#)
  - loader, [1459](#)
  - start, [1459](#)
  - trainer, [1459](#)
  - writer, [1459](#)
- Digikam::FacePipelineRetrain, [1460](#)
  - addMoreWorkers, [1463](#)
  - classifier, [1463](#)
  - extractor, [1463](#)
  - finder, [1463](#)
  - loader, [1463](#)
  - start, [1463](#)
  - trainer, [1463](#)
  - writer, [1463](#)
- Digikam::FacePreprocessor, [1464](#)
- Digikam::FacePreviewLoader, [1465](#)
- Digikam::FaceRejectionOverlay, [1471](#)
  - checkIndex, [1474](#)
  - createButton, [1474](#)
  - setActive, [1474](#)
  - updateButton, [1474](#)
  - widgetEnterEvent, [1475](#)
  - widgetLeaveEvent, [1475](#)
- Digikam::FaceRejectionOverlayButton, [1476](#)
  - icon, [1477](#)
  - sizeHint, [1477](#)
  - updateToolTip, [1478](#)
- Digikam::FaceScanSettings, [1478](#)
  - AlreadyScannedHandling, [1479](#)
  - ClearAll, [1479](#)
  - detectAccuracy, [1480](#)
  - DetectAndRecognize, [1479](#)
  - FaceDetectionModel, [1479](#)
  - FaceRecognitionModel, [1479](#)
  - OpenFace, [1479](#)
  - recognizeAccuracy, [1480](#)
  - RecognizeMarkedFaces, [1479](#)
  - RecognizeOnly, [1479](#)
  - Rescan, [1479](#)
  - RetrainAll, [1479](#)
  - ScanTask, [1479](#)
  - SFace, [1479](#)
  - Skip, [1479](#)
  - SSDMOBILENET, [1479](#)
  - YOLOv3, [1479](#)
  - YuNet, [1479](#)
- Digikam::FaceScanWidget, [1480](#)
  - doLoadState, [1482](#)
  - doSaveState, [1482](#)
- Digikam::FacesDetector, [1483](#)
- Digikam::FacesEngine, [1486](#)
- Digikam::FaceTags, [1489](#)
  - applyTagIdentityMapping, [1489](#)
  - ensureIsPerson, [1489](#)
  - getOrCreateTagForPerson, [1490](#)
  - tagForPerson, [1490](#)
- Digikam::FaceTagsEditor, [1491](#)
  - add, [1493](#)
  - addNormalTag, [1493](#)
  - changeRegion, [1493](#)
  - changeTag, [1493](#)
  - confirmName, [1493](#)
  - getSuggestedNames, [1494](#)
  - getTagRects, [1494](#)
  - removeFace, [1494](#)
  - removeNormalTag, [1494](#)
  - unconfirmedEntry, [1494](#)
  - unconfirmedFaceTagsIfaces, [1495](#)
  - unconfirmedNameFaceTagsIfaces, [1495](#)
- Digikam::FaceTagsIface, [1496](#)
  - fromVariant, [1498](#)
  - typeForAttribute, [1498](#)
- Digikam::FaceUtils, [1499](#)
  - addNormalTag, [1501](#)
  - faceRectToDisplayRect, [1501](#)
  - removeNormalTag, [1502](#)
  - removeNormalTags, [1502](#)
  - storeThumbnails, [1502](#)
  - toFaceTagsIfaces, [1502](#)
  - writeUnconfirmedResults, [1502](#)
- Digikam::FacialRecognitionWrapper, [1503](#)
  - addIdentity, [1504](#)
  - allIdentities, [1504](#)
  - findIdentity, [1504](#)
  - recognizeFaces, [1504](#)
  - setParameter, [1505](#)
  - train, [1505](#)
- Digikam::FFmpegBinary, [1506](#)
- Digikam::FFmpegConfigHelper, [1508](#)
  - getAudioCodecsProperties, [1509](#)



- getExtensionsProperties, 1509
- getVideoCodecsProperties, 1509
- Digikam::FFmpegLauncher, 1510
  - soundTrackLength, 1511
- Digikam::FieldQueryBuilder, 1512
- Digikam::FileActionItemInfoList, 1513
- Digikam::FileActionMngr, 1515
  - transform, 1516
- Digikam::FileActionMngrDatabaseWorker, 1517
  - applyMetadata, 1519
  - assignColorLabel, 1519
  - assignPickLabel, 1519
  - assignRating, 1519
  - assignTags, 1520
  - copyAttributes, 1520
  - editGroup, 1520
  - removeTags, 1520
  - setExifOrientation, 1520
- Digikam::FileActionMngrFileWorker, 1521
  - transform, 1523
  - writeMetadata, 1523
  - writeMetadataToFiles, 1523
  - writeOrientationToFiles, 1523
- Digikam::FileActionProgress, 1524
- Digikam::FileActionProgressItemContainer, 1527
- Digikam::FileActionProgressItemCreator, 1528
- Digikam::FilePropertiesOption, 1529
  - parseOperation, 1531
- Digikam::FileReadLocker, 1531
- Digikam::FileReadWriteLockKey, 1531
- Digikam::FileSaveConflictBox, 1532
- Digikam::FileSaveOptionsBox, 1533
  - discoverFormat, 1534
  - FileSaveOptionsBox, 1534
  - FORMAT, 1533
  - NONE, 1534
- Digikam::FileSaveOptionsDlg, 1535
- Digikam::FilesDownloader, 1536
- Digikam::FileWorkerInterface, 1537
- Digikam::FileWriteLocker, 1539
- Digikam::FilmContainer, 1539
- Digikam::FilmContainer::ListItem, 1540
- Digikam::FilmFilter, 1541
  - filterAction, 1545
  - filterIdentifier, 1545
  - readParameters, 1545
- Digikam::FilmGrainContainer, 1545
- Digikam::FilmGrainFilter, 1546
  - filterAction, 1550
  - filterIdentifier, 1550
  - readParameters, 1550
- Digikam::FilmGrainSettings, 1550
- Digikam::Filter, 1551
- Digikam::FilterAction, 1552
  - Category, 1554
  - ComplexFilter, 1555
  - description, 1555
  - DocumentedHistory, 1555
  - ExplicitBranch, 1555
  - Flag, 1555
  - hasParameters, 1555
  - identifier, 1555
  - m\_category, 1556
  - parameter, 1555
  - ReproducibleFilter, 1555
  - version, 1556
- Digikam::FilterActionFilter, 1557
  - appliedFilterActions, 1561
  - completelyApplied, 1561
  - filterAction, 1561
  - filterIdentifier, 1561
  - filterImage, 1562
  - isComplexAction, 1562
  - readParameters, 1562
  - setContinueOnError, 1562
- Digikam::FiltersHistoryWidget, 1563
- Digikam::FilterSideBarWidget, 1564
  - doLoadState, 1566
  - doSaveState, 1566
  - FilterSideBarWidget, 1566
  - setConfigGroup, 1567
  - signalTagFilterChanged, 1567
- Digikam::FilterStatusBar, 1568
- Digikam::FindDuplicatesAlbum, 1568
- Digikam::FindDuplicatesAlbumItem, 1570
- Digikam::FindDuplicatesView, 1571
- Digikam::FingerPrintsGenerator, 1572
  - FingerPrintsGenerator, 1575
  - setUseMultiCoreCPU, 1575
- Digikam::FingerprintsTask, 1576
- Digikam::FirstRunDlg, 1578
- Digikam::FocusPoint, 1578
  - FocusPoint, 1579
  - Inactive, 1579
  - InFocus, 1579
  - Selected, 1579
  - SelectedInFocus, 1579
  - setType, 1579
  - TypePoint, 1579
- Digikam::FocusPointGroup, 1580
- Digikam::FocusPointItem, 1582
- Digikam::FocusPointsExtractor, 1585
  - ListAFPoints, 1586
- Digikam::FocusPointsWriter, 1586
- Digikam::FrameOsd, 1586
- Digikam::FrameOsdSettings, 1587
- Digikam::FrameOsdWidget, 1588
- Digikam::FrameUtils, 1588
- Digikam::FreeRotationContainer, 1588
- Digikam::FreeRotationFilter, 1590
  - filterAction, 1594
  - filterIdentifier, 1594
  - readParameters, 1594
- Digikam::FreeRotationSettings, 1594
- Digikam::FreeSpaceToolTip, 1596
  - repositionRect, 1597

- tipContents, [1597](#)
- Digikam::FreeSpaceWidget, [1598](#)
- Digikam::FullObjectDetection, [1599](#)
- Digikam::FullScreenSettings, [1600](#)
- Digikam::FuzzySearchSideBarWidget, [1601](#)
  - applySettings, [1603](#)
  - changeAlbumFromHistory, [1603](#)
  - doLoadState, [1603](#)
  - doSaveState, [1603](#)
  - getCaption, [1603](#)
  - getIcon, [1604](#)
  - setActive, [1604](#)
- Digikam::FuzzySearchView, [1605](#)
  - doLoadState, [1607](#)
  - doSaveState, [1607](#)
  - setConfigGroup, [1607](#)
- Digikam::GeoCoordinates, [1607](#)
  - fromMarbleCoordinates, [1608](#)
- Digikam::GeodeticCalculator, [1609](#)
  - azimuth, [1611](#)
  - checkAzimuth, [1611](#)
  - checkLatitude, [1612](#)
  - checkLongitude, [1612](#)
  - checkOrthodromicDistance, [1612](#)
  - computeDirection, [1612](#)
  - destinationGeographicPoint, [1612](#)
  - fo, [1615](#)
  - GeodeticCalculator, [1611](#)
  - m\_destinationValid, [1615](#)
  - m\_directionValid, [1615](#)
  - m\_lat1, [1615](#)
  - m\_lat2, [1615](#)
  - m\_TOLERANCE\_CHECK, [1615](#)
  - meridianArcLength, [1613](#)
  - meridianArcLengthRadians, [1613](#)
  - orthodromicDistance, [1613](#)
  - setDestinationGeographicPoint, [1614](#)
  - setDirection, [1614](#)
  - setStartingGeographicPoint, [1614](#)
- Digikam::GeoDragDropHandler, [1616](#)
- Digikam::GeofaceCluster, [1616](#)
- Digikam::GeofaceGlobalObject, [1617](#)
- Digikam::GeofaceInternalWidgetInfo, [1619](#)
- Digikam::GeofaceSharedData, [1620](#)
  - hasRegionSelection, [1621](#)
- Digikam::GeolocationFilter, [1622](#)
- Digikam::GeolocationSettings, [1623](#)
  - instance, [1624](#)
  - mainMarbleWidget, [1624](#)
- Digikam::GeolocationSettingsContainer, [1624](#)
- Digikam::GeoModelHelper, [1625](#)
  - bestRepresentativeIndexFromList, [1626](#)
  - itemCoordinates, [1626](#)
  - itemIcon, [1627](#)
  - model, [1627](#)
  - onIndicesClicked, [1627](#)
  - pixmapFromRepresentativeIndex, [1627](#)
  - selectionModel, [1628](#)
- Digikam::GeoPluginAboutDlg, [1628](#)
- Digikam::GPCamera, [1629](#)
  - cameraAbout, [1631](#)
  - cameraDriverType, [1631](#)
  - cameraManual, [1631](#)
  - cameraMD5ID, [1632](#)
  - cameraSummary, [1632](#)
  - cancel, [1632](#)
  - capture, [1632](#)
  - deleteItem, [1632](#)
  - doConnect, [1632](#)
  - downloadItem, [1632](#)
  - getFolders, [1633](#)
  - getFreeSpace, [1633](#)
  - getItemInfo, [1633](#)
  - getItemInfoList, [1633](#)
  - getMetadata, [1633](#)
  - getPreview, [1633](#)
  - getThumbnail, [1634](#)
  - setLockItem, [1634](#)
  - uploadItem, [1634](#)
- Digikam::GPSBookmarkModelHelper, [1635](#)
  - itemCoordinates, [1636](#)
  - itemFlags, [1636](#)
  - itemIcon, [1637](#)
  - model, [1637](#)
  - modelFlags, [1637](#)
  - selectionModel, [1637](#)
  - snapItemsTo, [1637](#)
- Digikam::GPSBookmarkOwner, [1638](#)
- Digikam::GPSCorrelatorWidget, [1639](#)
- Digikam::GPSDataContainer, [1640](#)
- Digikam::GPSDBJobInfo, [1641](#)
- Digikam::GPSDBJobsThread, [1643](#)
  - GPSListing, [1645](#)
- Digikam::GPSGeofaceModelHelper, [1646](#)
  - bestRepresentativeIndexFromList, [1647](#)
  - itemCoordinates, [1647](#)
  - model, [1648](#)
  - modelFlags, [1648](#)
  - onIndicesMoved, [1648](#)
  - pixmapFromRepresentativeIndex, [1648](#)
  - selectionModel, [1648](#)
- Digikam::GPSItemContainer, [1649](#)
  - isTagListDirty, [1651](#)
  - loadImageData, [1651](#)
  - restoreGPSData, [1651](#)
  - saveChanges, [1651](#)
  - setTagList, [1651](#)
- Digikam::GPSItemDelegate, [1652](#)
- Digikam::GPSItemInfo, [1652](#)
- Digikam::GPSItemInfoSorter, [1653](#)
- Digikam::GPSItemList, [1654](#)
- Digikam::GPSItemListContextMenu, [1656](#)
- Digikam::GPSItemListDragDropHandler, [1657](#)
  - createMimeData, [1658](#)
- Digikam::GPSItemModel, [1658](#)
- Digikam::GPSItemSortProxyModel, [1660](#)



- Digikam::GPSJob, [1661](#)
- Digikam::GPSLinkItemSelectionModel, [1663](#)
- Digikam::GPSMarkerTiler, [1664](#)
  - bestRepresentativeIndexFromList, [1667](#)
  - getGlobalGroupState, [1667](#)
  - getTile, [1668](#)
  - getTileGroupState, [1668](#)
  - getTileMarkerCount, [1668](#)
  - getTileRepresentativeMarker, [1668](#)
  - getTileSelectedCount, [1669](#)
  - GPSMarkerTiler, [1667](#)
  - indicesEqual, [1669](#)
  - onIndicesClicked, [1669](#)
  - pixmapFromRepresentativeIndex, [1669](#)
  - prepareTiles, [1669](#)
  - regenerateTiles, [1670](#)
  - setActive, [1670](#)
  - setPositiveFilterIsActive, [1670](#)
  - slotNewModelData, [1670](#)
  - tileNew, [1670](#)
- Digikam::GPSModelIndexProxyMapper, [1671](#)
  - isConnected, [1672](#)
- Digikam::GPSSearchSideBarWidget, [1673](#)
  - applySettings, [1675](#)
  - changeAlbumFromHistory, [1675](#)
  - doLoadState, [1675](#)
  - doSaveState, [1675](#)
  - getCaption, [1675](#)
  - getIcon, [1676](#)
  - setActive, [1676](#)
- Digikam::GPSSearchView, [1677](#)
  - doLoadState, [1679](#)
  - doSaveState, [1679](#)
  - GPSSearchView, [1679](#)
  - setActive, [1679](#)
  - setConfigGroup, [1679](#)
- Digikam::GPSUndoCommand, [1680](#)
- Digikam::GPSUndoCommand::UndoInfo, [1681](#)
- Digikam::Graph< VertexProperties, EdgeProperties >, [1681](#)
  - AdjacencyFlags, [1686](#)
  - edgeDifference, [1686](#)
  - EdgesToLeaf, [1686](#)
  - leaves, [1686](#)
  - listPath, [1686](#)
  - longestPathTouching, [1687](#)
  - roots, [1687](#)
  - rootsOf, [1687](#)
  - shortestDistancesFrom, [1687](#)
  - shortestPath, [1687](#)
  - transitiveReduction, [1687](#)
  - vertexCount, [1688](#)
  - verticesBreadthFirst, [1688](#)
  - verticesDepthFirstSorted, [1688](#)
  - verticesDominatedBy, [1688](#)
  - verticesDominatedByDepthFirstSorted, [1688](#)
- Digikam::Graph< VertexProperties, EdgeProperties >::DominatorTree, [1689](#)
- Digikam::Graph< VertexProperties, EdgeProperties >::Edge, [1689](#)
- Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch, [1690](#)
  - depth\_first\_search\_sorted, [1690](#)
- Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch::BreadthFirstSearchVisitor, [1691](#)
- Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch::CommonVisitor, [1692](#)
- Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch::DepthFirstSearchVisitor, [1693](#)
- Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch::lessThanMapEdgeToTarget< GraphType, VertexLessThan >, [1694](#)
- Digikam::Graph< VertexProperties, EdgeProperties >::Path, [1694](#)
  - longestPath, [1695](#)
  - shortestPath, [1695](#)
- Digikam::Graph< VertexProperties, EdgeProperties >::Vertex, [1695](#)
- Digikam::GraphicsDImgItem, [1696](#)
  - setImage, [1697](#)
- Digikam::GraphicsDImgView, [1698](#)
  - scrollPointOnPoint, [1700](#)
  - setItem, [1700](#)
- Digikam::GreycstorationContainer, [1700](#)
- Digikam::GreycstorationFilter, [1701](#)
  - cancelFilter, [1705](#)
  - filterAction, [1705](#)
  - filterIdentifier, [1705](#)
  - GreycstorationFilter, [1705](#)
  - MODE, [1705](#)
  - readParameters, [1706](#)
  - SimpleResize, [1705](#)
- Digikam::GreycstorationSettings, [1706](#)
- Digikam::GroupedImagesFinder, [1707](#)
  - GroupedImagesFinder, [1707](#)
- Digikam::GroupIndicatorOverlay, [1708](#)
  - checkIndex, [1711](#)
  - createWidget, [1711](#)
  - setActive, [1711](#)
  - slotEntered, [1711](#)
  - visualChange, [1711](#)
- Digikam::GroupIndicatorOverlayWidget, [1712](#)
- Digikam::GroupingViewImplementation, [1713](#)
  - hasHiddenGroupedImages, [1714](#)
- Digikam::GroupItemFilterSettings, [1714](#)
- Digikam::GroupStateComputer, [1714](#)
- Digikam::Haar::Calculator, [1715](#)
  - calcHaar, [1715](#)
  - transform, [1715](#)
- Digikam::Haar::ImageData, [1715](#)
- Digikam::Haar::SignatureData, [1716](#)
- Digikam::Haar::SignatureMap, [1716](#)
- Digikam::Haar::WeightBin, [1716](#)
  - m\_bin, [1717](#)

- Digikam::Haar::Weights, [1717](#)
- Digikam::HaarIface, [1717](#)
  - bestMatchesForImageWithThreshold, [1719](#)
  - ExcludeFolder, [1719](#)
  - findDuplicates, [1719](#)
  - loadQImage, [1720](#)
  - NewerCreationDate, [1719](#)
  - NewerModificationDate, [1719](#)
  - OlderOrLarger, [1719](#)
  - PreferFolder, [1719](#)
  - rebuildDuplicatesAlbums, [1720](#)
  - ReflmageSelMethod, [1719](#)
  - retrieveSignatureFromDB, [1720](#)
  - setAlbumRootsToSearch, [1720](#)
  - signatureAsText, [1721](#)
- Digikam::HaarProgressObserver, [1721](#)
- Digikam::HidingStateChanger, [1722](#)
  - HidingStateChanger, [1725](#)
- Digikam::Highlighter, [1725](#)
- Digikam::HistogramBox, [1726](#)
- Digikam::HistogramPainter, [1727](#)
  - enableHistogramGuideByColor, [1728](#)
  - HistogramPainter, [1728](#)
  - initFrom, [1729](#)
  - render, [1729](#)
  - setChannelType, [1729](#)
  - setHighlightSelection, [1729](#)
  - setHistogram, [1730](#)
  - setRenderXGrid, [1730](#)
  - setScale, [1730](#)
  - setSelection, [1730](#)
- Digikam::HistogramWidget, [1731](#)
  - HistogramWidget, [1733](#)
- Digikam::HistoryEdgeProperties, [1733](#)
- Digikam::HistoryImageld, [1733](#)
  - Current, [1735](#)
  - Intermediate, [1735](#)
  - m\_originalUUID, [1735](#)
  - m\_uuid, [1735](#)
  - Original, [1735](#)
  - Source, [1735](#)
  - Type, [1735](#)
- Digikam::HistoryVertexProperties, [1735](#)
- Digikam::HotPixelContainer, [1736](#)
- Digikam::HotPixelFixer, [1737](#)
  - filterAction, [1741](#)
  - filterIdentifier, [1741](#)
  - readParameters, [1741](#)
- Digikam::HotPixelProps, [1741](#)
  - operator==, [1742](#)
- Digikam::HotPixelSettings, [1742](#)
- Digikam::HotPixelsWeights, [1743](#)
- Digikam::HoverButtonDelegateOverlay, [1744](#)
  - createButton, [1747](#)
  - createWidget, [1747](#)
  - setActive, [1747](#)
  - updateButton, [1747](#)
  - visualChange, [1747](#)
- Digikam::HSLContainer, [1748](#)
- Digikam::HSLFilter, [1749](#)
  - filterAction, [1753](#)
  - filterIdentifier, [1753](#)
  - readParameters, [1753](#)
- Digikam::HSLSettings, [1753](#)
- Digikam::HSPreviewWidget, [1754](#)
- Digikam::HTMLWidget, [1755](#)
- Digikam::HTMLWidgetPage, [1756](#)
- Digikam::lccManager, [1757](#)
  - lccManager, [1759](#)
  - needsPostLoadingManagement, [1759](#)
  - transformDefault, [1759](#)
  - transformForDisplay, [1759](#)
- Digikam::lccPostLoadingManager, [1760](#)
  - lccPostLoadingManager, [1762](#)
  - postLoadingManage, [1762](#)
- Digikam::ICCPreviewWidget, [1762](#)
- Digikam::lccProfile, [1763](#)
  - close, [1764](#)
  - data, [1764](#)
  - defaultSearchPaths, [1764](#)
  - description, [1764](#)
  - Display, [1764](#)
  - filePath, [1765](#)
  - Input, [1764](#)
  - InvalidType, [1764](#)
  - open, [1765](#)
  - operator==, [1765](#)
  - Output, [1764](#)
  - ProfileType, [1764](#)
  - sRGB, [1765](#)
  - type, [1765](#)
- Digikam::ICCProfileInfoDlg, [1766](#)
- Digikam::lccProfilesComboBox, [1767](#)
  - addProfileSqueezed, [1769](#)
  - lccProfilesComboBox, [1769](#)
  - setCurrentProfile, [1769](#)
- Digikam::lccProfilesMenuAction, [1770](#)
  - addProfile, [1771](#)
- Digikam::lccProfilesSettings, [1772](#)
- Digikam::ICCProfileWidget, [1774](#)
  - getMetadataTitle, [1776](#)
  - getTagDescription, [1776](#)
  - getTagTitle, [1776](#)
  - loadFromURL, [1776](#)
- Digikam::lccRenderingIntentComboBox, [1777](#)
- Digikam::lccSettings, [1778](#)
  - instance, [1779](#)
  - loadAllProfilesProperties, [1779](#)
  - monitorProfile, [1779](#)
- Digikam::ICCSettingsContainer, [1780](#)
  - BehaviorEnum, [1781](#)
  - InvalidBehavior, [1781](#)
  - KeepProfile, [1781](#)
  - LeaveFileUntagged, [1781](#)
  - PreserveEmbeddedProfile, [1781](#)
  - UseEmbeddedProfile, [1781](#)

- Digikam::lccTransform, 1781
  - apply, 1782
  - close, 1782
  - setDoNotEmbedOutputProfile, 1782
  - setEmbeddedProfile, 1782
  - willHaveEffect, 1783
- Digikam::lccTransformFilter, 1784
  - filterAction, 1788
  - filterIdentifier, 1788
  - filterImage, 1788
  - parametersSuccessfullyRead, 1788
  - progressInfo, 1788
  - readParameters, 1789
  - readParametersError, 1789
- Digikam::Identity, 1789
  - Identity, 1789
- Digikam::IdentityProvider, 1790
  - addIdentity, 1791
  - findIdentity, 1791
- Digikam::ImageChangeset, 1791
  - ImageChangeset, 1791
- Digikam::ImageCommonContainer, 1792
- Digikam::ImageCurves, 1792
  - channelToBinary, 1794
  - CURVE\_FREE, 1794
  - CURVE\_SMOOTH, 1794
  - CurveType, 1793
  - fillFromOtherCurves, 1794
  - setChannelFromBinary, 1794
  - setContainer, 1794
- Digikam::ImageDialog, 1795
- Digikam::ImageDialogIconProvider, 1796
- Digikam::ImageDialogPreview, 1797
- Digikam::ImageDialogToolTip, 1798
- Digikam::ImageGuideWidget, 1800
- Digikam::ImageHistogram, 1802
  - run, 1804
- Digikam::ImageHistoryEntry, 1804
- Digikam::ImageIface, 1805
  - FullImage, 1806
  - ImageIface, 1806
  - ImageSelection, 1806
  - original, 1806
  - paint, 1806
  - previewReference, 1806
  - PreviewType, 1806
  - setOriginal, 1807
  - setPreview, 1807
  - setPreviewSize, 1807
  - setPreviewType, 1807
  - setSelection, 1807
- Digikam::ImageLevels, 1808
- Digikam::ImageListProvider, 1808
- Digikam::ImageMetadataContainer, 1810
- Digikam::ImagePreviewItem, 1811
- Digikam::ImageQualityCalculator, 1812
- Digikam::ImageQualityCalculator::ResultDetection, 1813
- Digikam::ImageQualityConfSelector, 1813
  - CustomSettings, 1814
  - GlobalSettings, 1814
  - SettingsType, 1814
- Digikam::ImageQualityContainer, 1814
- Digikam::ImageQualityParser, 1815
- Digikam::ImageQualitySettings, 1816
- Digikam::ImageQualitySorter, 1817
  - AllItems, 1820
  - ImageQualitySorter, 1820
  - NonAssignedItems, 1820
  - QualityScanMode, 1819
  - setUseMultiCoreCPU, 1820
- Digikam::ImageQualityTask, 1821
- Digikam::ImageQualityThread, 1823
- Digikam::ImageQualityThreadPool, 1824
- Digikam::ImageRegionItem, 1825
- Digikam::ImageRegionWidget, 1827
  - getOriginalRegionImage, 1829
- Digikam::ImageRelation, 1829
- Digikam::ImageSortFilterModel, 1830
  - imageFilterModel, 1832
  - imageInfosSorted, 1832
  - mapListToSource, 1832
  - setDirectSourceItemModel, 1832
  - setSourceModel, 1832
- Digikam::ImageTagChangeset, 1833
  - Operation, 1833
  - operator<<, 1833
- Digikam::ImageTagProperty, 1834
- Digikam::ImageTagPropertyName, 1834
- Digikam::ImageWindow, 1835
  - infoIface, 1841
  - versionManager, 1841
- Digikam::ImageZoomSettings, 1841
  - fitToSize, 1842
  - originalImageSize, 1842
  - setImageSize, 1842
  - snappedZoomFactor, 1842
  - snappedZoomStep, 1843
  - zoomedSize, 1843
- Digikam::ImportCategorizedView, 1844
  - activated, 1849
  - addOverlay, 1849
  - camItemInfoActivated, 1850
  - deselected, 1850
  - dragDropHandler, 1850
  - filterModel, 1850
  - importFilterModel, 1850
  - indexActivated, 1850
  - nextIndexHint, 1851
  - nextInOrder, 1851
  - selected, 1851
  - showContextMenuOnIndex, 1851
- Digikam::ImportCategoryDrawer, 1852
  - categoryHeight, 1853
  - drawCategory, 1853
- Digikam::ImportContextMenuHelper, 1855

- addAction, [1856](#), [1857](#)
- addAssignTagsMenu, [1857](#)
- addGroupMenu, [1858](#)
- addLabelsAction, [1858](#)
- addRemoveTagsMenu, [1858](#)
- addRotateMenu, [1859](#)
- addServicesMenu, [1859](#)
- addSubMenu, [1859](#)
- exec, [1859](#)
- ImportContextMenuHelper, [1856](#)
- setImportFilterModel, [1860](#)
- Digikam::ImportCoordinatesOverlay, [1861](#)
  - checkIndex, [1863](#)
  - createWidget, [1863](#)
  - setActive, [1864](#)
  - slotEntered, [1864](#)
  - visualChange, [1864](#)
- Digikam::ImportDelegate, [1865](#)
  - acceptsActivation, [1869](#)
  - acceptsToolTip, [1869](#)
  - clearCaches, [1870](#)
  - imageInformationRect, [1870](#)
  - invalidatePaintingCache, [1870](#)
  - pixmapForDrag, [1870](#)
  - pixmapRect, [1870](#)
  - setDefaultViewOptions, [1870](#)
  - setSpacing, [1871](#)
  - updateContentWidth, [1871](#)
  - updateRects, [1871](#)
  - updateSizeRectsAndPixmaps, [1871](#)
- Digikam::ImportDownloadOverlay, [1872](#)
  - checkIndex, [1874](#)
  - createWidget, [1874](#)
  - setActive, [1875](#)
  - slotEntered, [1875](#)
  - visualChange, [1875](#)
- Digikam::ImportDragDropHandler, [1876](#)
  - accepts, [1877](#)
  - createMimeData, [1877](#)
  - dropEvent, [1877](#)
  - mimeTypes, [1877](#)
- Digikam::ImportFilterComboBox, [1878](#)
- Digikam::ImportFilterDlg, [1879](#)
- Digikam::ImportFilterModel, [1881](#)
  - camItemInfosAdded, [1884](#)
  - CategorizationModeRole, [1884](#)
  - CategoryDateRole, [1884](#)
  - CategoryFormatRole, [1884](#)
  - categoryIdentifier, [1884](#)
  - compareCategories, [1884](#)
  - importFilterModel, [1885](#)
  - ImportFilterModelPointerRole, [1884](#)
  - ImportFilterModelRoles, [1884](#)
  - infosLessThan, [1885](#)
  - setDirectSourceImportModel, [1885](#)
  - SortOrderRole, [1884](#)
  - subSortLessThan, [1885](#)
- Digikam::ImportIconView, [1887](#)
- activated, [1893](#)
- setThumbnailSize, [1893](#)
- showContextMenu, [1894](#)
- showContextMenuOnInfo, [1894](#)
- slotSetupChanged, [1894](#)
- Digikam::ImportItemModel, [1895](#)
  - addCamItemInfoSynchronously, [1899](#)
  - allRefreshingFinished, [1899](#)
  - camItemInfo, [1899](#)
  - ExtraDataDuplicateCount, [1898](#)
  - ExtraDataRole, [1898](#)
  - ImportItemModelPointerRole, [1898](#)
  - ImportItemModelRoles, [1898](#)
  - indexForUrl, [1899](#)
  - isRefreshing, [1899](#)
  - itemInfosAboutToBeAdded, [1899](#)
  - itemInfosAboutToBeRemoved, [1900](#)
  - itemInfosAdded, [1900](#)
  - itemInfosRemoved, [1900](#)
  - readyForIncrementalRefresh, [1900](#)
  - requestIncrementalRefresh, [1900](#)
  - setCameraThumbsController, [1900](#)
  - setKeepsFileUrlCache, [1900](#)
  - setSendRemovalSignals, [1901](#)
  - startIncrementalRefresh, [1901](#)
  - startRefresh, [1901](#)
  - ThumbnailRole, [1898](#)
- Digikam::ImportItemPropertiesSideBarImport, [1902](#)
  - applySettings, [1905](#)
  - doLoadState, [1905](#)
  - doSaveState, [1906](#)
- Digikam::ImportItemPropertiesTab, [1907](#)
- Digikam::ImportLockOverlay, [1909](#)
  - checkIndex, [1911](#)
  - createWidget, [1911](#)
  - setActive, [1912](#)
  - slotEntered, [1912](#)
  - visualChange, [1912](#)
- Digikam::ImportNormalDelegate, [1913](#)
  - updateRects, [1917](#)
- Digikam::ImportOverlayWidget, [1918](#)
- Digikam::ImportPreviewView, [1919](#)
  - acceptsMouseClicked, [1921](#)
- Digikam::ImportRatingOverlay, [1922](#)
  - createWidget, [1925](#)
  - hide, [1925](#)
  - setActive, [1925](#)
  - slotEntered, [1925](#)
  - visualChange, [1925](#)
  - widgetEnterEvent, [1926](#)
  - widgetLeaveEvent, [1926](#)
- Digikam::ImportRenameParser, [1927](#)
- Digikam::ImportRotateOverlay, [1928](#)
  - checkIndex, [1931](#)
  - createButton, [1931](#)
  - setActive, [1931](#)
  - updateButton, [1931](#)
  - widgetEnterEvent, [1932](#)

- widgetLeaveEvent, [1932](#)
- Digikam::ImportRotateOverlayButton, [1933](#)
  - icon, [1935](#)
  - sizeHint, [1935](#)
  - updateToolTip, [1935](#)
- Digikam::ImportSettings, [1936](#)
- Digikam::ImportSortFilterModel, [1939](#)
  - camItemInfosSorted, [1941](#)
  - importFilterModel, [1941](#)
  - mapToSourceImportModel, [1941](#)
  - setDirectSourceImportModel, [1941](#)
- Digikam::ImportStackedView, [1942](#)
  - PreviewCameraMode, [1943](#)
  - StackedViewMode, [1943](#)
- Digikam::ImportThumbnailBar, [1944](#)
  - setModelsFiltered, [1950](#)
  - slotSetupChanged, [1950](#)
- Digikam::ImportThumbnailDelegate, [1951](#)
  - acceptsActivation, [1955](#)
  - setDefaultViewOptions, [1955](#)
  - updateContentWidth, [1956](#)
  - updateRects, [1956](#)
- Digikam::ImportThumbnailModel, [1957](#)
  - data, [1961](#)
  - ImportThumbnailModel, [1961](#)
  - setCameraThumbsController, [1961](#)
  - setData, [1961](#)
  - setEmitDataChanged, [1961](#)
- Digikam::ImportUI, [1962](#)
  - infolface, [1965](#)
- Digikam::ImportView, [1966](#)
- Digikam::InfoDlg, [1968](#)
- Digikam::InfraredContainer, [1969](#)
- Digikam::InfraredFilter, [1970](#)
  - filterAction, [1974](#)
  - filterIdentifier, [1974](#)
  - readParameters, [1974](#)
- Digikam::InitializationObserver, [1975](#)
- Digikam::InsertBookmarksCommand, [1976](#)
- Digikam::InternalTagName, [1977](#)
- Digikam::InvertFilter, [1978](#)
  - filterAction, [1982](#)
  - filterIdentifier, [1982](#)
  - readParameters, [1982](#)
- Digikam::IOFileSettings, [1982](#)
  - JPEGSubSampling, [1983](#)
- Digikam::IOJob, [1983](#)
- Digikam::IOJobData, [1984](#)
- Digikam::IOJobsManager, [1986](#)
  - buildCollectionTrashCounters, [1986](#)
  - instance, [1986](#)
  - startDTrashItemsListingForCollection, [1987](#)
  - startIOJobs, [1987](#)
- Digikam::IOJobsThread, [1988](#)
  - copyOrMove, [1990](#)
  - deleteFiles, [1990](#)
  - emptyDTrashItems, [1990](#)
  - errorsList, [1990](#)
  - hasErrors, [1991](#)
  - isCanceled, [1991](#)
  - jobData, [1991](#)
  - listDTrashItems, [1991](#)
  - renameFile, [1991](#)
  - restoreDTrashItems, [1992](#)
- Digikam::IptcCoreContactInfo, [1992](#)
- Digikam::IptcCoreLocationInfo, [1992](#)
- Digikam::IptcMetaEngineMergeHelper, [1993](#)
- Digikam::IptcWidget, [1994](#)
  - getMetadataTitle, [1996](#)
  - getTagDescription, [1996](#)
  - getTagTitle, [1996](#)
  - loadFromURL, [1996](#)
- Digikam::ItemAlbumFilterModel, [1997](#)
  - compareInfosCategories, [2001](#)
  - setItemFilterSettings, [2001](#)
- Digikam::ItemAlbumModel, [2002](#)
  - openAlbum, [2008](#)
  - slotImageChange, [2008](#)
- Digikam::ItemAttributesWatch, [2008](#)
  - signalFileMetadataChanged, [2009](#)
  - signalImageRatingChanged, [2009](#)
  - signalImagesChanged, [2009](#)
  - signalImageTagsChanged, [2009](#)
- Digikam::ItemCategorizedView, [2010](#)
  - activated, [2016](#)
  - albumAt, [2016](#)
  - dragDropHandler, [2016](#)
  - filterModel, [2016](#)
  - indexActivated, [2016](#)
  - nextIndexHint, [2016](#)
  - nextInOrder, [2017](#)
  - showContextMenuOnIndex, [2017](#)
- Digikam::ItemCategoryDrawer, [2018](#)
  - categoryHeight, [2019](#)
  - drawCategory, [2019](#)
- Digikam::ItemChangeHint, [2020](#)
  - ChangeType, [2020](#)
  - ItemModified, [2021](#)
  - ItemRescan, [2021](#)
- Digikam::ItemComments, [2021](#)
  - addComment, [2023](#)
  - addHeadline, [2023](#)
  - addTitle, [2024](#)
  - apply, [2024](#)
  - changeComment, [2024](#)
  - commentForLanguage, [2024](#)
  - defaultComment, [2024](#)
  - ItemComments, [2023](#)
  - LanguageChoiceBehavior, [2022](#)
  - replaceComments, [2024](#)
  - ReturnMatchingDefaultOrFirstLanguage, [2023](#)
  - ReturnMatchingLanguageOnly, [2023](#)
  - ReturnMatchingOrDefaultLanguage, [2023](#)
  - setUniqueBehavior, [2025](#)
  - type, [2025](#)
  - UniqueBehavior, [2023](#)

- UniquePerLanguage, [2023](#)
- UniquePerLanguageAndAuthor, [2023](#)
- Digikam::ItemCoordinatesOverlay, [2026](#)
  - checkIndex, [2028](#)
  - createWidget, [2028](#)
  - setActive, [2029](#)
  - slotEntered, [2029](#)
  - visualChange, [2029](#)
- Digikam::ItemCopyMoveHint, [2029](#)
  - ItemCopyMoveHint, [2030](#)
- Digikam::ItemCopyright, [2030](#)
  - AddEntryToExisting, [2032](#)
  - contactInfo, [2032](#)
  - copyrightNotice, [2032](#)
  - creator, [2032](#)
  - creatorJobTitle, [2033](#)
  - fillTemplate, [2033](#)
  - instructions, [2033](#)
  - provider, [2033](#)
  - ReplaceAllEntries, [2032](#)
  - ReplaceLanguageEntry, [2032](#)
  - ReplaceMode, [2032](#)
  - rightsUsageTerms, [2033](#)
  - setCopyrightNotice, [2033](#)
  - setCreator, [2034](#)
  - setFromTemplate, [2034](#)
  - source, [2034](#)
- Digikam::ItemDelegate, [2035](#)
  - acceptsActivation, [2039](#)
  - acceptsToolTip, [2039](#)
  - clearCaches, [2040](#)
  - imageInformationRect, [2040](#)
  - invalidatePaintingCache, [2040](#)
  - pixmapForDrag, [2040](#)
  - pixmapRect, [2040](#)
  - setDefaultViewOptions, [2040](#)
  - setSpacing, [2041](#)
  - updateContentWidth, [2041](#)
  - updateRects, [2041](#)
  - updateSizeRectsAndPixmaps, [2041](#)
- Digikam::ItemDelegateOverlay, [2042](#)
  - affectsMultiple, [2043](#)
  - mouseMoved, [2043](#)
  - setActive, [2043](#)
  - visualChange, [2043](#)
- Digikam::ItemDelegateOverlayContainer, [2045](#)
  - asDelegate, [2046](#)
  - ItemDelegateOverlayContainer, [2046](#)
- Digikam::ItemDescEditTab, [2047](#)
- Digikam::ItemDragDropHandler, [2050](#)
  - accepts, [2051](#)
  - createMimeData, [2051](#)
  - dropEvent, [2051](#)
  - mimeTypes, [2052](#)
  - setReadOnlyDrop, [2052](#)
- Digikam::ItemExtendedProperties, [2052](#)
  - intellectualGenre, [2053](#)
  - jobId, [2053](#)
  - location, [2053](#)
  - scene, [2053](#)
  - similarityTo, [2053](#)
  - subjectCode, [2054](#)
- Digikam::ItemFaceDelegate, [2055](#)
  - thumbnailPixmap, [2060](#)
  - updateRects, [2060](#)
- Digikam::ItemFilterModel, [2061](#)
  - CategorizationModeRole, [2065](#)
  - CategoryAlbumIdRole, [2065](#)
  - CategoryDateRole, [2065](#)
  - CategoryFaceRole, [2065](#)
  - CategoryFormatRole, [2065](#)
  - categoryIdentifier, [2065](#)
  - compareCategories, [2065](#)
  - compareInfosCategories, [2066](#)
  - data, [2066](#)
  - filterMatchesForText, [2066](#)
  - GroupsOpenRole, [2065](#)
  - imageFilterModel, [2067](#)
  - infosLessThan, [2067](#)
  - ItemFilterModelRoles, [2065](#)
  - setDayFilter, [2067](#)
  - setDirectSourceItemModel, [2067](#)
  - setItemFilterSettings, [2067](#)
  - SortOrderRole, [2065](#)
  - subSortLessThan, [2067](#)
  - suggestedWatchFlags, [2068](#)
- Digikam::ItemFilterModelFilterer, [2069](#)
  - process, [2071](#)
- Digikam::ItemFilterModelPrepareHook, [2071](#)
- Digikam::ItemFilterModelPreparer, [2072](#)
  - process, [2074](#)
- Digikam::ItemFilterModelWorker, [2075](#)
- Digikam::ItemFilterSettings, [2077](#)
  - matches, [2078](#)
  - watchFlags, [2078](#)
- Digikam::ItemFiltersHistoryItemDelegate, [2079](#)
- Digikam::ItemFiltersHistoryModel, [2080](#)
- Digikam::ItemFiltersHistoryTreeItem, [2081](#)
- Digikam::ItemFullScreenOverlay, [2082](#)
  - checkIndex, [2085](#)
  - createButton, [2085](#)
  - setActive, [2085](#)
  - updateButton, [2085](#)
  - widgetEnterEvent, [2085](#)
  - widgetLeaveEvent, [2086](#)
- Digikam::ItemFullScreenOverlayButton, [2087](#)
  - icon, [2088](#)
  - sizeHint, [2088](#)
  - updateToolTip, [2089](#)
- Digikam::ItemGPS, [2090](#)
  - loadImageData, [2093](#)
  - saveChanges, [2093](#)
- Digikam::ItemGPSModelHelper, [2094](#)
  - bestRepresentativeIndexFromList, [2095](#)
  - itemCoordinates, [2095](#)
  - model, [2096](#)





- Digikam::ItemMetadataAdjustmentHint, 2158
  - AboutToEditMetadata, 2159
  - AdjustmentStatus, 2159
  - MetadataEditingAborted, 2159
  - MetadataEditingFinished, 2159
- Digikam::ItemModel, 2160
  - addItemInfo, 2165
  - addItemInfoSynchronously, 2165
  - allRefreshingFinished, 2165
  - CreationDateRole, 2164
  - ensureHasItemInfo, 2165
  - ExtraDataDuplicateCount, 2164
  - ExtraDataRole, 2164
  - FilterModelRoles, 2164
  - imageInfo, 2165
  - imageInfosAboutToBeAdded, 2165
  - imageInfosAboutToBeRemoved, 2166
  - imageInfosAdded, 2166
  - imageInfosCleared, 2166
  - imageInfosRemoved, 2166
  - indexForPath, 2166
  - isRefreshing, 2166
  - ItemModelPointerRole, 2164
  - ItemModelRoles, 2164
  - LTLeftPanelRole, 2164
  - LTRightPanelRole, 2164
  - readyForIncrementalRefresh, 2167
  - requestIncrementalRefresh, 2167
  - retrieveItemInfo, 2167
  - setKeepsFilePathCache, 2167
  - setPreprocessor, 2167
  - setSendRemovalSignals, 2167
  - setWatchFlags, 2168
  - startIncrementalRefresh, 2168
  - startRefresh, 2168
  - SubclassRoles, 2164
  - ThumbnailRole, 2164
- Digikam::ItemPosition, 2168
  - apply, 2170
  - isEmpty, 2170
  - ItemPosition, 2169
  - latitude, 2170
  - latitudeNumber, 2170
  - latitudeUserPresentableNumbers, 2170
  - remove, 2170
  - setLatitude, 2170, 2171
- Digikam::ItemPreviewCanvas, 2172
- Digikam::ItemPreviewView, 2175
  - acceptsMouseClicked, 2177
- Digikam::ItemPropertiesColorsTab, 2178
- Digikam::ItemPropertiesGPSTab, 2179
- Digikam::ItemPropertiesHistoryTab, 2180
- Digikam::ItemPropertiesMetadataTab, 2181
- Digikam::ItemPropertiesSideBar, 2182
  - doLoadState, 2186
  - doSaveState, 2186
- Digikam::ItemPropertiesSideBarDB, 2187
  - doLoadState, 2192
  - doSaveState, 2192
  - itemChanged, 2192
- Digikam::ItemPropertiesTab, 2193
  - humanReadableBytesCount, 2196
  - shortenedTagPaths, 2196
- Digikam::ItemPropertiesVersionsTab, 2197
- Digikam::ItemQueryBuilder, 2198
  - setImageTagPropertiesJoined, 2198
- Digikam::ItemQueryPostHook, 2198
- Digikam::ItemQueryPostHooks, 2199
  - addHook, 2199
  - checkPosition, 2199
- Digikam::ItemRatingOverlay, 2200
  - createWidget, 2203
  - hide, 2203
  - setActive, 2203
  - slotEntered, 2203
  - visualChange, 2203
  - widgetEnterEvent, 2204
  - widgetLeaveEvent, 2204
- Digikam::ItemRotateOverlay, 2205
  - checkIndex, 2208
  - createButton, 2208
  - setActive, 2208
  - updateButton, 2208
  - widgetEnterEvent, 2208
  - widgetLeaveEvent, 2209
- Digikam::ItemRotateOverlayButton, 2210
  - icon, 2212
  - sizeHint, 2212
  - updateToolTip, 2212
- Digikam::ItemScanInfo, 2212
- Digikam::ItemScanner, 2212
  - commit, 2216
  - copiedFrom, 2216
  - creationDateFromFilesystem, 2216
  - fileModified, 2216
  - fillCommonContainer, 2216
  - fillVideoMetadataContainer, 2216
  - iptcCorePropertyName, 2217
  - itemScanInfo, 2217
  - ItemScanner, 2215
  - loadFromDisk, 2217
  - newFileFullScan, 2217
  - resolvedImageHistory, 2217
  - resolveImageHistory, 2217
  - sameReferredImage, 2218
  - setCategory, 2218
  - sortByProximity, 2218
- Digikam::ItemSelectionOverlay, 2219
  - createButton, 2222
  - setActive, 2222
  - updateButton, 2222
- Digikam::ItemSelectionOverlayButton, 2223
  - icon, 2224
  - sizeHint, 2224
  - updateToolTip, 2225
- Digikam::ItemSelectionPropertiesTab, 2226



