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

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	url		

Dependency	Requirement	Qt5 Version	Qt6 Version	External Links	Remarks	Notes
ECM	X	>= 5.55.0	>= 5.240.0	url	Qt6 support implemented in KDE framework >= 5.91.0	
Qt::Core	X	>= 5.14	>= 6.4	url		
Qt::Gui	X	>= 5.14	>= 6.4	url		
Qt::Widgets	X	>= 5.14	>= 6.4	url		
Qt::Network	X	>= 5.14	>= 6.4	url		
Qt::NetworkAuth	X	>= 5.14	>= 6.4	url		
Qt::Sql	X	>= 5.14	>= 6.4	url	Including Qt::Sqlite and Qt::Mysql plugins	
Qt::Xml	X	>= 5.14	>= 6.4	url		
Qt::Concurrent	X	>= 5.14	>= 6.4	url		
Qt::PrintSupport	X	>= 5.14	>= 6.4	url		
Qt::Svg	X	>= 5.14	>= 6.4	url		
Qt::WebEngine	X	>= 5.14	>= 6.4	url	To render web contents (ENABLE_QWEBENGINE=on)	
Qt::XmlPatterns	optional	>= 5.14	—	url	To parse and validate Xml	Used by Rajce plugin. Module removed with Qt6.
Qt::X11Extras	optional	>= 5.14	—	url	For color management support under Linux	Module removed with Qt6.
Qt::DBus	optional	>= 5.14	>= 6.4	url	Optional: only for Linux Desktop	
Qt::OpenGL	optional	>= 5.14	>= 6.4	url	For Presentation tool	
Qt::OpenGLWidgets	optional	—	>= 6.4	url	For Presentation tool	With Qt6, OpenGL is separated in 2 modules: core and widgets.
Qt::Multimedia	optional	—	>= 6.4	url	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	url	To compile test codes (BUILD_↩ TESTING=on)	
Qt::Qml	optional	>= 5.14	>= 6.4	url	To compile test codes (BUILD_↩ TESTING=on) O2 unit tests	
Qt::WebView	optional	>= 5.14	>= 6.4	url	To compile test codes (BUILD_↩ TESTING=on) O2 unit tests	
KF::Config	X	>= 5.95.0	>= 5.240.0	url		
KF::XmlGui	X	>= 5.95.0	>= 5.240.0	url		
KF::I18n	X	>= 5.95.0	>= 5.240.0	url		
KF::↩ Window↩ System	X	>= 5.95.0	>= 5.240.0	url		
KF::Service	X	>= 5.95.0	>= 5.240.0	url	TODO: make optional for Linux desktop (DFile↩ Operations)	
KF::Solid	X	>= 5.95.0	>= 5.240.0	url		
KF::Core↩ Addons	X	>= 5.95.0	>= 5.240.0	url	Needs for KAbout↩ Data and KMemory↩ Info	
KF::Notify↩ Config	optional	>= 5.95.0	>= 5.240.0	url	For Linux desktop application notify configuration	
KF::↩ Notifications	optional	>= 5.95.0	>= 5.240.0	url	For Linux desktop notifications integrations	
KF::↩ Thread↩ Weaver	optional	>= 5.95.0	>= 5.240.0	url	For panorama tool	
KF::Icon↩ Themes	optional	>= 5.95.0	>= 5.240.0	url	Optional: only for Linux Desktop (KIcon↩ Dialog)	
KF::File↩ MetaData	optional	>= 5.95.0	>= 5.240.0	url	Plasma desktop files indexer support	(ENABLE_↩ KFILEMETADATASUPPORT=ON Disabled by default.