- Digikam::ItemShortInfo, [2228](#)
- Digikam::ItemSortCollator, [2228](#)
  - instance, [2229](#)
- Digikam::ItemSortSettings, [2229](#)
  - CategorizationMode, [2230](#)
  - compare, [2231](#)
  - compareCategories, [2231](#)
  - DefaultOrder, [2231](#)
  - lessThan, [2231](#)
  - lessThanByOrder, [2232](#)
  - NoCategories, [2230](#)
  - OneCategory, [2230](#)
  - SortByAspectRatio, [2231](#)
  - SortByFaces, [2231](#)
  - SortByImageSize, [2231](#)
  - SortOrder, [2230](#)
  - SortRole, [2231](#)
  - watchFlags, [2232](#)
- Digikam::ItemTagPair, [2232](#)
  - addProperty, [2233](#)
  - availablePairs, [2233](#)
  - ItemTagPair, [2233](#)
- Digikam::ItemThumbnailBar, [2234](#)
  - hasHiddenGroupedImages, [2240](#)
  - setModelsFiltered, [2240](#)
  - slotSetupChanged, [2241](#)
- Digikam::ItemThumbnailDelegate, [2242](#)
  - acceptsActivation, [2246](#)
  - setDefaultViewOptions, [2246](#)
  - updateContentWidth, [2247](#)
  - updateRects, [2247](#)
- Digikam::ItemThumbnailModel, [2248](#)
  - data, [2253](#)
  - imageInfosCleared, [2253](#)
  - ItemThumbnailModel, [2253](#)
  - preloadThumbnails, [2253](#)
  - setData, [2253](#)
  - setEmitDataChanged, [2253](#)
  - setPreloadThumbnails, [2254](#)
  - setThumbnailLoadThread, [2254](#)
- Digikam::ItemVersionsModel, [2255](#)
- Digikam::ItemViewCategorized, [2256](#)
  - clicked, [2260](#)
  - filterModel, [2260](#)
  - keyPressed, [2260](#)
  - mapIndexForDragDrop, [2260](#)
  - nextIndexHint, [2260](#)
  - pixmapForDrag, [2261](#)
  - rowsRemoved, [2261](#)
  - selectionChanged, [2261](#)
  - setScrollStepGranularity, [2261](#)
  - setSpacing, [2261](#)
  - showContextMenuOnIndex, [2261](#)
  - showToolTip, [2262](#)
- Digikam::ItemViewDelegate, [2263](#)
  - acceptsActivation, [2266](#)
  - acceptsToolTip, [2266](#)
  - asDelegate, [2266](#)
  - gridSize, [2267](#)
  - imageInformationRect, [2267](#)
  - mouseMoved, [2267](#)
  - pixmapRect, [2267](#)
  - setDefaultViewOptions, [2267](#)
  - setRatingEdited, [2267](#)
  - setSpacing, [2268](#)
  - setThumbnailSize, [2268](#)
- Digikam::ItemViewHoverButton, [2268](#)
  - icon, [2269](#)
  - sizeHint, [2269](#)
  - updateToolTip, [2269](#)
- Digikam::ItemViewImportDelegate, [2271](#)
  - acceptsActivation, [2274](#)
  - acceptsToolTip, [2274](#)
  - asDelegate, [2274](#)
  - gridSize, [2274](#)
  - imageInformationRect, [2275](#)
  - invalidatePaintingCache, [2275](#)
  - mouseMoved, [2275](#)
  - pixmapRect, [2275](#)
  - prepareRatingPixmaps, [2275](#)
  - setDefaultViewOptions, [2275](#)
  - setRatingEdited, [2276](#)
  - setSpacing, [2276](#)
  - setThumbnailSize, [2276](#)
- Digikam::ItemViewToolTip, [2277](#)
  - repositionRect, [2278](#)
  - show, [2278](#)
  - tipContents, [2278](#)
- Digikam::ItemViewUtilities, [2279](#)
- Digikam::ItemVisibilityController, [2281](#)
  - addItem, [2283](#)
  - createAnimation, [2284](#)
  - ExcludeFadingOut, [2283](#)
  - hasVisibleItems, [2284](#)
  - hideAndRemoveItem, [2284](#)
  - IncludeFadingOut, [2283](#)
  - IncludeFadingOutMode, [2283](#)
  - setItemThatShallBeShown, [2284](#)
  - show, [2284](#)
  - showItem, [2284](#)
  - State, [2283](#)
- Digikam::ItemVisibilityControllerPropertyObject, [2285](#)
  - ItemVisibilityControllerPropertyObject, [2286](#)
- Digikam::JPEGUtils::digikam\_source\_mgr, [2286](#)
- Digikam::JPEGUtils::JpegRotator, [2286](#)
  - autoExifTransform, [2287](#)
  - exifTransform, [2287](#)
  - JpegRotator, [2287](#)
  - setCurrentOrientation, [2287](#)
  - setDestinationFile, [2288](#)
  - setDocumentName, [2288](#)
- Digikam::KDNodeBase, [2289](#)
  - createNode, [2290](#)
- Digikam::KDNodeBase::NodeCompareResult, [2290](#)
- Digikam::KDNodeOpenFace, [2291](#)
  - createNode, [2292](#)

- nodeCompare, [2292](#)
- Digikam::KDNodeSFace, [2293](#)
  - createNode, [2294](#)
  - nodeCompare, [2294](#)
- Digikam::KDTreeBase, [2295](#)
  - add, [2296](#)
  - createNode, [2296](#)
  - getClosestNeighbors, [2296](#)
  - KDTreeBase, [2295](#)
- Digikam::KDTreeOpenFace, [2297](#)
- Digikam::KDTreeSFace, [2298](#)
- Digikam::KeywordSearchReader, [2299](#)
- Digikam::KeywordSearchWriter, [2301](#)
- Digikam::LabelsSideBarWidget, [2303](#)
  - applySettings, [2305](#)
  - changeAlbumFromHistory, [2305](#)
  - doLoadState, [2305](#)
  - doSaveState, [2305](#)
  - getCaption, [2305](#)
  - getIcon, [2306](#)
  - setActive, [2306](#)
- Digikam::LabelsTreeView, [2307](#)
  - colorRectPixmap, [2309](#)
  - doLoadState, [2309](#)
  - doSaveState, [2309](#)
  - goldenStarPixmap, [2309](#)
  - isCheckedable, [2309](#)
  - isLoadingState, [2309](#)
  - restoreSelectionFromHistory, [2310](#)
  - selectedLabels, [2310](#)
- Digikam::LanguagesList, [2311](#)
- Digikam::LcmsLock, [2311](#)
- Digikam::LensDistortionFilter, [2312](#)
  - filterAction, [2316](#)
  - filterIdentifier, [2316](#)
  - readParameters, [2316](#)
- Digikam::LensDistortionPixelAccess, [2316](#)
- Digikam::LensFunCameraSelector, [2317](#)
- Digikam::LensFunContainer, [2318](#)
- Digikam::LensFunFilter, [2319](#)
  - filterAction, [2323](#)
  - filterIdentifier, [2323](#)
  - readParameters, [2323](#)
- Digikam::LensFunIface, [2323](#)
- Digikam::LensFunSettings, [2324](#)
- Digikam::LevelsContainer, [2325](#)
- Digikam::LevelsFilter, [2326](#)
  - filterAction, [2330](#)
  - filterIdentifier, [2330](#)
  - readParameters, [2330](#)
- Digikam::LibsInfoDlg, [2331](#)
  - LibsInfoDlg, [2332](#)
- Digikam::LightTablePreview, [2333](#)
- Digikam::LightTableThumbBar, [2337](#)
- Digikam::LightTableView, [2345](#)
- Digikam::LightTableWindow, [2347](#)
  - infoIface, [2350](#)
  - loadItemInfos, [2350](#)
  - slotApplicationSettingsChanged, [2350](#)
- Digikam::ListItem, [2351](#)
  - containsItem, [2352](#)
- Digikam::ListViewComboBox, [2353](#)
  - installView, [2355](#)
  - ListViewComboBox, [2355](#)
  - sendViewportEventToView, [2355](#)
  - view, [2355](#)
- Digikam::LoadingCache, [2356](#)
  - addLoadingProcess, [2358](#)
  - fileChanged, [2358](#)
  - notifyFileChanged, [2358](#)
  - putImage, [2358](#)
  - retrieveThumbnail, [2358](#)
  - setCacheSize, [2358](#)
  - setFileWatch, [2358](#)
  - setThumbnailCacheSize, [2359](#)
- Digikam::LoadingCache::CacheLock, [2359](#)
- Digikam::LoadingCacheFileWatch, [2360](#)
  - notifyFileChanged, [2361](#)
- Digikam::LoadingCacheInterface, [2361](#)
  - cleanCache, [2361](#)
  - cleanThumbnailCache, [2361](#)
  - setCacheOptions, [2361](#)
- Digikam::LoadingDescription, [2362](#)
  - ApplyTransform, [2363](#)
  - ColorManagementSettings, [2363](#)
  - ConvertForDisplay, [2363](#)
  - ConvertForOutput, [2363](#)
  - LoadingDescription, [2364](#)
  - lookupCacheKeys, [2364](#)
  - needCheckRawDecoding, [2364](#)
  - RawDecodingCustomSettings, [2363](#)
  - RawDecodingDefaultSettings, [2363](#)
  - RawDecodingGlobalSettings, [2363](#)
  - RawDecodingHint, [2363](#)
  - RawDecodingTimeOptimized, [2363](#)
- Digikam::LoadingDescription::PostProcessingParameters, [2364](#)
- Digikam::LoadingDescription::PreviewParameters, [2365](#)
- Digikam::LoadingProcess, [2366](#)
- Digikam::LoadingProcessListener, [2367](#)
- Digikam::LoadingTask, [2368](#)
  - continueQuery, [2369](#)
  - execute, [2369](#)
  - progressInfo, [2370](#)
  - type, [2370](#)
- Digikam::LoadSaveFileInfoProvider, [2370](#)
  - dimensionsHint, [2371](#)
  - orientationHint, [2371](#)
- Digikam::LoadSaveNotifier, [2372](#)
  - thumbnailLoaded, [2373](#)
- Digikam::LoadSaveTask, [2374](#)
- Digikam::LoadSaveThread, [2376](#)
  - AccessMode, [2379](#)
  - AccessModeRead, [2379](#)
  - AccessModeReadWrite, [2379](#)
  - imageLoaded, [2379](#)

- imageSaved, [2379](#)
- imageStartedLoading, [2379](#)
- imageStartedSaving, [2380](#)
- loadingProgress, [2380](#)
- moreCompleteLoadingAvailable, [2380](#)
- NotificationPolicy, [2379](#)
- NotificationPolicyDirect, [2379](#)
- NotificationPolicyTimeLimited, [2379](#)
- run, [2380](#)
- savingProgress, [2380](#)
- signalImageLoaded, [2380](#)
- signalImageStartedLoading, [2381](#)
- signalLoadingProgress, [2381](#)
- signalMoreCompleteLoadingAvailable, [2381](#)
- thumbnailLoaded, [2381](#)
- Digikam::LocalContrastContainer, [2382](#)
- Digikam::LocalContrastFilter, [2383](#)
  - filterAction, [2387](#)
  - filterIdentifier, [2387](#)
  - readParameters, [2387](#)
- Digikam::LocalContrastSettings, [2387](#)
- Digikam::LocalizeConfig, [2388](#)
- Digikam::LocalizeContainer, [2389](#)
  - ignoredWords, [2389](#)
- Digikam::LocalizeSelector, [2390](#)
- Digikam::LocalizeSelectorList, [2391](#)
- Digikam::LocalizeSettings, [2392](#)
  - instance, [2393](#)
- Digikam::LookupAltitude, [2394](#)
- Digikam::LookupAltitude::Request, [2395](#)
- Digikam::LookupAltitudeGeonames, [2396](#)
  - backendHumanName, [2397](#)
  - backendName, [2397](#)
  - cancel, [2397](#)
  - errorMessage, [2397](#)
  - getRequest, [2398](#)
  - getRequests, [2398](#)
  - getStatus, [2398](#)
  - startLookup, [2398](#)
- Digikam::LookupFactory, [2398](#)
- Digikam::MaintenanceData, [2399](#)
- Digikam::MaintenanceDlg, [2399](#)
- Digikam::MaintenanceMngr, [2400](#)
- Digikam::MaintenanceSettings, [2400](#)
  - qualityScanMode, [2402](#)
- Digikam::MaintenanceThread, [2403](#)
  - signalAdvance, [2405](#)
- Digikam::MaintenanceTool, [2405](#)
  - setUseMultiCoreCPU, [2407](#)
- Digikam::MakerNoteWidget, [2408](#)
  - getMetadataTitle, [2410](#)
  - getTagDescription, [2410](#)
  - getTagTitle, [2410](#)
  - loadFromURL, [2410](#)
- Digikam::ManagedLoadSaveThread, [2411](#)
  - load, [2416](#)
  - LoadingMode, [2415](#)
  - LoadingModeNormal, [2415](#)
  - LoadingModeShared, [2415](#)
  - LoadingPolicy, [2415](#)
  - LoadingPolicyAppend, [2416](#)
  - LoadingPolicyFirstRemovePrevious, [2416](#)
  - LoadingPolicyPreload, [2416](#)
  - LoadingPolicyPrepend, [2416](#)
  - LoadingPolicySimpleAppend, [2416](#)
  - LoadingPolicySimplePrepend, [2416](#)
  - LoadingTaskFilter, [2416](#)
  - LoadingTaskFilterAll, [2416](#)
  - LoadingTaskFilterPreloading, [2416](#)
  - setLoadingPolicy, [2416](#)
  - stopLoading, [2417](#)
  - stopSaving, [2417](#)
  - TerminationPolicy, [2416](#)
  - TerminationPolicyTerminateAll, [2416](#)
  - TerminationPolicyTerminateLoading, [2416](#)
  - TerminationPolicyTerminatePreloading, [2416](#)
  - TerminationPolicyWait, [2416](#)
- Digikam::MapBackend, [2418](#)
  - centerOn, [2420](#)
  - mapWidget, [2420](#)
  - mouseModeChanged, [2420](#)
  - setActive, [2420](#)
- Digikam::MapDragData, [2420](#)
- Digikam::MapDragDropHandler, [2421](#)
  - accepts, [2422](#)
  - createMimeData, [2422](#)
  - dropEvent, [2422](#)
- Digikam::MapViewModelHelper, [2423](#)
  - bestRepresentativeIndexFromList, [2424](#)
  - itemCoordinates, [2425](#)
  - model, [2425](#)
  - onIndicesClicked, [2425](#)
  - pixmapFromRepresentativeIndex, [2426](#)
  - selectionModel, [2426](#)
- Digikam::MapWidget, [2426](#)
  - ~MapWidget, [2430](#)
  - addUngroupedModel, [2430](#)
  - adjustBoundariesToGroupedMarkers, [2430](#)
  - applyCacheToBackend, [2431](#)
  - convertZoomToBackendZoom, [2431](#)
  - dragEnterEvent, [2431](#)
  - getColorInfos, [2431](#)
  - getDecoratedPixmapForCluster, [2432](#)
  - removeUngroupedModel, [2432](#)
  - setBackend, [2432](#)
  - setGroupedModel, [2432](#)
  - setSortKey, [2432](#)
  - setThumbnailSize, [2432](#)
  - slotClustersClicked, [2432](#)
  - slotClustersMoved, [2432](#)
  - slotItemDisplaySettingsChanged, [2432](#)
  - slotMouseModeChanged, [2433](#)
  - slotNewSelectionFromMap, [2433](#)
  - slotUpdateActionsEnabled, [2433](#)
  - updateClusters, [2433](#)
- Digikam::MapWidgetView, [2433](#)

- currentCamItemInfo, [2436](#)
- currentItemInfo, [2436](#)
- doLoadState, [2436](#)
- doSaveState, [2436](#)
- getActiveState, [2436](#)
- MapViewWidgetView, [2436](#)
- setActive, [2436](#)
- Digikam::Mat, [2437](#)
- Digikam::Matrix, [145](#)
- Digikam::MdKeyListViewItem, [2437](#)
- Digikam::MediaPlayerView, [2438](#)
- Digikam::MetadataHub, [2439](#)
  - cleanupTags, [2441](#)
  - FullWrite, [2440](#)
  - FullWriteIfChanged, [2440](#)
  - load, [2441](#)
  - MetadataAvailable, [2440](#)
  - MetadataInvalid, [2440](#)
  - PartialWrite, [2440](#)
  - Status, [2440](#)
  - willWriteMetadata, [2441](#)
  - write, [2441](#), [2442](#)
  - WriteMode, [2440](#)
  - writeTags, [2443](#)
  - writeToBaloo, [2444](#)
  - writeToMetadata, [2444](#)
- Digikam::MetadataHubMgr, [2445](#)
- Digikam::MetadataKeys, [2446](#)
  - getDbValue, [2447](#)
- Digikam::MetadataListView, [2448](#)
- Digikam::MetadataListViewItem, [2449](#)
- Digikam::MetadataOption, [2450](#)
  - parseOperation, [2452](#)
- Digikam::MetadataOptionDialog, [2453](#)
- Digikam::MetadataPage, [2454](#)
- Digikam::MetadataPanel, [2455](#)
- Digikam::MetadataRemover, [2457](#)
  - MetadataRemover, [2460](#)
  - setUseMultiCoreCPU, [2460](#)
- Digikam::MetadataRemoveTask, [2461](#)
- Digikam::MetadataSelector, [2463](#)
- Digikam::MetadataSelectorItem, [2464](#)
- Digikam::MetadataSelectorView, [2465](#)
- Digikam::MetadataStatusBar, [2466](#)
- Digikam::MetadataSynchronizer, [2467](#)
  - MetadataSynchronizer, [2470](#)
  - setUseMultiCoreCPU, [2470](#)
- Digikam::MetadataSyncTask, [2471](#)
- Digikam::MetadataWidget, [2473](#)
- Digikam::MetaEngine, [2475](#)
  - addToXmpTagStringBag, [2485](#)
  - AltLangMap, [2484](#)
  - applyChanges, [2485](#)
  - Backend, [2484](#)
  - backendName, [2485](#)
  - convertDegreeAngleToDouble, [2485](#)
  - convertFromGPSCoordinateString, [2485](#), [2486](#)
  - convertToGPSCoordinateString, [2486](#)
  - convertToRational, [2486](#)
  - convertToRationalSmallDenominator, [2486](#)
  - convertToUserPresentableNumbers, [2487](#)
  - createExifUserStringFromValue, [2487](#)
  - detectLanguageAlt, [2487](#)
  - ExifToolBackend, [2484](#)
  - Exiv2Backend, [2484](#)
  - exportChanges, [2487](#)
  - FFMpegBackend, [2484](#)
  - getComments, [2487](#)
  - getCommentsDecoded, [2488](#)
  - getDigitizationDateTime, [2488](#)
  - getExifComment, [2488](#)
  - getExifEncoded, [2488](#)
  - getExifTagComment, [2488](#)
  - getExifTagData, [2488](#)
  - getExifTagLong, [2489](#)
  - getExifTagRational, [2489](#)
  - getExifTagsDataList, [2489](#)
  - getExifTagString, [2489](#)
  - getExifTagVariant, [2490](#)
  - getExifThumbnail, [2490](#)
  - getGPSInfo, [2490](#)
  - getGPSLatitudeNumber, [2490](#)
  - getGPSLatitudeString, [2490](#)
  - getIptc, [2490](#)
  - getIptcKeywords, [2491](#)
  - getIptcSubCategories, [2491](#)
  - getIptcSubjects, [2491](#)
  - getIptcTagData, [2491](#)
  - getIptcTagsDataList, [2491](#)
  - getIptcTagsStringList, [2491](#)
  - getIptcTagString, [2492](#)
  - getItemColorWorkSpace, [2492](#)
  - getItemDateTime, [2492](#)
  - getItemDimensions, [2492](#)
  - getItemOrientation, [2492](#)
  - getItemPreview, [2492](#)
  - getMimeType, [2492](#)
  - getPixelSize, [2493](#)
  - getXmp, [2493](#)
  - getXmpKeywords, [2493](#)
  - getXmpSubCategories, [2493](#)
  - getXmpSubjects, [2493](#)
  - getXmpTagsDataList, [2493](#)
  - getXmpTagString, [2494](#)
  - getXmpTagStringBag, [2494](#)
  - getXmpTagStringLangAlt, [2494](#)
  - getXmpTagStringListLangAlt, [2494](#)
  - getXmpTagStringSeq, [2494](#)
  - getXmpTagVariant, [2495](#)
  - ImageMagickBackend, [2484](#)
  - initializeExiv2, [2495](#)
  - LibHeifBackend, [2484](#)
  - LibRawBackend, [2484](#)
  - load, [2495](#)
  - loadFromData, [2495](#)
  - loadFromDataAndMerge, [2495](#)

- loadFromSidecarAndMerge, [2496](#)
- MetadataWritingMode, [2485](#)
- metadataWritingMode, [2496](#)
- NoBackend, [2484](#)
- registerXmpNameSpace, [2496](#)
- removeExifTag, [2496](#)
- removeFromXmpTagStringBag, [2496](#)
- removeGPSInfo, [2497](#)
- removeIptcTag, [2497](#)
- removeXmpKeywords, [2497](#)
- removeXmpSubCategories, [2497](#)
- removeXmpSubjects, [2497](#)
- removeXmpTag, [2497](#)
- rotateExifQImage, [2498](#)
- save, [2498](#)
- setComments, [2498](#)
- setExif, [2498](#)
- setExifComment, [2498](#)
- setExifTagData, [2498](#)
- setExifTagLong, [2499](#)
- setExifTagRational, [2499](#)
- setExifTagString, [2499](#)
- setExifTagURational, [2499](#)
- setExifTagUShort, [2499](#)
- setExifTagVariant, [2499](#)
- setExifThumbnail, [2500](#)
- setGPSInfo, [2500](#)
- setImageDateTime, [2500](#)
- setIptc, [2501](#)
- setIptcKeywords, [2501](#)
- setIptcSubCategories, [2501](#)
- setIptcSubjects, [2501](#)
- setIptcTagData, [2501](#)
- setIptcTagsStringList, [2501](#)
- setIptcTagString, [2502](#)
- setItemColorWorkSpace, [2502](#)
- setItemDimensions, [2502](#)
- setItemOrientation, [2502](#)
- setItemPreview, [2502](#)
- setItemProgramId, [2502](#)
- setMetadataWritingMode, [2503](#)
- setTiffThumbnail, [2503](#)
- setUpdateFileTimeStamp, [2503](#)
- setWriteRawFiles, [2503](#)
- setXmp, [2503](#)
- setXmpKeywords, [2503](#)
- setXmpSubCategories, [2504](#)
- setXmpSubjects, [2504](#)
- setXmpTagString, [2504](#)
- setXmpTagStringBag, [2504](#)
- setXmpTagStringLangAlt, [2504](#)
- setXmpTagStringListLangAlt, [2505](#)
- setXmpTagStringSeq, [2505](#)
- sidecarFilePathForFile, [2505](#)
- supportBmff, [2505](#)
- TagsMap, [2484](#)
- VideoMergeBackend, [2484](#)
- WRITE\_TO\_FILE\_ONLY, [2485](#)
- WRITE\_TO\_SIDECAR\_AND\_FILE, [2485](#)
- WRITE\_TO\_SIDECAR\_ONLY, [2485](#)
- WRITE\_TO\_SIDECAR\_ONLY\_FOR\_READ\_ONLY\_FILES, [2485](#)
- Digikam::MetaEngineData, [2505](#)
- Digikam::MetaEngineMergeHelper< Data, Key, KeyString, KeyStringList >, [2506](#)
  - exclusiveMerge, [2506](#)
  - mergeFields, [2506](#)
- Digikam::MetaEnginePreviews, [2507](#)
  - dataSize, [2507](#)
  - image, [2507](#)
- Digikam::MetaEngineRotation, [2508](#)
  - exifOrientation, [2510](#)
  - FlipHorizontal, [2510](#)
  - FlipVertical, [2510](#)
  - NoTransformation, [2510](#)
  - Rotate180, [2510](#)
  - Rotate270, [2510](#)
  - Rotate90, [2510](#)
  - TransformationAction, [2509](#)
  - transformations, [2510](#)
- Digikam::MetaEngineSettings, [2510](#)
  - instance, [2511](#)
- Digikam::MetaEngineSettingsContainer, [2511](#)
  - RotationBehaviorFlag, [2512](#)
- Digikam::MigrateFromDigikam4Page, [2513](#)
- Digikam::MimeFilter, [2514](#)
  - HEIFFiles, [2515](#)
  - RasterGraphics, [2515](#)
  - RAWFiles, [2515](#)
  - TypeMimeFilter, [2515](#)
- Digikam::MixerContainer, [2515](#)
- Digikam::MixerFilter, [2516](#)
  - filterAction, [2520](#)
  - filterIdentifier, [2520](#)
  - readParameters, [2520](#)
- Digikam::MixerSettings, [2520](#)
- Digikam::MLClassifierFoundation, [2522](#)
- Digikam::MLClassifierFoundation::VotingGroups, [2523](#)
- Digikam::MLClassifierFoundation::VotingGroups::VoteTally, [2523](#)
- Digikam::MLPipelineFoundation, [2524](#)
  - cancel, [2526](#)
  - Classifier, [2526](#)
  - Extractor, [2526](#)
  - Finder, [2526](#)
  - Loader, [2526](#)
  - MLPipelineStage, [2526](#)
  - None, [2526](#)
  - Trainer, [2526](#)
  - Writer, [2526](#)
- Digikam::MLPipelineFoundation::\_MLPipelinePerformanceProfile, [2527](#)
- Digikam::MLPipelinePackageFoundation, [2528](#)
- Digikam::MLPipelinePackageNotify, [2529](#)
- Digikam::ModelCompleter, [2530](#)
  - setItemModel, [2530](#)

- Digikam::ModelIndexBasedComboBox, [2532](#)
  - ModelIndexBasedComboBox, [2533](#)
- Digikam::ModelMenu, [2533](#)
  - prePopulated, [2535](#)
- Digikam::Modifier, [2535](#)
  - parseOperation, [2537](#)
- Digikam::MonthWidget, [2538](#)
- Digikam::MysqlAdminBinary, [2539](#)
- Digikam::MysqlInitBinary, [2542](#)
- Digikam::MysqlServerBinary, [2545](#)
- Digikam::MysqlUpgradeBinary, [2548](#)
- Digikam::NamespaceEditDlg, [2550](#)
- Digikam::NamespaceEntry, [2551](#)
- Digikam::NamespaceListView, [2552](#)
- Digikam::NetworkManager, [2553](#)
  - instance, [2554](#)
- Digikam::NewItemFinder, [2555](#)
  - CompleteCollectionScan, [2558](#)
  - FinderMode, [2557](#)
  - ScanDeferredFiles, [2558](#)
  - ScheduleCollectionScan, [2558](#)
- Digikam::NoDuplicatesImportFilterModel, [2559](#)
- Digikam::NoDuplicatesItemFilterModel, [2562](#)
- Digikam::NoiseDetector, [2564](#)
  - detect, [2565](#)
- Digikam::NonDeterministicRandomData, [2565](#)
  - NonDeterministicRandomData, [2566](#)
- Digikam::NormalizeFilter, [2567](#)
  - filterAction, [2571](#)
  - filterIdentifier, [2571](#)
  - readParameters, [2571](#)
- Digikam::NormalSearchTreeView, [2571](#)
  - addCustomContextMenuActions, [2578](#)
  - copySearch, [2578](#)
  - editSearch, [2578](#)
  - handleCustomContextMenuAction, [2578](#)
  - NormalSearchTreeView, [2577](#)
- Digikam::NRContainer, [2578](#)
  - thresholds, [2579](#)
- Digikam::NREstimate, [2580](#)
  - setLogFilesPath, [2584](#)
  - startAnalyse, [2584](#)
- Digikam::NRFilter, [2585](#)
  - filterAction, [2589](#)
  - filterIdentifier, [2589](#)
  - readParameters, [2589](#)
- Digikam::NRSettings, [2589](#)
- Digikam::OilPaintFilter, [2591](#)
  - filterAction, [2595](#)
  - filterIdentifier, [2595](#)
  - readParameters, [2595](#)
- Digikam::OnlineVersionChecker, [2595](#)
  - bundleProperties, [2596](#)
- Digikam::OnlineVersionDlg, [2597](#)
- Digikam::OnlineVersionDwnl, [2598](#)
- Digikam::OpenCVDNNFaceDetector, [2598](#)
  - detectFaces, [2599](#)
  - recommendedImageSizeForDetection, [2599](#)
- Digikam::OpenCVDNNFaceRecognizer, [2600](#)
  - Classifier, [2600](#)
  - DB, [2601](#)
  - OpenCV\_KNN, [2601](#)
  - recognize, [2601](#)
  - SVM, [2601](#)
  - Tree, [2601](#)
- Digikam::OpenfacePreprocessor, [2601](#)
- Digikam::OpenFilePage, [2602](#)
- Digikam::Option, [2603](#)
  - parseOperation, [2604](#)
- Digikam::OverlayWidget, [2605](#)
- Digikam::PackageLoadingDescriptionList, [2607](#)
- Digikam::PALbum, [2608](#)
  - databaseUrl, [2610](#)
- Digikam::PanIconFrame, [2611](#)
  - close, [2612](#)
  - resizeEvent, [2612](#)
  - setMainWidget, [2612](#)
- Digikam::PanIconWidget, [2613](#)
  - signalSelectionMoved, [2614](#)
- Digikam::ParallelAdapter< A >, [2615](#)
  - asQObject, [2617](#)
  - mocMetaObject, [2617](#)
  - ParallelAdapter, [2617](#)
  - staticMetacallPointer, [2617](#)
  - WorkerObjectQtMetacall, [2617](#)
- Digikam::ParallelPipes, [2618](#)
- Digikam::ParallelWorkers, [2620](#)
  - asQObject, [2622](#)
  - mocMetaObject, [2622](#)
  - ParallelWorkers, [2621](#)
  - WorkerObjectQtMetacall, [2622](#)
- Digikam::Parser, [2623](#)
  - parseStringsValid, [2624](#)
- Digikam::ParseResults, [2624](#)
- Digikam::ParseSettings, [2625](#)
- Digikam::PeopleSideBarWidget, [2626](#)
  - applySettings, [2628](#)
  - changeAlbumFromHistory, [2628](#)
  - doLoadState, [2628](#)
  - doSaveState, [2628](#)
  - getCaption, [2628](#)
  - getIcon, [2629](#)
  - setActive, [2629](#)
- Digikam::PersistentWidgetDelegateOverlay, [2630](#)
  - hide, [2633](#)
  - PersistentWidgetDelegateOverlay, [2633](#)
  - setActive, [2633](#)
  - setFocusOnWidget, [2633](#)
  - setPersistent, [2633](#)
  - showOnIndex, [2633](#)
  - slotEntered, [2634](#)
  - slotLayoutChanged, [2634](#)
  - slotReset, [2634](#)
  - slotRowsRemoved, [2634](#)
  - slotViewportEntered, [2634](#)
  - viewportLeaveEvent, [2634](#)



- Digikam::PhotoInfoContainer, 2635
- Digikam::PickLabelFilter, 2636
- Digikam::PickLabelMenuAction, 2638
- Digikam::PickLabelSelector, 2639
- Digikam::PickLabelWidget, 2640
  - setButtonsExclusive, 2641
  - setPickLabels, 2641
- Digikam::PlaceholderWidget, 2642
- Digikam::PointTransformAffine, 2642
- Digikam::PositionKeys, 2643
  - getDbValue, 2644
  - PositionKeys, 2644
- Digikam::PreviewList, 2645
- Digikam::PreviewListItem, 2646
- Digikam::PreviewLoadingTask, 2647
  - execute, 2649
- Digikam::PreviewLoadThread, 2650
  - load, 2655
  - loadFast, 2655
  - loadFastButLarge, 2655
  - loadFastSynchronously, 2655
  - loadHighQuality, 2656
  - PreviewLoadThread, 2655
- Digikam::PreviewPage, 2657
- Digikam::PreviewSettings, 2658
  - FastButLargePreview, 2658
  - FastPreview, 2658
  - HighQualityPreview, 2658
  - Quality, 2658
- Digikam::PreviewThreadWrapper, 2659
- Digikam::PreviewToolBar, 2660
  - NoPreviewMode, 2661
  - PreviewBothImagesHorz, 2661
  - PreviewBothImagesHorzCont, 2661
  - PreviewBothImagesVert, 2661
  - PreviewBothImagesVertCont, 2661
  - PreviewMode, 2661
  - PreviewOriginalImage, 2661
  - PreviewTargetImage, 2661
  - PreviewToggleOnMouseOver, 2661
- Digikam::ProcessLauncher, 2662
- Digikam::ProgressItem, 2663
  - advance, 2665
  - canBeCanceled, 2665
  - hasThumbnail, 2665
  - id, 2665
  - label, 2665
  - parent, 2665
  - progress, 2666
  - progressItemAdded, 2666
  - progressItemCanceled, 2666
  - progressItemCompleted, 2666
  - progressItemLabel, 2667
  - progressItemProgress, 2667
  - progressItemStatus, 2667
  - progressItemThumbnail, 2667
  - progressItemUsesBusyIndicator, 2668
  - setComplete, 2668
  - setLabel, 2668
  - setProgress, 2668
  - setShowAtStart, 2668
  - setStatus, 2669
  - setThumbnail, 2669
  - setUsesBusyIndicator, 2669
  - showAtStart, 2669
  - status, 2669
  - usesBusyIndicator, 2670
- Digikam::ProgressManager, 2670
  - addProgressItem, 2672
  - createProgressItem, 2672–2674
  - findItemById, 2674
  - getUniqueId, 2674
  - instance, 2675
  - isEmpty, 2675
  - progressItemAdded, 2675
  - progressItemCanceled, 2675
  - progressItemCompleted, 2675
  - progressItemLabel, 2676
  - progressItemProgress, 2676
  - progressItemStatus, 2676
  - progressItemThumbnail, 2676
  - progressItemUsesBusyIndicator, 2676
  - showProgressView, 2677
  - singleItem, 2677
  - slotAbortAll, 2677
  - slotStandardCancelHandler, 2677
- Digikam::ProgressView, 2678
- Digikam::ProxyClickLineEdit, 2680
  - ProxyClickLineEdit, 2682
- Digikam::ProxyLineEdit, 2683
  - ProxyLineEdit, 2684
- Digikam::QListImageListProvider, 2685
  - atEnd, 2686
  - image, 2686
  - images, 2686
  - proceed, 2686
  - setImages, 2686
  - setUnpairedImages, 2686
  - size, 2686
- Digikam::QMapForAdaptors< Key, Value >, 2687
- Digikam::QueueListView, 2688
  - All, 2689
  - ItemListType, 2689
  - Pending, 2689
  - Selected, 2689
- Digikam::QueueListViewItem, 2690
- Digikam::QueueMgrWindow, 2691
  - infolface, 2694
- Digikam::QueuePool, 2695
- Digikam::QueuePoolBar, 2697
- Digikam::QueueSettings, 2697
- Digikam::QueueSettingsView, 2698
- Digikam::QueueToolTip, 2699
- Digikam::RainDropFilter, 2701
  - filterAction, 2705
  - filterIdentifier, 2705

- readParameters, [2705](#)
- Digikam::RandomNumberGenerator, [2705](#)
  - currentSeed, [2706](#)
  - number, [2706](#)
  - RandomNumberGenerator, [2706](#)
  - reseed, [2706](#)
  - seed, [2706](#)
  - seedByTime, [2707](#)
  - seedNonDeterministic, [2707](#)
- Digikam::RangeDialog, [2707](#)
- Digikam::RangeModifier, [2709](#)
  - parseOperation, [2711](#)
- Digikam::RatingBox, [2712](#)
- Digikam::RatingComboBox, [2714](#)
  - Null, [2715](#)
  - RatingValue, [2715](#)
- Digikam::RatingComboBoxDelegate, [2716](#)
- Digikam::RatingComboBoxModel, [2717](#)
- Digikam::RatingComboBoxWidget, [2718](#)
- Digikam::RatingFilter, [2721](#)
- Digikam::RatingFilterWidget, [2723](#)
- Digikam::RatingMenuAction, [2725](#)
- Digikam::RatingStarDrawer, [2726](#)
- Digikam::RatingWidget, [2727](#)
- Digikam::RawCameraDlg, [2729](#)
- Digikam::RawPage, [2730](#)
- Digikam::RawProcessingFilter, [2731](#)
  - filterAction, [2737](#)
  - filterIdentifier, [2737](#)
  - filterImage, [2737](#)
  - RawProcessingFilter, [2736](#)
  - readParameters, [2737](#)
  - setObserver, [2737](#)
  - setSettings, [2737](#)
- Digikam::RecognitionBenchmark, [2738](#)
  - result, [2740](#)
- Digikam::RecognitionBenchmark::Statistics, [2740](#)
- Digikam::RecognitionPreprocessor, [2741](#)
  - preprocess, [2741](#)
- Digikam::RecognitionTrainingProvider, [2741](#)
  - images, [2742](#)
  - newImages, [2742](#)
- Digikam::RecognitionTrainingUpdateQueue, [2743](#)
- Digikam::RecognitionWorker, [2744](#)
  - aboutToDeactivate, [2746](#)
- Digikam::RedEye::RegressionTree, [2747](#)
  - operator(), [2747](#)
- Digikam::RedEye::ShapePredictor, [2747](#)
- Digikam::RedEye::SplitFeature, [2748](#)
- Digikam::RedEyeCorrectionContainer, [2748](#)
- Digikam::RedEyeCorrectionFilter, [2749](#)
  - filterAction, [2753](#)
  - filterIdentifier, [2753](#)
- Digikam::RedEyeCorrectionSettings, [2753](#)
- Digikam::RefocusFilter, [2755](#)
  - filterAction, [2759](#)
  - filterIdentifier, [2759](#)
  - readParameters, [2759](#)
- Digikam::RefocusMatrix, [2759](#)
- Digikam::RegionFrameItem, [2760](#)
  - setHudWidget, [2763](#)
  - setViewportRect, [2763](#)
- Digikam::RemoveBookmarksCommand, [2764](#)
- Digikam::RemoveDoublesModifier, [2765](#)
  - parseOperation, [2767](#)
- Digikam::RemoveFilterAction, [2768](#)
- Digikam::RenameCustomizer, [2769](#)
- Digikam::RenameFileJob, [2770](#)
- Digikam::ReplaceDialog, [2772](#)
- Digikam::ReplaceModifier, [2773](#)
  - parseOperation, [2775](#)
- Digikam::RestoreDTrashItemsJob, [2776](#)
- Digikam::RGBBackend, [2778](#)
  - backendName, [2778](#)
  - callRGBBackend, [2778](#)
  - getErrorMessage, [2779](#)
- Digikam::RGInfo, [2779](#)
- Digikam::RGTagModel, [2780](#)
  - addDataInTree, [2783](#)
  - addExternalTags, [2783](#)
  - addNewData, [2783](#)
  - addNewTag, [2784](#)
  - addSpacerTag, [2784](#)
  - branchFromIndex, [2784](#)
  - climbTreeAndGetSpacers, [2784](#)
  - deleteAllSpacersOrNewTags, [2785](#)
  - deleteTag, [2785](#)
  - findAndDeleteSpacersOrNewTags, [2785](#)
  - fromSourceIndex, [2785](#)
  - getSpacerAddress, [2786](#)
  - getSpacers, [2786](#)
  - getTagType, [2786](#)
  - readdNewTags, [2786](#)
  - readdTag, [2787](#)
  - RGTagModel, [2782](#)
  - toSourceIndex, [2787](#)
- Digikam::RGWidget, [2787](#)
  - readSettingsFromGroup, [2789](#)
  - RGWidget, [2789](#)
  - saveSettingsToGroup, [2789](#)
  - setUIEnabled, [2789](#)
  - signalProgressChanged, [2790](#)
  - signalSetUIEnabled, [2790](#)
  - signalUndoCommand, [2790](#)
- Digikam::RubberItem, [2791](#)
- Digikam::Rule, [2794](#)
  - addToken, [2795](#)
  - escapeToken, [2795](#)
  - isValid, [2796](#)
  - parseOperation, [2796](#)
  - regExp, [2796](#)
  - registerButton, [2797](#)
  - registerMenu, [2797](#)
  - reset, [2797](#)
  - setUseTokenMenu, [2798](#)
  - tokens, [2798](#)



- Digikam::RuleDialog, [2798](#)
- Digikam::SafeTemporaryFile, [2799](#)
- Digikam::SAlbum, [2799](#)
  - databaseUrl, [2802](#)
  - getTemporaryHaarTitle, [2802](#)
  - getTemporaryTitle, [2803](#)
  - isTemporarySearch, [2803](#)
- Digikam::SaveProperties, [2803](#)
- Digikam::SavingContext, [2804](#)
- Digikam::SavingTask, [2805](#)
  - continueQuery, [2806](#)
  - execute, [2806](#)
  - progressInfo, [2806](#)
  - type, [2806](#)
- Digikam::ScanController, [2807](#)
  - abortInitialization, [2810](#)
  - beginFileMetadataWrite, [2810](#)
  - cancelAllAndSuspendCollectionScan, [2810](#)
  - cancelCompleteScan, [2810](#)
  - completeCollectionScan, [2810](#)
  - databaseInitialization, [2810](#)
  - finishFileMetadataWrite, [2810](#)
  - hintAtModificationOfItems, [2811](#)
  - hintAtMoveOrCopyOfAlbum, [2811](#)
  - hintAtMoveOrCopyOfItems, [2811](#)
  - restartCollectionScan, [2811](#)
  - resumeCollectionScan, [2811](#)
  - scheduleCollectionScan, [2811](#)
  - scheduleCollectionScanExternal, [2812](#)
  - scheduleCollectionScanRelaxed, [2812](#)
  - shutDown, [2812](#)
  - suspendCollectionScan, [2812](#)
  - updateUniqueHash, [2812](#)
- Digikam::ScanController::FileMetadataWrite, [2812](#)
- Digikam::ScanStateFilter, [2814](#)
  - run, [2816](#)
- Digikam::ScriptingSettings, [2817](#)
- Digikam::SearchChangeset, [2817](#)
- Digikam::SearchesDBJobInfo, [2818](#)
- Digikam::SearchesDBJobsThread, [2820](#)
  - searchesListing, [2822](#)
- Digikam::SearchesJob, [2823](#)
- Digikam::SearchField, [2825](#)
  - createField, [2826](#)
  - isVisible, [2826](#)
  - setVisible, [2826](#)
  - write, [2826](#)
- Digikam::SearchFieldAlbum, [2827](#)
  - read, [2829](#)
  - reset, [2829](#)
  - setupValueWidgets, [2829](#)
  - setValueWidgetsVisible, [2829](#)
  - valueWidgetRects, [2829](#)
  - write, [2830](#)
- Digikam::SearchFieldCheckBox, [2831](#)
  - read, [2833](#)
  - reset, [2833](#)
  - setupValueWidgets, [2833](#)
  - setValueWidgetsVisible, [2833](#)
  - valueWidgetRects, [2833](#)
  - write, [2834](#)
- Digikam::SearchFieldChoice, [2835](#)
  - read, [2837](#)
  - reset, [2837](#)
  - setupValueWidgets, [2837](#)
  - setValueWidgetsVisible, [2837](#)
  - valueWidgetRects, [2838](#)
  - write, [2838](#)
- Digikam::SearchFieldColorDepth, [2839](#)
  - read, [2841](#)
  - setupValueWidgets, [2841](#)
- Digikam::SearchFieldComboBox, [2842](#)
  - reset, [2844](#)
  - setupValueWidgets, [2844](#)
  - setValueWidgetsVisible, [2844](#)
  - valueWidgetRects, [2844](#)
  - write, [2844](#)
- Digikam::SearchFieldGroup, [2845](#)
- Digikam::SearchFieldGroupLabel, [2846](#)
- Digikam::SearchFieldKeyword, [2848](#)
  - read, [2850](#)
  - write, [2850](#)
- Digikam::SearchFieldLabels, [2851](#)
  - read, [2853](#)
  - reset, [2853](#)
  - setupValueWidgets, [2853](#)
  - setValueWidgetsVisible, [2853](#)
  - valueWidgetRects, [2853](#)
  - write, [2854](#)
- Digikam::SearchFieldMonthDay, [2855](#)
  - read, [2857](#)
  - reset, [2857](#)
  - setupValueWidgets, [2857](#)
  - setValueWidgetsVisible, [2857](#)
  - valueWidgetRects, [2857](#)
  - write, [2858](#)
- Digikam::SearchFieldPageOrientation, [2859](#)
  - read, [2861](#)
  - setupValueWidgets, [2861](#)
- Digikam::SearchFieldRangeDate, [2862](#)
  - read, [2864](#)
  - reset, [2864](#)
  - setupValueWidgets, [2864](#)
  - setValueWidgetsVisible, [2864](#)
  - valueWidgetRects, [2865](#)
  - write, [2865](#)
- Digikam::SearchFieldRangeDouble, [2866](#)
  - read, [2868](#)
  - reset, [2868](#)
  - setupValueWidgets, [2868](#)
  - setValueWidgetsVisible, [2868](#)
  - valueWidgetRects, [2869](#)
  - write, [2869](#)
- Digikam::SearchFieldRangeInt, [2870](#)
  - read, [2872](#)
  - reset, [2872](#)

- setupValueWidgets, [2872](#)
  - setValueWidgetsVisible, [2872](#)
  - valueWidgetRects, [2873](#)
  - write, [2873](#)
- Digikam::SearchFieldRangeTime, [2874](#)
  - read, [2876](#)
  - reset, [2876](#)
  - setupValueWidgets, [2876](#)
  - setValueWidgetsVisible, [2876](#)
  - valueWidgetRects, [2876](#)
  - write, [2877](#)
- Digikam::SearchFieldRating, [2878](#)
  - read, [2880](#)
  - reset, [2880](#)
  - setupValueWidgets, [2880](#)
  - setValueWidgetsVisible, [2880](#)
  - valueWidgetRects, [2880](#)
  - write, [2881](#)
- Digikam::SearchFieldText, [2882](#)
  - read, [2884](#)
  - reset, [2884](#)
  - setupValueWidgets, [2884](#)
  - setValueWidgetsVisible, [2884](#)
  - valueWidgetRects, [2884](#)
  - write, [2885](#)
- Digikam::SearchFilterModel, [2885](#)
  - isFiltering, [2889](#)
  - matches, [2889](#)
- Digikam::SearchGroup, [2890](#)
  - addGroupToLayout, [2892](#)
  - createSearchGroup, [2892](#)
- Digikam::SearchGroupLabel, [2893](#)
- Digikam::SearchInfo, [2894](#)
- Digikam::SearchModel, [2895](#)
  - albumData, [2901](#)
  - albumForId, [2901](#)
  - setReplaceNames, [2901](#)
- Digikam::SearchModificationHelper, [2901](#)
  - createFuzzySearchFromDropped, [2903](#)
  - createFuzzySearchFromImage, [2904](#)
  - createFuzzySearchFromSketch, [2904](#)
  - SearchModificationHelper, [2903](#)
  - slotCreateFuzzySearchFromDropped, [2904](#)
  - slotCreateFuzzySearchFromImage, [2905](#)
  - slotCreateFuzzySearchFromSketch, [2905](#)
  - slotCreateTimeLineSearch, [2905](#)
  - slotSearchDelete, [2906](#)
  - slotSearchRename, [2906](#)
- Digikam::SearchSideBarWidget, [2907](#)
  - applySettings, [2909](#)
  - changeAlbumFromHistory, [2909](#)
  - doLoadState, [2909](#)
  - doSaveState, [2909](#)
  - getCaption, [2909](#)
  - getIcon, [2910](#)
  - setActive, [2910](#)
- Digikam::SearchTabHeader, [2911](#)
- Digikam::SearchTextBar, [2912](#)
  - doLoadState, [2914](#)
  - doSaveState, [2914](#)
  - getCurrentHighlightState, [2914](#)
  - HAS\_RESULT, [2914](#)
  - HighlightState, [2914](#)
  - NEUTRAL, [2914](#)
  - NO\_RESULT, [2914](#)
  - setCaseSensitive, [2915](#)
  - setHighlightOnResult, [2915](#)
- Digikam::SearchTextBarDb, [2915](#)
  - setFilterModel, [2918](#)
  - setModel, [2918, 2919](#)
- Digikam::SearchTextFilterSettings, [2919](#)
- Digikam::SearchTextSettings, [2920](#)
- Digikam::SearchTreeView, [2921](#)
- Digikam::SearchView, [2927](#)
  - addGroupToLayout, [2929](#)
  - bottomBarPixmap, [2929](#)
  - createSearchGroup, [2929](#)
  - groupLabelPixmap, [2929](#)
- Digikam::SearchViewBottomBar, [2930](#)
- Digikam::SearchViewThemedPartsCache, [2931](#)
- Digikam::SearchWindow, [2932](#)
  - readSearch, [2933](#)
  - reset, [2933](#)
  - searchEdited, [2933](#)
- Digikam::SearchXmlCachingReader, [2934](#)
- Digikam::SearchXmlReader, [2937](#)
  - defaultFieldOperator, [2938](#)
  - fieldOperator, [2938](#)
  - groupCaption, [2939](#)
  - groupOperator, [2939](#)
  - readNext, [2939](#)
  - readToStartOfElement, [2939](#)
  - value, [2939](#)
- Digikam::SearchXmlWriter, [2940](#)
  - finish, [2942](#)
  - finishField, [2942](#)
  - finishGroup, [2942](#)
  - setDefaultFieldOperator, [2942](#)
  - setGroupOperator, [2942](#)
  - writeField, [2942](#)
  - writeGroup, [2942](#)
  - xml, [2943](#)
- Digikam::SequenceNumberDialog, [2943](#)
- Digikam::SequenceNumberOption, [2945](#)
  - parseOperation, [2947](#)
- Digikam::Setup, [2948](#)
  - execDialog, [2951](#)
  - execSinglePage, [2951](#)
- Digikam::SetupAlbumView, [2951](#)
- Digikam::SetupCamera, [2952](#)
- Digikam::SetupCategory, [2953](#)
- Digikam::SetupCollectionDelegate, [2954](#)
  - createItemWidgets, [2956](#)
  - updateItemWidgets, [2956](#)
- Digikam::SetupCollectionModel, [2958](#)
  - AppendDecorationRole, [2960](#)

- CategoryButtonDisplayRole, [2960](#)
- DeleteDecorationRole, [2960](#)
- IsAppendRole, [2960](#)
- IsCategoryRole, [2960](#)
- IsDeleteRole, [2960](#)
- IsUpdateRole, [2960](#)
- SetupCollectionDataRole, [2960](#)
- SetupCollectionModel, [2961](#)
- slotAppendPressed, [2961](#)
- slotCategoryButtonPressed, [2961](#)
- UpdateDecorationRole, [2960](#)
- Digikam::SetupCollectionModel::Item, [2961](#)
- Digikam::SetupCollections, [2962](#)
- Digikam::SetupCollectionTreeView, [2963](#)
- Digikam::SetupDatabase, [2964](#)
- Digikam::SetupEditor, [2965](#)
- Digikam::SetupEditorIface, [2966](#)
- Digikam::SetupGeolocation, [2967](#)
- Digikam::SetupICC, [2968](#)
  - SetupICC, [2968](#)
- Digikam::SetupImageQualitySorter, [2969](#)
- Digikam::SetupIOFiles, [2970](#)
- Digikam::SetupLightTable, [2970](#)
- Digikam::SetupMetadata, [2971](#)
- Digikam::SetupMime, [2972](#)
- Digikam::SetupMisc, [2973](#)
- Digikam::SetupPlugins, [2974](#)
- Digikam::SetupRaw, [2975](#)
- Digikam::SetupTemplate, [2976](#)
- Digikam::SetupToolTip, [2977](#)
- Digikam::SetupVersioning, [2978](#)
- Digikam::SharedLoadingTask, [2979](#)
  - accessMode, [2981](#)
  - addListener, [2981](#)
  - cacheKey, [2981](#)
  - completed, [2981](#)
  - execute, [2981](#)
  - loadSaveNotifier, [2981](#)
  - notifyNewLoadingProcess, [2982](#)
  - progressInfo, [2982](#)
  - querySendNotifyEvent, [2982](#)
  - removeListener, [2982](#)
  - setResult, [2982](#)
- Digikam::SharedLoadSaveThread, [2983](#)
- Digikam::SharedQueue< T >, [2987](#)
- Digikam::SharpContainer, [2987](#)
- Digikam::SharpenFilter, [2989](#)
  - filterAction, [2993](#)
  - filterIdentifier, [2993](#)
  - readParameters, [2993](#)
- Digikam::SharpSettings, [2993](#)
- Digikam::ShearFilter, [2995](#)
  - filterAction, [2999](#)
  - filterIdentifier, [2999](#)
  - readParameters, [2999](#)
- Digikam::ShowHideVersionsOverlay, [3000](#)
  - checkIndex, [3003](#)
  - createButton, [3003](#)
  - setActive, [3003](#)
  - updateButton, [3003](#)
- Digikam::Sidebar, [3004](#)
  - activeNextTab, [3008](#)
  - activePreviousTab, [3008](#)
  - appendTab, [3008](#)
  - backup, [3008](#)
  - doLoadState, [3008](#)
  - doSaveState, [3008](#)
  - restore, [3008](#)
  - Sidebar, [3007](#)
- Digikam::SidebarSplitter, [3009](#)
  - restoreState, [3010](#)
  - saveState, [3010](#)
  - setSize, [3011](#)
- Digikam::SidebarWidget, [3011](#)
  - applySettings, [3013](#)
  - changeAlbumFromHistory, [3013](#)
  - getCaption, [3013](#)
  - getIcon, [3013](#)
  - setActive, [3013](#)
  - SidebarWidget, [3013](#)
- Digikam::SidecarFinder, [3014](#)
- Digikam::SimilarityDb, [3014](#)
  - clearImageSimilarity, [3015](#)
  - getDirtyOrMissingFingerprints, [3015](#)
  - getDirtyOrMissingFingerprintURLs, [3016](#)
  - getImageSimilarity, [3016](#)
  - getImageSimilarityAlgorithms, [3016](#)
  - getLegacySetting, [3017](#)
  - getSetting, [3017](#)
  - hasDirtyOrMissingFingerprint, [3017](#)
  - hasFingerprint, [3017](#)
  - hasFingerprints, [3018](#)
  - integrityCheck, [3018](#)
  - registeredImagelds, [3018](#)
  - removeImageFingerprint, [3019](#)
  - removeImageSimilarity, [3019](#)
  - setSetting, [3019](#)
- Digikam::SimilarityDbAccess, [3020](#)
  - checkReadyForUse, [3021](#)
  - initDbEngineErrorHandler, [3021](#)
  - isInitialized, [3021](#)
  - parameters, [3021](#)
  - setLastError, [3021](#)
  - setParameters, [3021](#)
  - SimilarityDbAccess, [3020](#)
- Digikam::SimilarityDbBackend, [3022](#)
  - initSchema, [3026](#)
- Digikam::SimilarityDbSchemaUpdater, [3026](#)
- Digikam::SimpleTreeModel, [3027](#)
- Digikam::SimpleTreeModel::Item, [3028](#)
- Digikam::SinglePhotoPreviewLayout, [3029](#)
  - addItem, [3030](#)
- Digikam::SketchWidget, [3031](#)
  - setSketchImageFromXML, [3032](#)
- Digikam::SlideVideo, [3033](#)
- Digikam::SoftProofDialog, [3034](#)

- Digikam::SolidHardwareDlg, [3035](#)
- Digikam::SpellCheckConfig, [3036](#)
- Digikam::SqueezedComboBox, [3036](#)
  - addSqueezedItem, [3038](#)
  - contains, [3039](#)
  - findOriginalText, [3039](#)
  - insertSqueezedItem, [3039](#)
  - insertSqueezedList, [3039](#)
  - item, [3040](#)
  - itemHighlighted, [3040](#)
  - setCurrent, [3040](#)
  - SqueezedComboBox, [3038](#)
- Digikam::StackedView, [3041](#)
- Digikam::StartScanPage, [3043](#)
- Digikam::StateSavingObject, [3044](#)
  - DIRECT\_CHILDREN, [3046](#)
  - doLoadState, [3046](#)
  - doSaveState, [3046](#)
  - entryName, [3046](#)
  - getConfigGroup, [3047](#)
  - getStateSavingDepth, [3047](#)
  - INSTANCE, [3046](#)
  - RECURSIVE, [3046](#)
  - setConfigGroup, [3047](#)
  - setEntryPrefix, [3048](#)
  - setStateSavingDepth, [3048](#)
  - StateSavingDepth, [3045](#)
  - StateSavingObject, [3046](#)
- Digikam::StatusBarProgressWidget, [3049](#)
- Digikam::StatusProgressBar, [3050](#)
- Digikam::StayPoppedUpComboBox, [3052](#)
  - installView, [3053](#)
  - sendViewportEventToView, [3053](#)
  - StayPoppedUpComboBox, [3053](#)
- Digikam::StretchFilter, [3055](#)
  - filterAction, [3059](#)
  - filterIdentifier, [3059](#)
  - readParameters, [3059](#)
- Digikam::StyleSheetDebugger, [3059](#)
  - StyleSheetDebugger, [3060](#)
- Digikam::SubjectData, [3060](#)
- Digikam::SubjectEdit, [3061](#)
- Digikam::SubjectWidget, [3063](#)
- Digikam::SyncJob, [3064](#)
- Digikam::SystemSettings, [3065](#)
  - HttpProxy, [3065](#)
  - ProxyType, [3065](#)
  - Socks5Proxy, [3065](#)
- Digikam::SystemSettingsWidget, [3066](#)
- Digikam::TableView, [3067](#)
  - doLoadState, [3070](#)
  - doSaveState, [3070](#)
  - invertSelection, [3070](#)
  - selectAll, [3070](#)
  - slotAwayFromSelection, [3070](#)
  - slotDeleteSelected, [3070](#)
  - slotSetCurrentWhenAvailable, [3070](#)
- Digikam::TableViewColumn, [3071](#)
  - columnAffectedByChangeset, [3072](#)
  - compare, [3072](#)
  - data, [3072](#)
  - getColumnFlags, [3072](#)
  - paint, [3073](#)
  - sizeHint, [3073](#)
  - updateThumbnailSize, [3073](#)
- Digikam::TableViewColumnConfiguration, [3073](#)
- Digikam::TableViewColumnConfigurationWidget, [3074](#)
- Digikam::TableViewColumnDescription, [3074](#)
- Digikam::TableViewColumnFactory, [3075](#)
- Digikam::TableViewColumnProfile, [3076](#)
  - loadSettings, [3076](#)
- Digikam::TableViewColumns::ColumnAudioVideoProperties, [3077](#)
  - compare, [3079](#)
  - data, [3079](#)
  - getColumnFlags, [3079](#)
  - getTitle, [3079](#)
  - setConfiguration, [3079](#)
- Digikam::TableViewColumns::ColumnDigikamProperties, [3081](#)
  - columnAffectedByChangeset, [3083](#)
  - compare, [3083](#)
  - data, [3083](#)
  - getColumnFlags, [3083](#)
  - getDescription, [3083](#)
  - getTitle, [3084](#)
- Digikam::TableViewColumns::ColumnFileConfigurationWidget, [3084](#)
  - getNewConfiguration, [3085](#)
- Digikam::TableViewColumns::ColumnFileProperties, [3086](#)
  - compare, [3088](#)
  - data, [3088](#)
  - getColumnFlags, [3088](#)
  - getConfigurationWidget, [3088](#)
  - getTitle, [3088](#)
  - setConfiguration, [3089](#)
- Digikam::TableViewColumns::ColumnGeoConfigurationWidget, [3089](#)
  - getNewConfiguration, [3090](#)
- Digikam::TableViewColumns::ColumnGeoProperties, [3091](#)
  - compare, [3093](#)
  - data, [3093](#)
  - getColumnFlags, [3093](#)
  - getConfigurationWidget, [3093](#)
  - getTitle, [3094](#)
  - setConfiguration, [3094](#)
- Digikam::TableViewColumns::ColumnItemProperties, [3095](#)
  - compare, [3097](#)
  - data, [3097](#)
  - getColumnFlags, [3097](#)
  - getTitle, [3097](#)
- Digikam::TableViewColumns::ColumnPhotoConfigurationWidget, [3098](#)

- getNewConfiguration, 3099
- Digikam::TableViewColumns::ColumnPhotoProperties, 3100
  - compare, 3102
  - data, 3102
  - getColumnFlags, 3102
  - getConfigurationWidget, 3102
  - getTitle, 3102
  - setConfiguration, 3103
- Digikam::TableViewColumns::ColumnThumbnail, 3104
  - data, 3106
  - getColumnFlags, 3106
  - getTitle, 3106
  - paint, 3106
  - sizeHint, 3106
  - updateThumbnailSize, 3106
- Digikam::TableViewConfigurationDialog, 3107
- Digikam::TableViewItemDelegate, 3108
  - sizeHint, 3108
- Digikam::TableViewModel, 3109
  - addColumnAt, 3111
  - flags, 3111
  - indexFromImageId, 3111
  - infoFromItem, 3111
  - loadColumnProfile, 3111
  - parent, 3111
  - sort, 3111
- Digikam::TableViewModel::Item, 3112
- Digikam::TableViewSelectionModelSyncer, 3112
  - TableViewSelectionModelSyncer, 3113
- Digikam::TableViewShared, 3113
- Digikam::TableViewTreeView, 3114
  - dragDropHandler, 3115
  - hasHiddenGroupedImages, 3115
  - mapIndexForDragDrop, 3115
  - pixmapForDrag, 3116
- Digikam::TagChangeset, 3116
  - Operation, 3116
  - PropertiesChanged, 3116
- Digikam::TagCheckView, 3117
  - addCustomContextMenuActions, 3123
  - checkedTagsChanged, 3124
  - doLoadState, 3124
  - doSaveState, 3124
  - setCheckNewTags, 3124
- Digikam::TagCompleter, 3125
  - setSupportingTagModel, 3125
- Digikam::TagData, 3126
- Digikam::TagDragDropHandler, 3126
  - accepts, 3127
  - createMimeData, 3127
  - dropEvent, 3127
  - mimeTypes, 3128
- Digikam::TagEditDlg, 3128
  - createTAlbum, 3129
- Digikam::TagFilterView, 3129
  - addCustomContextMenuActions, 3137
  - handleCustomContextMenuAction, 3137
  - TagFilterView, 3136
- Digikam::TagFolderView, 3138
  - addCustomContextMenuActions, 3144
  - contextMenuEvent, 3144
  - contextMenuTitle, 3144
  - handleCustomContextMenuAction, 3144
  - setContextMenuItems, 3145
  - setShowDeleteFaceTagsAction, 3145
  - setShowFindDuplicateAction, 3145
  - TagFolderView, 3143
- Digikam::TaggingAction, 3146
  - TaggingAction, 3146
- Digikam::TaggingActionFactory, 3146
  - MatchContainingFragment, 3147
  - MatchStartingWithFragment, 3147
  - NameMatchMode, 3147
  - setConstraintInterface, 3148
- Digikam::TaggingActionFactory::ConstraintInterface, 3148
- Digikam::TagInfo, 3149
- Digikam::TagList, 3149
  - restoreSettings, 3150
- Digikam::TagMngrListModel, 3150
  - addItem, 3151
  - dropMimeData, 3151
- Digikam::TagMngrListView, 3152
- Digikam::TagMngrTreeView, 3153
  - contextMenuEvent, 3159
  - setContextMenuItems, 3159
- Digikam::TagModel, 3160
  - albumData, 3166
  - albumForId, 3166
  - decorationRoleData, 3166
  - fontRoleData, 3166
- Digikam::TagModificationHelper, 3167
  - bindMultipleTags, 3169
  - bindTag, 3169
  - boundMultipleTags, 3169
  - boundTag, 3170
  - slotFaceTagDelete, 3170
  - slotMultipleFaceTagDel, 3170
  - slotMultipleTagDel, 3170
  - slotMultipleTagsToFaceTags, 3171
  - slotTagDelete, 3171
  - slotTagEdit, 3171
  - slotTagNew, 3171
  - slotTagToFaceTag, 3172
  - TagModificationHelper, 3169
- Digikam::TagProperties, 3172
  - addProperty, 3173
  - getOrCreate, 3173
  - TagProperties, 3173
  - value, 3173
- Digikam::TagPropertiesFilterModel, 3174
  - isFiltering, 3177
  - matches, 3177
- Digikam::TagProperty, 3178
- Digikam::TagPropertyName, 3178

- Digikam::TagPropWidget, 3178
- Digikam::TagRegion, 3179
  - absoluteToRelative, 3180
  - adjustToOrientation, 3180
  - intersects, 3181
  - reverseToOrientation, 3181
  - TagRegion, 3180
  - toVariant, 3181
- Digikam::TagsActionMngr, 3182
  - registerLabelsActions, 3183
  - registerTagsActionCollections, 3183
  - updateTagShortcut, 3183
- Digikam::TagsCache, 3184
  - canBeWrittenToMetadata, 3187
  - colorLabelForTag, 3187
  - colorLabelFromTags, 3187
  - createTag, 3187
  - getOrCreateTag, 3187
  - getOrCreateTagWithProperty, 3187
  - hasProperty, 3188
  - IncludeLeadingSlash, 3187
  - LeadingSlashPolicy, 3186
  - NoLeadingSlash, 3187
  - parentTags, 3188
  - pickLabelForTag, 3188
  - pickLabelFromTags, 3188
  - properties, 3188
  - propertyValue, 3188
  - shortenedTagPaths, 3189
  - tagAdded, 3189
  - tagForColorLabel, 3189
  - tagForName, 3189
  - tagForPath, 3189
  - tagForPickLabel, 3189
  - tagName, 3190
  - tagPath, 3190
  - tagsForName, 3190
  - tagsWithProperty, 3190
  - tagsWithPropertyCached, 3190
- Digikam::TagsDBJobInfo, 3191
- Digikam::TagsDBJobsThread, 3192
  - tagsListing, 3194
- Digikam::TagsEdit, 3195
- Digikam::TagShortInfo, 3196
- Digikam::TagsJob, 3196
- Digikam::TagsLineEditOverlay, 3198
  - createWidget, 3201
  - hide, 3201
  - setActive, 3201
  - slotEntered, 3201
  - visualChange, 3201
- Digikam::TagsManager, 3202
  - doLoadState, 3204
  - doSaveState, 3204
- Digikam::TagsManagerFilterModel, 3205
  - matches, 3208
- Digikam::TagsPopupMenu, 3209
  - DISPLAY, 3209
  - Mode, 3209
- Digikam::TagTreeView, 3210
- Digikam::TagTreeViewSelectComboBox, 3216
- Digikam::TagViewSideBarWidget, 3220
  - applySettings, 3222
  - changeAlbumFromHistory, 3222
  - doLoadState, 3222
  - doSaveState, 3222
  - getCaption, 3222
  - getIcon, 3223
  - setActive, 3223
- Digikam::TAlbum, 3223
  - databaseUrl, 3226
  - tagPath, 3226
- Digikam::Template, 3227
  - m\_templateTitle, 3228
- Digikam::TemplateList, 3228
- Digikam::TemplateListItem, 3229
- Digikam::TemplateManager, 3230
- Digikam::TemplatePanel, 3231
- Digikam::TemplateSelector, 3232
- Digikam::TemplateViewer, 3234
- Digikam::TextFilter, 3236
- Digikam::TextureContainer, 3237
- Digikam::TextureFilter, 3238
  - filterAction, 3242
  - filterIdentifier, 3242
  - readParameters, 3242
- Digikam::TextureSettings, 3242
- Digikam::ThemeManager, 3243
- Digikam::ThreadManager, 3244
- Digikam::ThumbBarDock, 3245
  - reinitialize, 3246
  - shouldBeVisible, 3246
- Digikam::ThumbnailAligningDelegate, 3247
- Digikam::ThumbnailCreator, 3247
  - deleteThumbnailsFromDisk, 3248
  - errorString, 3248
  - loadDetail, 3249
  - setExifRotate, 3249
  - setLoadingProperties, 3249
  - setOnlyLargeThumbnails, 3249
  - setRemoveAlphaChannel, 3249
  - setThumbnailSize, 3249
  - store, 3250
  - storedSize, 3250
  - ThumbnailCreator, 3248
- Digikam::ThumbnailIdentifier, 3250
- Digikam::ThumbnailImageCatcher, 3251
  - cancel, 3252
  - enqueue, 3252
  - setActive, 3252
  - ThumbnailImageCatcher, 3252
- Digikam::ThumbnailInfo, 3253
  - isAccessible, 3254
  - mimeType, 3254
  - modificationDate, 3254
  - orientationHint, 3254



- Digikam::ThumbnailInfoProvider, [3255](#)
- Digikam::ThumbnailLoadingTask, [3256](#)
  - execute, [3258](#)
  - postProcess, [3258](#)
- Digikam::ThumbnailLoadThread, [3259](#)
  - defaultThread, [3265](#)
  - deleteThumbnail, [3265](#)
  - find, [3265](#)
  - findGroup, [3265](#)
  - initializeNoThumbnailStorage, [3265](#)
  - initializeThumbnailDatabase, [3265](#)
  - lastDescriptions, [3266](#)
  - load, [3266](#)
  - maximumThumbnailSize, [3266](#)
  - pregenerateGroup, [3266](#)
  - preload, [3266](#)
  - setDisplayingWidget, [3266](#)
  - setHighlightPixmap, [3267](#)
  - setPixmapRequested, [3267](#)
  - setSendSurrogatePixmap, [3267](#)
  - setThumbnailSize, [3267](#)
  - signalThumbnailLoaded, [3267](#)
  - storeDetailThumbnail, [3268](#)
  - thumbnailCreator, [3268](#)
  - thumbnailLoaded, [3268](#)
  - thumbnailsAvailable, [3268](#)
  - thumbnailToPixmapSize, [3268](#)
- Digikam::ThumbnailSize, [3269](#)
  - Size, [3269](#)
  - Small, [3269](#)
- Digikam::ThumbsDb, [3270](#)
  - findByFilePath, [3270](#)
- Digikam::ThumbsDbAccess, [3271](#)
  - setLastError, [3271](#)
  - ThumbsDbAccess, [3271](#)
- Digikam::ThumbsDbBackend, [3272](#)
  - initSchema, [3276](#)
- Digikam::ThumbsDbInfo, [3276](#)
- Digikam::ThumbsDbInfoProvider, [3276](#)
  - thumbnailInfo, [3277](#)
- Digikam::ThumbsDbSchemaUpdater, [3277](#)
- Digikam::ThumbsGenerator, [3278](#)
  - setUseMultiCoreCPU, [3281](#)
  - ThumbsGenerator, [3280](#)
- Digikam::ThumbsTask, [3281](#)
- Digikam::TileGrouper, [3283](#)
  - updateClusters, [3283](#)
- Digikam::TileIndex, [3283](#)
- Digikam::TimeAdjustContainer, [3284](#)
- Digikam::TimeAdjustSettings, [3285](#)
  - detAdjustmentByClockPhotoUrl, [3286](#)
- Digikam::TimelineSideBarWidget, [3287](#)
  - applySettings, [3289](#)
  - changeAlbumFromHistory, [3289](#)
  - doLoadState, [3289](#)
  - doSaveState, [3289](#)
  - getCaption, [3289](#)
  - getIcon, [3290](#)
  - setActive, [3290](#)
- Digikam::TimeLineWidget, [3291](#)
  - FuzzySelection, [3293](#)
  - LinScale, [3292](#)
  - LogScale, [3292](#)
  - ScaleMode, [3292](#)
  - Selected, [3293](#)
  - SelectionMode, [3292](#)
  - Unselected, [3293](#)
- Digikam::TimeZoneComboBox, [3293](#)
- Digikam::Token, [3293](#)
  - action, [3295](#)
  - description, [3295](#)
  - id, [3295](#)
- Digikam::TonalityContainer, [3295](#)
- Digikam::TonalityFilter, [3296](#)
  - filterAction, [3300](#)
  - filterIdentifier, [3300](#)
  - readParameters, [3300](#)
- Digikam::ToolListViewGroup, [3300](#)
- Digikam::ToolListViewItem, [3301](#)
- Digikam::ToolSettingsView, [3302](#)
- Digikam::ToolsListView, [3303](#)
- Digikam::ToolsView, [3304](#)
- Digikam::TooltipCreator, [3305](#)
- Digikam::TooltipDialog, [3305](#)
- Digikam::TooltipsPage, [3306](#)
- Digikam::TrackCorrelator, [3307](#)
  - Digikam::TrackCorrelator::Correlation, [3308](#)
  - Digikam::TrackCorrelator::CorrelationOptions, [3308](#)
- Digikam::TrackCorrelatorThread, [3309](#)
- Digikam::TrackListModel, [3310](#)
  - headerData, [3311](#)
  - index, [3311](#)
- Digikam::TrackManager, [3311](#)
  - clear, [3312](#)
  - Id, [3312](#)
  - Digikam::TrackManager::Track, [3313](#)
  - Digikam::TrackManager::TrackPoint, [3313](#)
- Digikam::TrackReader, [3314](#)
  - Digikam::TrackReader::TrackReadResult, [3314](#)
- Digikam::TrainerWorker, [3315](#)
  - aboutToDeactivate, [3317](#)
- Digikam::TrainingDataProvider, [3318](#)
  - images, [3318](#)
  - newImages, [3318](#)
- Digikam::TransactionItem, [3320](#)
  - setStatus, [3321](#)
- Digikam::TransactionItemView, [3322](#)
- Digikam::TransitionMngr, [3323](#)
- Digikam::TransitionPreview, [3323](#)
- Digikam::TrashView, [3324](#)
  - getThumbnailSize, [3325](#)
  - lastSelectedItemUrl, [3325](#)
  - model, [3325](#)
  - setThumbnailSize, [3325](#)
  - statusBarText, [3325](#)
- Digikam::TreeBranch, [3326](#)

- Digikam::TreeProxyModel, [3326](#)
- Digikam::TreeViewComboBox, [3327](#)
  - installView, [3329](#)
  - sendViewportEventToView, [3329](#)
  - TreeViewComboBox, [3328](#)
  - view, [3329](#)
- Digikam::TreeViewLineEditComboBox, [3330](#)
  - installLineEdit, [3332](#)
  - installView, [3332](#)
  - setLineEditText, [3332](#)
  - TreeViewLineEditComboBox, [3332](#)
- Digikam::TrimmedModifier, [3333](#)
  - parseOperation, [3335](#)
- Digikam::TwoProgressItemsContainer, [3336](#)
- Digikam::UMSCamera, [3337](#)
  - cameraAbout, [3339](#)
  - cameraDriverType, [3339](#)
  - cameraManual, [3339](#)
  - cameraMD5ID, [3339](#)
  - cameraSummary, [3339](#)
  - cancel, [3340](#)
  - capture, [3340](#)
  - deleteItem, [3340](#)
  - doConnect, [3340](#)
  - downloadItem, [3340](#)
  - getFolders, [3340](#)
  - getFreeSpace, [3340](#)
  - getItemInfo, [3341](#)
  - getItemInfoList, [3341](#)
  - getMetadata, [3341](#)
  - getPreview, [3341](#)
  - getThumbnail, [3341](#)
  - setLockItem, [3342](#)
  - uploadItem, [3342](#)
- Digikam::UndoAction, [3343](#)
- Digikam::UndoActionIrreversible, [3344](#)
- Digikam::UndoActionReversible, [3345](#)
- Digikam::UndoCache, [3346](#)
- Digikam::UndoManager, [3346](#)
- Digikam::UndoMetadataContainer, [3346](#)
- Digikam::UndoState, [3347](#)
- Digikam::UniqueModifier, [3348](#)
  - parseOperation, [3350](#)
  - reset, [3350](#)
- Digikam::UnsharpMaskFilter, [3351](#)
  - filterAction, [3355](#)
  - filterIdentifier, [3355](#)
  - readParameters, [3355](#)
- Digikam::VersionFileInfo, [3355](#)
- Digikam::VersionFileOperation, [3355](#)
  - MoveToIntermediate, [3356](#)
  - NewFile, [3356](#)
  - Replace, [3356](#)
  - SaveAndDelete, [3356](#)
  - StoreIntermediates, [3356](#)
  - Task, [3356](#)
  - VersionFileOperation, [3356](#)
- Digikam::VersioningPromptUserSaveDialog, [3357](#)
- Digikam::VersionItemFilterSettings, [3357](#)
- Digikam::VersionManager, [3358](#)
- Digikam::VersionManagerSettings, [3358](#)
- Digikam::VersionNamingScheme, [3359](#)
  - baseName, [3360](#)
  - directory, [3360](#)
  - incrementedCounter, [3360](#)
  - initialCounter, [3361](#)
  - intermediateFileName, [3361](#)
  - versionFileName, [3361](#)
- Digikam::VersionsDelegate, [3362](#)
  - asDelegate, [3364](#)
  - requestNotification, [3364](#)
- Digikam::VersionsTreeView, [3365](#)
  - ~VersionsTreeView, [3367](#)
  - dragDropHandler, [3367](#)
  - mapIndexForDragDrop, [3367](#)
  - pixmapForDrag, [3367](#)
- Digikam::VersionsWidget, [3368](#)
- Digikam::VideoFrame, [3369](#)
- Digikam::VideoInfoContainer, [3369](#)
- Digikam::VideoMetadataContainer, [3369](#)
- Digikam::VideoStripFilter, [3370](#)
- Digikam::VideoThumbDecoder, [3370](#)
- Digikam::VideoThumbnailer, [3370](#)
- Digikam::VideoThumbWriter, [3370](#)
- Digikam::VidPlayerDlg, [3371](#)
- Digikam::VidSlideSettings, [3371](#)
  - AVI, [3375](#)
  - BLUERAY, [3376](#)
  - CVD1, [3375](#)
  - CVD2, [3375](#)
  - DVD1, [3375](#)
  - DVD2, [3375](#)
  - DVGA, [3375](#)
  - EDTV1, [3375](#)
  - EDTV2, [3375](#)
  - EGA, [3375](#)
  - FLASH, [3374](#)
  - HDPLUS, [3376](#)
  - HDTV, [3376](#)
  - HSXGA, [3376](#)
  - HUXGA, [3376](#)
  - HVGA, [3375](#)
  - HXGA, [3376](#)
  - MJPEG, [3374](#)
  - MKV, [3375](#)
  - MP4, [3375](#)
  - MPEG2, [3374](#)
  - MPEG4, [3374](#)
  - MPG, [3375](#)
  - NTSC, [3375](#)
  - PAL, [3375](#)
  - QSXGA, [3376](#)
  - QSXGAPLUS, [3376](#)
  - QUXGA, [3376](#)
  - QVGA, [3375](#)
  - QXGA, [3376](#)



SDTV1, [3375](#)  
SDTV2, [3375](#)  
SDTV3, [3375](#)  
SVCD1, [3375](#)  
SVCD2, [3375](#)  
SVGA, [3375](#)  
SXGA, [3376](#)  
SXGAPLUS, [3376](#)  
THEORA, [3374](#)  
TXGA, [3376](#)  
UHD4K, [3376](#)  
UHD5K, [3376](#)  
UHD6K, [3376](#)  
UHD8K, [3376](#)  
UW10K, [3376](#)  
UW16K, [3376](#)  
UWFHD, [3376](#)  
UXGA, [3376](#)  
VBR04, [3374](#)  
VBR05, [3374](#)  
VBR10, [3374](#)  
VBR12, [3374](#)  
VBR15, [3374](#)  
VBR20, [3374](#)  
VBR25, [3374](#)  
VBR30, [3374](#)  
VBR40, [3374](#)  
VBR45, [3374](#)  
VBR50, [3374](#)  
VBR60, [3374](#)  
VBR80, [3374](#)  
VCD1, [3375](#)  
VCD2, [3375](#)  
VGA, [3375](#)  
VidBitRate, [3374](#)  
VidCodec, [3374](#)  
VidFormat, [3374](#)  
VidStd, [3375](#)  
VidType, [3375](#)  
WEBMVP8, [3374](#)  
WHSXGA, [3376](#)  
WHUXGA, [3376](#)  
WHXGA, [3376](#)  
WMV7, [3374](#)  
WMV8, [3374](#)  
WMV9, [3374](#)  
WQHD, [3376](#)  
WQSXGA, [3376](#)  
WQUXGA, [3376](#)  
WQXGA, [3376](#)  
WQXGAPLUS, [3376](#)  
WSXGA, [3376](#)  
WSXGAPLUS, [3376](#)  
WUXGA, [3376](#)  
WVGA, [3375](#)  
WXGA1, [3376](#)  
WXGA2, [3376](#)  
X264, [3374](#)  
XVGA, [3375](#)  
Digikam::VidSlideTask, [3377](#)  
Digikam::VidSlideThread, [3379](#)  
Digikam::VisibilityController, [3381](#)  
    addObject, [3382](#)  
Digikam::VisibilityObject, [3382](#)  
Digikam::WBContainer, [3383](#)  
Digikam::WBFilter, [3384](#)  
    autoWBAdjustementFromColor, [3388](#)  
    filterAction, [3388](#)  
    filterIdentifier, [3388](#)  
    filterImage, [3388](#)  
    readParameters, [3388](#)  
Digikam::WBSettings, [3389](#)  
Digikam::WebBrowserDlg, [3390](#)  
Digikam::WebWidget, [3391](#)  
Digikam::WelcomePage, [3392](#)  
Digikam::WelcomePageView, [3393](#)  
Digikam::WelcomePageViewPage, [3394](#)  
Digikam::WorkerObject, [3395](#)  
    aboutToDeactivate, [3397](#)  
    aboutToQuitLoop, [3397](#)  
    connectAndSchedule, [3397](#)  
    deactivate, [3397](#)  
    DeactivatingMode, [3396](#)  
    FlushSignals, [3396](#)  
    KeepSignals, [3396](#)  
    PhaseOut, [3396](#)  
    setPriority, [3397](#)  
    shutDown, [3397](#)  
    WorkerObject, [3396](#)  
Digikam::Workflow, [3398](#)  
Digikam::WorkflowDlg, [3398](#)  
Digikam::WorkflowItem, [3399](#)  
Digikam::WorkflowList, [3400](#)  
Digikam::WorkflowManager, [3401](#)  
    load, [3402](#)  
Digikam::WorkingWidget, [3402](#)  
Digikam::WSAlbum, [3403](#)  
Digikam::WSCoboBoxIntermediate, [3403](#)  
    setIntermediate, [3404](#)  
Digikam::WSLoginDialog, [3404](#)  
Digikam::WSNewAlbumDialog, [3405](#)  
Digikam::WSSelectUserDlg, [3406](#)  
Digikam::WSSettings, [3407](#)  
Digikam::WSSettingsWidget, [3409](#)  
Digikam::WSToolDialog, [3411](#)  
Digikam::WSToolUtils, [3412](#)  
Digikam::XbelReader, [3412](#)  
Digikam::XbelWriter, [3413](#)  
Digikam::XmpMetaEngineMergeHelper, [3414](#)  
Digikam::XmpWidget, [3415](#)  
    getMetadataTitle, [3417](#)  
    getTagDescription, [3417](#)  
    getTagTitle, [3417](#)  
    loadFromURL, [3417](#)  
dimensions  
    Digikam::ItemInfo, [2118](#)

- dimensionsHint
  - Digikam::DatabaseLoadSaveFileInfoProvider, 729
  - Digikam::LoadSaveFileInfoProvider, 2371
- DImg
  - Digikam::DImg, 986, 987
- DImgBuiltinFilter
  - Digikam::DImgBuiltinFilter, 996
- DImgChildItem
  - Digikam::DImgChildItem, 1000
- DImgThreadedAnalyser
  - Digikam::DImgThreadedAnalyser, 1018
- DImgThreadedFilter
  - Digikam::DImgThreadedFilter, 1022
- DIRECT\_CHILDREN
  - Digikam::StateSavingObject, 3046
- DirectMatch
  - Digikam::AlbumFilterModel, 270
- directory
  - Digikam::DefaultVersionNamingScheme, 918
  - Digikam::VersionNamingScheme, 3360
- discoverFormat
  - Digikam::FileSaveOptionsBox, 1534
- DISPLAY
  - Digikam::TagsPopupMenu, 3209
- Display
  - Digikam::IccProfile, 1764
- displayableName
  - Digikam::BasicDImgFilterGenerator< T >, 456
  - Digikam::DImgFilterGenerator, 1003
  - Digikam::DImgFilterManager, 1005
- DisplayFlag
  - Digikam::DFontProperties, 944
- DItemsListsLessThanHandler
  - Digikam, 134
- DNG\_SDK\_INTERNAL\_ERROR
  - Digikam::DNGWriter, 1115
- DNGVersion
  - Digikam::DRawInfo, 1271
- DNNDetectorSSD
  - Digikam, 134
- DNNDetectorYOLOv3
  - Digikam, 134
- DNNDetectorYuNet
  - Digikam, 134
- DNotificationWrapper
  - Digikam, 136
- doConnect
  - Digikam::GPCamera, 1632
  - Digikam::UMSCamera, 3340
- DocumentedHistory
  - Digikam::FilterAction, 1555
- doLoadState
  - Digikam::AbstractAlbumTreeView, 159
  - Digikam::AbstractCheckableAlbumTreeView, 182
  - Digikam::AlbumFolderViewSideBarWidget, 276
  - Digikam::AutoTagsScanWidget, 427
  - Digikam::DateFolderView, 756
  - Digikam::DateFolderViewSideBarWidget, 759
- Digikam::FaceScanWidget, 1482
- Digikam::FilterSideBarWidget, 1566
- Digikam::FuzzySearchSideBarWidget, 1603
- Digikam::FuzzySearchView, 1607
- Digikam::GPSSearchSideBarWidget, 1675
- Digikam::GPSSearchView, 1679
- Digikam::ImportItemPropertiesSideBarImport, 1905
- Digikam::ItemPropertiesSideBar, 2186
- Digikam::ItemPropertiesSideBarDB, 2192
- Digikam::LabelsSideBarWidget, 2305
- Digikam::LabelsTreeView, 2309
- Digikam::MapWidgetView, 2436
- Digikam::PeopleSideBarWidget, 2628
- Digikam::SearchSideBarWidget, 2909
- Digikam::SearchTextBar, 2914
- Digikam::Sidebar, 3008
- Digikam::StateSavingObject, 3046
- Digikam::TableView, 3070
- Digikam::TagCheckView, 3124
- Digikam::TagsManager, 3204
- Digikam::TagViewSideBarWidget, 3222
- Digikam::TimelineSideBarWidget, 3289
- ShowFoto::ShowfotoFolderViewSideBar, 3465
- ShowFoto::ShowfotoStackViewSideBar, 3516
- DOnlineTranslator
  - Digikam::DOnlineTranslator, 1171
- DOnlineTts
  - Digikam::DOnlineTts, 1182
- DontStretchPixels
  - Digikam::DRawDecoderSettings, 1261
- doSaveState
  - Digikam::AbstractAlbumTreeView, 159
  - Digikam::AbstractCheckableAlbumTreeView, 182
  - Digikam::AlbumFolderViewSideBarWidget, 277
  - Digikam::AutoTagsScanWidget, 427
  - Digikam::DateFolderView, 756
  - Digikam::DateFolderViewSideBarWidget, 759
  - Digikam::FaceScanWidget, 1482
  - Digikam::FilterSideBarWidget, 1566
  - Digikam::FuzzySearchSideBarWidget, 1603
  - Digikam::FuzzySearchView, 1607
  - Digikam::GPSSearchSideBarWidget, 1675
  - Digikam::GPSSearchView, 1679
  - Digikam::ImportItemPropertiesSideBarImport, 1906
  - Digikam::ItemPropertiesSideBar, 2186
  - Digikam::ItemPropertiesSideBarDB, 2192
  - Digikam::LabelsSideBarWidget, 2305
  - Digikam::LabelsTreeView, 2309
  - Digikam::MapWidgetView, 2436
  - Digikam::PeopleSideBarWidget, 2628
  - Digikam::SearchSideBarWidget, 2909
  - Digikam::SearchTextBar, 2914
  - Digikam::Sidebar, 3008
  - Digikam::StateSavingObject, 3046
  - Digikam::TableView, 3070
  - Digikam::TagCheckView, 3124

- Digikam::TagsManager, [3204](#)
- Digikam::TagViewSideBarWidget, [3222](#)
- Digikam::TimelineSideBarWidget, [3289](#)
- ShowFoto::ShowfotoFolderViewSideBar, [3465](#)
- ShowFoto::ShowfotoStackViewSideBar, [3516](#)
- downloaded
  - Digikam::CamItemInfo, [542](#)
- DownloadedNo
  - Digikam::CamItemInfo, [541](#)
- DownloadedYes
  - Digikam::CamItemInfo, [541](#)
- DownloadFailed
  - Digikam::CamItemInfo, [541](#)
- downloadItem
  - Digikam::GPCamera, [1632](#)
  - Digikam::UMSCamera, [3340](#)
- DownloadStarted
  - Digikam::CamItemInfo, [541](#)
- DownloadStatus
  - Digikam::CamItemInfo, [541](#)
- DownloadUnknown
  - Digikam::CamItemInfo, [541](#)
- DPlainTextEdit
  - Digikam::DPlainTextEdit, [1188](#)
- DPopupFrame
  - Digikam::DPopupFrame, [1237](#)
- dragDropFlags
  - Digikam::DragDropModelImplementation, [1248](#)
- dragDropFlagsV2
  - Digikam::DragDropModelImplementation, [1248](#)
- dragDropHandler
  - Digikam::DragDropViewImplementation, [1250](#)
  - Digikam::ImportCategorizedView, [1850](#)
  - Digikam::ItemCategorizedView, [2016](#)
  - Digikam::TableViewTreeView, [3115](#)
  - Digikam::VersionsTreeView, [3367](#)
  - ShowFoto::ShowfotoCategorizedView, [3432](#)
- DragDropModelImplementation
  - Digikam::DragDropModelImplementation, [1248](#)
- dragEnterEvent
  - Digikam::MapWidget, [2431](#)
- drawCategory
  - Digikam::DCategoryDrawer, [829](#)
  - Digikam::ImportCategoryDrawer, [1853](#)
  - Digikam::ItemCategoryDrawer, [2019](#)
- drawContents
  - Digikam::DColorValueSelector, [841](#)
  - Digikam::DHueSaturationSelector, [956](#)
  - Digikam::DPointSelect, [1233](#)
  - Digikam::DSelector, [1277](#)
- DRawDecoderWidget
  - Digikam::DRawDecoderWidget, [1265](#)
- DRawDecoding
  - Digikam::DRawDecoding, [1266](#)
- DRawInfo
  - Digikam::DRawInfo, [1270](#)
- dropEvent
  - Digikam::AbstractItemDragDropHandler, [195](#)
  - Digikam::AlbumDragDropHandler, [266](#)
  - Digikam::AlbumModelDragDropHandler, [311](#)
  - Digikam::ImportDragDropHandler, [1877](#)
  - Digikam::ItemDragDropHandler, [2051](#)
  - Digikam::MapDragDropHandler, [2422](#)
  - Digikam::TagDragDropHandler, [3127](#)
  - ShowFoto::ShowfotoDragDropHandler, [3447](#)
- dropMimeData
  - Digikam::TagMngrListModel, [3151](#)
- DTextEdit
  - Digikam::DTextEdit, [1288](#)
- DVD1
  - Digikam::VidSlideSettings, [3375](#)
- DVD2
  - Digikam::VidSlideSettings, [3375](#)
- DVGA
  - Digikam::VidSlideSettings, [3375](#)
- DWItemDelegate
  - Digikam::DWItemDelegate, [1308](#)
- DWItemDelegatePool
  - Digikam::DWItemDelegatePool, [1311](#)
- DynamicThread
  - Digikam::DynamicThread, [1320](#)
- eccentricity
  - Digikam::Ellipsoid, [1355](#)
- edgeDifference
  - Digikam::Graph< VertexProperties, EdgeProperties >, [1686](#)
- EdgesToLeaf
  - Digikam::Graph< VertexProperties, EdgeProperties >, [1686](#)
- EditableSearchTreeView
  - Digikam::EditableSearchTreeView, [1330](#)
- editGroup
  - Digikam::FileActionMngrDatabaseWorker, [1520](#)
- editKeyboardShortcuts
  - Digikam::DXmlGuiWindow, [1316](#)
- Editor
  - Digikam::DPluginAction, [1197](#)
- EditorColors
  - Digikam::DPluginAction, [1197](#)
- EditorDecorate
  - Digikam::DPluginAction, [1197](#)
- EditorEnhance
  - Digikam::DPluginAction, [1197](#)
- EditorFile
  - Digikam::DPluginAction, [1197](#)
- EditorFilters
  - Digikam::DPluginAction, [1197](#)
- EditorTransform
  - Digikam::DPluginAction, [1197](#)
- editRegion
  - Digikam::FacePipeline, [1424](#)
- editSearch
  - Digikam::NormalSearchTreeView, [2578](#)
- editTag
  - Digikam::FacePipeline, [1424](#)
- EDTV1

- Digikam::VidSlideSettings, [3375](#)
- EDTV2
  - Digikam::VidSlideSettings, [3375](#)
- EffectType
  - Digikam::EffectMgr, [1351](#)
- EGA
  - Digikam::VidSlideSettings, [3375](#)
- Ellipsoid
  - Digikam::Ellipsoid, [1354](#)
- Emotion
  - Digikam::DOnlineTts, [1181](#)
- emotion
  - Digikam::DOnlineTts, [1182](#)
- emotionCode
  - Digikam::DOnlineTts, [1182](#)
- emptyDTrashItems
  - Digikam::IOJobsThread, [1990](#)
- enableColumn
  - Digikam::DFontProperties, [945](#)
- enabled
  - Digikam::DConfigDlgWdgltem, [881](#)
- enableHistogramGuideByColor
  - Digikam::HistogramPainter, [1728](#)
- EnhanceTool
  - Digikam::BatchTool, [460](#)
- enqueue
  - Digikam::FacePipelineBase, [1429](#)
  - Digikam::ThumbnailImageCatcher, [3252](#)
- ensureHasItemInfo
  - Digikam::ItemModel, [2165](#)
- ensureIsPerson
  - Digikam::FaceTags, [1489](#)
- entries
  - Digikam::DImageHistory, [978](#)
- entryName
  - Digikam::StateSavingObject, [3046](#)
- error
  - Digikam::DBJobsThread, [805](#)
  - Digikam::DOnlineTranslator, [1172](#)
  - Digikam::DOnlineTts, [1183](#)
  - Digikam::ItemListerJobReceiver, [2142](#)
  - Digikam::ItemListerValueListReceiver, [2146](#)
- ErrorMessage
  - Digikam::DConfigDlgTitle, [862](#)
- errorMessage
  - Digikam::LookupAltitudeGeonames, [2397](#)
- errorsList
  - Digikam::DBJobsThread, [806](#)
  - Digikam::IOJobsThread, [1990](#)
- errorString
  - Digikam::DOnlineTranslator, [1172](#)
  - Digikam::DOnlineTts, [1183](#)
  - Digikam::ThumbnailCreator, [3248](#)
- escapeToken
  - Digikam::Rule, [2795](#)
- eventFilter
  - Digikam::BackendMarble, [444](#)
- excludeChildrenCount
  - Digikam::AbstractCountingAlbumModel, [188](#)
- ExcludeFadingOut
  - Digikam::ItemVisibilityController, [2283](#)
- ExcludeFolder
  - Digikam::Haarface, [1719](#)
- exclusiveMerge
  - Digikam::MetaEngineMergeHelper< Data, Key, KeyString, KeyStringList >, [2506](#)
- exec
  - Digikam::ContextMenuHelper, [636](#)
  - Digikam::ImportContextMenuHelper, [1859](#)
- execDBAction
  - Digikam::BdEngineBackend, [476](#)
- execDBActionQuery
  - Digikam::BdEngineBackend, [476](#)
- execDialog
  - Digikam::Setup, [2951](#)
- execDirectSql
  - Digikam::BdEngineBackend, [476](#)
- execDirectSqlWithResult
  - Digikam::BdEngineBackend, [476](#)
- execQuery
  - Digikam::BdEngineBackend, [477](#)
- execSinglePage
  - Digikam::Setup, [2951](#)
  - ShowFoto::ShowfotoSetup, [3497](#)
- execSql
  - Digikam::BdEngineBackend, [477](#)
- execUpsertDBAction
  - Digikam::BdEngineBackend, [477](#)
- execute
  - Digikam::LoadingTask, [2369](#)
  - Digikam::PreviewLoadingTask, [2649](#)
  - Digikam::SavingTask, [2806](#)
  - Digikam::SharedLoadingTask, [2981](#)
  - Digikam::ThumbnailLoadingTask, [3258](#)
- ExifHumanList
  - Digikam, [141](#)
- exifOrientation
  - Digikam::MetaEngineRotation, [2510](#)
- ExifToolBackend
  - Digikam::MetaEngine, [2484](#)
- ExifToolData
  - Digikam::ExifToolParser, [1384](#)
- exifTransform
  - Digikam::JPEGUtils::JpegRotator, [2287](#)
- Exiv2Backend
  - Digikam::MetaEngine, [2484](#)
- expandEverything
  - Digikam::AbstractAlbumTreeView, [159](#)
- expandMatches
  - Digikam::AbstractAlbumTreeView, [160](#)
- ExplicitBranch
  - Digikam::FilterAction, [1555](#)
- expoCorrectionHighlight
  - Digikam::DRawDecoderSettings, [1261](#)
- expoCorrectionShift
  - Digikam::DRawDecoderSettings, [1261](#)

- exportChanges
  - Digikam::MetaEngine, [2487](#)
- exportWidget
  - Digikam::DPluginDImg, [1216](#)
  - Digikam::DPluginLoader, [1226](#)
- exposureIndex
  - Digikam::DRawInfo, [1271](#)
- exposureIndicatorMode
  - Digikam::ExposureSettingsContainer, [1399](#)
- exposureProgram
  - Digikam::DRawInfo, [1271](#)
- extraAboutData
  - Digikam::DPlugin, [1192](#)
  - Digikam::DPluginDImg, [1216](#)
- extraAboutDataRowTitles
  - Digikam::DPlugin, [1192](#)
  - Digikam::DPluginDImg, [1216](#)
- extraAboutDataTitle
  - Digikam::DPlugin, [1192](#)
  - Digikam::DPluginDImg, [1216](#)
- extractJsonForItem
  - Digikam::DTrash, [1294](#)
- Extractor
  - Digikam::MLPipelineFoundation, [2526](#)
- extractor
  - Digikam::FacePipelineDetect, [1433](#)
  - Digikam::FacePipelineDetectRecognize, [1437](#)
  - Digikam::FacePipelineEdit, [1441](#)
  - Digikam::FacePipelineRecognize, [1455](#)
  - Digikam::FacePipelineReset, [1459](#)
  - Digikam::FacePipelineRetrain, [1463](#)
- extractRAWData
  - Digikam::DRawDecoder, [1255](#)
- extraData
  - Digikam::Album, [257](#)
- ExtraDataDuplicateCount
  - Digikam::ImportItemModel, [1898](#)
  - Digikam::ItemModel, [2164](#)
  - ShowFoto::ShowfotoItemModel, [3475](#)
- ExtraDataRole
  - Digikam::ImportItemModel, [1898](#)
  - Digikam::ItemModel, [2164](#)
  - ShowFoto::ShowfotoItemModel, [3475](#)
- ExtraRoles
  - Digikam::CategorizedItemModel, [558](#)
- FACE
  - Digikam::Album, [256](#)
- FACE\_TEMPLATE
  - Digikam, [142](#)
- faceCount
  - Digikam::ItemInfo, [2119](#)
- FaceDbAccess
  - Digikam::FaceDbAccess, [1405](#)
- FaceDbAccessUnlock
  - Digikam::FaceDbAccessUnlock, [1406](#)
- FaceDetectionModel
  - Digikam::FaceScanSettings, [1479](#)
- FaceDetector
  - Digikam::FaceDetector, [1412](#)
- faceenum2size
  - Digikam, [142](#)
- FaceRecognitionModel
  - Digikam::FaceScanSettings, [1479](#)
- faceRectToDisplayRect
  - Digikam::FaceUtils, [1501](#)
- FaceType
  - Digikam::DConfigDlg, [847](#)
  - Digikam::DConfigDlgView, [869](#)
- FastButLargePreview
  - Digikam::PreviewSettings, [2658](#)
- fastNumberToString
  - Digikam, [137](#)
- FastPreview
  - Digikam::PreviewSettings, [2658](#)
- FavoriteFolder
  - ShowFoto::ShowfotoStackViewFavoriteItem, [3506](#)
- FavoriteItem
  - ShowFoto::ShowfotoStackViewFavoriteItem, [3506](#)
- FavoriteRoot
  - ShowFoto::ShowfotoStackViewFavoriteItem, [3506](#)
- FavoriteType
  - ShowFoto::ShowfotoStackViewFavoriteItem, [3506](#)
- FFmpegBackend
  - Digikam::MetaEngine, [2484](#)
- fieldOperator
  - Digikam::SearchXmlReader, [2938](#)
- FILE\_NOT\_SUPPORTED
  - Digikam::DNGWriter, [1115](#)
- fileChanged
  - Digikam::LoadingCache, [2358](#)
- FileDate
  - ShowFoto::ShowfotoFolderViewList, [3462](#)
  - ShowFoto::ShowfotoStackViewList, [3513](#)
- fileModified
  - Digikam::ItemScanner, [2216](#)
- fileOriginData
  - Digikam::DImg, [989](#)
- filePath
  - Digikam::IccProfile, [1765](#)
- FileSaveOptionsBox
  - Digikam::FileSaveOptionsBox, [1534](#)
- FileScanMode
  - Digikam::CollectionScanner, [599](#)
- fileSize
  - Digikam::ItemInfo, [2119](#)
- fileUrl
  - Digikam::ItemInfo, [2119](#)
- fill
  - Digikam::DImg, [990](#)
- fillCommonContainer
  - Digikam::ItemScanner, [2216](#)
- fillFromOtherCurves
  - Digikam::ImageCurves, [1794](#)
- fillTemplate
  - Digikam::ItemCopyright, [2033](#)
- fillVideoMetadataContainer

- Digikam::ItemScanner, 2216
- filterAction
  - Digikam::AntiVignettingFilter, 376
  - Digikam::AutoExpoFilter, 413
  - Digikam::AutoLevelsFilter, 419
  - Digikam::BCGFilter, 470
  - Digikam::BlurFilter, 491
  - Digikam::BlurFXFilter, 496
  - Digikam::BorderFilter, 509
  - Digikam::BWSepiaFilter, 521
  - Digikam::CBFilter, 563
  - Digikam::CharcoalFilter, 570
  - Digikam::ColorFXFilter, 609
  - Digikam::ContentAwareFilter, 627
  - Digikam::CurvesFilter, 706
  - Digikam::DImgBuiltinFilter, 997
  - Digikam::DImgThreadedFilter, 1023
  - Digikam::DistortionFXFilter, 1053
  - Digikam::EmbossFilter, 1363
  - Digikam::EqualizeFilter, 1372
  - Digikam::FilmFilter, 1545
  - Digikam::FilmGrainFilter, 1550
  - Digikam::FilterActionFilter, 1561
  - Digikam::FreeRotationFilter, 1594
  - Digikam::GreycstorationFilter, 1705
  - Digikam::HotPixelFixer, 1741
  - Digikam::HSLFilter, 1753
  - Digikam::IccTransformFilter, 1788
  - Digikam::InfraredFilter, 1974
  - Digikam::InvertFilter, 1982
  - Digikam::LensDistortionFilter, 2316
  - Digikam::LensFunFilter, 2323
  - Digikam::LevelsFilter, 2330
  - Digikam::LocalContrastFilter, 2387
  - Digikam::MixerFilter, 2520
  - Digikam::NormalizeFilter, 2571
  - Digikam::NRFilter, 2589
  - Digikam::OilPaintFilter, 2595
  - Digikam::RainDropFilter, 2705
  - Digikam::RawProcessingFilter, 2737
  - Digikam::RedEyeCorrectionFilter, 2753
  - Digikam::RefocusFilter, 2759
  - Digikam::SharpenFilter, 2993
  - Digikam::ShearFilter, 2999
  - Digikam::StretchFilter, 3059
  - Digikam::TextureFilter, 3242
  - Digikam::TonalityFilter, 3300
  - Digikam::UnsharpMaskFilter, 3355
  - Digikam::WBFilter, 3388
- filterAlbum
  - Digikam::AbstractAlbumModel, 152
- FilterBehavior
  - Digikam::AlbumFilterModel, 270
- filterIcon
  - Digikam::DImgFilterManager, 1005
- filterIdentifier
  - Digikam::AntiVignettingFilter, 376
  - Digikam::AutoExpoFilter, 413
  - Digikam::AutoLevelsFilter, 419
  - Digikam::BCGFilter, 470
  - Digikam::BlurFilter, 491
  - Digikam::BlurFXFilter, 496
  - Digikam::BorderFilter, 509
  - Digikam::BWSepiaFilter, 521
  - Digikam::CBFilter, 563
  - Digikam::CharcoalFilter, 570
  - Digikam::ColorFXFilter, 609
  - Digikam::ContentAwareFilter, 627
  - Digikam::CurvesFilter, 706
  - Digikam::DImgThreadedFilter, 1023
  - Digikam::DistortionFXFilter, 1053
  - Digikam::EmbossFilter, 1363
  - Digikam::EqualizeFilter, 1372
  - Digikam::FilmFilter, 1545
  - Digikam::FilmGrainFilter, 1550
  - Digikam::FilterActionFilter, 1561
  - Digikam::FreeRotationFilter, 1594
  - Digikam::GreycstorationFilter, 1705
  - Digikam::HotPixelFixer, 1741
  - Digikam::HSLFilter, 1753
  - Digikam::IccTransformFilter, 1788
  - Digikam::InfraredFilter, 1974
  - Digikam::InvertFilter, 1982
  - Digikam::LensDistortionFilter, 2316
  - Digikam::LensFunFilter, 2323
  - Digikam::LevelsFilter, 2330
  - Digikam::LocalContrastFilter, 2387
  - Digikam::MixerFilter, 2520
  - Digikam::NormalizeFilter, 2571
  - Digikam::NRFilter, 2589
  - Digikam::OilPaintFilter, 2595
  - Digikam::RainDropFilter, 2705
  - Digikam::RawProcessingFilter, 2737
  - Digikam::RedEyeCorrectionFilter, 2753
  - Digikam::RefocusFilter, 2759
  - Digikam::SharpenFilter, 2993
  - Digikam::ShearFilter, 2999
  - Digikam::StretchFilter, 3059
  - Digikam::TextureFilter, 3242
  - Digikam::TonalityFilter, 3300
  - Digikam::UnsharpMaskFilter, 3355
  - Digikam::WBFilter, 3388
- filterImage
  - Digikam::DImgThreadedFilter, 1024
  - Digikam::FilterActionFilter, 1562
  - Digikam::IccTransformFilter, 1788
  - Digikam::RawProcessingFilter, 2737
  - Digikam::WBFilter, 3388
- filterMatchesForText
  - Digikam::ItemFilterModel, 2066
- FilterMode
  - Digikam::FacePipeline, 1423
  - Digikam::FacePipelineBase, 1428
- filterModel
  - Digikam::ImportCategorizedView, 1850
  - Digikam::ItemCategorizedView, 2016



- Digikam::ItemViewCategorized, [2260](#)
- ShowFoto::ShowfotoCategorizedView, [3433](#)
- FilterModelRoles
  - Digikam::ItemModel, [2164](#)
- FilterSideBarWidget
  - Digikam::FilterSideBarWidget, [1566](#)
- FiltersTool
  - Digikam::BatchTool, [460](#)
- find
  - Digikam::ThumbnailLoadThread, [3265](#)
- findAlbum
  - Digikam::AlbumManager, [292](#)
- findAndDeleteSpacersOrNewTags
  - Digikam::RGTagModel, [2785](#)
- findByFilePath
  - Digikam::ThumbsDb, [3270](#)
- findDAlbum
  - Digikam::AlbumManager, [292](#)
- findDuplicates
  - Digikam::Haarlface, [1719](#)
- Finder
  - Digikam::MLPipelineFoundation, [2526](#)
- finder
  - Digikam::FacePipelineDetect, [1433](#)
  - Digikam::FacePipelineDetectRecognize, [1437](#)
  - Digikam::FacePipelineEdit, [1441](#)
  - Digikam::FacePipelineRecognize, [1455](#)
  - Digikam::FacePipelineReset, [1459](#)
  - Digikam::FacePipelineRetrain, [1463](#)
- FinderMode
  - Digikam::NewItemFinder, [2557](#)
- findExecutable
  - Digikam::DFileOperations, [939](#)
- findGroup
  - Digikam::ThumbnailLoadThread, [3265](#)
- findIdentity
  - Digikam::FacialRecognitionWrapper, [1504](#)
  - Digikam::IdentityProvider, [1791](#)
- findImageld
  - Digikam::CoreDB, [657](#)
- findInDownloadHistory
  - Digikam::CoreDB, [657](#)
- findItembyId
  - Digikam::ProgressManager, [2674](#)
- findOrCreateTAlbums
  - Digikam::AlbumManager, [293](#)
- findOriginalText
  - Digikam::SqueezedComboBox, [3039](#)
- findPAlbum
  - Digikam::AlbumManager, [293](#)
- findSAlbum
  - Digikam::AlbumManager, [294](#)
- findSAlbumsBySearchType
  - Digikam::AlbumManager, [294](#)
- findTAlbum
  - Digikam::AlbumManager, [295](#)
- findWidgets
  - Digikam::DWItemDelegatePool, [1311](#)
- FingerPrintsGenerator
  - Digikam::FingerPrintsGenerator, [1575](#)
- finish
  - Digikam::SearchXmlWriter, [2942](#)
- finishCompleteScan
  - Digikam::CollectionScanner, [600](#)
- finished
  - Digikam::DImgThreadedFilter, [1024](#)
- finishedScanningAlbumRoot
  - Digikam::CollectionScanner, [600](#)
- finishField
  - Digikam::SearchXmlWriter, [2942](#)
- finishFileMetadataWrite
  - Digikam::ScanController, [2810](#)
- finishGroup
  - Digikam::SearchXmlWriter, [2942](#)
- firstChild
  - Digikam::Album, [258](#)
- fitToSize
  - Digikam::ImageZoomSettings, [1842](#)
- Flag
  - Digikam::AbstractAlbumTreeView, [158](#)
  - Digikam::FilterAction, [1555](#)
- flags
  - Digikam::TableModel, [3111](#)
- FLASH
  - Digikam::VidSlideSettings, [3374](#)
- flashUsed
  - Digikam::DRawInfo, [1271](#)
- FlipHorizontal
  - Digikam::MetaEngineRotation, [2510](#)
- FlipVertical
  - Digikam::MetaEngineRotation, [2510](#)
- FlushSignals
  - Digikam::WorkerObject, [3396](#)
- fo
  - Digikam::GeodeticCalculator, [1615](#)
- focusedIndex
  - Digikam::DWItemDelegate, [1309](#)
- FocusPoint
  - Digikam::FocusPoint, [1579](#)
- FolderViewRole
  - ShowFoto::ShowfotoFolderViewList, [3461](#)
- font
  - Digikam::DFontProperties, [946](#)
- FontColumn
  - Digikam::DFontProperties, [944](#)
- FontDiff
  - Digikam::DFontProperties, [944](#)
- fontDiffFlags
  - Digikam::DFontProperties, [946](#)
- FontListCriteria
  - Digikam::DFontProperties, [944](#)
- fontRoleData
  - Digikam::AbstractAlbumModel, [152](#)
  - Digikam::TagModel, [3166](#)
- FORMAT
  - Digikam::DImg, [984](#)

- Digikam::FileSaveOptionsBox, [1533](#)
- format
  - Digikam::DImg, [990](#)
- ForRecognition
  - Digikam::FacePipelineFaceTagsIface, [1447](#)
- fromAlbumAndName
  - Digikam::CoreDbUrl, [693](#)
- fromDateForMonth
  - Digikam::CoreDbUrl, [694](#)
- fromDateForYear
  - Digikam::CoreDbUrl, [694](#)
- fromDateRange
  - Digikam::CoreDbUrl, [694](#)
- fromFileUrl
  - Digikam::CoreDbUrl, [694](#)
- fromInfo
  - Digikam::ItemHistoryGraph, [2098](#)
- fromMarbleCoordinates
  - Digikam::GeoCoordinates, [1608](#)
- fromSourceIndex
  - Digikam::RGTagModel, [2785](#)
- fromTagIds
  - Digikam::CoreDbUrl, [694](#)
- fromVariant
  - Digikam::FaceTagsIface, [1498](#)
- FS\_ALBUMGUI
  - Digikam, [134](#)
- FS\_EDITOR
  - Digikam, [134](#)
- FS\_IMPORTUI
  - Digikam, [134](#)
- FS\_LIGHTTABLE
  - Digikam, [134](#)
- FS\_NONE
  - Digikam, [134](#)
- FS\_SIDEBARS
  - Digikam, [134](#)
- FS\_STATUSBAR
  - Digikam, [134](#)
- FS\_THUMBBAR
  - Digikam, [134](#)
- FS\_TOOLBARS
  - Digikam, [134](#)
- FULL\_SIZE
  - Digikam::DNGWriter, [1116](#)
- FullFiltering
  - Digikam::AlbumFilterModel, [270](#)
- FullImage
  - Digikam::ImageIface, [1806](#)
- FullImageHistogram
  - Digikam, [135](#)
- FullScreenOptions
  - Digikam, [134](#)
- FullWrite
  - Digikam::DisjointMetadata, [1044](#)
  - Digikam::MetadataHub, [2440](#)
- FullWritelfChanged
  - Digikam::DisjointMetadata, [1044](#)
- Digikam::MetadataHub, [2440](#)
- FuzzySelection
  - Digikam::TimeLineWidget, [3293](#)
- generatedName
  - Digikam::AlbumLabelsSearchHandler, [281](#)
- generateTagsList
  - Digikam::AutoTagsAssign, [419](#)
- generateUrls
  - Digikam::DOnlineTts, [1183](#)
- Generic
  - Digikam::DPluginAction, [1197](#)
- GenericExport
  - Digikam::DPluginAction, [1197](#)
- GenericImport
  - Digikam::DPluginAction, [1197](#)
- GenericMetadata
  - Digikam::DPluginAction, [1197](#)
- GenericTool
  - Digikam::DPluginAction, [1197](#)
- GenericView
  - Digikam::DPluginAction, [1197](#)
- geoCoordinates
  - Digikam::BackendGoogleMaps, [437](#)
  - Digikam::BackendMarble, [444](#)
- GeodeticCalculator
  - Digikam::GeodeticCalculator, [1611](#)
- GeoGroupStateEnum
  - Digikam, [135](#)
- GeofaceHelperParseLatLonString
  - Digikam, [137](#)
- GeofaceMinMarkerGroupingRadius
  - Digikam, [142](#)
- GeoPainter\_drawPixmapAtCoordinates
  - Digikam::BackendMarble, [445](#)
- getActiveState
  - Digikam::MapWidgetView, [2436](#)
- getAlbumAndSubalbumsForPath
  - Digikam::CoreDB, [657](#)
- getAlbumAverageDate
  - Digikam::CoreDB, [658](#)
- getAlbumForPath
  - Digikam::CoreDB, [658](#)
- getAlbumHighestDate
  - Digikam::CoreDB, [658](#)
- getAlbumLowestDate
  - Digikam::CoreDB, [659](#)
- getAlbumModificationDate
  - Digikam::CoreDB, [659](#)
- getAlbumModificationMap
  - Digikam::CoreDB, [659](#)
- getAlbumRelativePath
  - Digikam::CoreDB, [659](#)
- getAlbumRootId
  - Digikam::CoreDB, [660](#)
- getAlbumRoots
  - Digikam::CoreDB, [660](#)
- getAlbumsOnAlbumRoot
  - Digikam::CoreDB, [660](#)



- getAlbumThumbnail
  - Digikam::AlbumThumbnailLoader, [352](#)
- getAlbumThumbnailDirectly
  - Digikam::AlbumThumbnailLoader, [352](#)
- getAllItemsWithAlbum
  - Digikam::CoreDB, [660](#)
- getAudioCodecsProperties
  - Digikam::FFMpegConfigHelper, [1509](#)
- getCaption
  - Digikam::AlbumFolderViewSideBarWidget, [277](#)
  - Digikam::DateFolderViewSideBarWidget, [759](#)
  - Digikam::FuzzySearchSideBarWidget, [1603](#)
  - Digikam::GPSSearchSideBarWidget, [1675](#)
  - Digikam::LabelsSideBarWidget, [2305](#)
  - Digikam::PeopleSideBarWidget, [2628](#)
  - Digikam::SearchSideBarWidget, [2909](#)
  - Digikam::SidebarWidget, [3013](#)
  - Digikam::TagViewSideBarWidget, [3222](#)
  - Digikam::TimelineSideBarWidget, [3289](#)
- getCenter
  - Digikam::BackendGoogleMaps, [437](#)
  - Digikam::BackendMarble, [445](#)
- getClosestNeighbors
  - Digikam::KDTreeBase, [2296](#)
- getColorInfos
  - Digikam::MapWidget, [2431](#)
- getColumnFlags
  - Digikam::TableViewColumn, [3072](#)
  - Digikam::TableViewColumns::ColumnAudioVideoProperties, [3079](#)
  - Digikam::TableViewColumns::ColumnDigikamProperties, [3083](#)
  - Digikam::TableViewColumns::ColumnFileProperties, [3088](#)
  - Digikam::TableViewColumns::ColumnGeoProperties, [3093](#)
  - Digikam::TableViewColumns::ColumnItemProperties, [3097](#)
  - Digikam::TableViewColumns::ColumnPhotoProperties, [3102](#)
  - Digikam::TableViewColumns::ColumnThumbnail, [3106](#)
- getComments
  - Digikam::MetaEngine, [2487](#)
- getCommentsDecoded
  - Digikam::MetaEngine, [2488](#)
- getComposer
  - Digikam::DColorComposer, [837](#)
- getConfigGroup
  - Digikam::StateSavingObject, [3047](#)
- getConfigurationWidget
  - Digikam::TableViewColumns::ColumnFileProperties, [3088](#)
  - Digikam::TableViewColumns::ColumnGeoProperties, [3093](#)
  - Digikam::TableViewColumns::ColumnPhotoProperties, [3102](#)
- getCopyrightInformation
  - Digikam::DMetadata, [1090](#)
- getCoreDatabaseNameOrDir
  - Digikam::DbEngineParameters, [787](#)
- getCurrentHighlightState
  - Digikam::SearchTextBar, [2914](#)
- getCustomProperty
  - Digikam::DConfigDlgMngr, [855](#)
- getCustomPropertyChangedSignal
  - Digikam::DConfigDlgMngr, [856](#)
- getDAAlbumsCount
  - Digikam::AlbumManager, [295](#)
- getDatabaseEncoding
  - Digikam::CoreDB, [660](#)
- getDatabaseFieldsRaw
  - Digikam::ItemInfo, [2119](#)
- getDbValue
  - Digikam::CommonKeys, [620](#)
  - Digikam::DbKeysCollection, [808](#)
  - Digikam::MetadataKeys, [2447](#)
  - Digikam::PositionKeys, [2644](#)
- getDecoratedPixmapForCluster
  - Digikam::MapWidget, [2432](#)
- getDescription
  - Digikam::TableViewColumns::ColumnDigikamProperties, [3083](#)
- getDigitizationDateTime
  - Digikam::MetaEngine, [2488](#)
- getDirtyOrMissingFacelImageUrls
  - Digikam::CoreDB, [661](#)
- getDirtyOrMissingFingerprints
  - Digikam::SimilarityDb, [3015](#)
- getDirtyOrMissingFingerprintURLs
  - Digikam::SimilarityDb, [3016](#)
- getErrorMessage
  - Digikam::BackendGeonamesRG, [430](#)
  - Digikam::BackendGeonamesUSRG, [433](#)
  - Digikam::BackendOsmRG, [452](#)
  - Digikam::RGBBackend, [2779](#)
- getExifComment
  - Digikam::MetaEngine, [2488](#)
- getExifEncoded
  - Digikam::MetaEngine, [2488](#)
- getExifTagComment
  - Digikam::MetaEngine, [2488](#)
- getExifTagData
  - Digikam::MetaEngine, [2488](#)
- getExifTagLong
  - Digikam::MetaEngine, [2489](#)
- getExifTagRational
  - Digikam::MetaEngine, [2489](#)
- getExifTagsDataList
  - Digikam::MetaEngine, [2489](#)
- getExifTagString
  - Digikam::MetaEngine, [2489](#)
- getExifTagVariant
  - Digikam::MetaEngine, [2490](#)
- getExifThumbnail
  - Digikam::MetaEngine, [2490](#)

- getExtensionsProperties
  - Digikam::FFMpegConfigHelper, [1509](#)
- getFaceCount
  - Digikam::AlbumManager, [295](#)
- getFaceEmbedding
  - Digikam::DNNOpenFaceExtractor, [1140](#)
  - Digikam::DNNSFaceExtractor, [1145](#)
- getFilterSettings
  - Digikam::CoreDB, [661](#)
- getFolders
  - Digikam::GPCamera, [1633](#)
  - Digikam::UMSCamera, [3340](#)
- getFontList
  - Digikam::DFontProperties, [946](#)
- getFreeSpace
  - Digikam::DKCamera, [1068](#)
  - Digikam::GPCamera, [1633](#)
  - Digikam::UMSCamera, [3340](#)
- getGlobalGroupState
  - Digikam::GPSMarkerTiler, [1667](#)
  - Digikam::ItemMarkerTiler, [2155](#)
- getGPSInfo
  - Digikam::MetaEngine, [2490](#)
- getGPSLatitudeNumber
  - Digikam::MetaEngine, [2490](#)
- getGPSLatitudeString
  - Digikam::MetaEngine, [2490](#)
- getGroupingOperateOnAll
  - Digikam::ApplicationSettings, [383](#)
- getHSL
  - Digikam::DColor, [834](#)
- getIccProfile
  - Digikam::DMetadata, [1090](#)
- getIcon
  - Digikam::AlbumFolderViewSideBarWidget, [277](#)
  - Digikam::DateFolderViewSideBarWidget, [760](#)
  - Digikam::FuzzySearchSideBarWidget, [1604](#)
  - Digikam::GPSSearchSideBarWidget, [1676](#)
  - Digikam::LabelsSideBarWidget, [2306](#)
  - Digikam::PeopleSideBarWidget, [2629](#)
  - Digikam::SearchSideBarWidget, [2910](#)
  - Digikam::SidebarWidget, [3013](#)
  - Digikam::TagViewSideBarWidget, [3223](#)
  - Digikam::TimelineSideBarWidget, [3290](#)
- getIdenticalFiles
  - Digikam::CoreDB, [661](#)
- getImageId
  - Digikam::CoreDB, [661](#)
- getImageIds
  - Digikam::CoreDB, [661](#), [662](#)
- getImageMetadata
  - Digikam::CoreDB, [663](#)
- getImageFields
  - Digikam::CoreDB, [663](#)
- getImageSimilarity
  - Digikam::SimilarityDb, [3016](#)
- getImageSimilarityAlgorithms
  - Digikam::SimilarityDb, [3016](#)
- getImagesRelatedFrom
  - Digikam::CoreDB, [663](#)
- getImagesRelatingTo
  - Digikam::CoreDB, [663](#)
- getImageTagProperties
  - Digikam::CoreDB, [663](#)
- getIptc
  - Digikam::MetaEngine, [2490](#)
- getIptcKeywords
  - Digikam::MetaEngine, [2491](#)
- getIptcSubCategories
  - Digikam::MetaEngine, [2491](#)
- getIptcSubjects
  - Digikam::MetaEngine, [2491](#)
- getIptcTagData
  - Digikam::MetaEngine, [2491](#)
- getIptcTagsDataList
  - Digikam::MetaEngine, [2491](#)
- getIptcTagsStringList
  - Digikam::MetaEngine, [2491](#)
- getIptcTagString
  - Digikam::MetaEngine, [2492](#)
- getItemAlbum
  - Digikam::CoreDB, [663](#)
- getItemColorWorkSpace
  - Digikam::MetaEngine, [2492](#)
- getItemCommonTagIDs
  - Digikam::CoreDB, [664](#)
- getItemCopyright
  - Digikam::CoreDB, [664](#)
- getItemDateTime
  - Digikam::MetaEngine, [2492](#)
- getItemDimensions
  - Digikam::MetaEngine, [2492](#)
- getItemFacesMap
  - Digikam::DMetadata, [1091](#)
- getItemFromAlbum
  - Digikam::AlbumManager, [295](#)
  - Digikam::CoreDB, [664](#)
- getItemIDsAndURLsInAlbum
  - Digikam::CoreDB, [664](#)
- getItemIDsInAlbum
  - Digikam::CoreDB, [665](#)
- getItemIDsInTag
  - Digikam::CoreDB, [665](#)
- getItemInfo
  - Digikam::GPCamera, [1633](#)
  - Digikam::UMSCamera, [3341](#)
- getItemInformation
  - Digikam::CoreDB, [665](#)
- getItemName
  - Digikam::CoreDB, [666](#)
- getItemNamesInAlbum
  - Digikam::CoreDB, [666](#)
- getItemOrientation
  - Digikam::MetaEngine, [2492](#)
- getItemPosition
  - Digikam::CoreDB, [666](#)

- getItemPreview
  - Digikam::MetaEngine, [2492](#)
- getItemInfoList
  - Digikam::DKCamera, [1068](#)
  - Digikam::GPCamera, [1633](#)
  - Digikam::UMSCamera, [3341](#)
- getItemTagIDs
  - Digikam::CoreDB, [666](#)
- getItemTagIDs
  - Digikam::CoreDB, [666](#)
- getItemTagNames
  - Digikam::CoreDB, [667](#)
- getItemURLsInAlbum
  - Digikam::CoreDB, [667](#)
- getItemURLsInTag
  - Digikam::CoreDB, [667](#)
- getLegacySetting
  - Digikam::SimilarityDb, [3017](#)
- getLensDescription
  - Digikam::DMetadata, [1091](#)
- getMarkerModelLevel
  - Digikam::BackendGoogleMaps, [437](#)
  - Digikam::BackendMarble, [445](#)
- getMetadata
  - Digikam::GPCamera, [1633](#)
  - Digikam::UMSCamera, [3341](#)
- getMetadataField
  - Digikam::DMetadata, [1091](#)
- getMetadataTitle
  - Digikam::ExifWidget, [1397](#)
  - Digikam::ICCPProfileWidget, [1776](#)
  - Digikam::IptcWidget, [1996](#)
  - Digikam::MakerNoteWidget, [2410](#)
  - Digikam::XmpWidget, [3417](#)
- getMimeType
  - Digikam::MetaEngine, [2492](#)
- getModel
  - Digikam::DNNModelManager, [1133](#)
- getNewConfiguration
  - Digikam::TableViewColumns::ColumnFileConfigurationWidget, [3085](#)
  - Digikam::TableViewColumns::ColumnGeoConfigurationWidget, [3090](#)
  - Digikam::TableViewColumns::ColumnPhotoConfigurationWidget, [3099](#)
- getNormalizedBounds
  - Digikam::BackendGoogleMaps, [437](#)
  - Digikam::BackendMarble, [445](#)
- getNumberOfAllItemsAndAlbums
  - Digikam::CoreDB, [668](#)
- getNumberOfItemsInAlbum
  - Digikam::CoreDB, [668](#)
- getOneRelatedImageEach
  - Digikam::CoreDB, [668](#)
- getOrCreate
  - Digikam::TagProperties, [3173](#)
- getOrCreateTag
  - Digikam::TagsCache, [3187](#)
- getOrCreateTagForPerson
  - Digikam::FaceTags, [1490](#)
- getOrCreateTagWithProperty
  - Digikam::TagsCache, [3187](#)
- getOriginalRegionImage
  - Digikam::ImageRegionWidget, [1829](#)
- getPALbumsCount
  - Digikam::AlbumManager, [296](#)
- getPixelColor
  - Digikam::DImg, [990](#)
- getPixelSize
  - Digikam::MetaEngine, [2493](#)
- getPreview
  - Digikam::DKCamera, [1068](#)
  - Digikam::GPCamera, [1633](#)
  - Digikam::UMSCamera, [3341](#)
- getProjection
  - Digikam::BackendMarble, [445](#)
- getRecentlyAssignedTags
  - Digikam::AlbumManager, [296](#)
  - Digikam::CoreDB, [668](#)
- getRelationCloud
  - Digikam::CoreDB, [669](#)
- getRequest
  - Digikam::LookupAltitudeGeonames, [2398](#)
- getRequests
  - Digikam::LookupAltitudeGeonames, [2398](#)
- getSemanticInfo
  - Digikam::BalooWrap, [454](#)
- getSetting
  - Digikam::CoreDB, [669](#)
  - Digikam::SimilarityDb, [3017](#)
- getSpacerAddress
  - Digikam::RGTagModel, [2786](#)
- getSpacers
  - Digikam::RGTagModel, [2786](#)
- getStandardTagIcon
  - Digikam::AlbumThumbnailLoader, [352](#)
- getStateSavingDepth
  - Digikam::StateSavingObject, [3047](#)
- getStatus
  - Digikam::LookupAltitudeGeonames, [2398](#)
- getStringComparisonType
  - Digikam::ApplicationSettings, [384](#)
- getSuggestedNames
  - Digikam::FaceTagsEditor, [1494](#)
  - Digikam::ItemInfo, [2119](#)
- getTagDescription
  - Digikam::ExifWidget, [1397](#)
  - Digikam::ICCPProfileWidget, [1776](#)
  - Digikam::IptcWidget, [1996](#)
  - Digikam::MakerNoteWidget, [2410](#)
  - Digikam::XmpWidget, [3417](#)
- getTagRects
  - Digikam::FaceTagsEditor, [1494](#)
- getTagsWithProperty
  - Digikam::CoreDB, [669](#)
- getTagThumbnail

- Digikam::AlbumThumbnailLoader, [352](#)
- getTagThumbnailDirectly
  - Digikam::AlbumThumbnailLoader, [352](#)
- getTagTitle
  - Digikam::ExifWidget, [1397](#)
  - Digikam::ICCPProfileWidget, [1776](#)
  - Digikam::IptcWidget, [1996](#)
  - Digikam::MakerNoteWidget, [2410](#)
  - Digikam::XmpWidget, [3417](#)
- getTagType
  - Digikam::RGTagModel, [2786](#)
- getTAlbumsCount
  - Digikam::AlbumManager, [296](#)
- getTemporaryHaarTitle
  - Digikam::SAlbum, [2802](#)
- getTemporaryTitle
  - Digikam::SAlbum, [2803](#)
- getThreshold
  - Digikam::DNNFaceExtractorBase, [1128](#)
  - Digikam::DNNModelBase, [1130](#)
  - Digikam::DNNOpenFaceExtractor, [1140](#)
  - Digikam::DNNSFaceExtractor, [1145](#)
- getThumbInfo
  - Digikam::CameraThumbsCtrl, [539](#)
- getThumbnail
  - Digikam::GPCamera, [1634](#)
  - Digikam::UMSCamera, [3341](#)
- getThumbnailSize
  - Digikam::TrashView, [3325](#)
- getTile
  - Digikam::AbstractMarkerTiler, [197](#)
  - Digikam::GPSMarkerTiler, [1668](#)
  - Digikam::ItemMarkerTiler, [2155](#)
- getTileGroupState
  - Digikam::AbstractMarkerTiler, [198](#)
  - Digikam::GPSMarkerTiler, [1668](#)
  - Digikam::ItemMarkerTiler, [2155](#)
- getTileMarkerCount
  - Digikam::GPSMarkerTiler, [1668](#)
  - Digikam::ItemMarkerTiler, [2155](#)
- getTileRepresentativeMarker
  - Digikam::AbstractMarkerTiler, [198](#)
  - Digikam::GPSMarkerTiler, [1668](#)
  - Digikam::ItemMarkerTiler, [2155](#)
- getTileSelectedCount
  - Digikam::GPSMarkerTiler, [1669](#)
  - Digikam::ItemMarkerTiler, [2155](#)
- getTitle
  - Digikam::TableViewColumns::ColumnAudioVideoProperties, [3079](#)
  - Digikam::TableViewColumns::ColumnDigikamProperties, [3084](#)
  - Digikam::TableViewColumns::ColumnFileProperties, [3088](#)
  - Digikam::TableViewColumns::ColumnGeoProperties, [3094](#)
  - Digikam::TableViewColumns::ColumnItemProperties, [3097](#)
  - Digikam::TableViewColumns::ColumnPhotoProperties, [3102](#)
  - Digikam::TableViewColumns::ColumnThumbnail, [3106](#)
- getUnconfirmedFaceCount
  - Digikam::AlbumManager, [296](#)
- getUniqueHash
  - Digikam::DImg, [990](#)
- getUniqueHashVersion
  - Digikam::CoreDB, [669](#)
  - Digikam::DImg, [990](#)
- getUniqueID
  - Digikam::ProgressManager, [2674](#)
- getUserFilterSettings
  - Digikam::CoreDB, [669](#)
- getValue
  - Digikam::DbKeysCollection, [808](#)
- getVideoCodecsProperties
  - Digikam::FFMpegConfigHelper, [1509](#)
- getVideoMetadata
  - Digikam::CoreDB, [669](#)
- getXmp
  - Digikam::MetaEngine, [2493](#)
- getXmpKeywords
  - Digikam::DMetadata, [1092](#)
  - Digikam::MetaEngine, [2493](#)
- getXmpSubCategories
  - Digikam::DMetadata, [1092](#)
  - Digikam::MetaEngine, [2493](#)
- getXmpSubjects
  - Digikam::DMetadata, [1092](#)
  - Digikam::MetaEngine, [2493](#)
- getXmpTagsDataList
  - Digikam::MetaEngine, [2493](#)
- getXmpTagString
  - Digikam::MetaEngine, [2494](#)
- getXmpTagStringBag
  - Digikam::MetaEngine, [2494](#)
- getXmpTagStringLangAlt
  - Digikam::MetaEngine, [2494](#)
- getXmpTagStringListLangAlt
  - Digikam::MetaEngine, [2494](#)
- getXmpTagStringSeq
  - Digikam::MetaEngine, [2494](#)
- getXmpTagVariant
  - Digikam::MetaEngine, [2495](#)
- getYCbCr
  - Digikam::DColor, [834](#)
- getZoom
  - Digikam::BackendGoogleMaps, [437](#)
  - Digikam::BackendMarble, [445](#)
- GivenAsArgument
  - Digikam::FacePipelineFaceTagsIface, [1447](#)
- globalID
  - Digikam::Album, [258](#)
- GlobalSettings
  - Digikam::ImageQualityConfSelector, [1814](#)
- goldenStarPixmap

- Digikam::LabelsTreeView, [2309](#)
- GPSListing
  - Digikam::GPSDBJobsThread, [1645](#)
- GPSMarkerTiler
  - Digikam::GPSMarkerTiler, [1667](#)
- GPSSearchView
  - Digikam::GPSSearchView, [1679](#)
- granularity
  - Digikam::DImgLoaderObserver, [1010](#)
- GreycstorationFilter
  - Digikam::GreycstorationFilter, [1705](#)
- gridSize
  - Digikam::DItemDelegate, [1055](#)
  - Digikam::ItemViewDelegate, [2267](#)
  - Digikam::ItemViewImportDelegate, [2274](#)
  - ShowFoto::ShowfotoItemViewDelegate, [3484](#)
- groupCaption
  - Digikam::SearchXmlReader, [2939](#)
- GroupedImagesFinder
  - Digikam::GroupedImagesFinder, [1707](#)
- groupImage
  - Digikam::ItemInfo, [2119](#)
- GroupsOpenRole
  - Digikam::ItemFilterModel, [2065](#)
- groupLabelPixmap
  - Digikam::SearchView, [2929](#)
- groupOperator
  - Digikam::SearchXmlReader, [2939](#)
- halfSizeColorImage
  - Digikam::DRawDecoderSettings, [1261](#)
- handbookChapter
  - Digikam::DPlugin, [1192](#)
- handbookReference
  - Digikam::DPlugin, [1193](#)
- handbookSection
  - Digikam::DPlugin, [1193](#)
- handleCustomContextMenuAction
  - Digikam::AbstractAlbumTreeView, [160](#)
  - Digikam::AlbumSelectTreeView, [345](#)
  - Digikam::EditableSearchTreeView, [1330](#)
  - Digikam::NormalSearchTreeView, [2578](#)
  - Digikam::TagFilterView, [3137](#)
  - Digikam::TagFolderView, [3144](#)
- handleQueryResult
  - Digikam::BdEngineBackend, [477](#)
- HAS\_RESULT
  - Digikam::SearchTextBar, [2914](#)
- hasActions
  - Digikam::DImageHistory, [978](#)
- hasCloseButton
  - Digikam::DDatePicker, [896](#)
- hasDirtyOrMissingFingerprint
  - Digikam::SimilarityDb, [3017](#)
- hasEdges
  - Digikam::ItemHistoryGraph, [2098](#)
- hasErrors
  - Digikam::DBJobsThread, [806](#)
  - Digikam::IOJobsThread, [1991](#)
- hasFingerprint
  - Digikam::SimilarityDb, [3017](#)
- hasFingerprints
  - Digikam::SimilarityDb, [3018](#)
- hasHiddenGroupedImages
  - Digikam::DigikamItemView, [975](#)
  - Digikam::GroupingViewImplementation, [1714](#)
  - Digikam::ItemThumbnailBar, [2240](#)
  - Digikam::TableViewTreeView, [3115](#)
- hasParameters
  - Digikam::FilterAction, [1555](#)
- hasProperty
  - Digikam::TagsCache, [3188](#)
- hasRegionSelection
  - Digikam::GeofaceSharedData, [1621](#)
- hasSearchResult
  - Digikam::AlbumFilterModel, [271](#)
- hasTags
  - Digikam::CoreDB, [670](#)
- hasThumbnail
  - Digikam::ProgressItem, [2665](#)
- hasTransparentPixels
  - Digikam::DImg, [991](#)
- hasVisibilityProperty
  - Digikam::DPlugin, [1193](#)
  - Digikam::DPluginBqm, [1201](#)
  - Digikam::DPluginDImg, [1216](#)
- hasVisibleItems
  - Digikam::ItemVisibilityController, [2284](#)
- HDPLUS
  - Digikam::VidSlideSettings, [3376](#)
- HDTV
  - Digikam::VidSlideSettings, [3376](#)
- headerData
  - Digikam::TrackListModel, [3311](#)
- HeaderRole
  - Digikam::DConfigDlgModel, [859](#)
- HEIFFiles
  - Digikam::MimeFilter, [2515](#)
- heightForWidth
  - Digikam::DNotificationWidget, [1163](#)
- hide
  - Digikam::AbstractWidgetDelegateOverlay, [209](#)
  - Digikam::ImportRatingOverlay, [1925](#)
  - Digikam::ItemRatingOverlay, [2203](#)
  - Digikam::PersistentWidgetDelegateOverlay, [2633](#)
  - Digikam::TagsLineEditOverlay, [3201](#)
- hideAndRemoveItem
  - Digikam::ItemVisibilityController, [2284](#)
- hideAnimationFinished
  - Digikam::DNotificationWidget, [1163](#)
- HidingStateChanger
  - Digikam::HidingStateChanger, [1725](#)
- hierarchyFromParent
  - ShowFoto::ShowfotoStackViewFavoriteItem, [3506](#)
- HighlightState
  - Digikam::SearchTextBar, [2914](#)
- HighQualityPreview

- Digikam::PreviewSettings, 2658
- hintAtModificationOfItems
  - Digikam::ScanController, 2811
- hintAtMoveOrCopyOfAlbum
  - Digikam::ScanController, 2811
- hintAtMoveOrCopyOfItems
  - Digikam::ScanController, 2811
- HistogramPainter
  - Digikam::HistogramPainter, 1728
- HistogramRenderingType
  - Digikam, 135
- HistogramScale
  - Digikam, 135
- HistogramWidget
  - Digikam::HistogramWidget, 1733
- HistoryLoadingFlag
  - Digikam::ItemHistoryGraph, 2097
- hover
  - Digikam::ActionItemModel, 216
- HS\_None
  - Digikam, 135
- HSXGA
  - Digikam::VidSlideSettings, 3376
- HttpProxy
  - Digikam::SystemSettings, 3065
- HudSide
  - Digikam, 135
- hue
  - Digikam::DColorValueSelector, 841
  - Digikam::DHueSaturationSelector, 957
- humanReadableBytesCount
  - Digikam::ItemPropertiesTab, 2196
- HUXGA
  - Digikam::VidSlideSettings, 3376
- HVGA
  - Digikam::VidSlideSettings, 3375
- HXGA
  - Digikam::VidSlideSettings, 3376
- IccManager
  - Digikam::IccManager, 1759
- IccPostLoadingManager
  - Digikam::IccPostLoadingManager, 1762
- IccProfilesComboBox
  - Digikam::IccProfilesComboBox, 1769
- icon
  - Digikam::DNotificationWidget, 1163
  - Digikam::DPlugin, 1193
  - Digikam::FaceRejectionOverlayButton, 1477
  - Digikam::ImportRotateOverlayButton, 1935
  - Digikam::ItemFullScreenOverlayButton, 2088
  - Digikam::ItemRotateOverlayButton, 2212
  - Digikam::ItemSelectionOverlayButton, 2224
  - Digikam::ItemViewHoverButton, 2269
- Id
  - Digikam::TrackManager, 3312
- id
  - Digikam::Album, 259
  - Digikam::ItemInfo, 2120
  - Digikam::ProgressItem, 2665
  - Digikam::Token, 3295
- identifier
  - Digikam::FilterAction, 1555
- Identity
  - Digikam::Identity, 1789
- ids
  - Digikam::CollectionImageChangeset, 586
  - Digikam::DbKeysCollection, 808
- ifacelid
  - Digikam::DPlugin, 1193
  - Digikam::DPluginBqm, 1201
  - Digikam::DPluginDImg, 1216
  - Digikam::DPluginEditor, 1220
  - Digikam::DPluginGeneric, 1223
  - Digikam::DPluginRawImport, 1230
- ignoredCharacters
  - Digikam::DPlainTextEdit, 1189
  - Digikam::DTextEdit, 1288
- ignoredWords
  - Digikam::LocalizeContainer, 2389
- IgnoreRootAlbum
  - Digikam::AbstractAlbumModel, 151
- iid
  - Digikam::DPlugin, 1193
- image
  - Digikam::EmptyImageListProvider, 1367
  - Digikam::MetaEnginePreviews, 2507
  - Digikam::QListImageListProvider, 2686
- image2Mat
  - Digikam, 137
- image2Mat\_shared
  - Digikam, 138
- ImageAlignment
  - Digikam::DConfigDlgTitle, 861
- imageChange
  - Digikam::CoreDbWatch, 698
- ImageChangeset
  - Digikam::ImageChangeset, 1791
- imageComments
  - Digikam::ItemInfo, 2120
- imageCopyright
  - Digikam::ItemInfo, 2120
- imageExtendedProperties
  - Digikam::ItemInfo, 2120
- imageFilterModel
  - Digikam::ImageSortFilterModel, 1832
  - Digikam::ItemFilterModel, 2067
- imageHistory
  - Digikam::ItemInfo, 2120
- ImageIface
  - Digikam::ImageIface, 1806
- imageInfo
  - Digikam::ItemModel, 2165
- imageInformationRect
  - Digikam::ImportDelegate, 1870
  - Digikam::ItemDelegate, 2040
  - Digikam::ItemViewDelegate, 2267



- Digikam::ItemViewImportDelegate, 2275
- ShowFoto::ShowfotoDelegate, 3444
- ShowFoto::ShowfotoItemViewDelegate, 3484
- imageInfosAboutToBeAdded
  - Digikam::ItemModel, 2165
- imageInfosAboutToBeRemoved
  - Digikam::ItemModel, 2166
- imageInfosAdded
  - Digikam::ItemModel, 2166
- imageInfosCleared
  - Digikam::ItemModel, 2166
  - Digikam::ItemThumbnailModel, 2253
- imageInfosRemoved
  - Digikam::ItemModel, 2166
- imageInfosSorted
  - Digikam::ImageSortFilterModel, 1832
- ImageLeft
  - Digikam::DConfigDlgTitle, 862
- imageLoaded
  - Digikam::LoadSaveThread, 2379
- ImageMagickBackend
  - Digikam::MetaEngine, 2484
- imageModel
  - Digikam::ItemHistoryGraphModel, 2107
- imageModelIndex
  - Digikam::ItemHistoryGraphModel, 2107
- imageProcessed
  - Digikam::DuplicatesProgressObserver, 1304
- ImageQualitySorter
  - Digikam::ImageQualitySorter, 1820
- ImageRight
  - Digikam::DConfigDlgTitle, 862
- images
  - Digikam::EmptyImageListProvider, 1367
  - Digikam::QListImageListProvider, 2686
  - Digikam::RecognitionTrainingProvider, 2742
  - Digikam::TrainingDataProvider, 3318
- imageSaved
  - Digikam::LoadSaveThread, 2379
- imageSavedAs
  - Digikam::DImg, 991
- ImageSelection
  - Digikam::Imageface, 1806
- ImageSelectionHistogram
  - Digikam, 135
- imageStartedLoading
  - Digikam::LoadSaveThread, 2379
- imageStartedSaving
  - Digikam::LoadSaveThread, 2380
- imageUrls
  - Digikam::AlbumLabelsSearchHandler, 282
- ImportContextMenuHelper
  - Digikam::ImportContextMenuHelper, 1856
- importFilterModel
  - Digikam::ImportCategorizedView, 1850
  - Digikam::ImportFilterModel, 1885
  - Digikam::ImportSortFilterModel, 1941
- ImportFilterModelPointerRole
  - Digikam::ImportFilterModel, 1884
- ImportFilterModelRoles
  - Digikam::ImportFilterModel, 1884
- ImportItemModelPointerRole
  - Digikam::ImportItemModel, 1898
- ImportItemModelRoles
  - Digikam::ImportItemModel, 1898
- ImportThumbnailModel
  - Digikam::ImportThumbnailModel, 1961
- Inactive
  - Digikam::FocusPoint, 1579
- includeChildrenCount
  - Digikam::AbstractCountingAlbumModel, 188
- IncludeFadingOut
  - Digikam::ItemVisibilityController, 2283
- IncludeFadingOutMode
  - Digikam::ItemVisibilityController, 2283
- IncludeLeadingSlash
  - Digikam::TagsCache, 3187
- IncludeRootAlbum
  - Digikam::AbstractAlbumModel, 151
- incrementedCounter
  - Digikam::DefaultVersionNamingScheme, 918
  - Digikam::VersionNamingScheme, 3360
- indent
  - Digikam::DSelector, 1277
- index
  - Digikam::DConfigDlgWdgModel, 886
  - Digikam::TrackListModel, 3311
- indexActivated
  - Digikam::ImportCategorizedView, 1850
  - Digikam::ItemCategorizedView, 2016
  - ShowFoto::ShowfotoCategorizedView, 3433
- indexForInfo
  - Digikam::ItemHistoryGraphModel, 2107
- indexForPath
  - Digikam::ItemModel, 2166
- indexForUrl
  - Digikam::ImportItemModel, 1899
  - ShowFoto::ShowfotoItemModel, 3476
- indexFromImageId
  - Digikam::TableViewModel, 3111
- indexVisuallyAt
  - Digikam::AbstractAlbumTreeView, 160
- indicesEqual
  - Digikam::AbstractMarkerTiler, 198
  - Digikam::GPSTiler, 1669
  - Digikam::ItemMarkerTiler, 2156
- InFocus
  - Digikam::FocusPoint, 1579
- infoForId
  - Digikam::ItemInfoCache, 2125
- infoForPath
  - Digikam::ItemInfoCache, 2125
- infoFromItem
  - Digikam::TableViewModel, 3111
- infoface
  - Digikam::DigikamApp, 962

- Digikam::DXmlGuiWindow, [1316](#)
- Digikam::ImageWindow, [1841](#)
- Digikam::ImportUI, [1965](#)
- Digikam::LightTableWindow, [2350](#)
- Digikam::QueueMgrWindow, [2694](#)
- ShowFoto::Showfoto, [3426](#)
- InfoMessage
  - Digikam::DConfigDlgTitle, [862](#)
- infosLessThan
  - Digikam::ImportFilterModel, [1885](#)
  - Digikam::ItemFilterModel, [2067](#)
  - ShowFoto::ShowfotoFilterModel, [3453](#)
- init
  - Digikam::DConfigDlgMgr, [856](#)
  - Digikam::DPluginLoader, [1226](#)
- initDbEngineErrorHandler
  - Digikam::SimilarityDbAccess, [3021](#)
- initExifTool
  - Digikam::ExifToolProcess, [1391](#)
- initFilter
  - Digikam::DImgThreadedFilter, [1024](#)
- initFrom
  - Digikam::HistogramPainter, [1729](#)
- initialCounter
  - Digikam::DefaultVersionNamingScheme, [918](#)
  - Digikam::VersionNamingScheme, [3361](#)
- initializeExiv2
  - Digikam::MetaEngine, [2495](#)
- initializeNoThumbnailStorage
  - Digikam::ThumbnailLoadThread, [3265](#)
- initializeThumbnailDatabase
  - Digikam::ThumbnailLoadThread, [3265](#)
- initSchema
  - Digikam::CoreDbBackend, [686](#)
  - Digikam::FaceDbBackend, [1411](#)
  - Digikam::SimilarityDbBackend, [3026](#)
  - Digikam::ThumbsDbBackend, [3276](#)
- initSlave
  - Digikam::DImgThreadedFilter, [1024](#)
- Input
  - Digikam::IccProfile, [1764](#)
- InputColorSpace
  - Digikam::DRawDecoderSettings, [1260](#)
- inputColorSpace
  - Digikam::DRawDecoderSettings, [1261](#)
- insertFaceVector
  - Digikam::FaceDb, [1404](#)
- insertItem
  - Digikam::DExpanderBox, [933](#)
- insertPage
  - Digikam::DConfigDlg, [850](#), [851](#)
  - Digikam::DConfigDlgWdg, [876](#)
  - Digikam::DConfigDlgWdgModel, [886](#)
- insertSqueezedItem
  - Digikam::SqueezedComboBox, [3039](#)
- insertSqueezedList
  - Digikam::SqueezedComboBox, [3039](#)
- installLineEdit
  - Digikam::TreeViewLineEditComboBox, [3332](#)
- installView
  - Digikam::AbstractAlbumTreeViewSelectComboBox, [167](#)
  - Digikam::AlbumSelectComboBox, [328](#)
  - Digikam::ChoiceSearchComboBox, [577](#)
  - Digikam::ListViewComboBox, [2355](#)
  - Digikam::StayPoppedUpComboBox, [3053](#)
  - Digikam::TreeViewComboBox, [3329](#)
  - Digikam::TreeViewLineEditComboBox, [3332](#)
- INSTANCE
  - Digikam::StateSavingObject, [3046](#)
- instance
  - Digikam::AlbumThumbnailLoader, [353](#)
  - Digikam::DatabaseServerStarter, [738](#)
  - Digikam::DBJobsManager, [802](#)
  - Digikam::DMetadataSettings, [1096](#)
  - Digikam::DNNModelManager, [1133](#)
  - Digikam::DPluginLoader, [1226](#)
  - Digikam::GeolocationSettings, [1624](#)
  - Digikam::IccSettings, [1779](#)
  - Digikam::IOJobsManager, [1986](#)
  - Digikam::ItemSortCollator, [2229](#)
  - Digikam::LocalizeSettings, [2393](#)
  - Digikam::MetaEngineSettings, [2511](#)
  - Digikam::NetworkManager, [2554](#)
  - Digikam::ProgressManager, [2675](#)
- instructions
  - Digikam::ItemCopyright, [2033](#)
- integrityCheck
  - Digikam::SimilarityDb, [3018](#)
- intellectualGenre
  - Digikam::ItemExtendedProperties, [2053](#)
- Intermediate
  - Digikam::HistoryImageld, [1735](#)
- intermediateDirectory
  - Digikam::DefaultVersionNamingScheme, [919](#)
- intermediateFileName
  - Digikam::DefaultVersionNamingScheme, [919](#)
  - Digikam::VersionNamingScheme, [3361](#)
- intersects
  - Digikam::TagRegion, [3181](#)
- invalidatePaintingCache
  - Digikam::ImportDelegate, [1870](#)
  - Digikam::ItemDelegate, [2040](#)
  - Digikam::ItemViewImportDelegate, [2275](#)
- InvalidBehavior
  - Digikam::ICCSettingsContainer, [1781](#)
- InvalidType
  - Digikam::DPluginAction, [1197](#)
  - Digikam::IccProfile, [1764](#)
- inverseFlattening
  - Digikam::Ellipsoid, [1355](#)
- invertSelection
  - Digikam::TableView, [3070](#)
- iptcCorePropertyName
  - Digikam::ItemScanner, [2217](#)
- IptcHumanList



- Digikam, [142](#)
- isAccessible
  - Digikam::ThumbnailInfo, [3254](#)
- isAlbumRoot
  - Digikam::CollectionManager, [593](#)
- isAlbumUrl
  - Digikam::CoreDbUrl, [695](#)
- isAncestorOf
  - Digikam::Album, [259](#)
- IsAppendRole
  - Digikam::SetupCollectionModel, [2960](#)
- isCanceled
  - Digikam::DuplicatesProgressObserver, [1304](#)
  - Digikam::IOJobsThread, [1991](#)
- isCancelled
  - Digikam::BatchTool, [461](#)
- isCategorizedModel
  - Digikam::DCategorizedSortFilterProxyModel, [821](#)
- IsCategoryRole
  - Digikam::SetupCollectionModel, [2960](#)
- isCheckable
  - Digikam::LabelsTreeView, [2309](#)
- isClearButtonEnabled
  - Digikam::DPlainTextEdit, [1189](#)
  - Digikam::DTextEdit, [1288](#)
- isCloseButtonVisible
  - Digikam::DNotificationWidget, [1164](#)
- isComplexAction
  - Digikam::FilterActionFilter, [1562](#)
- isConnected
  - Digikam::GPSModelIndexProxyMapper, [1672](#)
- IsDeleteRole
  - Digikam::SetupCollectionModel, [2960](#)
- isEmpty
  - Digikam::CurvesContainer, [701](#)
  - Digikam::DTrashItemModel, [1297](#)
  - Digikam::ItemPosition, [2170](#)
  - Digikam::ProgressManager, [2675](#)
- isFiltering
  - Digikam::AlbumFilterModel, [271](#)
  - Digikam::CheckableAlbumFilterModel, [574](#)
  - Digikam::SearchFilterModel, [2889](#)
  - Digikam::TagPropertiesFilterModel, [3177](#)
- isHideAnimationRunning
  - Digikam::DNotificationWidget, [1164](#)
- isInitialized
  - Digikam::SimilarityDbAccess, [3021](#)
- isInTransaction
  - Digikam::BdEngineBackend, [478](#)
- isIvfDefinitive
  - Digikam::Ellipsoid, [1355](#)
- isLoadingState
  - Digikam::LabelsTreeView, [2309](#)
- isMovingAlbum
  - Digikam::AlbumManager, [296](#)
- isReadOnly
  - Digikam::DDateEdit, [891](#)
  - Digikam::DImg, [991](#)
- isReady
  - Digikam::BackendGoogleMaps, [438](#)
  - Digikam::BackendMarble, [446](#)
- isRefreshing
  - Digikam::ImportItemModel, [1899](#)
  - Digikam::ItemModel, [2166](#)
- isRestoreCheckState
  - Digikam::AbstractCheckableAlbumTreeView, [182](#)
- isRestoringSelectionFromHistory
  - Digikam::AlbumLabelsSearchHandler, [282](#)
- isRoot
  - Digikam::Album, [259](#)
- isRunning
  - Digikam::DOnlineTranslator, [1172](#)
- isShowAnimationRunning
  - Digikam::DNotificationWidget, [1164](#)
- isSourceTranscriptionEnabled
  - Digikam::DOnlineTranslator, [1172](#)
- isSourceTranslitEnabled
  - Digikam::DOnlineTranslator, [1173](#)
- isSphere
  - Digikam::Ellipsoid, [1356](#)
- isStoredLosslessly
  - Digikam::CurvesContainer, [701](#)
- isSupported
  - Digikam::DImgFilterGenerator, [1003](#)
  - Digikam::DImgFilterManager, [1006](#)
- isSupportTranslation
  - Digikam::DOnlineTranslator, [1173](#)
- isTagListDirty
  - Digikam::GPSItemContainer, [1651](#)
- isTemporarySearch
  - Digikam::SAlbum, [2803](#)
- isTranslationOptionsEnabled
  - Digikam::DOnlineTranslator, [1173](#)
- isTranslationTranslitEnabled
  - Digikam::DOnlineTranslator, [1173](#)
- isTrashAlbum
  - Digikam::Album, [259](#)
- IsUpdateRole
  - Digikam::SetupCollectionModel, [2960](#)
- isUsedByLabelsTree
  - Digikam::Album, [259](#)
- isValid
  - Digikam::Rule, [2796](#)
- isValue
  - Digikam::DbEngineActionType, [780](#)
- isVisible
  - Digikam::SearchField, [2826](#)
- item
  - Digikam::DConfigDlgWdgModel, [887](#)
  - Digikam::SqueezedComboBox, [3040](#)
- itemChanged
  - Digikam::ItemPropertiesSideBarDB, [2192](#)
- ItemComments
  - Digikam::ItemComments, [2023](#)
- itemCoordinates
  - Digikam::GeoModelHelper, [1626](#)

- Digikam::GPSBookmarkModelHelper, [1636](#)
- Digikam::GPSGeofaceModelHelper, [1647](#)
- Digikam::ItemGPSModelHelper, [2095](#)
- Digikam::MapViewModelHelper, [2425](#)
- ItemCopyMoveHint
  - Digikam::ItemCopyMoveHint, [2030](#)
- ItemDelegateOverlayContainer
  - Digikam::ItemDelegateOverlayContainer, [2046](#)
- ItemFilterModelRoles
  - Digikam::ItemFilterModel, [2065](#)
- itemFlags
  - Digikam::GPSBookmarkModelHelper, [1636](#)
- itemForAction
  - Digikam::ActionItemModel, [216](#)
- itemHighlighted
  - Digikam::SqueezedComboBox, [3040](#)
- itemIcon
  - Digikam::GeoModelHelper, [1627](#)
  - Digikam::GPSBookmarkModelHelper, [1637](#)
- ItemInfo
  - Digikam::ItemInfo, [2117](#)
- itemInfo
  - Digikam::DBInfolface, [796](#)
  - Digikam::DMetalInfolface, [1100](#)
- itemInfosAboutToBeAdded
  - Digikam::ImportItemModel, [1899](#)
  - ShowFoto::ShowfotoItemModel, [3476](#)
- itemInfosAboutToBeRemoved
  - Digikam::ImportItemModel, [1900](#)
  - ShowFoto::ShowfotoItemModel, [3476](#)
- itemInfosAdded
  - Digikam::ImportItemModel, [1900](#)
  - ShowFoto::ShowfotoItemModel, [3476](#)
- itemInfosRemoved
  - Digikam::ImportItemModel, [1900](#)
  - ShowFoto::ShowfotoItemModel, [3476](#)
- ItemListType
  - Digikam::QueueListView, [2689](#)
- ItemModelPointerRole
  - Digikam::ItemModel, [2164](#)
- ItemModelRoles
  - Digikam::ItemModel, [2164](#)
- ItemModified
  - Digikam::ItemChangeHint, [2021](#)
- ItemOrderRole
  - Digikam::CategorizedItemModel, [558](#)
- ItemPosition
  - Digikam::ItemPosition, [2169](#)
- ItemRescan
  - Digikam::ItemChangeHint, [2021](#)
- items
  - Digikam::DDatePickerPopup, [900](#)
- itemScanInfo
  - Digikam::ItemScanner, [2217](#)
- ItemScanner
  - Digikam::ItemScanner, [2215](#)
- ItemTagPair
  - Digikam::ItemTagPair, [2233](#)
- ItemThumbnailModel
  - Digikam::ItemThumbnailModel, [2253](#)
- itemView
  - Digikam::DWItemDelegate, [1309](#)
- ItemVisibilityControllerPropertyObject
  - Digikam::ItemVisibilityControllerPropertyObject, [2286](#)
- jobData
  - Digikam::IOJobsThread, [1991](#)
- jobId
  - Digikam::ItemExtendedProperties, [2053](#)
- JPEGPreview
  - Digikam::DNGWriter, [1115](#)
- JpegRotator
  - Digikam::JPEGUtils::JpegRotator, [2287](#)
- JPEGSubSampling
  - Digikam::IOFileSettings, [1983](#)
- KDTreeBase
  - Digikam::KDTreeBase, [2295](#)
- KeepProfile
  - Digikam::ICCSettingsContainer, [1781](#)
- KeepSignals
  - Digikam::WorkerObject, [3396](#)
- keyPressed
  - Digikam::ItemViewCategorized, [2260](#)
- keywords
  - Digikam::DisjointMetadata, [1045](#)
- label
  - Digikam::ProgressItem, [2665](#)
- language
  - Digikam::DOnlineTranslator, [1174](#)
- LanguageChoiceBehavior
  - Digikam::ItemComments, [2022](#)
- languageCode
  - Digikam::DOnlineTranslator, [1174](#)
- languageName
  - Digikam::DOnlineTranslator, [1175](#)
- lastChild
  - Digikam::Album, [260](#)
- lastDescriptions
  - Digikam::ThumbnailLoadThread, [3266](#)
- lastError
  - Digikam::BdEngineBackend, [478](#)
- lastSavedFilePath
  - Digikam::DImg, [991](#)
- lastSelectedItemUrl
  - Digikam::TrashView, [3325](#)
- lastSQLException
  - Digikam::BdEngineBackend, [478](#)
- latitude
  - Digikam::ItemPosition, [2170](#)
- latitudeNumber
  - Digikam::ItemPosition, [2170](#)
- latitudeUserPresentableNumbers
  - Digikam::ItemPosition, [2170](#)
- LeadingSlashPolicy

- Digikam::TagsCache, [3186](#)
- leafImages
  - Digikam::ItemHistoryGraph, [2099](#)
- LeaveFileUntagged
  - Digikam::ICCSettingsContainer, [1781](#)
- leaves
  - Digikam::Graph< VertexProperties, EdgeProperties >, [1686](#)
- leftMargin
  - Digikam::DCategoryDrawer, [829](#)
- lessThan
  - Digikam::AlbumFilterModel, [271](#)
  - Digikam::CamItemSortSettings, [544](#)
  - Digikam::DCategorizedSortFilterProxyModel, [821](#)
  - Digikam::ItemSortSettings, [2231](#)
  - ShowFoto::ShowfotoItemSortSettings, [3479](#)
- lessThanByOrder
  - Digikam::ItemSortSettings, [2232](#)
- LibHeifBackend
  - Digikam::MetaEngine, [2484](#)
- libraryFileName
  - Digikam::DPlugin, [1194](#)
- LibRawBackend
  - Digikam::MetaEngine, [2484](#)
- librawUseGomp
  - Digikam::DRawDecoder, [1255](#)
- LibsInfoDlg
  - Digikam::LibsInfoDlg, [2332](#)
- lift
  - Digikam::CoreDbOperationGroup, [689](#)
  - Digikam::FaceDbOperationGroup, [1411](#)
- linkActivated
  - Digikam::DNotificationWidget, [1164](#)
- linkHovered
  - Digikam::DNotificationWidget, [1165](#)
- LinScale
  - Digikam::TimeLineWidget, [3292](#)
- LinScaleHistogram
  - Digikam, [135](#)
- ListAFPoints
  - Digikam::FocusPointsExtractor, [1586](#)
- listDTrashItems
  - Digikam::IOJobsThread, [1991](#)
- listHaarSearch
  - Digikam::ItemLister, [2135](#)
- listImageTagPropertySearch
  - Digikam::ItemLister, [2135](#)
- listPAlbum
  - Digikam::ItemLister, [2136](#)
- listPath
  - Digikam::Graph< VertexProperties, EdgeProperties >, [1686](#)
- listSearch
  - Digikam::ItemLister, [2136](#)
- ListViewComboBox
  - Digikam::ListViewComboBox, [2355](#)
- load
  - Digikam::DMetadata, [1092](#)
  - Digikam::ExifToolParser, [1386](#)
  - Digikam::ManagedLoadSaveThread, [2416](#)
  - Digikam::MetadataHub, [2441](#)
  - Digikam::MetaEngine, [2495](#)
  - Digikam::PreviewLoadThread, [2655](#)
  - Digikam::ThumbnailLoadThread, [3266](#)
  - Digikam::WorkflowManager, [3402](#)
- LOAD\_CHUNKS
  - Digikam::ExifToolProcess, [1390](#)
- LOAD\_METADATA
  - Digikam::ExifToolProcess, [1390](#)
- LoadAll
  - Digikam::DImgLoader, [1008](#)
- loadAllProfilesProperties
  - Digikam::IccSettings, [1779](#)
- loadChunk
  - Digikam::ExifToolParser, [1386](#)
- loadColumnProfile
  - Digikam::TableViewModel, [3111](#)
- loadDetail
  - Digikam::ThumbnailCreator, [3249](#)
- loadEmbeddedPreview
  - Digikam::DRawDecoder, [1255](#), [1256](#)
- Loader
  - Digikam::MLPipelineFoundation, [2526](#)
- loader
  - Digikam::FacePipelineDetect, [1433](#)
  - Digikam::FacePipelineDetectRecognize, [1437](#)
  - Digikam::FacePipelineEdit, [1442](#)
  - Digikam::FacePipelineRecognize, [1455](#)
  - Digikam::FacePipelineReset, [1459](#)
  - Digikam::FacePipelineRetrain, [1463](#)
- loaderName
  - Digikam::DPluginDImg, [1216](#)
- loadFast
  - Digikam::PreviewLoadThread, [2655](#)
- loadFastButLarge
  - Digikam::PreviewLoadThread, [2655](#)
- loadFastSynchronously
  - Digikam::PreviewLoadThread, [2655](#)
- LoadFlag
  - Digikam::DImgLoader, [1008](#)
- loadFromData
  - Digikam::MetaEngine, [2495](#)
- loadFromDataAndMerge
  - Digikam::MetaEngine, [2495](#)
- loadFromDisk
  - Digikam::ItemScanner, [2217](#)
- loadFromSidecarAndMerge
  - Digikam::MetaEngine, [2496](#)
- loadFromURL
  - Digikam::ExifWidget, [1397](#)
  - Digikam::ICCPProfileWidget, [1776](#)
  - Digikam::IptcWidget, [1996](#)
  - Digikam::MakerNoteWidget, [2410](#)
  - Digikam::XmpWidget, [3417](#)
- loadFullImage
  - Digikam::DRawDecoder, [1256](#)

- loadHalfPreview
  - Digikam::DRawDecoder, [1256](#)
- loadHighQuality
  - Digikam::PreviewLoadThread, [2656](#)
- LoadICCDData
  - Digikam::DImgLoader, [1008](#)
- LoadImageData
  - Digikam::DImgLoader, [1008](#)
- loadImageData
  - Digikam::GPSItemContainer, [1651](#)
  - Digikam::ItemGPS, [2093](#)
- LoadImageHistory
  - Digikam::DImgLoader, [1008](#)
- LoadingDescription
  - Digikam::LoadingDescription, [2364](#)
- LoadingMode
  - Digikam::ManagedLoadSaveThread, [2415](#)
- LoadingModeNormal
  - Digikam::ManagedLoadSaveThread, [2415](#)
- LoadingModeShared
  - Digikam::ManagedLoadSaveThread, [2415](#)
- LoadingPolicy
  - Digikam::ManagedLoadSaveThread, [2415](#)
- LoadingPolicyAppend
  - Digikam::ManagedLoadSaveThread, [2416](#)
- LoadingPolicyFirstRemovePrevious
  - Digikam::ManagedLoadSaveThread, [2416](#)
- LoadingPolicyPreload
  - Digikam::ManagedLoadSaveThread, [2416](#)
- LoadingPolicyPrepend
  - Digikam::ManagedLoadSaveThread, [2416](#)
- LoadingPolicySimpleAppend
  - Digikam::ManagedLoadSaveThread, [2416](#)
- LoadingPolicySimplePrepend
  - Digikam::ManagedLoadSaveThread, [2416](#)
- loadingProgress
  - Digikam::LoadSaveThread, [2380](#)
- LoadingTaskFilter
  - Digikam::ManagedLoadSaveThread, [2416](#)
- LoadingTaskFilterAll
  - Digikam::ManagedLoadSaveThread, [2416](#)
- LoadingTaskFilterPreloading
  - Digikam::ManagedLoadSaveThread, [2416](#)
- LoadItemInfo
  - Digikam::DImgLoader, [1008](#)
- loadItemInfo
  - Digikam::DImg, [991](#)
- loadItemInfos
  - Digikam::LightTableWindow, [2350](#)
- loadItemsForCollection
  - Digikam::DTrashItemModel, [1297](#)
- LoadLeavesHistory
  - Digikam::ItemHistoryGraph, [2098](#)
- LoadMetadata
  - Digikam::DImgLoader, [1008](#)
- loadModels
  - Digikam::DNNFaceExtractorBase, [1128](#)
  - Digikam::DNNOpenFaceExtractor, [1141](#)
  - Digikam::DNNResnetDetector, [1143](#)
  - Digikam::DNNSFaceExtractor, [1146](#)
  - Digikam::DNNYoloDetector, [1149](#)
- loadPlugins
  - Digikam::DPluginConfViewBqm, [1205](#)
  - Digikam::DPluginConfViewDImg, [1207](#)
  - Digikam::DPluginConfViewEditor, [1209](#)
  - Digikam::DPluginConfViewGeneric, [1211](#)
- LoadPreview
  - Digikam::DImgLoader, [1008](#)
- loadQImage
  - Digikam::Haarface, [1720](#)
- loadRawPreview
  - Digikam::DRawDecoder, [1256](#), [1257](#)
- LoadRelationCloud
  - Digikam::ItemHistoryGraph, [2098](#)
- loadSaveNotifier
  - Digikam::SharedLoadingTask, [2981](#)
- loadSettings
  - Digikam::TableViewColumnProfile, [3076](#)
- loadState
  - Digikam::AlbumSelectors, [338](#)
- LoadSubjectHistory
  - Digikam::ItemHistoryGraph, [2098](#)
- loadTrainingData
  - Digikam::FaceClassifier, [1401](#)
- LoadUniqueHash
  - Digikam::DImgLoader, [1008](#)
- localFileRename
  - Digikam::DFileOperations, [939](#)
- location
  - Digikam::ItemExtendedProperties, [2053](#)
- LocationAllRight
  - Digikam::CollectionManager, [592](#)
- LocationAvailable
  - Digikam::CollectionLocation, [587](#)
- LocationCheckResult
  - Digikam::CollectionManager, [591](#)
- LocationDeleted
  - Digikam::CollectionLocation, [587](#)
- locationForAlbumRoot
  - Digikam::CollectionManager, [593](#)
- locationForUrl
  - Digikam::CollectionManager, [593](#)
- LocationHasProblems
  - Digikam::CollectionManager, [592](#)
- LocationHidden
  - Digikam::CollectionLocation, [587](#)
- LocationInvalidCheck
  - Digikam::CollectionManager, [592](#)
- LocationNotAllowed
  - Digikam::CollectionManager, [592](#)
- LocationNull
  - Digikam::CollectionLocation, [587](#)
- locationStatusChanged
  - Digikam::CollectionManager, [594](#)
- LocationUnavailable
  - Digikam::CollectionLocation, [587](#)

- LogScale
  - Digikam::TimeLineWidget, [3292](#)
- LogScaleHistogram
  - Digikam, [135](#)
- longestPath
  - Digikam::Graph< VertexProperties, EdgeProperties >::Path, [1695](#)
- longestPathTouching
  - Digikam::Graph< VertexProperties, EdgeProperties >, [1687](#)
- longitudeNumber
  - Digikam::ItemInfo, [2120](#)
- lookupCacheKeys
  - Digikam::LoadingDescription, [2364](#)
- LTLeftPanelRole
  - Digikam::ItemModel, [2164](#)
- LTRightPanelRole
  - Digikam::ItemModel, [2164](#)
- m\_bin
  - Digikam::Haar::WeightBin, [1717](#)
- m\_cancel
  - Digikam::DRawDecoder, [1258](#)
- m\_category
  - Digikam::FilterAction, [1556](#)
- m\_decoderSettings
  - Digikam::DRawDecoder, [1258](#)
- m\_destinationValid
  - Digikam::GeodeticCalculator, [1615](#)
- m\_directionValid
  - Digikam::GeodeticCalculator, [1615](#)
- m\_inverseFlattening
  - Digikam::Ellipsoid, [1357](#)
- m\_ivfDefinitive
  - Digikam::Ellipsoid, [1357](#)
- m\_lat1
  - Digikam::GeodeticCalculator, [1615](#)
- m\_lat2
  - Digikam::GeodeticCalculator, [1615](#)
- m\_master
  - Digikam::DImgThreadedFilter, [1026](#)
- m\_originalUUID
  - Digikam::HistoryImageId, [1735](#)
- m\_semiMajorAxis
  - Digikam::Ellipsoid, [1358](#)
- m\_semiMinorAxis
  - Digikam::Ellipsoid, [1358](#)
- m\_slave
  - Digikam::DImgThreadedFilter, [1026](#)
- m\_templateTitle
  - Digikam::Template, [3228](#)
- m\_TOLERANCE\_CHECK
  - Digikam::GeodeticCalculator, [1615](#)
- m\_transformQue
  - Digikam::EditorWindow, [1351](#)
- m\_uuid
  - Digikam::HistoryImageId, [1735](#)
- mainMarbleWidget
  - Digikam::GeolocationSettings, [1624](#)
- makeColumnVisible
  - Digikam::DFontProperties, [946](#)
- makeQMapFromXML
  - Digikam::BackendGeonamesRG, [430](#)
  - Digikam::BackendGeonamesUSRG, [433](#)
  - Digikam::BackendOsmRG, [452](#)
- makeStaleAlbum
  - Digikam::CoreDB, [670](#)
- mapIndexForDragDrop
  - Digikam::DragDropViewImplementation, [1250](#)
  - Digikam::ItemViewCategorized, [2260](#)
  - Digikam::TableViewTreeView, [3115](#)
  - Digikam::VersionsTreeView, [3367](#)
- mapListToSource
  - Digikam::ImageSortFilterModel, [1832](#)
- mapSize
  - Digikam::BackendGoogleMaps, [438](#)
  - Digikam::BackendMarble, [446](#)
- mapToSourceImportModel
  - Digikam::ImportSortFilterModel, [1941](#)
- mapToSourceShowfotoModel
  - ShowFoto::ShowfotoSortFilterModel, [3504](#)
- mapWidget
  - Digikam::BackendGoogleMaps, [438](#)
  - Digikam::BackendMarble, [446](#)
  - Digikam::MapBackend, [2420](#)
- mapWidgetDocked
  - Digikam::BackendGoogleMaps, [438](#)
  - Digikam::BackendMarble, [446](#)
- MapWidgetView
  - Digikam::MapWidgetView, [2436](#)
- marbleCustomPaint
  - Digikam::BackendMarble, [446](#)
- MatchContainingFragment
  - Digikam::TaggingActionFactory, [3147](#)
- matches
  - Digikam::AlbumFilterModel, [271](#)
  - Digikam::CheckableAlbumFilterModel, [574](#)
  - Digikam::ItemFilterSettings, [2078](#)
  - Digikam::SearchFilterModel, [2889](#)
  - Digikam::TagPropertiesFilterModel, [3177](#)
  - Digikam::TagsManagerFilterModel, [3208](#)
- MatchResult
  - Digikam::AlbumFilterModel, [270](#)
- matchResult
  - Digikam::AlbumFilterModel, [271](#), [272](#)
- MatchStartingWithFragment
  - Digikam::TaggingActionFactory, [3147](#)
- maximumBoundValues
  - Digikam::BdEngineBackend, [478](#)
- maximumThumbnailSize
  - Digikam::ThumbnailLoadThread, [3266](#)
- maxValue
  - Digikam::DIntRangeBox, [1036](#)
- MeaningOfDirection
  - Digikam, [135](#)
- media
  - Digikam::DOnlineTts, [1184](#)

- MEDIUM
  - Digikam::DNGWriter, [1116](#)
- MenuCategoryFlag
  - Digikam::ActionItemModel, [215](#)
- mergeFields
  - Digikam::MetaEngineMergeHelper< Data, Key, KeyString, KeyStringList >, [2506](#)
- mergeTAlbum
  - Digikam::AlbumManager, [297](#)
- meridianArcLength
  - Digikam::GeodeticCalculator, [1613](#)
- meridianArcLengthRadians
  - Digikam::GeodeticCalculator, [1613](#)
- message
  - Digikam::DNotificationPopup, [1154–1157](#)
- MessageType
  - Digikam::DConfigDlgTitle, [862](#)
  - Digikam::DNotificationWidget, [1162](#)
- messageType
  - Digikam::DNotificationWidget, [1165](#)
- MetadataAvailable
  - Digikam::DisjointMetadataDataFields, [1048](#)
  - Digikam::MetadataHub, [2440](#)
- MetadataDisjoint
  - Digikam::DisjointMetadataDataFields, [1048](#)
- MetadataEditingAborted
  - Digikam::ItemMetadataAdjustmentHint, [2159](#)
- MetadataEditingFinished
  - Digikam::ItemMetadataAdjustmentHint, [2159](#)
- MetadataInvalid
  - Digikam::DisjointMetadataDataFields, [1048](#)
  - Digikam::MetadataHub, [2440](#)
- MetadataRemover
  - Digikam::MetadataRemover, [2460](#)
- MetadataSynchronizer
  - Digikam::MetadataSynchronizer, [2470](#)
- metadataTemplate
  - Digikam::DisjointMetadata, [1045](#)
- MetadataTool
  - Digikam::BatchTool, [460](#)
- MetadataWritingMode
  - Digikam::MetaEngine, [2485](#)
- metadataWritingMode
  - Digikam::MetaEngine, [2496](#)
- meteringMode
  - Digikam::DRawInfo, [1271](#)
- middleButtonPressed
  - Digikam::AbstractCheckableAlbumTreeView, [182](#)
- migrateAlbumRoot
  - Digikam::CoreDB, [670](#)
- migrateToVolume
  - Digikam::CollectionManager, [594](#)
- mimeType
  - Digikam::ThumbnailInfo, [3254](#)
- mimeTypes
  - Digikam::AbstractItemDragDropHandler, [195](#)
  - Digikam::AlbumDragDropHandler, [266](#)
  - Digikam::AlbumModelDragDropHandler, [311](#)
  - Digikam::ImportDragDropHandler, [1877](#)
  - Digikam::ItemDragDropHandler, [2052](#)
  - Digikam::TagDragDropHandler, [3128](#)
  - ShowFoto::ShowfotoDragDropHandler, [3447](#)
- minValue
  - Digikam::DIntRangeBox, [1036](#)
- MJPEG
  - Digikam::VidSlideSettings, [3374](#)
- MKV
  - Digikam::VidSlideSettings, [3375](#)
- MLPipelineStage
  - Digikam::MLPipelineFoundation, [2526](#)
- mocMetaObject
  - Digikam::ParallelAdapter< A >, [2617](#)
  - Digikam::ParallelWorkers, [2622](#)
- modDateTime
  - Digikam::ItemInfo, [2120](#)
- MODE
  - Digikam::GreycstorationFilter, [1705](#)
- Mode
  - Digikam::TagsPopupMenu, [3209](#)
- model
  - Digikam::AlbumSelectComboBox, [328](#)
  - Digikam::GeoModelHelper, [1627](#)
  - Digikam::GPSBookmarkModelHelper, [1637](#)
  - Digikam::GPSGeofaceModelHelper, [1648](#)
  - Digikam::ItemGPSModelHelper, [2096](#)
  - Digikam::MapViewModelHelper, [2425](#)
  - Digikam::TrashView, [3325](#)
- modelFlags
  - Digikam::GPSBookmarkModelHelper, [1637](#)
  - Digikam::GPSGeofaceModelHelper, [1648](#)
- ModelIndexedComboBox
  - Digikam::ModelIndexedComboBox, [2533](#)
- modificationDate
  - Digikam::ThumbnailInfo, [3254](#)
- ModifiedScan
  - Digikam::CollectionScanner, [600](#)
- modulateProgress
  - Digikam::DImgThreadedFilter, [1024](#)
- monitorProfile
  - Digikam::IccSettings, [1779](#)
- monthIndexForDate
  - Digikam::DateAlbumModel, [753](#)
- moreCompleteLoadingAvailable
  - Digikam::LoadSaveThread, [2380](#)
- mouseButtonDoubleClicked
  - Digikam::DCategoryDrawer, [829](#)
- mouseButtonPressed
  - Digikam::DCategoryDrawer, [830](#)
- mouseButtonReleased
  - Digikam::DCategoryDrawer, [830](#)
- mouseLeft
  - Digikam::DCategoryDrawer, [830](#)
- mouseModeChanged
  - Digikam::BackendGoogleMaps, [438](#)
  - Digikam::BackendMarble, [446](#)
  - Digikam::MapBackend, [2420](#)



- mouseMoved
  - Digikam::DCategoryDrawer, [831](#)
  - Digikam::DItemDelegate, [1055](#)
  - Digikam::ItemDelegateOverlay, [2043](#)
  - Digikam::ItemViewDelegate, [2267](#)
  - Digikam::ItemViewImportDelegate, [2275](#)
  - ShowFoto::ShowfotoItemViewDelegate, [3485](#)
- Moved
  - Digikam::CollectionImageChangeset, [585](#)
- moveItem
  - Digikam::CoreDB, [670](#)
- moveNear
  - Digikam::DNotificationPopup, [1157](#)
- moveTAlbum
  - Digikam::AlbumManager, [297](#)
- MoveToIntermediate
  - Digikam::VersionFileOperation, [3356](#)
- MP4
  - Digikam::VidSlideSettings, [3375](#)
- MPEG2
  - Digikam::VidSlideSettings, [3374](#)
- MPEG4
  - Digikam::VidSlideSettings, [3374](#)
- MPG
  - Digikam::VidSlideSettings, [3375](#)
- mSecTimeStamp
  - Digikam::DMetadata, [1092](#)
- multithreadedSteps
  - Digikam::DImgThreadedFilter, [1025](#)
- name
  - Digikam::CoreDbUrl, [695](#)
  - Digikam::DPlugin, [1194](#)
  - Digikam::ItemInfo, [2121](#)
- NameMatchMode
  - Digikam::TaggingActionFactory, [3147](#)
- namespaceTitleDefinitions
  - Digikam, [143](#)
- Natural
  - Digikam::ApplicationSettings, [383](#)
- needCheckRawDecoding
  - Digikam::LoadingDescription, [2364](#)
- needsPostLoadingManagement
  - Digikam::lccManager, [1759](#)
- Network
  - Digikam::CollectionLocation, [588](#)
- NetworkError
  - Digikam::DOnlineTranslator, [1171](#)
- NEUTRAL
  - Digikam::SearchTextBar, [2914](#)
- NewerCreationDate
  - Digikam::HaarIface, [1719](#)
- NewerModificationDate
  - Digikam::HaarIface, [1719](#)
- NewFile
  - Digikam::VersionFileOperation, [3356](#)
- newFileFullScan
  - Digikam::ItemScanner, [2217](#)
- newImages
  - Digikam::RecognitionTrainingProvider, [2742](#)
  - Digikam::TrainingDataProvider, [3318](#)
- NewPicture
  - Digikam::CamItemInfo, [541](#)
- next
  - Digikam::Album, [260](#)
- nextIndexHint
  - Digikam::ImportCategorizedView, [1851](#)
  - Digikam::ItemCategorizedView, [2016](#)
  - Digikam::ItemViewCategorized, [2260](#)
  - ShowFoto::ShowfotoCategorizedView, [3433](#)
- nextInOrder
  - Digikam::ImportCategorizedView, [1851](#)
  - Digikam::ItemCategorizedView, [2017](#)
  - ShowFoto::ShowfotoCategorizedView, [3433](#)
- NO\_ACTION
  - Digikam::ExifToolProcess, [1390](#)
- NO\_RESULT
  - Digikam::SearchTextBar, [2914](#)
- NoBackend
  - Digikam::MetaEngine, [2484](#)
- NoCategories
  - Digikam::ItemSortSettings, [2230](#)
- nodeCompare
  - Digikam::KDNodeOpenFace, [2292](#)
  - Digikam::KDNodeSFace, [2294](#)
- NoError
  - Digikam::DOnlineTranslator, [1171](#)
  - Digikam::DOnlineTts, [1182](#)
- NoErrors
  - Digikam::BdEngineBackend, [475](#)
  - Digikam::DatabaseServerError, [737](#)
- NoiseReduction
  - Digikam::DRawDecoderSettings, [1260](#)
- NoLeadingSlash
  - Digikam::TagsCache, [3187](#)
- NoMatch
  - Digikam::AlbumFilterModel, [270](#)
- NonAssignedItems
  - Digikam::AutoTagsScanSettings, [425](#)
  - Digikam::ImageQualitySorter, [1820](#)
- NonDeterministicRandomData
  - Digikam::NonDeterministicRandomData, [2566](#)
- NONE
  - Digikam::DNGWriter, [1116](#)
  - Digikam::FileSaveOptionsBox, [1534](#)
- None
  - Digikam::EffectMngr, [1352](#)
  - Digikam::MLPipelineFoundation, [2526](#)
- NoPreviewMode
  - Digikam::PreviewToolBar, [2661](#)
- NoPreviewZoomCtrl
  - Digikam::DZoomBar, [1324](#)
- Normal
  - Digikam::ApplicationSettings, [383](#)
- NormalScan
  - Digikam::CollectionScanner, [600](#)
- NormalSearchTreeView

- Digikam::NormalSearchTreeView, [2577](#)
- NormalWrite
  - Digikam::FacePipeline, [1424](#)
  - Digikam::FacePipelineBase, [1429](#)
- NotificationPolicy
  - Digikam::LoadSaveThread, [2379](#)
- NotificationPolicyDirect
  - Digikam::LoadSaveThread, [2379](#)
- NotificationPolicyTimeLimited
  - Digikam::LoadSaveThread, [2379](#)
- notifyFileChanged
  - Digikam::LoadingCache, [2358](#)
  - Digikam::LoadingCacheFileWatch, [2361](#)
- notifyNewLoadingProcess
  - Digikam::SharedLoadingTask, [2982](#)
- NoTransformation
  - Digikam::MetaEngineRotation, [2510](#)
- NotSupported
  - Digikam::DatabaseServerError, [737](#)
- NRThreshold
  - Digikam::DRawDecoderSettings, [1262](#)
- NTSC
  - Digikam::VidSlideSettings, [3375](#)
- Null
  - Digikam::RatingComboBox, [2715](#)
- number
  - Digikam::RandomNumberGenerator, [2706](#)
- OlderOrLarger
  - Digikam::HaarIface, [1719](#)
- oneAlbumRoot
  - Digikam::CollectionManager, [594](#)
- OneCategory
  - Digikam::ItemSortSettings, [2230](#)
- onIndicesClicked
  - Digikam::AbstractMarkerTiler, [198](#)
  - Digikam::GeoModelHelper, [1627](#)
  - Digikam::GPSMarkerTiler, [1669](#)
  - Digikam::ItemMarkerTiler, [2156](#)
  - Digikam::MapViewModelHelper, [2425](#)
- onIndicesMoved
  - Digikam::GPSGeofaceModelHelper, [1648](#)
  - Digikam::ItemMarkerTiler, [2156](#)
- Open
  - Digikam::BdEngineBackend, [475](#)
- open
  - Digikam::BdEngineBackend, [478](#)
  - Digikam::IccProfile, [1765](#)
- openAlbum
  - Digikam::ItemAlbumModel, [2008](#)
- OpenCV\_KNN
  - Digikam::OpenCVDNNFaceRecognizer, [2601](#)
- OpenFace
  - Digikam::FaceScanSettings, [1479](#)
- openOnlineDocumentation
  - Digikam, [138](#)
- OpenSchemaChecked
  - Digikam::BdEngineBackend, [475](#)
- openSetupPage
  - Digikam::DBInfoIface, [797](#)
  - Digikam::DInfoInterface, [1032](#)
  - ShowFoto::ShowfotoInfoIface, [3470](#)
- Operation
  - Digikam::CollectionImageChangeset, [585](#)
  - Digikam::ImageTagChangeset, [1833](#)
  - Digikam::TagChangeset, [3116](#)
- OperationType
  - Digikam, [136](#)
- operationTypeExplanation
  - Digikam::ApplicationSettings, [384](#)
- operationTypeTitle
  - Digikam::ApplicationSettings, [384](#)
- operator<<
  - Digikam, [138](#)
  - Digikam::CollectionImageChangeset, [586](#)
  - Digikam::DImageHistory, [978](#)
  - Digikam::ImageTagChangeset, [1833](#)
- operator()
  - Digikam::RedEye::RegressionTree, [2747](#)
- operator==
  - Digikam::BatchToolSet, [464](#)
  - Digikam::ChoiceSearchModel::Entry, [580](#)
  - Digikam::DImg, [992](#)
  - Digikam::HotPixelProps, [1742](#)
  - Digikam::IccProfile, [1765](#)
- orientationHint
  - Digikam::DatabaseLoadSaveFileInfoProvider, [729](#)
  - Digikam::LoadSaveFileInfoProvider, [2371](#)
  - Digikam::ThumbnailInfo, [3254](#)
- Original
  - Digikam::HistoryImageId, [1735](#)
- original
  - Digikam::ImageIface, [1806](#)
- originalBitDepth
  - Digikam::DImg, [992](#)
- originalColorModel
  - Digikam::DImg, [992](#)
- originalFilePath
  - Digikam::DImg, [992](#)
- originalImageSize
  - Digikam::ImageZoomSettings, [1842](#)
- originalRect
  - Digikam::DImgChildItem, [1000](#)
- orthodromicDistance
  - Digikam::Ellipsoid, [1356](#)
  - Digikam::GeodeticCalculator, [1613](#)
- Output
  - Digikam::IccProfile, [1764](#)
- OutputColorSpace
  - Digikam::DRawDecoderSettings, [1260](#)
- outputColorSpace
  - Digikam::DRawDecoderSettings, [1262](#)
- outputSuffix
  - Digikam::BatchTool, [461](#)
- OverwriteAllFaces
  - Digikam::FacePipeline, [1424](#)
  - Digikam::FacePipelineBase, [1429](#)



- OverwriteUnconfirmed
  - Digikam::FacePipeline, [1424](#)
  - Digikam::FacePipelineBase, [1429](#)
- pageRemoved
  - Digikam::DConfigDlg, [851](#)
  - Digikam::DConfigDlgWdg, [877](#)
- pageToggled
  - Digikam::DConfigDlgWdg, [877](#)
- paint
  - Digikam::Imageface, [1806](#)
  - Digikam::TableViewColumn, [3073](#)
  - Digikam::TableViewColumns::ColumnThumbnail, [3106](#)
- PAL
  - Digikam::VidSlideSettings, [3375](#)
- ParallelAdapter
  - Digikam::ParallelAdapter< A >, [2617](#)
- ParallelWorkers
  - Digikam::ParallelWorkers, [2621](#)
- parameter
  - Digikam::FilterAction, [1555](#)
- parameters
  - Digikam::CoreDbUrl, [695](#)
  - Digikam::SimilarityDbAccess, [3021](#)
- ParametersError
  - Digikam::DOnlineTranslator, [1171](#)
- parametersSuccessfullyRead
  - Digikam::DImgThreadedFilter, [1025](#)
  - Digikam::IccTransformFilter, [1788](#)
- parent
  - Digikam::Album, [260](#)
  - Digikam::ProgressItem, [2665](#)
  - Digikam::TableViewModel, [3111](#)
- ParentMatch
  - Digikam::AlbumFilterModel, [270](#)
- ParentMenuCategory
  - Digikam::ActionItemModel, [216](#)
- parentTags
  - Digikam::TagsCache, [3188](#)
- ParentToChild
  - Digikam, [136](#)
- parse
  - Digikam::AdvancedRenameWidget, [249](#)
- parseAlbumItemsRecursive
  - Digikam::DBInfolface, [797](#)
  - Digikam::DMetalInfolface, [1100](#)
- parseChildren
  - Digikam::DConfigDlgMgr, [856](#)
- parseOperation
  - Digikam::CameraNameOption, [537](#)
  - Digikam::CaseModifier, [556](#)
  - Digikam::DatabaseOption, [733](#)
  - Digikam::DateOption, [763](#)
  - Digikam::DefaultValueModifier, [916](#)
  - Digikam::DirectoryNameOption, [1041](#)
  - Digikam::FilePropertiesOption, [1531](#)
  - Digikam::MetadataOption, [2452](#)
  - Digikam::Modifier, [2537](#)
  - Digikam::Option, [2604](#)
  - Digikam::RangeModifier, [2711](#)
  - Digikam::RemoveDoublesModifier, [2767](#)
  - Digikam::ReplaceModifier, [2775](#)
  - Digikam::Rule, [2796](#)
  - Digikam::SequenceNumberOption, [2947](#)
  - Digikam::TrimmedModifier, [3335](#)
  - Digikam::UniqueModifier, [3350](#)
- parseStringIsValid
  - Digikam::Parser, [2624](#)
- ParsingError
  - Digikam::DOnlineTranslator, [1171](#)
- partialScan
  - Digikam::CollectionScanner, [600](#)
- PartialWrite
  - Digikam::DisjointMetadata, [1044](#)
  - Digikam::MetadataHub, [2440](#)
- passShortcutActionsToWidget
  - Digikam::DBInfolface, [797](#)
  - Digikam::DInfoInterface, [1032](#)
- Pending
  - Digikam::QueueListView, [2689](#)
- PersistentWidgetDelegateOverlay
  - Digikam::PersistentWidgetDelegateOverlay, [2633](#)
- PhaseOut
  - Digikam::WorkerObject, [3396](#)
- PHYSICAL
  - Digikam::Album, [256](#)
- pickLabel
  - Digikam::DisjointMetadata, [1045](#)
- pickLabelForTag
  - Digikam::TagsCache, [3188](#)
- pickLabelFromTags
  - Digikam::TagsCache, [3188](#)
- pickLabelInterval
  - Digikam::DisjointMetadata, [1046](#)
- pixelAntiAliasing
  - Digikam::DPixelsAliasFilter, [1186](#)
- pixelAntiAliasing16
  - Digikam::DPixelsAliasFilter, [1186](#)
- pixelAspectRatio
  - Digikam::DRawInfo, [1271](#)
- pixmap
  - Digikam::DConfigDlgTitle, [862](#)
- pixmapForDrag
  - Digikam::AbstractAlbumTreeView, [160](#)
  - Digikam::DragDropViewImplementation, [1250](#)
  - Digikam::ImportDelegate, [1870](#)
  - Digikam::ItemDelegate, [2040](#)
  - Digikam::ItemViewCategorized, [2261](#)
  - Digikam::TableViewTreeView, [3116](#)
  - Digikam::VersionsTreeView, [3367](#)
  - ShowFoto::ShowfotoDelegate, [3444](#)
- pixmapForItem
  - Digikam::DTrashItemModel, [1297](#)
- pixmapFromRepresentativeIndex
  - Digikam::AbstractMarkerTiler, [198](#)
  - Digikam::GeoModelHelper, [1627](#)

- Digikam::GPSGeofaceModelHelper, [1648](#)
- Digikam::GPSMarkerTiler, [1669](#)
- Digikam::ItemGPSModelHelper, [2096](#)
- Digikam::ItemMarkerTiler, [2156](#)
- Digikam::MapViewModelHelper, [2426](#)
- pixmapRect
  - Digikam::ImportDelegate, [1870](#)
  - Digikam::ItemDelegate, [2040](#)
  - Digikam::ItemViewDelegate, [2267](#)
  - Digikam::ItemViewImportDelegate, [2275](#)
  - ShowFoto::ShowfotoDelegate, [3444](#)
  - ShowFoto::ShowfotoItemViewDelegate, [3485](#)
- PlainTextMessage
  - Digikam::DConfigDlgTitle, [862](#)
- plugDatabaseFilter
  - Digikam::FacePipeline, [1424](#)
- pluginAction
  - Digikam::DPluginLoader, [1226](#)
- pluginActions
  - Digikam::DPluginLoader, [1227](#)
- pluginsActions
  - Digikam::DPluginLoader, [1227](#)
- PopupStyle
  - Digikam::DNotificationPopup, [1153](#)
- posFromDate
  - Digikam::DDateTable, [903](#)
- position
  - Digikam::DMultiTabBar, [1105](#)
- positionChanged
  - Digikam::DImgChildItem, [1000](#)
- PositionKeys
  - Digikam::PositionKeys, [2644](#)
- positionOnImageChanged
  - Digikam::DImgChildItem, [1000](#)
- positionSelf
  - Digikam::DNotificationPopup, [1158](#)
- possibleValuesForEnumField
  - Digikam::DMetadata, [1092](#)
- postLoadingManage
  - Digikam::IccPostLoadingManager, [1762](#)
- postProcess
  - Digikam::ThumbnailLoadingTask, [3258](#)
- predict
  - Digikam::FaceClassifier, [1401](#)
- PreferFolder
  - Digikam::Haarface, [1719](#)
- pregenerateGroup
  - Digikam::ThumbnailLoadThread, [3266](#)
- preload
  - Digikam::ThumbnailLoadThread, [3266](#)
- preloadThumbnails
  - Digikam::ItemThumbnailModel, [2253](#)
- premultiply
  - Digikam::DColor, [834](#)
- PrepareMetadataFlag
  - Digikam::DImg, [984](#)
- prepareMetadataToSave
  - Digikam::DImg, [992](#)
- prepareRatingPixmap
  - Digikam::ItemViewImportDelegate, [2275](#)
- prepareTiles
  - Digikam::AbstractMarkerTiler, [198](#)
  - Digikam::GPSMarkerTiler, [1669](#)
  - Digikam::ItemMarkerTiler, [2156](#)
- prePopulated
  - Digikam::BookmarksMenu, [502](#)
  - Digikam::ModelMenu, [2535](#)
- preprocess
  - Digikam::RecognitionPreprocessor, [2741](#)
- PreserveEmbeddedProfile
  - Digikam::IccSettingsContainer, [1781](#)
- prev
  - Digikam::Album, [260](#)
- PreviewBothImagesHorz
  - Digikam::PreviewToolBar, [2661](#)
- PreviewBothImagesHorzCont
  - Digikam::PreviewToolBar, [2661](#)
- PreviewBothImagesVert
  - Digikam::PreviewToolBar, [2661](#)
- PreviewBothImagesVertCont
  - Digikam::PreviewToolBar, [2661](#)
- PreviewCameraMode
  - Digikam::ImportStackedView, [1943](#)
- PreviewLoadThread
  - Digikam::PreviewLoadThread, [2655](#)
- PreviewMode
  - Digikam::PreviewToolBar, [2661](#)
- PreviewOriginalImage
  - Digikam::PreviewToolBar, [2661](#)
- previewReference
  - Digikam::Imagelface, [1806](#)
- PreviewTargetImage
  - Digikam::PreviewToolBar, [2661](#)
- PreviewToggleOnMouseOver
  - Digikam::PreviewToolBar, [2661](#)
- PreviewType
  - Digikam::Imagelface, [1806](#)
- PreviewZoomCtrl
  - Digikam::DZoomBar, [1324](#)
- proceed
  - Digikam::EmptyImageListProvider, [1367](#)
  - Digikam::QListImageListProvider, [2686](#)
- process
  - Digikam::FacePipeline, [1425](#)
  - Digikam::ItemFilterModelFilterer, [2071](#)
  - Digikam::ItemFilterModelPreparer, [2074](#)
- PROCESS\_CANCELED
  - Digikam::DNGWriter, [1115](#)
- PROCESS\_COMPLETE
  - Digikam::DNGWriter, [1115](#)
- PROCESS\_CONTINUE
  - Digikam::DNGWriter, [1115](#)
- PROCESS\_FAILED
  - Digikam::DNGWriter, [1115](#)
- ProfileType
  - Digikam::IccProfile, [1764](#)

- progress
  - Digikam::ProgressItem, 2666
- progressInfo
  - Digikam::DImgLoaderObserver, 1010
  - Digikam::IccTransformFilter, 1788
  - Digikam::LoadingTask, 2370
  - Digikam::SavingTask, 2806
  - Digikam::SharedLoadingTask, 2982
- progressItemAdded
  - Digikam::ProgressItem, 2666
  - Digikam::ProgressManager, 2675
- progressItemCanceled
  - Digikam::ProgressItem, 2666
  - Digikam::ProgressManager, 2675
- progressItemCompleted
  - Digikam::ProgressItem, 2666
  - Digikam::ProgressManager, 2675
- progressItemLabel
  - Digikam::ProgressItem, 2667
  - Digikam::ProgressManager, 2676
- progressItemProgress
  - Digikam::ProgressItem, 2667
  - Digikam::ProgressManager, 2676
- progressItemStatus
  - Digikam::ProgressItem, 2667
  - Digikam::ProgressManager, 2676
- progressItemThumbnail
  - Digikam::ProgressItem, 2667
  - Digikam::ProgressManager, 2676
- progressItemUsesBusyIndicator
  - Digikam::ProgressItem, 2668
  - Digikam::ProgressManager, 2676
- progressScheduled
  - Digikam::DProgressWdg, 1246
- properties
  - Digikam::TagsCache, 3188
- PropertiesChanged
  - Digikam::TagChangeset, 3116
- propertyValue
  - Digikam::TagsCache, 3188
- provider
  - Digikam::ItemCopyright, 2033
- ProxyClickLineEdit
  - Digikam::ProxyClickLineEdit, 2682
- ProxyLineEdit
  - Digikam::ProxyLineEdit, 2684
- ProxyType
  - Digikam::SystemSettings, 3065
- pureColorMask
  - Digikam::DImg, 992
- purgePathFromReferredImages
  - Digikam::DImageHistory, 978
- putImage
  - Digikam::LoadingCache, 2358
- putImageData
  - Digikam::DImg, 993
- QIMAGE
  - Digikam::DImg, 984
- QPointSquareDistance
  - Digikam, 139
- QSXGA
  - Digikam::VidSlideSettings, 3376
- QSXGAPLUS
  - Digikam::VidSlideSettings, 3376
- Quality
  - Digikam::PreviewSettings, 2658
- QualityScanMode
  - Digikam::ImageQualitySorter, 1819
- qualityScanMode
  - Digikam::MaintenanceSettings, 2402
- queryErrorHandling
  - Digikam::BdEngineBackend, 478
- querySendNotifyEvent
  - Digikam::SharedLoadingTask, 2982
- QueryStateEnum
  - Digikam::BdEngineBackend, 474
- QUXGA
  - Digikam::VidSlideSettings, 3376
- QVGA
  - Digikam::VidSlideSettings, 3375
- QXGA
  - Digikam::VidSlideSettings, 3376
- radiusOfCurvature
  - Digikam::Ellipsoid, 1356
- RandomNumberGenerator
  - Digikam::RandomNumberGenerator, 2706
- RasterGraphics
  - Digikam::MimeFilter, 2515
- rating
  - Digikam::DisjointMetadata, 1046
- ratingInterval
  - Digikam::DisjointMetadata, 1046
- RatingValue
  - Digikam::RatingComboBox, 2715
- RawDecodingCustomSettings
  - Digikam::LoadingDescription, 2363
- RawDecodingDefaultSettings
  - Digikam::LoadingDescription, 2363
- RawDecodingGlobalSettings
  - Digikam::LoadingDescription, 2363
- RawDecodingHint
  - Digikam::LoadingDescription, 2363
- rawDecodingSettings
  - Digikam::DImg, 993
- RawDecodingTimeOptimized
  - Digikam::LoadingDescription, 2363
- rawFileIdentify
  - Digikam::DRawDecoder, 1257
- RAWFiles
  - Digikam::MimeFilter, 2515
- rawFilesVersion
  - Digikam::DRawDecoder, 1257
- RawProcessingFilter
  - Digikam::RawProcessingFilter, 2736
- RAWQuality
  - Digikam::DRawDecoderSettings, 1262

- read
  - Digikam::SearchFieldAlbum, [2829](#)
  - Digikam::SearchFieldCheckBox, [2833](#)
  - Digikam::SearchFieldChoice, [2837](#)
  - Digikam::SearchFieldColorDepth, [2841](#)
  - Digikam::SearchFieldKeyword, [2850](#)
  - Digikam::SearchFieldLabels, [2853](#)
  - Digikam::SearchFieldMonthDay, [2857](#)
  - Digikam::SearchFieldPageOrientation, [2861](#)
  - Digikam::SearchFieldRangeDate, [2864](#)
  - Digikam::SearchFieldRangeDouble, [2868](#)
  - Digikam::SearchFieldRangeInt, [2872](#)
  - Digikam::SearchFieldRangeTime, [2876](#)
  - Digikam::SearchFieldRating, [2880](#)
  - Digikam::SearchFieldText, [2884](#)
- READ\_FORMATS
  - Digikam::ExifToolProcess, [1390](#)
- readableFormats
  - Digikam::ExifToolParser, [1386](#)
- ReadConfirmedFaces
  - Digikam::FacePipeline, [1423](#)
- readdNewTags
  - Digikam::RGTagModel, [2786](#)
- readdTag
  - Digikam::RGTagModel, [2787](#)
- ReadFacesForTraining
  - Digikam::FacePipeline, [1423](#)
- readFromConfig
  - Digikam::DbEngineParameters, [788](#)
- README, [29](#)
- readMsgBoxShouldBeShown
  - Digikam::ApplicationSettings, [384](#)
  - Digikam::DMessageBox, [1072](#)
- readNext
  - Digikam::SearchXmlReader, [2939](#)
- readParameters
  - Digikam::AntiVignettingFilter, [376](#)
  - Digikam::AutoExpoFilter, [414](#)
  - Digikam::AutoLevelsFilter, [419](#)
  - Digikam::BCGFilter, [470](#)
  - Digikam::BlurFilter, [491](#)
  - Digikam::BlurFXFilter, [496](#)
  - Digikam::BorderFilter, [509](#)
  - Digikam::BWSepiaFilter, [521](#)
  - Digikam::CBFilter, [563](#)
  - Digikam::CharcoalFilter, [570](#)
  - Digikam::ColorFXFilter, [609](#)
  - Digikam::ContentAwareFilter, [627](#)
  - Digikam::CurvesFilter, [706](#)
  - Digikam::DistortionFXFilter, [1053](#)
  - Digikam::EmbossFilter, [1363](#)
  - Digikam::EqualizeFilter, [1372](#)
  - Digikam::FilmFilter, [1545](#)
  - Digikam::FilmGrainFilter, [1550](#)
  - Digikam::FilterActionFilter, [1562](#)
  - Digikam::FreeRotationFilter, [1594](#)
  - Digikam::GreycstorationFilter, [1706](#)
  - Digikam::HotPixelFixer, [1741](#)
  - Digikam::HSLFilter, [1753](#)
  - Digikam::IccTransformFilter, [1789](#)
  - Digikam::InfraredFilter, [1974](#)
  - Digikam::InvertFilter, [1982](#)
  - Digikam::LensDistortionFilter, [2316](#)
  - Digikam::LensFunFilter, [2323](#)
  - Digikam::LevelsFilter, [2330](#)
  - Digikam::LocalContrastFilter, [2387](#)
  - Digikam::MixerFilter, [2520](#)
  - Digikam::NormalizeFilter, [2571](#)
  - Digikam::NRFilter, [2589](#)
  - Digikam::OilPaintFilter, [2595](#)
  - Digikam::RainDropFilter, [2705](#)
  - Digikam::RawProcessingFilter, [2737](#)
  - Digikam::RefocusFilter, [2759](#)
  - Digikam::SharpenFilter, [2993](#)
  - Digikam::ShearFilter, [2999](#)
  - Digikam::StretchFilter, [3059](#)
  - Digikam::TextureFilter, [3242](#)
  - Digikam::TonalityFilter, [3300](#)
  - Digikam::UnsharpMaskFilter, [3355](#)
  - Digikam::WBFilter, [3388](#)
- readParametersError
  - Digikam::IccTransformFilter, [1789](#)
- readSearch
  - Digikam::SearchWindow, [2933](#)
- readSettings
  - Digikam::DRawDecoderWidget, [1265](#)
- readSettingsFromGroup
  - Digikam::BackendGoogleMaps, [438](#)
  - Digikam::BackendMarble, [446](#)
  - Digikam::RGWidget, [2789](#)
- readToList
  - Digikam::BdEngineBackend, [479](#)
- readToStartOfElement
  - Digikam::SearchXmlReader, [2939](#)
- ReadUnconfirmedFaces
  - Digikam::FacePipeline, [1423](#)
- readyForIncrementalRefresh
  - Digikam::ImportItemModel, [1900](#)
  - Digikam::ItemModel, [2167](#)
  - ShowFoto::ShowfotoItemModel, [3476](#)
- rebuildDuplicatesAlbums
  - Digikam::HaarIface, [1720](#)
- receive
  - Digikam::ItemLISTERJobGrowingPartsSendingReceiver, [2138](#)
  - Digikam::ItemLISTERJobPartsSendingReceiver, [2140](#)
  - Digikam::ItemLISTERValueListReceiver, [2146](#)
- recognize
  - Digikam::OpenCVDNNFaceRecognizer, [2601](#)
- recognizeAccuracy
  - Digikam::FaceScanSettings, [1480](#)
- recognizeFaces
  - Digikam::FacialRecognitionWrapper, [1504](#)
- RecognizeMarkedFaces
  - Digikam::FaceScanSettings, [1479](#)

RecognizeOnly  
  Digikam::FaceScanSettings, [1479](#)

recommendedImageSize  
  Digikam::FaceDetector, [1413](#)

recommendedImageSizeForDetection  
  Digikam::OpenCVDNNFaceDetector, [2599](#)

rect  
  Digikam::DImgChildItem, [1000](#)

RECURSIVE  
  Digikam::StateSavingObject, [3046](#)

reduceEdges  
  Digikam::ItemHistoryGraph, [2099](#)

ReflImageSelMethod  
  Digikam::HaarIface, [1719](#)

refresh  
  Digikam::AlbumManager, [297](#)  
  Digikam::CollectionManager, [594](#)

refreshThumbnails  
  Digikam::DTrashItemModel, [1297](#)

regenerateTiles  
  Digikam::GPSMarkerTiler, [1670](#)  
  Digikam::ItemMarkerTiler, [2156](#)

regExp  
  Digikam::Rule, [2796](#)

regionSelectionChanged  
  Digikam::BackendGoogleMaps, [438](#)  
  Digikam::BackendMarble, [446](#)

registerButton  
  Digikam::Rule, [2797](#)

registeredImageIds  
  Digikam::SimilarityDb, [3018](#)

registerExtraPluginsActions  
  Digikam::EditorWindow, [1350](#)

registerLabelsActions  
  Digikam::TagsActionMngr, [3183](#)

registerMenu  
  Digikam::Rule, [2797](#)

registerPluginsActions  
  Digikam::DXmlGuiWindow, [1317](#)

registerSettingsWidget  
  Digikam::BatchTool, [462](#)

registerTagsActionCollections  
  Digikam::TagsActionMngr, [3183](#)

registerXmpNameSpace  
  Digikam::MetaEngine, [2496](#)

reInitialize  
  Digikam::ThumbBarDock, [3246](#)

rejected  
  Digikam::AssignNameWidget, [398](#)

rejectFaces  
  Digikam::DigikamItemView, [975](#)

relationCloud  
  Digikam::ItemHistoryGraph, [2099](#)

relativeRect  
  Digikam::DImgChildItem, [1001](#)

RelativeSize  
  Digikam::AlbumThumbnailLoader, [352](#)

releaseWidget  
  Digikam::BackendGoogleMaps, [439](#)  
  Digikam::BackendMarble, [447](#)

reload  
  Digikam::BackendGoogleMaps, [439](#)  
  Digikam::BackendMarble, [447](#)

remove  
  Digikam::ItemPosition, [2170](#)

removeAction  
  Digikam::DNotificationWidget, [1165](#)

removeAlphaChannel  
  Digikam::DImg, [993](#)

removeAndCopyFile  
  Digikam::DFileOperations, [939](#)

Removed  
  Digikam::CollectionImageChangeset, [585](#)

RemovedAll  
  Digikam::CollectionImageChangeset, [585](#)

RemovedDeleted  
  Digikam::CollectionImageChangeset, [585](#)

removeExifTag  
  Digikam::MetaEngine, [2496](#)

removeExtraData  
  Digikam::Album, [260](#)

removeFace  
  Digikam::FaceTagsEditor, [1494](#)

removeFaces  
  Digikam::DigikamItemView, [976](#)

removeFaceVector  
  Digikam::FaceDb, [1404](#)

removeFromXmpTagStringBag  
  Digikam::DMetadata, [1093](#)  
  Digikam::MetaEngine, [2496](#)

removeGPSInfo  
  Digikam::MetaEngine, [2497](#)

removeImageFingerprint  
  Digikam::SimilarityDb, [3019](#)

removeImageRelation  
  Digikam::CoreDB, [671](#)

removeImageSimilarity  
  Digikam::SimilarityDb, [3019](#)

removeImageTagProperties  
  Digikam::CoreDB, [671](#)

removeIptcTag  
  Digikam::MetaEngine, [2497](#)

removeItemAllTags  
  Digikam::CoreDB, [671](#)

removeItemCopyrightProperties  
  Digikam::CoreDB, [671](#)

removeItems  
  Digikam::CoreDB, [672](#)  
  Digikam::DTrashItemModel, [1298](#)

removeItemsFromAlbum  
  Digikam::CoreDB, [672](#)

removeItemsPermanently  
  Digikam::CoreDB, [672](#)

removeItemTag  
  Digikam::CoreDB, [673](#)

removeListener

- Digikam::SharedLoadingTask, 2982
- removeLocation
  - Digikam::CollectionManager, 594
- removeMarkerIndexFromGrid
  - Digikam::ItemMarkerTiler, 2157
- removeNormalTag
  - Digikam::FaceTagsEditor, 1494
  - Digikam::FaceUtils, 1502
- removeNormalTags
  - Digikam::FaceUtils, 1502
- RemoveOldMetadataPreviews
  - Digikam::DImg, 986
- removePage
  - Digikam::DConfigDlg, 851
  - Digikam::DConfigDlgWdg, 877
  - Digikam::DConfigDlgWdgModel, 887
- removeTag
  - Digikam::ItemInfo, 2121
- removeTagProperties
  - Digikam::CoreDB, 673
- removeTags
  - Digikam::FileActionMngrDatabaseWorker, 1520
- removeUngroupedModel
  - Digikam::MapWidget, 2432
- removeXmpKeywords
  - Digikam::DMetadata, 1093
  - Digikam::MetaEngine, 2497
- removeXmpSubCategories
  - Digikam::DMetadata, 1093
  - Digikam::MetaEngine, 2497
- removeXmpSubjects
  - Digikam::DMetadata, 1093
  - Digikam::MetaEngine, 2497
- removeXmpTag
  - Digikam::MetaEngine, 2497
- renameFile
  - Digikam::IOJobsThread, 1991
- renameItem
  - Digikam::CoreDB, 673
- renamePAlbum
  - Digikam::AlbumManager, 298
- renameTAlbum
  - Digikam::AlbumManager, 298
- render
  - Digikam::HistogramPainter, 1729
- Replace
  - Digikam::VersionFileOperation, 3356
- ReplaceAllEntries
  - Digikam::ItemCopyright, 2032
- replaceColorLabel
  - Digikam::DisjointMetadata, 1046
- replaceComments
  - Digikam::ItemComments, 2024
- ReplaceLanguageEntry
  - Digikam::ItemCopyright, 2032
- ReplaceMode
  - Digikam::ItemCopyright, 2032
- repositionRect
  - Digikam::BlackFrameToolTip, 484
  - Digikam::FreeSpaceToolTip, 1597
  - Digikam::ItemViewToolTip, 2278
- ReproducibleFilter
  - Digikam::FilterAction, 1555
- requestIncrementalRefresh
  - Digikam::ImportItemModel, 1900
  - Digikam::ItemModel, 2167
  - ShowFoto::ShowfotoItemModel, 3477
- requestNotification
  - Digikam::VersionsDelegate, 3364
- Rescan
  - Digikam::CollectionScanner, 600
  - Digikam::FaceScanSettings, 1479
- reseed
  - Digikam::RandomNumberGenerator, 2706
- reset
  - Digikam::Rule, 2797
  - Digikam::SearchFieldAlbum, 2829
  - Digikam::SearchFieldCheckBox, 2833
  - Digikam::SearchFieldChoice, 2837
  - Digikam::SearchFieldComboBox, 2844
  - Digikam::SearchFieldLabels, 2853
  - Digikam::SearchFieldMonthDay, 2857
  - Digikam::SearchFieldRangeDate, 2864
  - Digikam::SearchFieldRangeDouble, 2868
  - Digikam::SearchFieldRangeInt, 2872
  - Digikam::SearchFieldRangeTime, 2876
  - Digikam::SearchFieldRating, 2880
  - Digikam::SearchFieldText, 2884
  - Digikam::SearchWindow, 2933
  - Digikam::UniqueModifier, 3350
- ResetExifOrientationTag
  - Digikam::DImg, 986
- Resize
  - Digikam::DImgBuiltinFilter, 996
- resizeEvent
  - Digikam::DPopupFrame, 1237
  - Digikam::PanIconFrame, 2612
- RESNET50
  - Digikam, 136
- resolvedImageHistory
  - Digikam::ItemScanner, 2217
- resolveImageHistory
  - Digikam::ItemScanner, 2217
- restartCollectionScan
  - Digikam::ScanController, 2811
- restore
  - Digikam::Sidebar, 3008
- RESTORE\_PREVIEW
  - Digikam::ExifToolProcess, 1390
- restoreCurve
  - Digikam::CurvesWidget, 710
- restoreDTrashItems
  - Digikam::IOJobsThread, 1992
- restoreGPSData
  - Digikam::GPSItemContainer, 1651
- restoreSelectionFromHistory



- Digikam::AlbumLabelsSearchHandler, 282
- Digikam::LabelsTreeView, 2310
- restoreSettings
  - Digikam::TagList, 3150
- restoreState
  - Digikam::SidebarSplitter, 3010
- result
  - Digikam::DetectionBenchmark, 929
  - Digikam::RecognitionBenchmark, 2740
- resumeCollectionScan
  - Digikam::ScanController, 2811
- retrain
  - Digikam::FaceClassifier, 1401
- RetrainAll
  - Digikam::FaceScanSettings, 1479
- retrieveAlbum
  - Digikam::AbstractAlbumModel, 152
- retrieveItemInfo
  - Digikam::ItemModel, 2167
- retrieveSignatureFromDB
  - Digikam::HaarIface, 1720
- retrieveThumbnail
  - Digikam::LoadingCache, 2358
- ReturnMatchingDefaultOrFirstLanguage
  - Digikam::ItemComments, 2023
- ReturnMatchingLanguageOnly
  - Digikam::ItemComments, 2023
- ReturnMatchingOrDefaultLanguage
  - Digikam::ItemComments, 2023
- returnPressed
  - Digikam::DPlainTextEdit, 1189
  - Digikam::DTextEdit, 1288
- reverseFilter
  - Digikam::DImgBuiltinFilter, 997
- reverseToOrientation
  - Digikam::TagRegion, 3181
- RGTagModel
  - Digikam::RGTagModel, 2782
- RGWidget
  - Digikam::RGWidget, 2789
- rightMargin
  - Digikam::DCategoryDrawer, 831
- rightsUsageTerms
  - Digikam::ItemCopyright, 2033
- Role
  - Digikam::DConfigDlgModel, 859
  - Digikam::FacePipelineFaceTagsIface, 1447
- rootAlbumAvailable
  - Digikam::AbstractAlbumModel, 152
- RootAlbumBehavior
  - Digikam::AbstractAlbumModel, 151
- rootAlbumIndex
  - Digikam::AbstractAlbumModel, 152
- rootImages
  - Digikam::ItemHistoryGraph, 2099
- roots
  - Digikam::Graph< VertexProperties, EdgeProperties >, 1687
- rootsOf
  - Digikam::Graph< VertexProperties, EdgeProperties >, 1687
- Rotate180
  - Digikam::MetaEngineRotation, 2510
- Rotate270
  - Digikam::MetaEngineRotation, 2510
- Rotate90
  - Digikam::MetaEngineRotation, 2510
- rotateAndFlip
  - Digikam::DImg, 993
- rotateExifQImage
  - Digikam::MetaEngine, 2498
- RotationBehaviorFlag
  - Digikam::MetaEngineSettingsContainer, 2512
- rowFromAlbum
  - Digikam::Album, 261
- rowsRemoved
  - Digikam::ItemViewCategorized, 2261
- run
  - Digikam::DImgThreadedFilter, 1025
  - Digikam::DynamicThread, 1320
  - Digikam::ImageHistogram, 1804
  - Digikam::LoadSaveThread, 2380
  - Digikam::ScanStateFilter, 2816
- s\_inlineTranslateString
  - Digikam, 139
- s\_rawFileExtensionsdWithDesc
  - Digikam, 139
- s\_rawFileExtensionsVersion
  - Digikam, 140
- s\_setXmpTagStringFromEntry
  - Digikam, 140
- sameReferredImage
  - Digikam::ItemScanner, 2218
- sampleText
  - Digikam::DFontProperties, 947
- saturation
  - Digikam::DColorValueSelector, 842
  - Digikam::DHueSaturationSelector, 957
- save
  - Digikam::MetaEngine, 2498
- SaveAndDelete
  - Digikam::VersionFileOperation, 3356
- saveChanges
  - Digikam::GPSItemContainer, 1651
  - Digikam::ItemGPS, 2093
- saveCurve
  - Digikam::CurvesWidget, 711
- saveDestinationUrl
  - Digikam::EditorWindow, 1350
- savedFormat
  - Digikam::DImg, 993
- savefromDImg
  - Digikam::BatchTool, 462
- saveMsgBoxShouldBeShown
  - Digikam::ApplicationSettings, 385
  - Digikam::DMessageBox, 1072

- saveSettingsToGroup
  - Digikam::BackendGoogleMaps, [439](#)
  - Digikam::BackendMarble, [447](#)
  - Digikam::RGWidget, [2789](#)
- saveState
  - Digikam::AlbumSelectors, [338](#)
  - Digikam::SidebarSplitter, [3010](#)
- savingProgress
  - Digikam::LoadSaveThread, [2380](#)
- ScaleMode
  - Digikam::TimeLineWidget, [3292](#)
- scanAlbums
  - Digikam::CoreDB, [673](#)
- ScanAll
  - Digikam::FacePipeline, [1423](#)
  - Digikam::FacePipelineBase, [1428](#)
- ScanDeferredFiles
  - Digikam::NewItemFinder, [2558](#)
- scanFile
  - Digikam::CollectionScanner, [601](#)
- ScanMode
  - Digikam::AutoTagsScanSettings, [425](#)
- scannedFiles
  - Digikam::CollectionScanner, [601](#)
- ScanNew
  - Digikam::FacePipelineBase, [1428](#)
- scanSearches
  - Digikam::CoreDB, [673](#)
- scanTags
  - Digikam::CoreDB, [673](#)
- ScanTask
  - Digikam::FaceScanSettings, [1479](#)
- scene
  - Digikam::ItemExtendedProperties, [2053](#)
- ScheduleCollectionScan
  - Digikam::NewItemFinder, [2558](#)
- scheduleCollectionScan
  - Digikam::ScanController, [2811](#)
- scheduleCollectionScanExternal
  - Digikam::ScanController, [2812](#)
- scheduleCollectionScanRelaxed
  - Digikam::ScanController, [2812](#)
- screenCoordinates
  - Digikam::BackendGoogleMaps, [439](#)
  - Digikam::BackendMarble, [447](#)
- scrollPointOnPoint
  - Digikam::GraphicsDImgView, [1700](#)
- SDTV1
  - Digikam::VidSlideSettings, [3375](#)
- SDTV2
  - Digikam::VidSlideSettings, [3375](#)
- SDTV3
  - Digikam::VidSlideSettings, [3375](#)
- SEARCH
  - Digikam::Album, [256](#)
- searchEdited
  - Digikam::SearchWindow, [2933](#)
- searchesListing
  - Digikam::SearchesDBJobsThread, [2822](#)
- searchId
  - Digikam::CoreDbUrl, [695](#)
- SearchModificationHelper
  - Digikam::SearchModificationHelper, [2903](#)
- searchTextSettings
  - Digikam::AlbumFilterModel, [272](#)
- searchTextSettingsAboutToChange
  - Digikam::AlbumFilterModel, [272](#)
- searchTextSettingsChanged
  - Digikam::AlbumFilterModel, [272](#)
- seed
  - Digikam::RandomNumberGenerator, [2706](#)
- seedByTime
  - Digikam::RandomNumberGenerator, [2707](#)
- seedNonDeterministic
  - Digikam::RandomNumberGenerator, [2707](#)
- selectAll
  - Digikam::TableView, [3070](#)
- selectBbox
  - Digikam::DNNFaceDetectorBase, [1120](#)
- Selected
  - Digikam::FocusPoint, [1579](#)
  - Digikam::QueueListView, [2689](#)
  - Digikam::TimeLineWidget, [3293](#)
- selected
  - Digikam::AssignNameWidget, [398](#)
  - Digikam::ImportCategorizedView, [1851](#)
  - ShowFoto::ShowfotoCategorizedView, [3433](#)
- selectedAlbumsChanged
  - Digikam::AbstractAlbumTreeView, [160](#)
- SelectedInFocus
  - Digikam::FocusPoint, [1579](#)
- selectedLabels
  - Digikam::LabelsTreeView, [2310](#)
- selectedUrls
  - Digikam::ItemIconView, [2112](#)
- selectionChanged
  - Digikam::ItemViewCategorized, [2261](#)
- SelectionMode
  - Digikam::TimeLineWidget, [3292](#)
- selectionModel
  - Digikam::GeoModelHelper, [1628](#)
  - Digikam::GPSBookmarkModelHelper, [1637](#)
  - Digikam::GPSGeoifaceModelHelper, [1648](#)
  - Digikam::ItemGPSModelHelper, [2096](#)
  - Digikam::MapViewModelHelper, [2426](#)
- semiMajorAxis
  - Digikam::Ellipsoid, [1357](#)
- semiMinorAxis
  - Digikam::Ellipsoid, [1357](#)
- sendViewportEventToView
  - Digikam::AbstractAlbumTreeViewSelectComboBox, [167](#)
  - Digikam::ListViewComboBox, [2355](#)
  - Digikam::StayPoppedUpComboBox, [3053](#)
  - Digikam::TreeViewComboBox, [3329](#)
- ServiceError



- Digikam::DOnlineTranslator, [1171](#)
- setActive
  - Digikam::AbstractMarkerTiler, [199](#)
  - Digikam::AbstractWidgetDelegateOverlay, [209](#)
  - Digikam::ActionVersionsOverlay, [229](#)
  - Digikam::AlbumFolderViewSideBarWidget, [277](#)
  - Digikam::AssignNameOverlay, [393](#)
  - Digikam::BackendGoogleMaps, [439](#)
  - Digikam::BackendMarble, [447](#)
  - Digikam::DateFolderViewSideBarWidget, [760](#)
  - Digikam::FaceRejectionOverlay, [1474](#)
  - Digikam::FuzzySearchSideBarWidget, [1604](#)
  - Digikam::GPSMarkerTiler, [1670](#)
  - Digikam::GPSSearchSideBarWidget, [1676](#)
  - Digikam::GPSSearchView, [1679](#)
  - Digikam::GroupIndicatorOverlay, [1711](#)
  - Digikam::HoverButtonDelegateOverlay, [1747](#)
  - Digikam::ImportCoordinatesOverlay, [1864](#)
  - Digikam::ImportDownloadOverlay, [1875](#)
  - Digikam::ImportLockOverlay, [1912](#)
  - Digikam::ImportRatingOverlay, [1925](#)
  - Digikam::ImportRotateOverlay, [1931](#)
  - Digikam::ItemCoordinatesOverlay, [2029](#)
  - Digikam::ItemDelegateOverlay, [2043](#)
  - Digikam::ItemFullScreenOverlay, [2085](#)
  - Digikam::ItemMarkerTiler, [2158](#)
  - Digikam::ItemRatingOverlay, [2203](#)
  - Digikam::ItemRotateOverlay, [2208](#)
  - Digikam::ItemSelectionOverlay, [2222](#)
  - Digikam::LabelsSideBarWidget, [2306](#)
  - Digikam::MapBackend, [2420](#)
  - Digikam::MapWidgetView, [2436](#)
  - Digikam::PeopleSideBarWidget, [2629](#)
  - Digikam::PersistentWidgetDelegateOverlay, [2633](#)
  - Digikam::SearchSideBarWidget, [2910](#)
  - Digikam::ShowHideVersionsOverlay, [3003](#)
  - Digikam::SidebarWidget, [3013](#)
  - Digikam::TagsLineEditOverlay, [3201](#)
  - Digikam::TagViewSideBarWidget, [3223](#)
  - Digikam::ThumbnailImageCatcher, [3252](#)
  - Digikam::TimelineSideBarWidget, [3290](#)
  - ShowFoto::ShowfotoCoordinatesOverlay, [3438](#)
- setAlbumCaption
  - Digikam::CoreDB, [674](#)
- setAlbumCategory
  - Digikam::CoreDB, [674](#)
- setAlbumDate
  - Digikam::CoreDB, [674](#)
- setAlbumIcon
  - Digikam::CoreDB, [674](#)
- setAlbumManagerCurrentAlbum
  - Digikam::AbstractAlbumTreeView, [161](#)
- setAlbumModel
  - Digikam::ContextMenuHelper, [636](#)
- setAlbumModels
  - Digikam::AddTagsComboBox, [236](#)
- setAlbumModificationDate
  - Digikam::CoreDB, [675](#)
- setAlbumRootLabel
  - Digikam::CoreDB, [675](#)
- setAlbumRootPath
  - Digikam::CoreDB, [675](#)
- setAlbumRootsToSearch
  - Digikam::Haarface, [1720](#)
- setAlbumSelected
  - Digikam::AlbumSelectors, [338](#)
- setAnchor
  - Digikam::DNotificationPopup, [1158](#)
- setAuthorsList
  - Digikam::CaptionsMap, [551](#)
- setAutoDelete
  - Digikam::DNotificationPopup, [1158](#)
- setAutoHideTimeout
  - Digikam::DConfigDlgTitle, [863](#)
- setAutoSuggest
  - Digikam::FaceGroup, [1416](#)
- setBackend
  - Digikam::MapWidget, [2432](#)
- setBlockedEventTypes
  - Digikam::DWItemDelegate, [1309](#)
- setBlockUpdateSignalOnDrag
  - Digikam::DAbstractSliderSpinBox, [717](#)
- setBuddy
  - Digikam::DConfigDlgTitle, [863](#)
- setButtonBox
  - Digikam::DConfigDlg, [852](#)
- setButtonsExclusive
  - Digikam::ColorLabelWidget, [616](#)
  - Digikam::PickLabelWidget, [2641](#)
- setCacheOptions
  - Digikam::LoadingCacheInterface, [2361](#)
- setCacheSize
  - Digikam::LoadingCache, [2358](#)
- setCameraThumbsController
  - Digikam::ImportItemModel, [1900](#)
  - Digikam::ImportThumbnailModel, [1961](#)
- setCaseSensitive
  - Digikam::SearchTextBar, [2915](#)
- setCategorizedModel
  - Digikam::DCategorizedSortFilterProxyModel, [822](#)
- setCategory
  - Digikam::ItemScanner, [2218](#)
- setCenter
  - Digikam::BackendGoogleMaps, [439](#)
  - Digikam::BackendMarble, [447](#)
- setChannelFromBinary
  - Digikam::ImageCurves, [1794](#)
- setChannelType
  - Digikam::HistogramPainter, [1729](#)
- setCheckable
  - Digikam::AlbumSelectComboBox, [328](#)
  - Digikam::DConfigDlgWdgItem, [880](#)
- setChecked
  - Digikam::ChoiceSearchModel, [579](#)
- setCheckNewTags
  - Digikam::TagCheckView, [3124](#)

- setChoice
  - Digikam::ChoiceSearchModel, 580
- setChooserMode
  - Digikam::DColorValueSelector, 842
  - Digikam::DHueSaturationSelector, 957
- setCloseButton
  - Digikam::DDatePicker, 896
- setCloseButtonVisible
  - Digikam::DNotificationWidget, 1165
- setCloseOnActivate
  - Digikam::AlbumSelectComboBox, 328
- setColor
  - Digikam::DColor, 834
- setColorLabels
  - Digikam::ColorLabelWidget, 616
- setColorValue
  - Digikam::DColorValueSelector, 842
  - Digikam::DHueSaturationSelector, 957
- setComment
  - Digikam::DConfigDlgTitle, 863
- setComments
  - Digikam::MetaEngine, 2498
- setComplete
  - Digikam::ProgressItem, 2668
- setConfigGroup
  - Digikam::DateFolderView, 756
  - Digikam::FilterSideBarWidget, 1567
  - Digikam::FuzzySearchView, 1607
  - Digikam::GPSSearchView, 1679
  - Digikam::StateSavingObject, 3047
- setConfiguration
  - Digikam::TableViewColumns::ColumnAudioVideoProperties, 3079
  - Digikam::TableViewColumns::ColumnFileProperties, 3089
  - Digikam::TableViewColumns::ColumnGeoProperties, 3094
  - Digikam::TableViewColumns::ColumnPhotoProperties, 3103
- setConstraintInterface
  - Digikam::TaggingActionFactory, 3148
- setContainer
  - Digikam::ImageCurves, 1794
- setContextMenuItems
  - Digikam::TagFolderView, 3145
  - Digikam::TagMngrTreeView, 3159
- setContextMenuIcon
  - Digikam::AbstractAlbumTreeView, 161
- setContinueOnError
  - Digikam::FilterActionFilter, 1562
- setControlButtonsPlacement
  - Digikam::DItemsList, 1061
- setControlWidgets
  - Digikam::AdvancedRenameWidget, 249
- setCopyrightNotice
  - Digikam::ItemCopyright, 2033
- setCountHash
  - Digikam::AbstractCountingAlbumModel, 188
- setCreator
  - Digikam::ItemCopyright, 2034
- setCurrent
  - Digikam::SqueezedComboBox, 3040
- setCurrentAlbums
  - Digikam::AbstractAlbumTreeView, 161
  - Digikam::AlbumManager, 299
- setCurrentLanguage
  - Digikam::DPlainTextEdit, 1189
  - Digikam::DTextEdit, 1288
- setCurrentOrientation
  - Digikam::JPEGUtils::JpegRotator, 2287
- setCurrentPage
  - Digikam::DConfigDlg, 852
  - Digikam::DConfigDlgView, 869
  - Digikam::DConfigDlgWdg, 877
- setCurrentProfile
  - Digikam::IccProfilesComboBox, 1769
- setData
  - Digikam::AbstractCheckableAlbumModel, 175
  - Digikam::ImportThumbnailModel, 1961
  - Digikam::ItemThumbnailModel, 2253
  - ShowFoto::ShowfotoThumbnailModel, 3535
- setDatabase
  - Digikam::AlbumManager, 299
- setDate
  - Digikam::DDateEdit, 891
  - Digikam::DDatePicker, 896
- setDateTime
  - Digikam::ItemInfo, 2121
- setDayFilter
  - Digikam::ItemFilterModel, 2067
- setDbEngineErrorHandler
  - Digikam::BdEngineBackend, 479
- setDefaultAlbumModel
  - Digikam::AlbumSelectComboBox, 328
- setDefaultFieldOperator
  - Digikam::SearchXmlWriter, 2942
- setDefaultMaximumNumberOfThreads
  - Digikam::ActionThreadBase, 225
- setDefaultViewOptions
  - Digikam::DItemDelegate, 1056
  - Digikam::ImportDelegate, 1870
  - Digikam::ImportThumbnailDelegate, 1955
  - Digikam::ItemDelegate, 2040
  - Digikam::ItemThumbnailDelegate, 2246
  - Digikam::ItemViewDelegate, 2267
  - Digikam::ItemViewImportDelegate, 2275
  - ShowFoto::ShowfotoDelegate, 3445
  - ShowFoto::ShowfotoItemViewDelegate, 3485
  - ShowFoto::ShowfotoThumbnailDelegate, 3530
- setDestinationFile
  - Digikam::JPEGUtils::JpegRotator, 2288
- setDestinationGeographicPoint
  - Digikam::GeodeticCalculator, 1614
- setDirection
  - Digikam::GeodeticCalculator, 1614
- setDirectSourceImportModel

- Digikam::ImportFilterModel, [1885](#)
- Digikam::ImportSortFilterModel, [1941](#)
- setDirectSourceItemModel
  - Digikam::ImageSortFilterModel, [1832](#)
  - Digikam::ItemFilterModel, [2067](#)
- setDirectSourceShowfotoModel
  - ShowFoto::ShowfotoFilterModel, [3454](#)
  - ShowFoto::ShowfotoSortFilterModel, [3504](#)
- setDisplayingWidget
  - Digikam::ThumbnailLoadThread, [3266](#)
- setDocumentName
  - Digikam::JPEGUtils::JpegRotator, [2288](#)
- setDoNotEmbedOutputProfile
  - Digikam::IccTransform, [1782](#)
- setDrawDraggedItems
  - Digikam::DCategorizedView, [826](#)
- setEmbeddedProfile
  - Digikam::IccTransform, [1782](#)
- setEmitDataChanged
  - Digikam::ImportThumbnailModel, [1961](#)
  - Digikam::ItemThumbnailModel, [2253](#)
  - ShowFoto::ShowfotoThumbnailModel, [3535](#)
- setEnabledContextMenu
  - Digikam::AbstractAlbumTreeView, [161](#)
- setEnabled
  - Digikam::DIntRangeBox, [1036](#)
- setEnabledDrag
  - Digikam::AbstractAlbumModel, [153](#)
- setEngineApiKey
  - Digikam::DOnlineTranslator, [1175](#)
- setEngineUrl
  - Digikam::DOnlineTranslator, [1175](#)
- setEntryPrefix
  - Digikam::StateSavingObject, [3048](#)
- setExif
  - Digikam::MetaEngine, [2498](#)
- setExifComment
  - Digikam::MetaEngine, [2498](#)
- setExifOrientation
  - Digikam::FileActionMngrDatabaseWorker, [1520](#)
- setExifRotate
  - Digikam::ThumbnailCreator, [3249](#)
- setExifTagData
  - Digikam::MetaEngine, [2498](#)
- setExifTagLong
  - Digikam::MetaEngine, [2499](#)
- setExifTagRational
  - Digikam::MetaEngine, [2499](#)
- setExifTagString
  - Digikam::MetaEngine, [2499](#)
- setExifTagURational
  - Digikam::MetaEngine, [2499](#)
- setExifTagUShort
  - Digikam::MetaEngine, [2499](#)
- setExifTagVariant
  - Digikam::MetaEngine, [2499](#)
- setExifThumbnail
  - Digikam::MetaEngine, [2500](#)
- setExifToolProgram
  - Digikam::ExifToolProcess, [1391](#)
- setExifXmpTagDataVariant
  - Digikam, [140](#)
- setExtraData
  - Digikam::Album, [261](#)
- setFaceDetectionSize
  - Digikam::DNNFaceDetectorYuNet, [1126](#)
- setFileWatch
  - Digikam::LoadingCache, [2358](#)
- setFilter
  - Digikam::DPluginConfView, [1203](#)
  - ShowFoto::ShowfotoStackViewFavoriteList, [3509](#)
- setFilterBehavior
  - Digikam::AlbumFilterModel, [272](#)
- setFilterModel
  - Digikam::AddTagsLineEdit, [238](#)
  - Digikam::SearchTextBarDb, [2918](#)
- setFilterSettings
  - Digikam::CoreDB, [675](#)
- setFilterVersion
  - Digikam::DImgThreadedFilter, [1025](#)
- setFocusOnWidget
  - Digikam::AssignNameOverlay, [394](#)
  - Digikam::PersistentWidgetDelegateOverlay, [2633](#)
- setFont
  - Digikam::DFontProperties, [947](#)
- setForeignKeyChecks
  - Digikam::BdEngineBackend, [479](#)
- setFromTemplate
  - Digikam::ItemCopyright, [2034](#)
- setGPSInfo
  - Digikam::MetaEngine, [2500](#)
- setGroupedModel
  - Digikam::MapWidget, [2432](#)
- setGroupingOperateOnAll
  - Digikam::ApplicationSettings, [385](#)
- setGroupList
  - Digikam::ExifToolListView, [1379](#)
- setGroupOperator
  - Digikam::SearchXmlWriter, [2942](#)
- setHeader
  - Digikam::DConfigDlgWdgItem, [881](#)
- setHighlightOnResult
  - Digikam::SearchTextBar, [2915](#)
- setHighlightPixmap
  - Digikam::ThumbnailLoadThread, [3267](#)
- setHighlightSelection
  - Digikam::HistogramPainter, [1729](#)
- setHistogram
  - Digikam::HistogramPainter, [1730](#)
- setHistory
  - Digikam::ItemHistoryGraphModel, [2107](#)
- setHistoryBranchAfter
  - Digikam::DImg, [994](#)
- setHSL
  - Digikam::DColor, [835](#)
- setHudWidget

- Digikam::RegionFrameItem, [2763](#)
- setHue
  - Digikam::DColorValueSelector, [842](#)
  - Digikam::DHueSaturationSelector, [957](#)
- setIcon
  - Digikam::DConfigDlgWdgItem, [881](#)
- setImage
  - Digikam::ClockPhotoDialog, [584](#)
  - Digikam::GraphicsDImgItem, [1697](#)
- setImageComment
  - Digikam::CoreDB, [675](#)
- setImageDateTime
  - Digikam::MetaEngine, [2500](#)
- setImages
  - Digikam::EmptyImageListProvider, [1367](#)
  - Digikam::QListImageListProvider, [2686](#)
- setImageSize
  - Digikam::ImageZoomSettings, [1842](#)
- setImageTagPropertiesJoined
  - Digikam::ItemQueryBuilder, [2198](#)
- setImportFilterModel
  - Digikam::ImportContextMenuHelper, [1860](#)
- setIndent
  - Digikam::DSelector, [1277](#)
- setIntermediate
  - Digikam::WSComboBoxIntermediate, [3404](#)
- setInternalValue
  - Digikam::DAbstractSliderSpinBox, [717](#)
  - Digikam::DDoubleSliderSpinBox, [911](#)
  - Digikam::DSliderSpinBox, [1281](#)
- setInterval
  - Digikam::DIntRangeBox, [1036](#)
- setIptc
  - Digikam::MetaEngine, [2501](#)
- setIptcKeywords
  - Digikam::MetaEngine, [2501](#)
- setIptcSubCategories
  - Digikam::MetaEngine, [2501](#)
- setIptcSubjects
  - Digikam::MetaEngine, [2501](#)
- setIptcTagData
  - Digikam::MetaEngine, [2501](#)
- setIptcTagsStringList
  - Digikam::MetaEngine, [2501](#)
- setIptcTagString
  - Digikam::MetaEngine, [2502](#)
- setIsLessThanHandler
  - Digikam::DItemsList, [1061](#)
- setItem
  - Digikam::GraphicsDImgView, [1700](#)
- setItemAlbum
  - Digikam::CoreDB, [676](#)
- setItemColorWorkSpace
  - Digikam::MetaEngine, [2502](#)
- setItemDelegate
  - Digikam::DConfigDlgView, [870](#)
- setItemDimensions
  - Digikam::MetaEngine, [2502](#)
- setItemFacesMap
  - Digikam::DMetadata, [1093](#)
- setItemFilterModel
  - Digikam::ContextMenuHelper, [637](#)
- setItemFilterSettings
  - Digikam::ItemAlbumFilterModel, [2001](#)
  - Digikam::ItemFilterModel, [2067](#)
- setItemInfo
  - Digikam::DBInfolface, [797](#)
  - Digikam::DMetalInfolface, [1100](#)
- setItemModel
  - Digikam::ModelCompleter, [2530](#)
- setItemOrientation
  - Digikam::MetaEngine, [2502](#)
- setItemPreview
  - Digikam::MetaEngine, [2502](#)
- setItemProgramId
  - Digikam::MetaEngine, [2502](#)
- setItemStatus
  - Digikam::CoreDB, [676](#)
- setItemThatShallBeShown
  - Digikam::ItemVisibilityController, [2284](#)
- setKeepsFilePathCache
  - Digikam::ItemModel, [2167](#)
- setKeepsFileUriCache
  - Digikam::ImportItemModel, [1900](#)
  - ShowFoto::ShowfotoItemModel, [3477](#)
- setLabel
  - Digikam::ProgressItem, [2668](#)
- setLastError
  - Digikam::CoreDbAccess, [680](#)
  - Digikam::FaceDbAccess, [1406](#)
  - Digikam::SimilarityDbAccess, [3021](#)
  - Digikam::ThumbsDbAccess, [3271](#)
- setLatitude
  - Digikam::ItemPosition, [2170](#), [2171](#)
- setLayoutStyle
  - Digikam::AdvancedRenameWidget, [250](#)
- setLibraryFileName
  - Digikam::DPlugin, [1194](#)
- setLineEditText
  - Digikam::TreeViewLineEditComboBox, [3332](#)
- setLinesVisible
  - Digikam::AltLangStrEdit, [366](#)
  - Digikam::DPlainTextEdit, [1189](#)
  - Digikam::DTextEdit, [1288](#)
- setListOnlyAvailable
  - Digikam::ItemLister, [2136](#)
- setLoadingPolicy
  - Digikam::ManagedLoadSaveThread, [2416](#)
- setLoadingProperties
  - Digikam::ThumbnailCreator, [3249](#)
- setLockItem
  - Digikam::GPCamera, [1634](#)
  - Digikam::UMSCamera, [3342](#)
- setLogFilePath
  - Digikam::NREstimate, [2584](#)
- setMainWidget

- Digikam::DPopupFrame, [1238](#)
- Digikam::PanIconFrame, [2612](#)
- setMarkerColor
  - Digikam::DPointSelect, [1234](#)
- setMarkerPixmap
  - Digikam::BackendGoogleMaps, [439](#)
- setMaxLength
  - Digikam::DPlainTextEdit, [1189](#)
  - Digikam::DTextEdit, [1289](#)
- setMessageType
  - Digikam::DNotificationWidget, [1166](#)
- setMetadataTemplate
  - Digikam::ItemInfo, [2121](#)
- setMetadataWritingMode
  - Digikam::MetaEngine, [2503](#)
- setModDateTime
  - Digikam::ItemInfo, [2121](#)
- setMode
  - Digikam::AssignNameWidget, [398](#)
- setModel
  - Digikam::DConfigDlgView, [870](#)
  - Digikam::SearchTextBarDb, [2918](#), [2919](#)
- setModelsFiltered
  - Digikam::ImportThumbnailBar, [1950](#)
  - Digikam::ItemThumbnailBar, [2240](#)
  - ShowFoto::ShowfotoThumbnailBar, [3525](#)
- setModificationTime
  - Digikam::DFileOperations, [939](#)
- setName
  - Digikam::ItemInfo, [2122](#)
- setNeedFileCount
  - Digikam::CollectionScanner, [601](#)
- setNoSelectionText
  - Digikam::AlbumSelectComboBox, [328](#)
- setObserver
  - Digikam::RawProcessingFilter, [2737](#)
- setOnlyLargeThumbnails
  - Digikam::ThumbnailCreator, [3249](#)
- setOriginal
  - Digikam::Imageface, [1807](#)
- setOriginalPos
  - Digikam::DImgChildItem, [1001](#)
- setOutputUrlFromInputUrl
  - Digikam::BatchTool, [462](#)
- setPageWidget
  - Digikam::DConfigDlg, [852](#)
- setParameter
  - Digikam::FaceDetector, [1413](#)
  - Digikam::FacialRecognitionWrapper, [1505](#)
- setParameters
  - Digikam::CoreDbAccess, [680](#)
  - Digikam::SimilarityDbAccess, [3021](#)
- setParentTag
  - Digikam::AddTagsLineEdit, [238](#)
- setParser
  - Digikam::AdvancedRenameWidget, [250](#)
- setParseString
  - Digikam::AdvancedRenameWidget, [250](#)
- setPerformFastScan
  - Digikam::CollectionScanner, [601](#)
- setPersistent
  - Digikam::PersistentWidgetDelegateOverlay, [2633](#)
- setPickLabels
  - Digikam::PickLabelWidget, [2641](#)
- setPixel
  - Digikam::DColor, [835](#)
- setPixmap
  - Digikam::DConfigDlgTitle, [864](#), [865](#)
- setPixmapRequested
  - Digikam::ThumbnailLoadThread, [3267](#)
- setPopupMenuEnabled
  - Digikam::DDateTable, [904](#)
- setPopupStyle
  - Digikam::DNotificationPopup, [1158](#)
- setPos
  - Digikam::DImgChildItem, [1001](#)
- setPosition
  - Digikam::DMultiTabBar, [1105](#)
  - Digikam::DMultiTabBarTab, [1112](#)
- setPositiveFilterIsActive
  - Digikam::GPSMarkerTiler, [1670](#)
- setPreloadThumbnails
  - Digikam::ItemThumbnailModel, [2254](#)
  - ShowFoto::ShowfotoThumbnailModel, [3535](#)
- setPreprocessor
  - Digikam::ItemModel, [2167](#)
- setPreview
  - Digikam::Imageface, [1807](#)
- setPreviewSize
  - Digikam::Imageface, [1807](#)
- setPreviewType
  - Digikam::Imageface, [1807](#)
- setPriority
  - Digikam::DynamicThread, [1320](#)
  - Digikam::FacePipeline, [1425](#)
  - Digikam::WorkerObject, [3397](#)
- setProgress
  - Digikam::ProgressItem, [2668](#)
- setProgressMessage
  - Digikam::EditorToolThreaded, [1344](#)
- setRange
  - Digikam::DIntRangeBox, [1036](#)
- setRatingEdited
  - Digikam::ItemViewDelegate, [2267](#)
  - Digikam::ItemViewImportDelegate, [2276](#)
- setReadOnly
  - Digikam::DDateEdit, [891](#)
- setReadOnlyDrop
  - Digikam::ItemDragDropHandler, [2052](#)
- setRecursive
  - Digikam::ItemLister, [2136](#)
- setRelativePos
  - Digikam::DImgChildItem, [1001](#)
- setRemoveAlphaChannel
  - Digikam::ThumbnailCreator, [3249](#)
- setRenderXGrid

- Digikam::HistogramPainter, 1730
- setReplaceNames
  - Digikam::SearchModel, 2901
- setRestoreCheckState
  - Digikam::AbstractCheckableAlbumTreeView, 182
- setResult
  - Digikam::SharedLoadingTask, 2982
- setRootCheckable
  - Digikam::AbstractCheckableAlbumModel, 175
- setSampleBoxVisible
  - Digikam::DFontProperties, 947
- setSampleText
  - Digikam::DFontProperties, 947
- setSaturation
  - Digikam::DColorValueSelector, 843
  - Digikam::DHueSaturationSelector, 958
- setScale
  - Digikam::HistogramPainter, 1730
- setScrollStepGranularity
  - Digikam::ItemViewCategorized, 2261
- setSearchModel
  - Digikam::ChoiceSearchComboBox, 577
- setSearchTextSettings
  - Digikam::AlbumFilterModel, 272
- setSelectAlbumOnClick
  - Digikam::AbstractAlbumTreeView, 161
- setSelection
  - Digikam::HistogramPainter, 1730
  - Digikam::Imageface, 1807
- setSelectionArea
  - Digikam::DPreviewImage, 1240
  - Digikam::DPreviewManager, 1243
- setSelectOnContextMenu
  - Digikam::AbstractAlbumTreeView, 162
- setSemanticInfo
  - Digikam::BalooWrap, 454
- setSendRemovalSignals
  - Digikam::ImportItemModel, 1901
  - Digikam::ItemModel, 2167
  - ShowFoto::ShowfotoItemModel, 3477
- setSendSurrogatePixmap
  - Digikam::ThumbnailLoadThread, 3267
- setSetting
  - Digikam::CoreDB, 676
  - Digikam::SimilarityDb, 3019
- setSettings
  - Digikam::BatchTool, 462
  - Digikam::RawProcessingFilter, 2737
- setShallBeShown
  - Digikam::AnimatedClearButton, 369
- setShouldLoaded
  - Digikam::DPlugin, 1194
- setShowAtStart
  - Digikam::ProgressItem, 2668
- setShowCheckStateSummary
  - Digikam::AlbumSelectComboBox, 328
- setShowDeleteFaceTagsAction
  - Digikam::TagFolderView, 3145
- setShowFindDuplicateAction
  - Digikam::TagFolderView, 3145
- setSignalsEnabled
  - Digikam::CollectionScanner, 601
- setSize
  - Digikam::SidebarSplitter, 3011
- setSizelsRelative
  - Digikam::DFontProperties, 948
- setSketchImageFromXML
  - Digikam::SketchWidget, 3032
- setSlave
  - Digikam::DImgThreadedFilter, 1025
- setSortCategoriesByNaturalComparison
  - Digikam::DCategorizedSortFilterProxyModel, 822
- setSortKey
  - Digikam::MapWidget, 2432
- setSourceAlbumModel
  - Digikam::AlbumFilterModel, 273
- setSourceFilterModel
  - Digikam::AlbumFilterModel, 273
- setSourceModel
  - Digikam::AlbumFilterModel, 273
  - Digikam::ImageSortFilterModel, 1832
- setSourceTranscriptionEnabled
  - Digikam::DOnlineTranslator, 1175
- setSourceTranslitEnabled
  - Digikam::DOnlineTranslator, 1176
- setSpacing
  - Digikam::ImportDelegate, 1871
  - Digikam::ItemDelegate, 2041
  - Digikam::ItemViewCategorized, 2261
  - Digikam::ItemViewDelegate, 2268
  - Digikam::ItemViewImportDelegate, 2276
  - ShowFoto::ShowfotoItemViewDelegate, 3485
- setStartingGeographicPoint
  - Digikam::GeodeticCalculator, 1614
- setState
  - Digikam::DMultiTabBarTab, 1112
- setStateSavingDepth
  - Digikam::StateSavingObject, 3048
- setStatus
  - Digikam::ProgressItem, 2669
  - Digikam::TransactionItem, 3321
- setStringComparisonType
  - Digikam::ApplicationSettings, 385
- setStyle
  - Digikam::DMultiTabBarTab, 1112
- setSuffix
  - Digikam::DIntRangeBox, 1037
- setSuggestedValues
  - Digikam::CustomStepsDoubleSpinBox, 713
  - Digikam::CustomStepsIntSpinBox, 714
- setSupportingTagModel
  - Digikam::TagCompleter, 3125
- setTab
  - Digikam::DMultiTabBar, 1106
- setTag
  - Digikam::ItemInfo, 2122



- setTagIcon
  - Digikam::CoreDB, [677](#)
- setTagList
  - Digikam::GPSItemContainer, [1651](#)
- setTagName
  - Digikam::CoreDB, [677](#)
- setTagParentID
  - Digikam::CoreDB, [677](#)
- setTagSelected
  - Digikam::AlbumSelectors, [338](#)
- setText
  - Digikam::DConfigDlgTitle, [865](#)
  - Digikam::DNotificationWidget, [1166](#)
- setThumbnail
  - Digikam::ProgressItem, [2669](#)
- setThumbnailCacheSize
  - Digikam::LoadingCache, [2359](#)
- setThumbnailLoadThread
  - Digikam::ItemThumbnailModel, [2254](#)
  - ShowFoto::ShowfotoThumbnailModel, [3536](#)
- setThumbnailSize
  - Digikam::AlbumThumbnailLoader, [353](#)
  - Digikam::DigikamItemView, [976](#)
  - Digikam::DItemDelegate, [1056](#)
  - Digikam::ImportIconView, [1893](#)
  - Digikam::ItemViewDelegate, [2268](#)
  - Digikam::ItemViewImportDelegate, [2276](#)
  - Digikam::ThumbnailCreator, [3249](#)
  - Digikam::ThumbnailLoadThread, [3267](#)
  - Digikam::TrashView, [3325](#)
  - ShowFoto::ShowfotoItemViewDelegate, [3485](#)
- setThumbnailSize
  - Digikam::MapWidget, [2432](#)
- setTiffThumbnail
  - Digikam::MetaEngine, [2503](#)
- setTimeout
  - Digikam::DNotificationPopup, [1158](#)
- settingsChanged
  - Digikam::DConfigDlgMgr, [856](#)
- SettingsType
  - Digikam::ImageQualityConfSelector, [1814](#)
- settingsWidget
  - Digikam::BatchTool, [462](#)
- setTitle
  - Digikam::AltLangStrEdit, [366](#)
- setTitleWidget
  - Digikam::AltLangStrEdit, [366](#)
- setTranslationOptionsEnabled
  - Digikam::DOnlineTranslator, [1176](#)
- setTranslationTranslitEnabled
  - Digikam::DOnlineTranslator, [1176](#)
- setTreeView
  - Digikam::AbstractAlbumTreeViewSelectComboBox, [168](#)
- setTristate
  - Digikam::AbstractCheckableAlbumModel, [175](#)
- setType
  - Digikam::FocusPoint, [1579](#)
- setUIEnabled
  - Digikam::RGWidget, [2789](#)
- setUniqueBehavior
  - Digikam::ItemComments, [2025](#)
- setUnpairedImages
  - Digikam::EmptyImageListProvider, [1367](#)
  - Digikam::QListImageListProvider, [2686](#)
- SetupCollectionDataRole
  - Digikam::SetupCollectionModel, [2960](#)
- SetupCollectionModel
  - Digikam::SetupCollectionModel, [2961](#)
- setUpdateFileTimeStamp
  - Digikam::MetaEngine, [2503](#)
- setupFilter
  - Digikam::DImgThreadedFilter, [1025](#)
- SetupICC
  - Digikam::SetupICC, [2968](#)
- setupKeywords
  - Digikam::DDateEdit, [892](#)
- setValueWidgets
  - Digikam::SearchFieldAlbum, [2829](#)
  - Digikam::SearchFieldCheckBox, [2833](#)
  - Digikam::SearchFieldChoice, [2837](#)
  - Digikam::SearchFieldColorDepth, [2841](#)
  - Digikam::SearchFieldComboBox, [2844](#)
  - Digikam::SearchFieldLabels, [2853](#)
  - Digikam::SearchFieldMonthDay, [2857](#)
  - Digikam::SearchFieldPageOrientation, [2861](#)
  - Digikam::SearchFieldRangeDate, [2864](#)
  - Digikam::SearchFieldRangeDouble, [2868](#)
  - Digikam::SearchFieldRangeInt, [2872](#)
  - Digikam::SearchFieldRangeTime, [2876](#)
  - Digikam::SearchFieldRating, [2880](#)
  - Digikam::SearchFieldText, [2884](#)
- setUsedByLabelsTree
  - Digikam::Album, [261](#)
- setUseMultiCoreCPU
  - Digikam::AutotagsAssignment, [423](#)
  - Digikam::DbCleaner, [778](#)
  - Digikam::FingerPrintsGenerator, [1575](#)
  - Digikam::ImageQualitySorter, [1820](#)
  - Digikam::MaintenanceTool, [2407](#)
  - Digikam::MetadataRemover, [2460](#)
  - Digikam::MetadataSynchronizer, [2470](#)
  - Digikam::ThumbsGenerator, [3281](#)
- setUserData
  - Digikam::AssignNameWidget, [399](#)
- setUserFilterSettings
  - Digikam::CoreDB, [677](#)
- setUsesBusyIndicator
  - Digikam::ProgressItem, [2669](#)
- setUseTokenMenu
  - Digikam::Rule, [2798](#)
- setValue
  - Digikam::DbEngineActionType, [780](#)
- setValues
  - Digikam::DPointSelect, [1234](#)
- setValueWidgetsVisible

- Digikam::SearchFieldAlbum, [2829](#)
- Digikam::SearchFieldCheckBox, [2833](#)
- Digikam::SearchFieldChoice, [2837](#)
- Digikam::SearchFieldComboBox, [2844](#)
- Digikam::SearchFieldLabels, [2853](#)
- Digikam::SearchFieldMonthDay, [2857](#)
- Digikam::SearchFieldRangeDate, [2864](#)
- Digikam::SearchFieldRangeDouble, [2868](#)
- Digikam::SearchFieldRangeInt, [2872](#)
- Digikam::SearchFieldRangeTime, [2876](#)
- Digikam::SearchFieldRating, [2880](#)
- Digikam::SearchFieldText, [2884](#)
- setViewportRect
  - Digikam::RegionFrameItem, [2763](#)
- setVisible
  - Digikam::DPlugin, [1194](#)
  - Digikam::DPluginBqm, [1202](#)
  - Digikam::DPluginDImg, [1217](#)
  - Digikam::DPluginEditor, [1220](#)
  - Digikam::DPluginGeneric, [1223](#)
  - Digikam::DPluginRawImport, [1230](#)
  - Digikam::SearchField, [2826](#)
- setWaitingDataProgress
  - Digikam::DRawDecoder, [1257](#)
- setWatchDisabled
  - Digikam::CollectionManager, [594](#)
- setWatchFlags
  - Digikam::ItemModel, [2168](#)
- setWidget
  - Digikam::DConfigDlgTitle, [866](#)
- setWordWrap
  - Digikam::DNotificationWidget, [1166](#)
- setWriteRawFiles
  - Digikam::MetaEngine, [2503](#)
- setXmp
  - Digikam::MetaEngine, [2503](#)
- setXmpKeywords
  - Digikam::DMetadata, [1094](#)
  - Digikam::MetaEngine, [2503](#)
- setXmpSubCategories
  - Digikam::DMetadata, [1094](#)
  - Digikam::MetaEngine, [2504](#)
- setXmpSubjects
  - Digikam::DMetadata, [1094](#)
  - Digikam::MetaEngine, [2504](#)
- setXmpTagString
  - Digikam::MetaEngine, [2504](#)
- setXmpTagStringBag
  - Digikam::MetaEngine, [2504](#)
- setXmpTagStringLangAlt
  - Digikam::MetaEngine, [2504](#)
- setXmpTagStringListLangAlt
  - Digikam::MetaEngine, [2505](#)
- setXmpTagStringSeq
  - Digikam::MetaEngine, [2505](#)
- setXValue
  - Digikam::DPointSelect, [1234](#)
- setYCbCr
  - Digikam::DColor, [835](#)
- setYValue
  - Digikam::DPointSelect, [1234](#)
- setZoom
  - Digikam::BackendGoogleMaps, [440](#)
  - Digikam::BackendMarble, [447](#)
- SFace
  - Digikam::FaceScanSettings, [1479](#)
- shortenedTagPaths
  - Digikam::ItemPropertiesTab, [2196](#)
  - Digikam::TagsCache, [3189](#)
- shortestDistancesFrom
  - Digikam::Graph< VertexProperties, EdgeProperties >, [1687](#)
- shortestPath
  - Digikam::Graph< VertexProperties, EdgeProperties >, [1687](#)
  - Digikam::Graph< VertexProperties, EdgeProperties >::Path, [1695](#)
- shouldBeVisible
  - Digikam::ThumbBarDock, [3246](#)
- shouldLoaded
  - Digikam::DPlugin, [1194](#)
- show
  - Digikam::ItemViewToolTip, [2278](#)
  - Digikam::ItemVisibilityController, [2284](#)
- showAnimationFinished
  - Digikam::DNotificationWidget, [1166](#)
- showAtStart
  - Digikam::ProgressItem, [2669](#)
- showContextMenu
  - Digikam::DigikamItemView, [976](#)
  - Digikam::ImportIconView, [1894](#)
- showContextMenuAt
  - Digikam::AbstractAlbumTreeView, [162](#)
- showContextMenuOnIndex
  - Digikam::ImportCategorizedView, [1851](#)
  - Digikam::ItemCategorizedView, [2017](#)
  - Digikam::ItemViewCategorized, [2261](#)
  - ShowFoto::ShowfotoCategorizedView, [3433](#)
- showContextMenuOnInfo
  - Digikam::DigikamItemView, [976](#)
  - Digikam::ImportIconView, [1894](#)
- showContinueCancel
  - Digikam::DMessageBox, [1073](#)
- showContinueCancelList
  - Digikam::DMessageBox, [1073](#)
- showContinueCancelWidget
  - Digikam::DMessageBox, [1073](#)
- ShowCountAccordingToSettings
  - Digikam::AbstractAlbumTreeView, [158](#)
- ShowFoto::NoDuplicatesShowfotoFilterModel, [3418](#)
- ShowFoto::Showfoto, [3421](#)
  - infoface, [3426](#)
- ShowFoto::ShowfotoCategorizedView, [3427](#)
  - addOverlay, [3432](#)
  - deselected, [3432](#)
  - dragDropHandler, [3432](#)



- filterModel, [3433](#)
- indexActivated, [3433](#)
- nextIndexHint, [3433](#)
- nextInOrder, [3433](#)
- selected, [3433](#)
- showContextMenuOnIndex, [3433](#)
- showfotoFilterModel, [3434](#)
- showfotoItemInfoActivated, [3434](#)
- ShowFoto::ShowfotoCoordinatesOverlay, [3435](#)
  - checkIndex, [3437](#)
  - createWidget, [3437](#)
  - setActive, [3438](#)
  - slotEntered, [3438](#)
  - visualChange, [3438](#)
- ShowFoto::ShowfotoCoordinatesOverlayWidget, [3438](#)
- ShowFoto::ShowfotoDelegate, [3440](#)
  - acceptsActivation, [3444](#)
  - acceptsToolTip, [3444](#)
  - clearCaches, [3444](#)
  - imageInformationRect, [3444](#)
  - pixmapForDrag, [3444](#)
  - pixmapRect, [3444](#)
  - setDefaultViewOptions, [3445](#)
  - updateContentWidth, [3445](#)
  - updateRects, [3445](#)
  - updateSizeRectsAndPixmaps, [3445](#)
- ShowFoto::ShowfotoDragDropHandler, [3446](#)
  - accepts, [3447](#)
  - createMimeData, [3447](#)
  - dropEvent, [3447](#)
  - mimeTypes, [3447](#)
- ShowFoto::ShowfotoFilterModel, [3449](#)
  - CategorizationModeRole, [3453](#)
  - CategoryFormatRole, [3453](#)
  - categoryIdentifier, [3453](#)
  - compareCategories, [3453](#)
  - infosLessThan, [3453](#)
  - setDirectSourceShowfotoModel, [3454](#)
  - showfotoFilterModel, [3454](#)
  - ShowfotoFilterModelPointerRole, [3453](#)
  - ShowfotoFilterModelRoles, [3452](#)
  - SortOrderRole, [3453](#)
  - subSortLessThan, [3454](#)
- ShowFoto::ShowfotoFolderViewBar, [3455](#)
- ShowFoto::ShowfotoFolderViewBookmarkDlg, [3457](#)
- ShowFoto::ShowfotoFolderViewBookmarkItem, [3458](#)
- ShowFoto::ShowfotoFolderViewBookmarkList, [3459](#)
- ShowFoto::ShowfotoFolderViewBookmarks, [3460](#)
- ShowFoto::ShowfotoFolderViewList, [3461](#)
  - FileDate, [3462](#)
  - FolderViewRole, [3461](#)
- ShowFoto::ShowfotoFolderViewModel, [3462](#)
- ShowFoto::ShowfotoFolderViewSideBar, [3463](#)
  - doLoadState, [3465](#)
  - doSaveState, [3465](#)
- ShowFoto::ShowfotoFolderViewToolTip, [3466](#)
- ShowFoto::ShowfotoFolderViewUndo, [3467](#)
- ShowFoto::ShowfotoInfoIface, [3468](#)
  - openSetupPage, [3470](#)
- ShowFoto::ShowfotoItemInfo, [3470](#)
  - size, [3471](#)
- ShowFoto::ShowfotoItemModel, [3472](#)
  - addShowfotoItemInfoSynchronously, [3475](#)
  - allRefreshingFinished, [3476](#)
  - ExtraDataDuplicateCount, [3475](#)
  - ExtraDataRole, [3475](#)
  - indexForUrl, [3476](#)
  - itemInfosAboutToBeAdded, [3476](#)
  - itemInfosAboutToBeRemoved, [3476](#)
  - itemInfosAdded, [3476](#)
  - itemInfosRemoved, [3476](#)
  - readyForIncrementalRefresh, [3476](#)
  - requestIncrementalRefresh, [3477](#)
  - setKeepsFileUrlCache, [3477](#)
  - setSendRemovalSignals, [3477](#)
  - showfotoItemInfo, [3477](#)
  - showfotoItemInfosCleared, [3477](#)
  - ShowfotoItemModelPointerRole, [3475](#)
  - ShowfotoItemModelRoles, [3475](#)
  - startIncrementalRefresh, [3477](#)
  - ThumbnailRole, [3475](#)
- ShowFoto::ShowfotoItemSortSettings, [3478](#)
  - compare, [3479](#)
  - compareCategories, [3479](#)
  - DefaultOrder, [3479](#)
  - lessThan, [3479](#)
  - SortOrder, [3479](#)
- ShowFoto::ShowfotoItemViewDelegate, [3481](#)
  - acceptsActivation, [3484](#)
  - acceptsToolTip, [3484](#)
  - asDelegate, [3484](#)
  - gridSize, [3484](#)
  - imageInformationRect, [3484](#)
  - mouseMoved, [3485](#)
  - pixmapRect, [3485](#)
  - setDefaultViewOptions, [3485](#)
  - setSpacing, [3485](#)
  - setThumbnailSize, [3485](#)
- ShowFoto::ShowfotoKineticScroller, [3486](#)
- ShowFoto::ShowfotoNormalDelegate, [3487](#)
  - updateRects, [3491](#)
- ShowFoto::ShowfotoSettings, [3492](#)
- ShowFoto::ShowfotoSetup, [3494](#)
  - execSinglePage, [3497](#)
- ShowFoto::ShowfotoSetupMetadata, [3497](#)
- ShowFoto::ShowfotoSetupMisc, [3498](#)
- ShowFoto::ShowfotoSetupPlugins, [3499](#)
- ShowFoto::ShowfotoSetupRaw, [3500](#)
- ShowFoto::ShowfotoSetupToolTip, [3501](#)
- ShowFoto::ShowfotoSortFilterModel, [3502](#)
  - mapToSourceShowfotoModel, [3504](#)
  - setDirectSourceShowfotoModel, [3504](#)
  - showfotoFilterModel, [3504](#)
  - showfotoItemInfosSorted, [3504](#)
- ShowFoto::ShowfotoStackViewFavoriteItem, [3505](#)
  - FavoriteFolder, [3506](#)

- FavoritelItem, [3506](#)
- FavoriteRoot, [3506](#)
- FavoriteType, [3506](#)
- hierarchyFromParent, [3506](#)
- ShowFoto::ShowfotoStackViewFavoritelItemDlg, [3507](#)
- ShowFoto::ShowfotoStackViewFavoriteList, [3508](#)
  - setFilter, [3509](#)
  - signalSearchResult, [3509](#)
- ShowFoto::ShowfotoStackViewFavorites, [3510](#)
- ShowFoto::ShowfotoStackViewItem, [3511](#)
- ShowFoto::ShowfotoStackViewList, [3512](#)
  - FileDate, [3513](#)
  - StackViewRole, [3513](#)
- ShowFoto::ShowfotoStackViewSideBar, [3514](#)
  - doLoadState, [3516](#)
  - doSaveState, [3516](#)
- ShowFoto::ShowfotoStackViewToolTip, [3517](#)
- ShowFoto::ShowfotoThumbnailBar, [3519](#)
  - setModelsFiltered, [3525](#)
- ShowFoto::ShowfotoThumbnailDelegate, [3526](#)
  - acceptsActivation, [3530](#)
  - setDefaultViewOptions, [3530](#)
  - updateContentWidth, [3530](#)
  - updateRects, [3530](#)
- ShowFoto::ShowfotoThumbnailModel, [3531](#)
  - data, [3535](#)
  - setData, [3535](#)
  - setEmitDataChanged, [3535](#)
  - setPreloadThumbnails, [3535](#)
  - setThumbnailLoadThread, [3536](#)
  - showfotoItemInfosCleared, [3536](#)
  - ShowfotoThumbnailModel, [3535](#)
- showfotoFilterModel
  - ShowFoto::ShowfotoCategorizedView, [3434](#)
  - ShowFoto::ShowfotoFilterModel, [3454](#)
  - ShowFoto::ShowfotoSortFilterModel, [3504](#)
- ShowfotoFilterModelPointerRole
  - ShowFoto::ShowfotoFilterModel, [3453](#)
- ShowfotoFilterModelRoles
  - ShowFoto::ShowfotoFilterModel, [3452](#)
- showfotoItemInfo
  - ShowFoto::ShowfotoItemModel, [3477](#)
- showfotoItemInfoActivated
  - ShowFoto::ShowfotoCategorizedView, [3434](#)
- showfotoItemInfosCleared
  - ShowFoto::ShowfotoItemModel, [3477](#)
  - ShowFoto::ShowfotoThumbnailModel, [3536](#)
- showfotoItemInfosSorted
  - ShowFoto::ShowfotoSortFilterModel, [3504](#)
- ShowfotoItemModelPointerRole
  - ShowFoto::ShowfotoItemModel, [3475](#)
- ShowfotoItemModelRoles
  - ShowFoto::ShowfotoItemModel, [3475](#)
- ShowfotoThumbnailModel
  - ShowFoto::ShowfotoThumbnailModel, [3535](#)
- showItem
  - Digikam::ItemVisibilityController, [2284](#)
- showOnIndex
  - Digikam::AssignNameOverlay, [394](#)
  - Digikam::PersistentWidgetDelegateOverlay, [2633](#)
- showPageHeader
  - Digikam::DConfigDlgView, [870](#)
- showProgressView
  - Digikam::ProgressManager, [2677](#)
- showSideBars
  - Digikam::DXmlGuiWindow, [1317](#)
- showThumbBar
  - Digikam::DXmlGuiWindow, [1317](#)
- showToolTip
  - Digikam::ItemViewCategorized, [2262](#)
- showYesNo
  - Digikam::DMessageBox, [1073](#)
- showYesNoList
  - Digikam::DMessageBox, [1073](#)
- showYesNoWidget
  - Digikam::DMessageBox, [1073](#)
- shutDown
  - Digikam::DynamicThread, [1320](#)
  - Digikam::ScanController, [2812](#)
  - Digikam::WorkerObject, [3397](#)
- shutDownExifTool
  - Digikam::ExifToolProcess, [1391](#)
- SideBar
  - Digikam::Sidebar, [3007](#)
- SidebarWidget
  - Digikam::SidebarWidget, [3013](#)
- sidecarFilePathForFile
  - Digikam::MetaEngine, [2505](#)
- signalAdvance
  - Digikam::MaintenanceThread, [2405](#)
- signalAlbumAboutToBeMoved
  - Digikam::AlbumManager, [299](#)
- signalAlbumHasBeenDeleted
  - Digikam::AlbumManager, [299](#)
- signalAlbumMoved
  - Digikam::AlbumManager, [299](#)
- signalAssignSettings2Widget
  - Digikam::BatchTool, [462](#)
- signalClicked
  - Digikam::DMultiTabBarButton, [1108](#)
- signalFailed
  - Digikam::AlbumThumbnailLoader, [353](#)
- signalFileMetadataChanged
  - Digikam::ItemAttributesWatch, [2009](#)
- signalFindDuplicates
  - Digikam::AlbumSelectionTreeView, [335](#)
- signalFinished
  - Digikam::DOnlineTranslator, [1176](#)
- signalImageLoaded
  - Digikam::LoadSaveThread, [2380](#)
- signalImageRatingChanged
  - Digikam::ItemAttributesWatch, [2009](#)
- signalImagesChanged
  - Digikam::ItemAttributesWatch, [2009](#)
- signalImageStartedLoading
  - Digikam::LoadSaveThread, [2381](#)

- signalImageTagsChanged
  - Digikam::ItemAttributesWatch, [2009](#)
- signalLoadingProgress
  - Digikam::LoadSaveThread, [2381](#)
- signalMoreCompleteLoadingAvailable
  - Digikam::LoadSaveThread, [2381](#)
- signalProgressChanged
  - Digikam::RGWidget, [2790](#)
- signalReloadThumbnails
  - Digikam::AlbumThumbnailLoader, [353](#)
- signalSearchResult
  - Digikam::DPluginConfView, [1203](#)
  - ShowFoto::ShowfotoStackViewFavoriteList, [3509](#)
- signalSelectionMoved
  - Digikam::PanIconWidget, [2614](#)
- signalSetUIEnabled
  - Digikam::RGWidget, [2790](#)
- signalShowOnlyAvailableAlbumsChanged
  - Digikam::AlbumManager, [299](#)
- signalTagFilterChanged
  - Digikam::FilterSideBarWidget, [1567](#)
- signalThumbnail
  - Digikam::AlbumThumbnailLoader, [353](#)
- signalThumbnailLoaded
  - Digikam::ThumbnailLoadThread, [3267](#)
- signalUndoCommand
  - Digikam::RGWidget, [2790](#)
- signatureAsText
  - Digikam::HaarIface, [1721](#)
- SimilarityDbAccess
  - Digikam::SimilarityDbAccess, [3020](#)
- similarityTo
  - Digikam::ItemExtendedProperties, [2053](#)
- SimpleFiltering
  - Digikam::AlbumFilterModel, [270](#)
- SimpleResize
  - Digikam::GreycstorationFilter, [1705](#)
- singleGroupMainItem
  - Digikam::ItemInfoList, [2129](#)
- singleItem
  - Digikam::ProgressManager, [2677](#)
- Size
  - Digikam::ThumbnailSize, [3269](#)
- size
  - Digikam::CamItemInfo, [542](#)
  - Digikam::EmptyImageListProvider, [1367](#)
  - Digikam::QListImageListProvider, [2686](#)
  - ShowFoto::ShowfotoItemInfo, [3471](#)
- sizeHint
  - Digikam::DDatePicker, [897](#)
  - Digikam::DDateTable, [904](#)
  - Digikam::FaceRejectionOverlayButton, [1477](#)
  - Digikam::ImportRotateOverlayButton, [1935](#)
  - Digikam::ItemFullScreenOverlayButton, [2088](#)
  - Digikam::ItemRotateOverlayButton, [2212](#)
  - Digikam::ItemSelectionOverlayButton, [2224](#)
  - Digikam::ItemViewHoverButton, [2269](#)
  - Digikam::TableViewColumn, [3073](#)
  - Digikam::TableViewColumns::ColumnThumbnail, [3106](#)
  - Digikam::TableViewItemDelegate, [3108](#)
- sizelsRelative
  - Digikam::DFontProperties, [948](#)
- Skip
  - Digikam::FaceScanSettings, [1479](#)
- SkipAlreadyScanned
  - Digikam::FacePipeline, [1423](#)
- slotAbortAll
  - Digikam::ProgressManager, [2677](#)
- slotAlbumDelete
  - Digikam::AlbumModificationHelper, [314](#)
- slotAlbumEdit
  - Digikam::AlbumModificationHelper, [314](#)
- slotAlbumNew
  - Digikam::AlbumModificationHelper, [314](#)
- slotAlbumRename
  - Digikam::AlbumModificationHelper, [314](#)
- slotAppendPressed
  - Digikam::SetupCollectionModel, [2961](#)
- slotApplicationSettingsChanged
  - Digikam::LightTableWindow, [2350](#)
- slotAssignSettings2Widget
  - Digikam::BatchTool, [462](#)
- slotAwayFromSelection
  - Digikam::TableView, [3070](#)
- slotCategoryButtonPressed
  - Digikam::SetupCollectionModel, [2961](#)
- slotClustersClicked
  - Digikam::MapWidget, [2432](#)
- slotClustersMoved
  - Digikam::MapWidget, [2432](#)
- slotCreateFuzzySearchFromDropped
  - Digikam::SearchModificationHelper, [2904](#)
- slotCreateFuzzySearchFromImage
  - Digikam::SearchModificationHelper, [2905](#)
- slotCreateFuzzySearchFromSketch
  - Digikam::SearchModificationHelper, [2905](#)
- slotCreateTimeLineSearch
  - Digikam::SearchModificationHelper, [2905](#)
- slotDateTimeForUrl
  - Digikam::DInfoInterface, [1032](#)
  - Digikam::DMetaInfoIface, [1100](#)
- slotDeleteSelected
  - Digikam::TableView, [3070](#)
- slotEnabledInternalWidgets
  - Digikam::AltLangStrEdit, [366](#)
- slotEntered
  - Digikam::AbstractWidgetDelegateOverlay, [209](#)
  - Digikam::GroupIndicatorOverlay, [1711](#)
  - Digikam::ImportCoordinatesOverlay, [1864](#)
  - Digikam::ImportDownloadOverlay, [1875](#)
  - Digikam::ImportLockOverlay, [1912](#)
  - Digikam::ImportRatingOverlay, [1925](#)
  - Digikam::ItemCoordinatesOverlay, [2029](#)
  - Digikam::ItemRatingOverlay, [2203](#)
  - Digikam::PersistentWidgetDelegateOverlay, [2634](#)

- Digikam::TagsLineEditOverlay, [3201](#)
- ShowFoto::ShowfotoCoordinatesOverlay, [3438](#)
- slotFaceTagDelete
  - Digikam::TagModificationHelper, [3170](#)
- slotFitToWindow
  - Digikam::ItemIconView, [2112](#)
- slotImageChange
  - Digikam::ItemAlbumModel, [2008](#)
- slotImageQualitySorter
  - Digikam::ItemIconView, [2112](#)
- slotItemDisplaySettingsChanged
  - Digikam::MapWidget, [2432](#)
- slotLayoutChanged
  - Digikam::PersistentWidgetDelegateOverlay, [2634](#)
- slotMetadataChangedForUrl
  - Digikam::DInfoInterface, [1033](#)
  - Digikam::DMetaInfoInterface, [1100](#)
- slotMouseMoveModeChanged
  - Digikam::MapWidget, [2433](#)
- slotMultipleFaceTagDel
  - Digikam::TagModificationHelper, [3170](#)
- slotMultipleTagDel
  - Digikam::TagModificationHelper, [3170](#)
- slotMultipleTagsToFaceTags
  - Digikam::TagModificationHelper, [3171](#)
- slotNewModelData
  - Digikam::GPSMarkerTiler, [1670](#)
- slotNewSelectionFromMap
  - Digikam::MapWidget, [2433](#)
- slotRemoveTag
  - Digikam::ItemIconView, [2113](#)
- slotReset
  - Digikam::AbstractWidgetDelegateOverlay, [209](#)
  - Digikam::PersistentWidgetDelegateOverlay, [2634](#)
- slotRowsRemoved
  - Digikam::PersistentWidgetDelegateOverlay, [2634](#)
- slotScheduleUpdate
  - Digikam::BackendMarble, [448](#)
- slotSearchDelete
  - Digikam::SearchModificationHelper, [2906](#)
- slotSearchRename
  - Digikam::SearchModificationHelper, [2906](#)
- slotSetCurrentWhenAvailable
  - Digikam::TableView, [3070](#)
- slotSetHighlightArea
  - Digikam::DPreviewImage, [1241](#)
- slotSetHighlightShown
  - Digikam::DPreviewImage, [1241](#)
- slotSetSelection
  - Digikam::DPreviewImage, [1241](#)
- slotSetupChanged
  - Digikam::DigikamItemView, [976](#)
  - Digikam::ImportIconView, [1894](#)
  - Digikam::ImportThumbnailBar, [1950](#)
  - Digikam::ItemThumbnailBar, [2241](#)
- slotStandardCancelHandler
  - Digikam::ProgressManager, [2677](#)
- slotTagDelete
  - Digikam::TagModificationHelper, [3171](#)
- slotTagEdit
  - Digikam::TagModificationHelper, [3171](#)
- slotTagNew
  - Digikam::TagModificationHelper, [3171](#)
- slotTagToFaceTag
  - Digikam::TagModificationHelper, [3172](#)
- slotUpdateActionsEnabled
  - Digikam::MapWidget, [2433](#)
- slotViewportEntered
  - Digikam::PersistentWidgetDelegateOverlay, [2634](#)
- Small
  - Digikam::ThumbnailSize, [3269](#)
- smoothScale
  - Digikam::DImg, [994](#)
- smoothScaleClipped
  - Digikam::DImg, [994](#)
- snapItemsTo
  - Digikam::GPSBookmarkModelHelper, [1637](#)
- snappedZoomFactor
  - Digikam::ImageZoomSettings, [1842](#)
- snappedZoomStep
  - Digikam::ImageZoomSettings, [1843](#)
- Socks5Proxy
  - Digikam::SystemSettings, [3065](#)
- sort
  - Digikam::DCategorizedSortFilterProxyModel, [822](#)
  - Digikam::TableViewModel, [3111](#)
- SortByAspectRatio
  - Digikam::ItemSortSettings, [2231](#)
- SortByFaces
  - Digikam::ItemSortSettings, [2231](#)
- SortByImageSize
  - Digikam::ItemSortSettings, [2231](#)
- sortByProximity
  - Digikam::ItemScanner, [2218](#)
- SortCategoriesAlphabetically
  - Digikam::ActionItemModel, [216](#)
- SortCategoriesByInsertionOrder
  - Digikam::ActionItemModel, [216](#)
- sortCategoriesByNaturalComparison
  - Digikam::DCategorizedSortFilterProxyModel, [822](#)
- sortColumn
  - Digikam::DCategorizedSortFilterProxyModel, [823](#)
- SortOrder
  - Digikam::CamItemSortSettings, [543](#)
  - Digikam::ItemSortSettings, [2230](#)
  - ShowFoto::ShowfotoItemSortSettings, [3479](#)
- sortOrder
  - Digikam::DCategorizedSortFilterProxyModel, [823](#)
- SortOrderRole
  - Digikam::ImportFilterModel, [1884](#)
  - Digikam::ItemFilterModel, [2065](#)
  - ShowFoto::ShowfotoFilterModel, [3453](#)
- SortRole
  - Digikam::ItemSortSettings, [2231](#)
- sortRoleData
  - Digikam::AbstractAlbumModel, [153](#)

- Digikam::DateAlbumModel, [753](#)
- soundTrackLength
  - Digikam::FFmpegLauncher, [1511](#)
- Source
  - Digikam::HistoryImageld, [1735](#)
- source
  - Digikam::DOnlineTranslator, [1176](#)
  - Digikam::ItemCopyright, [2034](#)
- sourceLanguage
  - Digikam::DOnlineTranslator, [1177](#)
- sourceLanguageName
  - Digikam::DOnlineTranslator, [1177](#)
- sourceTranscription
  - Digikam::DOnlineTranslator, [1177](#)
- sourceTranslit
  - Digikam::DOnlineTranslator, [1177](#)
- SpecialMatch
  - Digikam::AlbumFilterModel, [270](#)
- spectral\_chromaticity
  - Digikam, [143](#)
- spellCheckSettings
  - Digikam::DPlainTextEdit, [1190](#)
  - Digikam::DTextEdit, [1289](#)
- SPHERE
  - Digikam::Ellipsoid, [1357](#)
- SQLException
  - Digikam::BdEngineBackend, [475](#)
- SQLiteDatabaseType
  - Digikam::DbEngineParameters, [788](#)
- SqueezedComboBox
  - Digikam::SqueezedComboBox, [3038](#)
- sRGB
  - Digikam::lccProfile, [1765](#)
- SSDMOBILENET
  - Digikam::FaceScanSettings, [1479](#)
- StackedViewMode
  - Digikam::ImportStackedView, [1943](#)
- StackViewRole
  - ShowFoto::ShowfotoStackViewList, [3513](#)
- standardView
  - Digikam::DNotificationPopup, [1159](#)
- start
  - Digikam::DynamicThread, [1321](#)
  - Digikam::FacePipelineDetect, [1433](#)
  - Digikam::FacePipelineDetectRecognize, [1437](#)
  - Digikam::FacePipelineEdit, [1442](#)
  - Digikam::FacePipelineRecognize, [1455](#)
  - Digikam::FacePipelineReset, [1459](#)
  - Digikam::FacePipelineRetrain, [1463](#)
- startAlbumsJobThread
  - Digikam::DBJobsManager, [802](#)
- startAnalyse
  - Digikam::AutoCrop, [408](#)
  - Digikam::DImgThreadedAnalyser, [1019](#)
  - Digikam::NREstimate, [2584](#)
- startDate
  - Digikam::CoreDbUrl, [695](#)
- startDatesJobThread
  - Digikam::DBJobsManager, [802](#)
- startDTrashItemsListingForCollection
  - Digikam::IOJobsManager, [1987](#)
- started
  - Digikam::ClickDragReleaseItem, [583](#)
- startEditing
  - Digikam::ComboBoxDelegate, [618](#)
- StartError
  - Digikam::DatabaseServerError, [737](#)
- startGPSJobThread
  - Digikam::DBJobsManager, [802](#)
- startIncrementalRefresh
  - Digikam::ImportItemModel, [1901](#)
  - Digikam::ItemModel, [2168](#)
  - ShowFoto::ShowfotoItemModel, [3477](#)
- startIOJobs
  - Digikam::IOJobsManager, [1987](#)
- startLookup
  - Digikam::LookupAltitudeGeonames, [2398](#)
- startRefresh
  - Digikam::ImportItemModel, [1901](#)
  - Digikam::ItemModel, [2168](#)
- startScan
  - Digikam::AlbumManager, [299](#)
- startSearchesJobThread
  - Digikam::DBJobsManager, [803](#)
- startTagsJobThread
  - Digikam::DBJobsManager, [803](#)
- State
  - Digikam::ItemVisibilityController, [2283](#)
- StateSavingDepth
  - Digikam::StateSavingObject, [3045](#)
- StateSavingObject
  - Digikam::StateSavingObject, [3046](#)
- staticMetacallPointer
  - Digikam::ParallelAdapter< A >, [2617](#)
- Status
  - Digikam::BdEngineBackend, [475](#)
  - Digikam::CollectionLocation, [587](#)
  - Digikam::DisjointMetadataDataFields, [1048](#)
  - Digikam::MetadataHub, [2440](#)
- status
  - Digikam::CollectionLocation, [588](#)
  - Digikam::CoreDbDownloadHistory, [687](#)
  - Digikam::ProgressItem, [2669](#)
- statusBarText
  - Digikam::TrashView, [3325](#)
- StayPoppedUpComboBox
  - Digikam::StayPoppedUpComboBox, [3053](#)
- stayVisibleWhenAnimatedOut
  - Digikam::AnimatedClearButton, [369](#)
- stopLoading
  - Digikam::ManagedLoadSaveThread, [2417](#)
- stopSaving
  - Digikam::ManagedLoadSaveThread, [2417](#)
- store
  - Digikam::ThumbnailCreator, [3250](#)
- storeDetailThumbnail



- Digikam::ThumbnailLoadThread, [3268](#)
- storedSize
  - Digikam::ThumbnailCreator, [3250](#)
- StoreIntermediates
  - Digikam::VersionFileOperation, [3356](#)
- storeThumbnails
  - Digikam::FaceUtils, [1502](#)
- StrictFiltering
  - Digikam::AlbumFilterModel, [270](#)
- StringComparisonType
  - Digikam::ApplicationSettings, [383](#)
- striplmageData
  - Digikam::DImg, [994](#)
- StyleSheetDebugger
  - Digikam::StyleSheetDebugger, [3060](#)
- SubclassRoles
  - Digikam::ItemModel, [2164](#)
- subjectCode
  - Digikam::ItemExtendedProperties, [2054](#)
- subSortLessThan
  - Digikam::DCategorizedSortFilterProxyModel, [823](#)
  - Digikam::ImportFilterModel, [1885](#)
  - Digikam::ItemFilterModel, [2067](#)
  - ShowFoto::ShowfotoFilterModel, [3454](#)
- suggestedWatchFlags
  - Digikam::ItemFilterModel, [2068](#)
- supportAlbums
  - Digikam::DBInfolface, [797](#)
  - Digikam::DMetalInfolface, [1100](#)
- supportBmff
  - Digikam::MetaEngine, [2505](#)
- supportedDropActions
  - Digikam::DragDropModelImplementation, [1248](#)
- supportedFilters
  - Digikam::BasicDImgFilterGenerator< T >, [456](#)
  - Digikam::DImgFilterGenerator, [1003](#)
  - Digikam::DImgFilterManager, [1006](#)
- supportedImageMimeTypes
  - Digikam, [140](#)
- supportedVersions
  - Digikam::BasicDImgFilterGenerator< T >, [456](#)
  - Digikam::DImgFilterGenerator, [1003](#)
  - Digikam::DImgFilterManager, [1006](#)
- suspendCollectionScan
  - Digikam::ScanController, [2812](#)
- SVCD1
  - Digikam::VidSlideSettings, [3375](#)
- SVCD2
  - Digikam::VidSlideSettings, [3375](#)
- SVGA
  - Digikam::VidSlideSettings, [3375](#)
- SVM
  - Digikam::OpenCVDNNFaceRecognizer, [2601](#)
- SXGA
  - Digikam::VidSlideSettings, [3376](#)
- SXGAPLUS
  - Digikam::VidSlideSettings, [3376](#)
- TableViewSelectionModeSyncer, [3113](#)
- tabStyle
  - Digikam::DMultiTabBar, [1106](#)
- TAG
  - Digikam::Album, [256](#)
- tagAdded
  - Digikam::TagsCache, [3189](#)
- tagFilterModel
  - Digikam::DBInfolface, [797](#)
  - Digikam::DInfoInterface, [1033](#)
- TagFilterView
  - Digikam::TagFilterView, [3136](#)
- TagFolderView
  - Digikam::TagFolderView, [3143](#)
- tagForColorLabel
  - Digikam::TagsCache, [3189](#)
- tagForName
  - Digikam::TagsCache, [3189](#)
- tagForPath
  - Digikam::TagsCache, [3189](#)
- tagForPerson
  - Digikam::FaceTags, [1490](#)
- tagForPickLabel
  - Digikam::TagsCache, [3189](#)
- TaggingAction
  - Digikam::TaggingAction, [3146](#)
- taggingActionSelected
  - Digikam::AddTagsComboBox, [236](#)
  - Digikam::AddTagsLineEdit, [238](#)
- tagId
  - Digikam::CoreDbUrl, [695](#)
- tagIds
  - Digikam::ItemInfo, [2122](#)
- TagModificationHelper
  - Digikam::TagModificationHelper, [3169](#)
- tagName
  - Digikam::TagsCache, [3190](#)
- tagNames
  - Digikam::AlbumManager, [300](#)
- tagPath
  - Digikam::TagsCache, [3190](#)
  - Digikam::TAlbum, [3226](#)
- tagPaths
  - Digikam::AlbumManager, [300, 301](#)
- TagProperties
  - Digikam::TagProperties, [3173](#)
- TagRegion
  - Digikam::TagRegion, [3180](#)
- tags
  - Digikam::DisjointMetadata, [1046](#)
- TAGS\_DATABASE
  - Digikam::ExifToolProcess, [1390](#)
- tagsDatabase
  - Digikam::ExifToolParser, [1386](#)
- tagsDbToOrderedMap
  - Digikam::ExifToolParser, [1386](#)
- tagsForName
  - Digikam::TagsCache, [3190](#)

- tagsListing
  - Digikam::TagsDBJobsThread, [3194](#)
- TagsMap
  - Digikam::MetaEngine, [2484](#)
- tagsWithProperty
  - Digikam::TagsCache, [3190](#)
- tagsWithPropertyCached
  - Digikam::TagsCache, [3190](#)
- Task
  - Digikam::VersionFileOperation, [3356](#)
- TerminationPolicy
  - Digikam::ManagedLoadSaveThread, [2416](#)
- TerminationPolicyTerminateAll
  - Digikam::ManagedLoadSaveThread, [2416](#)
- TerminationPolicyTerminateLoading
  - Digikam::ManagedLoadSaveThread, [2416](#)
- TerminationPolicyTerminatePreloading
  - Digikam::ManagedLoadSaveThread, [2416](#)
- TerminationPolicyWait
  - Digikam::ManagedLoadSaveThread, [2416](#)
- text
  - Digikam::DConfigDlgTitle, [866](#)
  - Digikam::DNotificationWidget, [1167](#)
  - Digikam::DPlainTextEdit, [1190](#)
  - Digikam::DTextEdit, [1289](#)
- TextStyle
  - Digikam::DMultiTabBar, [1104](#)
- THEORA
  - Digikam::VidSlideSettings, [3374](#)
- threadMutex
  - Digikam::DynamicThread, [1321](#)
- thresholds
  - Digikam::NRContainer, [2579](#)
- thumbbarVisibility
  - Digikam::DXmlGuiWindow, [1317](#)
- ThumbnailCreator
  - Digikam::ThumbnailCreator, [3248](#)
- thumbnailCreator
  - Digikam::ThumbnailLoadThread, [3268](#)
- ThumbnailImageCatcher
  - Digikam::ThumbnailImageCatcher, [3252](#)
- thumbnailInfo
  - Digikam::ThumbsDbInfoProvider, [3277](#)
- thumbnailLoaded
  - Digikam::LoadSaveNotifier, [2373](#)
  - Digikam::LoadSaveThread, [2381](#)
  - Digikam::ThumbnailLoadThread, [3268](#)
- thumbnailPixmap
  - Digikam::ItemFaceDelegate, [2060](#)
- ThumbnailRole
  - Digikam::ImportItemModel, [1898](#)
  - Digikam::ItemModel, [2164](#)
  - ShowFoto::ShowfotoItemModel, [3475](#)
- thumbnailsAvailable
  - Digikam::ThumbnailLoadThread, [3268](#)
- thumbnailToPixmapSize
  - Digikam::ThumbnailLoadThread, [3268](#)
- ThumbsDbAccess
  - Digikam::ThumbsDbAccess, [3271](#)
- ThumbsGenerator
  - Digikam::ThumbsGenerator, [3280](#)
- ThumbsSizeCtrl
  - Digikam::DZoomBar, [1324](#)
- tileNew
  - Digikam::GPSMarkerTiler, [1670](#)
  - Digikam::ItemMarkerTiler, [2158](#)
- tilerFlags
  - Digikam::AbstractMarkerTiler, [199](#)
  - Digikam::ItemMarkerTiler, [2158](#)
- tipContents
  - Digikam::BlackFrameToolTip, [484](#)
  - Digikam::DItemToolTip, [1065](#)
  - Digikam::FreeSpaceToolTip, [1597](#)
  - Digikam::ItemViewToolTip, [2278](#)
- title
  - Digikam::Album, [262](#)
  - Digikam::ItemInfo, [2122](#)
- titles
  - Digikam::DisjointMetadata, [1046](#)
- titleLabel
  - Digikam::AltLangStrEdit, [367](#)
- toFaceTagsIfaces
  - Digikam::FaceUtils, [1502](#)
- toggled
  - Digikam::DConfigDlgWdgItem, [881](#)
  - Digikam::DConfigDlgWdgModel, [887](#)
- toggleZoomActions
  - Digikam::EditorWindow, [1351](#)
- toJson
  - Digikam::DOnlineTranslator, [1177](#)
  - Digikam::DOnlineTranslatorOption, [1180](#)
- tokens
  - Digikam::Rule, [2798](#)
- toolGroup
  - Digikam::BatchTool, [463](#)
- toolOperations
  - Digikam::BatchTool, [463](#)
- toolVersion
  - Digikam::BatchTool, [463](#)
- ToplevelMenuCategory
  - Digikam::ActionItemModel, [216](#)
- toSourceIndex
  - Digikam::RGTagModel, [2787](#)
- toString
  - Digikam::DPluginAction, [1198](#)
  - Digikam::DPluginAuthor, [1198](#)
- totalFilesToScan
  - Digikam::CollectionScanner, [602](#)
- toVariant
  - Digikam::TagRegion, [3181](#)
- toXml
  - Digikam::DImageHistory, [978](#)
- train
  - Digikam::FacialRecognitionWrapper, [1505](#)
- TrainAll
  - Digikam::FacePipelineBase, [1428](#)

- trainData
  - Digikam::FaceDb, [1405](#)
- Trainer
  - Digikam::MLPipelineFoundation, [2526](#)
- trainer
  - Digikam::FacePipelineDetect, [1433](#)
  - Digikam::FacePipelineDetectRecognize, [1437](#)
  - Digikam::FacePipelineEdit, [1442](#)
  - Digikam::FacePipelineRecognize, [1455](#)
  - Digikam::FacePipelineReset, [1459](#)
  - Digikam::FacePipelineRetrain, [1463](#)
- TrainNew
  - Digikam::FacePipelineBase, [1428](#)
- TrainRemove
  - Digikam::FacePipelineBase, [1428](#)
- TrainReset
  - Digikam::FacePipelineBase, [1428](#)
- TRANS\_ALL\_EXIF
  - Digikam::ExifToolProcess, [1390](#)
- TRANS\_ALL\_IPTC
  - Digikam::ExifToolProcess, [1390](#)
- TRANS\_ALL\_XMP
  - Digikam::ExifToolProcess, [1390](#)
- TRANS\_TAGS
  - Digikam::ExifToolProcess, [1390](#)
- transform
  - Digikam::DImg, [994](#)
  - Digikam::FileActionMgr, [1516](#)
  - Digikam::FileActionMgrFileWorker, [1523](#)
  - Digikam::Haar::Calculator, [1715](#)
- TransformationAction
  - Digikam::MetaEngineRotation, [2509](#)
- transformations
  - Digikam::MetaEngineRotation, [2510](#)
- transformDefault
  - Digikam::lccManager, [1759](#)
- transformForDisplay
  - Digikam::lccManager, [1759](#)
- TransformTool
  - Digikam::BatchTool, [460](#)
- transitiveReduction
  - Digikam::Graph< VertexProperties, EdgeProperties >, [1687](#)
- translate
  - Digikam::DOnlineTranslator, [1177](#)
- translateTags
  - Digikam::ExifToolParser, [1386](#)
- TranslateTagsOps
  - Digikam::ExifToolProcess, [1390](#)
- translation
  - Digikam::DOnlineTranslator, [1178](#)
- TranslationError
  - Digikam::DOnlineTranslator, [1171](#)
- translationLanguage
  - Digikam::DOnlineTranslator, [1178](#)
- translationLanguageName
  - Digikam::DOnlineTranslator, [1178](#)
- translationOptions
  - Digikam::DOnlineTranslator, [1178](#)
- TRANSLATIONS\_LIST
  - Digikam::ExifToolProcess, [1390](#)
- translationsList
  - Digikam::ExifToolParser, [1387](#)
- translationTranslit
  - Digikam::DOnlineTranslator, [1178](#)
- Tree
  - Digikam::OpenCVDNNFaceRecognizer, [2601](#)
- TreeViewComboBox
  - Digikam::TreeViewComboBox, [3328](#)
- TreeViewLineEditComboBox
  - Digikam::TreeViewLineEditComboBox, [3332](#)
- TtsError
  - Digikam::DOnlineTts, [1181](#)
- TXGA
  - Digikam::VidSlideSettings, [3376](#)
- Type
  - Digikam::Album, [256](#)
  - Digikam::CollectionLocation, [587](#)
  - Digikam::DImgBuiltinFilter, [996](#)
  - Digikam::HistoryImageld, [1735](#)
- type
  - Digikam::Album, [262](#)
  - Digikam::CollectionLocation, [588](#)
  - Digikam::lccProfile, [1765](#)
  - Digikam::ItemComments, [2025](#)
  - Digikam::LoadingTask, [2370](#)
  - Digikam::SavingTask, [2806](#)
- typeForAttribute
  - Digikam::FaceTagsIface, [1498](#)
- TypeMimeFilter
  - Digikam::MimeFilter, [2515](#)
- typeMimes
  - Digikam::DPluginDImg, [1217](#)
- TypePoint
  - Digikam::FocusPoint, [1579](#)
- UHD4K
  - Digikam::VidSlideSettings, [3376](#)
- UHD5K
  - Digikam::VidSlideSettings, [3376](#)
- UHD6K
  - Digikam::VidSlideSettings, [3376](#)
- UHD8K
  - Digikam::VidSlideSettings, [3376](#)
- uiConfidenceThreshold
  - Digikam::DNNBaseDetectorModel, [1118](#)
- Unavailable
  - Digikam::BdEngineBackend, [475](#)
- unclipColors
  - Digikam::DRawDecoderSettings, [1262](#)
- unconfirmedEntry
  - Digikam::FaceTagsEditor, [1494](#)
- unconfirmedFaceCount
  - Digikam::ItemInfo, [2122](#)
- unconfirmedFaceTagsIfaces
  - Digikam::FaceTagsEditor, [1495](#)
- unconfirmedNameFaceTagsIfaces



- Digikam::FaceTagsEditor, [1495](#)
- Undefined
  - Digikam::CollectionLocation, [588](#)
- UniqueBehavior
  - Digikam::ItemComments, [2023](#)
- uniqueHash
  - Digikam::ItemInfo, [2123](#)
- UniquePerLanguage
  - Digikam::ItemComments, [2023](#)
- UniquePerLanguageAndAuthor
  - Digikam::ItemComments, [2023](#)
- UnknownCaseSensitivity
  - Digikam::CollectionLocation, [587](#)
- Unselected
  - Digikam::TimeLineWidget, [3293](#)
- UnspecifiedOps
  - Digikam, [136](#)
- UnsupportedEmotion
  - Digikam::DOnlineTts, [1182](#)
- UnsupportedEngine
  - Digikam::DOnlineTts, [1182](#)
- UnsupportedLanguage
  - Digikam::DOnlineTts, [1182](#)
- UnsupportedVoice
  - Digikam::DOnlineTts, [1182](#)
- updateActionAvailability
  - Digikam::BackendGoogleMaps, [440](#)
  - Digikam::BackendMarble, [448](#)
- updateButton
  - Digikam::ActionVersionsOverlay, [230](#)
  - Digikam::FaceRejectionOverlay, [1474](#)
  - Digikam::HoverButtonDelegateOverlay, [1747](#)
  - Digikam::ImportRotateOverlay, [1931](#)
  - Digikam::ItemFullScreenOverlay, [2085](#)
  - Digikam::ItemRotateOverlay, [2208](#)
  - Digikam::ItemSelectionOverlay, [2222](#)
  - Digikam::ShowHideVersionsOverlay, [3003](#)
- updateClusters
  - Digikam::BackendGoogleMaps, [440](#)
  - Digikam::BackendMarble, [448](#)
  - Digikam::MapWidget, [2433](#)
  - Digikam::TileGroupier, [3283](#)
- updateContentWidth
  - Digikam::ImportDelegate, [1871](#)
  - Digikam::ImportThumbnailDelegate, [1956](#)
  - Digikam::ItemDelegate, [2041](#)
  - Digikam::ItemThumbnailDelegate, [2247](#)
  - ShowFoto::ShowfotoDelegate, [3445](#)
  - ShowFoto::ShowfotoThumbnailDelegate, [3530](#)
- updateData
  - Digikam::CurvesWidget, [711](#)
- UpdateDecorationRole
  - Digikam::SetupCollectionModel, [2960](#)
- updateFace
  - Digikam::AssignNameOverlay, [394](#)
- updateItem
  - Digikam::CoreDB, [678](#)
- updateItemWidgets
  - Digikam::DItemsListViewItem, [1064](#)
  - Digikam::DWItemDelegate, [1310](#)
  - Digikam::SetupCollectionDelegate, [2956](#)
- updateMarkers
  - Digikam::BackendGoogleMaps, [440](#)
  - Digikam::BackendMarble, [448](#)
- updatePALbumIcon
  - Digikam::AlbumManager, [301](#)
- updateRects
  - Digikam::DigikamItemDelegate, [967](#)
  - Digikam::ImportDelegate, [1871](#)
  - Digikam::ImportNormalDelegate, [1917](#)
  - Digikam::ImportThumbnailDelegate, [1956](#)
  - Digikam::ItemDelegate, [2041](#)
  - Digikam::ItemFaceDelegate, [2060](#)
  - Digikam::ItemThumbnailDelegate, [2247](#)
  - ShowFoto::ShowfotoDelegate, [3445](#)
  - ShowFoto::ShowfotoNormalDelegate, [3491](#)
  - ShowFoto::ShowfotoThumbnailDelegate, [3530](#)
- updateSAlbum
  - Digikam::AlbumManager, [301](#)
- updateSearch
  - Digikam::CoreDB, [678](#)
- updateSettings
  - Digikam::DConfigDlgMngr, [857](#)
- updateSizeRectsAndPixmaps
  - Digikam::ImportDelegate, [1871](#)
  - Digikam::ItemDelegate, [2041](#)
  - ShowFoto::ShowfotoDelegate, [3445](#)
- updateTagShortcut
  - Digikam::TagsActionMngr, [3183](#)
- updateTAlbumIcon
  - Digikam::AlbumManager, [302](#)
- updateText
  - Digikam::AlbumSelectComboBox, [329](#)
- updateThumbnailSize
  - Digikam::TableViewColumn, [3073](#)
  - Digikam::TableViewColumns::ColumnThumbnail, [3106](#)
- updateToolTip
  - Digikam::FaceRejectionOverlayButton, [1478](#)
  - Digikam::ImportRotateOverlayButton, [1935](#)
  - Digikam::ItemFullScreenOverlayButton, [2089](#)
  - Digikam::ItemRotateOverlayButton, [2212](#)
  - Digikam::ItemSelectionOverlayButton, [2225](#)
  - Digikam::ItemViewHoverButton, [2269](#)
- updateUniqueHash
  - Digikam::ScanController, [2812](#)
- updateWidgets
  - Digikam::DConfigDlgMngr, [857](#)
- updateWidgetsDefault
  - Digikam::DConfigDlgMngr, [857](#)
- uploadItem
  - Digikam::GPCamera, [1634](#)
  - Digikam::UMSCamera, [3342](#)
- uploadUrl
  - Digikam::DBInfoIface, [797](#)
  - Digikam::DMetaInfoIface, [1101](#)

- uploadWidget
  - Digikam::DBInfoInterface, [798](#)
  - Digikam::DInfoInterface, [1033](#)
  - Digikam::DMetaInfoInterface, [1101](#)
- UseEmbeddedProfile
  - Digikam::ICCSettingsContainer, [1781](#)
- userLoadingHint
  - Digikam::DImgPreviewItem, [1014](#)
- usesBusyIndicator
  - Digikam::ProgressItem, [2670](#)
- UW10K
  - Digikam::VidSlideSettings, [3376](#)
- UW16K
  - Digikam::VidSlideSettings, [3376](#)
- UWFHD
  - Digikam::VidSlideSettings, [3376](#)
- UXGA
  - Digikam::VidSlideSettings, [3376](#)
- value
  - Digikam::SearchXmlReader, [2939](#)
  - Digikam::TagProperties, [3173](#)
- valueChanged
  - Digikam::DPointSelect, [1235](#)
- valueString
  - Digikam::DDoubleSliderSpinBox, [911](#)
  - Digikam::DSliderSpinBox, [1281](#)
- valueToString
  - Digikam::DMetadata, [1094](#)
- valueWidgetRects
  - Digikam::SearchFieldAlbum, [2829](#)
  - Digikam::SearchFieldCheckBox, [2833](#)
  - Digikam::SearchFieldChoice, [2838](#)
  - Digikam::SearchFieldComboBox, [2844](#)
  - Digikam::SearchFieldLabels, [2853](#)
  - Digikam::SearchFieldMonthDay, [2857](#)
  - Digikam::SearchFieldRangeDate, [2865](#)
  - Digikam::SearchFieldRangeDouble, [2869](#)
  - Digikam::SearchFieldRangeInt, [2873](#)
  - Digikam::SearchFieldRangeTime, [2876](#)
  - Digikam::SearchFieldRating, [2880](#)
  - Digikam::SearchFieldText, [2884](#)
- VBR04
  - Digikam::VidSlideSettings, [3374](#)
- VBR05
  - Digikam::VidSlideSettings, [3374](#)
- VBR10
  - Digikam::VidSlideSettings, [3374](#)
- VBR12
  - Digikam::VidSlideSettings, [3374](#)
- VBR15
  - Digikam::VidSlideSettings, [3374](#)
- VBR20
  - Digikam::VidSlideSettings, [3374](#)
- VBR25
  - Digikam::VidSlideSettings, [3374](#)
- VBR30
  - Digikam::VidSlideSettings, [3374](#)
- VBR40
  - Digikam::VidSlideSettings, [3374](#)
- VBR45
  - Digikam::VidSlideSettings, [3374](#)
- VBR50
  - Digikam::VidSlideSettings, [3374](#)
- VBR60
  - Digikam::VidSlideSettings, [3374](#)
- VBR80
  - Digikam::VidSlideSettings, [3374](#)
- VCD1
  - Digikam::VidSlideSettings, [3375](#)
- VCD2
  - Digikam::VidSlideSettings, [3375](#)
- version
  - Digikam::DPlugin, [1194](#)
  - Digikam::ExifToolParser, [1387](#)
  - Digikam::FilterAction, [1556](#)
- VERSION\_STRING
  - Digikam::ExifToolProcess, [1390](#)
- versionFileName
  - Digikam::DefaultVersionNamingScheme, [919](#)
  - Digikam::VersionNamingScheme, [3361](#)
- VersionFileOperation
  - Digikam::VersionFileOperation, [3356](#)
- versionManager
  - Digikam::ImageWindow, [1841](#)
- vertexCount
  - Digikam::Graph< VertexProperties, EdgeProperties >, [1688](#)
- verticesBreadthFirst
  - Digikam::Graph< VertexProperties, EdgeProperties >, [1688](#)
- verticesDepthFirstSorted
  - Digikam::Graph< VertexProperties, EdgeProperties >, [1688](#)
- verticesDominatedBy
  - Digikam::Graph< VertexProperties, EdgeProperties >, [1688](#)
- verticesDominatedByDepthFirstSorted
  - Digikam::Graph< VertexProperties, EdgeProperties >, [1688](#)
- VGA
  - Digikam::VidSlideSettings, [3375](#)
- VidBitRate
  - Digikam::VidSlideSettings, [3374](#)
- VidCodec
  - Digikam::VidSlideSettings, [3374](#)
- VIDEOCOLORMODEL
  - Digikam::DMetadata, [1090](#)
- VideoMergeBackend
  - Digikam::MetaEngine, [2484](#)
- videoStrip16
  - Digikam, [143](#)
- videoStrip4
  - Digikam, [144](#)
- videoStrip8
  - Digikam, [144](#)
- VidFormat

- Digikam::VidSlideSettings, [3374](#)
- VidStd
  - Digikam::VidSlideSettings, [3375](#)
- VidType
  - Digikam::VidSlideSettings, [3375](#)
- view
  - Digikam::DCategoryDrawer, [831](#)
  - Digikam::ListViewComboBox, [2355](#)
  - Digikam::TreeViewComboBox, [3329](#)
- viewportLeaveEvent
  - Digikam::AbstractWidgetDelegateOverlay, [209](#)
  - Digikam::AssignNameOverlay, [394](#)
  - Digikam::PersistentWidgetDelegateOverlay, [2634](#)
- viewPosition
  - Digikam::DConfigDlgView, [870](#)
- visualChange
  - Digikam::AssignNameOverlay, [394](#)
  - Digikam::GroupIndicatorOverlay, [1711](#)
  - Digikam::HoverButtonDelegateOverlay, [1747](#)
  - Digikam::ImportCoordinatesOverlay, [1864](#)
  - Digikam::ImportDownloadOverlay, [1875](#)
  - Digikam::ImportLockOverlay, [1912](#)
  - Digikam::ImportRatingOverlay, [1925](#)
  - Digikam::ItemCoordinatesOverlay, [2029](#)
  - Digikam::ItemDelegateOverlay, [2043](#)
  - Digikam::ItemRatingOverlay, [2203](#)
  - Digikam::TagsLineEditOverlay, [3201](#)
  - ShowFoto::ShowfotoCoordinatesOverlay, [3438](#)
- Voice
  - Digikam::DOnlineTts, [1182](#)
- voice
  - Digikam::DOnlineTts, [1184](#)
- voiceCode
  - Digikam::DOnlineTts, [1184](#)
- VolumeHardWired
  - Digikam::CollectionLocation, [588](#)
- VolumeRemovable
  - Digikam::CollectionLocation, [588](#)
- wait
  - Digikam::DynamicThread, [1321](#)
- waitForExifToolResult
  - Digikam::ExifToolProcess, [1391](#)
- warningContinueCancelList
  - Digikam::CameraMessageBox, [534](#)
- WarningMessage
  - Digikam::DConfigDlgTitle, [862](#)
- wasExifRotated
  - Digikam::DImg, [995](#)
- watchFlags
  - Digikam::ItemFilterSettings, [2078](#)
  - Digikam::ItemSortSettings, [2232](#)
- WEBMVP8
  - Digikam::VidSlideSettings, [3374](#)
- WGS84
  - Digikam::Ellipsoid, [1357](#)
- WhiteBalance
  - Digikam::DRawDecoderSettings, [1261](#)
- whiteBalance
  - Digikam::DRawDecoderSettings, [1262](#)
- WHSXGA
  - Digikam::VidSlideSettings, [3376](#)
- WHUXGA
  - Digikam::VidSlideSettings, [3376](#)
- WHXGA
  - Digikam::VidSlideSettings, [3376](#)
- widgetEnterEvent
  - Digikam::AbstractWidgetDelegateOverlay, [210](#)
  - Digikam::AssignNameOverlay, [394](#)
  - Digikam::FaceRejectionOverlay, [1475](#)
  - Digikam::ImportRatingOverlay, [1926](#)
  - Digikam::ImportRotateOverlay, [1932](#)
  - Digikam::ItemFullScreenOverlay, [2085](#)
  - Digikam::ItemRatingOverlay, [2204](#)
  - Digikam::ItemRotateOverlay, [2208](#)
- widgetLeaveEvent
  - Digikam::AssignNameOverlay, [395](#)
  - Digikam::FaceRejectionOverlay, [1475](#)
  - Digikam::ImportRatingOverlay, [1926](#)
  - Digikam::ImportRotateOverlay, [1932](#)
  - Digikam::ItemFullScreenOverlay, [2086](#)
  - Digikam::ItemRatingOverlay, [2204](#)
  - Digikam::ItemRotateOverlay, [2209](#)
- widgetModified
  - Digikam::DConfigDlgMgr, [857](#)
- WidgetRole
  - Digikam::DConfigDlgModel, [859](#)
- willHaveEffect
  - Digikam::lccTransform, [1783](#)
- willWriteMetadata
  - Digikam::MetadataHub, [2441](#)
- WMV7
  - Digikam::VidSlideSettings, [3374](#)
- WMV8
  - Digikam::VidSlideSettings, [3374](#)
- WMV9
  - Digikam::VidSlideSettings, [3374](#)
- wordWrap
  - Digikam::DNotificationWidget, [1167](#)
- WorkerObject
  - Digikam::WorkerObject, [3396](#)
- WorkerObjectQtMetacall
  - Digikam::ParallelAdapter< A >, [2617](#)
  - Digikam::ParallelWorkers, [2622](#)
- WQHD
  - Digikam::VidSlideSettings, [3376](#)
- WQSXGA
  - Digikam::VidSlideSettings, [3376](#)
- WQUXGA
  - Digikam::VidSlideSettings, [3376](#)
- WQXGA
  - Digikam::VidSlideSettings, [3376](#)
- WQXGAPLUS
  - Digikam::VidSlideSettings, [3376](#)
- writableFormats
  - Digikam::ExifToolParser, [1387](#)
- write

- Digikam::DisjointMetadata, [1047](#)
- Digikam::MetadataHub, [2441](#), [2442](#)
- Digikam::SearchField, [2826](#)
- Digikam::SearchFieldAlbum, [2830](#)
- Digikam::SearchFieldCheckBox, [2834](#)
- Digikam::SearchFieldChoice, [2838](#)
- Digikam::SearchFieldComboBox, [2844](#)
- Digikam::SearchFieldKeyword, [2850](#)
- Digikam::SearchFieldLabels, [2854](#)
- Digikam::SearchFieldMonthDay, [2858](#)
- Digikam::SearchFieldRangeDate, [2865](#)
- Digikam::SearchFieldRangeDouble, [2869](#)
- Digikam::SearchFieldRangeInt, [2873](#)
- Digikam::SearchFieldRangeTime, [2877](#)
- Digikam::SearchFieldRating, [2881](#)
- Digikam::SearchFieldText, [2885](#)
- WRITE\_EXISTING\_TAGS
  - Digikam::ExifToolProcess, [1391](#)
- WRITE\_FORMATS
  - Digikam::ExifToolProcess, [1390](#)
- WRITE\_TO\_FILE\_ONLY
  - Digikam::MetaEngine, [2485](#)
- WRITE\_TO\_SIDECAR\_AND\_FILE
  - Digikam::MetaEngine, [2485](#)
- WRITE\_TO\_SIDECAR\_ONLY
  - Digikam::MetaEngine, [2485](#)
- WRITE\_TO\_SIDECAR\_ONLY\_FOR\_READ\_ONLY\_FILES
  - Digikam::MetaEngine, [2485](#)
- writeField
  - Digikam::SearchXmlWriter, [2942](#)
- writeGroup
  - Digikam::SearchXmlWriter, [2942](#)
- writeMetadata
  - Digikam::FileActionMngrFileWorker, [1523](#)
- writeMetadataToFiles
  - Digikam::FileActionMngrFileWorker, [1523](#)
- WriteMode
  - Digikam::DisjointMetadata, [1044](#)
  - Digikam::FacePipeline, [1423](#)
  - Digikam::FacePipelineBase, [1429](#)
  - Digikam::MetadataHub, [2440](#)
- writeOrientationToFiles
  - Digikam::FileActionMngrFileWorker, [1523](#)
- Writer
  - Digikam::MLPipelineFoundation, [2526](#)
- writer
  - Digikam::FacePipelineDetect, [1433](#)
  - Digikam::FacePipelineDetectRecognize, [1437](#)
  - Digikam::FacePipelineEdit, [1442](#)
  - Digikam::FacePipelineRecognize, [1455](#)
  - Digikam::FacePipelineReset, [1459](#)
  - Digikam::FacePipelineRetrain, [1463](#)
- writeSettings
  - Digikam::DRawDecoderWidget, [1265](#)
- writeTags
  - Digikam::MetadataHub, [2443](#)
- writeToBaloo
  - Digikam::MetadataHub, [2444](#)
- writeToMetadata
  - Digikam::MetadataHub, [2444](#)
- writeUnconfirmedResults
  - Digikam::FaceUtils, [1502](#)
- WritingTagsMode
  - Digikam::ExifToolProcess, [1391](#)
- WSXGA
  - Digikam::VidSlideSettings, [3376](#)
- WSXGAPLUS
  - Digikam::VidSlideSettings, [3376](#)
- WUXGA
  - Digikam::VidSlideSettings, [3376](#)
- WVGA
  - Digikam::VidSlideSettings, [3375](#)
- WXGA1
  - Digikam::VidSlideSettings, [3376](#)
- WXGA2
  - Digikam::VidSlideSettings, [3376](#)
- X264
  - Digikam::VidSlideSettings, [3374](#)
- xml
  - Digikam::SearchXmlWriter, [2943](#)
- XmpHumanList
  - Digikam, [144](#)
- xValue
  - Digikam::DPointSelect, [1235](#)
- XVGA
  - Digikam::VidSlideSettings, [3375](#)
- YOLOv3
  - Digikam::FaceScanSettings, [1479](#)
- YOLOV5NANO
  - Digikam, [136](#)
  - Digikam::AutoTagsScanSettings, [425](#)
- YOLOV5XLARGE
  - Digikam, [136](#)
  - Digikam::AutoTagsScanSettings, [425](#)
- YoloVersions
  - Digikam, [136](#)
- YuNet
  - Digikam::FaceScanSettings, [1479](#)
- yValue
  - Digikam::DPointSelect, [1235](#)
- zoomedSize
  - Digikam::ImageZoomSettings, [1843](#)
- zoomIn
  - Digikam::BackendGoogleMaps, [440](#)
  - Digikam::BackendMarble, [448](#)
- zoomOut
  - Digikam::BackendGoogleMaps, [440](#)
  - Digikam::BackendMarble, [448](#)