Dependency	Requirement	Qt5 Version	Qt6 Version	External Links	Remarks	Notes
KF::↩ Calendar↩ Core	optional	>= 5.95.0	>= 5.240.0	url	For calendar tool to setup ical special events	
KF::KIO	optional	>= 5.95.0	>= 5.240.0	url	Optional: only for Linux Desktop	
KF::Sonnet	optional	>= 5.95.0	>= 5.240.0	url	To perform spell-cheking in text widget (aka caption)	
KF::↩ Akonadi↩ Contact	optional	>= 5.95.0	>= 5.240.0	url	Plasma desktop addess-book support	(ENABLE_↩ AKONADICONTACTSUPPORT= Disabled by default.
libopencv	X	>= 4.8		url	OpenCV 4 recommended	DNN module required for face management
libtiff	X	>= 4.0		url	For DImg TIFF image loader	
libpng	X	>= 1.6		url	For DImg PNG image loader	
libjpeg	X	>= 8		url	jpeglib >= 8.0 required by Raw↩ Engine for DNG supporrt	
libboost	X	>= 1.55.0		url	For Versioning support	
liblcms	X	>= 2.x		url	For Color Management support	
libexpat	X	>= 2.1.0		url	For RAW to DNG converter	
libexiv2	X	>= 0.27.0		url	Metadata low level management.	
libjpegxl	optional	>= 0.7		url	For DNG-Writer and RawEngine	To decode and encode DNG files
libheif	optional	>= 1.6.0		url	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		url	For HEIC encoding support	
libxml2	optional	>= 2.7.0		url	For Html↩ Gallery tool	
libxslt	optional	>= 1.1.0		url	For Html↩ Gallery tool	
Flex	optional	>= 2.5.0		url	For Panorama tool	
Bison	optional	>= 2.5.0		url	For Panorama tool	
libmesa	optional	>= 11.0		url	For Presentation tools (Linux only)	
libksane	optional	>= 21.12.0	22.04.2	url	Digital scanner support	
libjpasper	optional	>= 1.900.1		url	For JPEG-2000 support	
libeigen3	optional	>= 3.2		url	For Refocus tool	See if Clapack from OpenCV can be used instead
liblensfun	optional	>= 0.2.8		url	For Lens↩ Correction tool	
libglib2	optional	>= 2.0.0		url	For Liquid rescale tool	
libgphoto2	optional	>= 2.5		url	Digital camera drivers support. Need libusb-1	
libgomp	optional	>= 5.0		url	OpenMP support for RawEngine	
libimagemagick	optional	>= 6.7.0		url	Image↩ Magick codecs support for DImg image loader	Version >= 7.0 recommended
libffmpeg	optional	>= 5.x		url	To play video and audio (ENABLE_↩ MEDIAPLAYER=on)	libavformat, libavutil, libavcodec used to extract video metadata. QtAVPlayer
libvaapi	optional	>= 2.4		url	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 dabatase 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.aspx

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_FACESENGINE_LOG,             "digikam.facesengine",             QtInfoMsg)
Q_LOGGING_CATEGORY(DIGIKAM_AUTOTAGSENGINE_LOG,          "digikam.autotagsengine",          QtInfoMsg)
Q_LOGGING_CATEGORY(DIGIKAM_GEOIFACE_LOG,                "digikam.geoiface",                QtInfoMsg)
Q_LOGGING_CATEGORY(DIGIKAM_GEOENGINE_LOG,                 "digikam.geocore",                 QtInfoMsg)
Q_LOGGING_CATEGORY(DIGIKAM_DNNMODELMNGR_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.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})
    ENDIF(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 "tagmngrlistitem.h"
```

instead of :

```
#include "models/tagmngrlistitem.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

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;                          // 4 spaces from beginning
....for (int i = 0 ; i < 10 ; ++i)  // 4 spaces from beginning
....{                               // 4 spaces from beginning
........a = i;                      // 4 spaces from previous indent block
```

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

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-col1-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

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

It is also worth to follow most of QTTest 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\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/

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

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 **developper 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")
```

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")
```

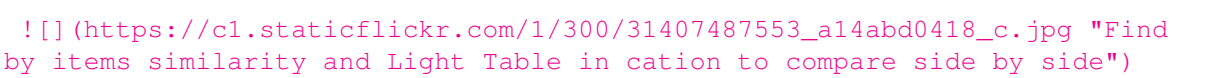
digiKam organizes photos, RAW files, and videos into albums. But the application also features powerful tagging tools that allow you to assign tags, ratings, and labels to photos and raw files. You can then use filtering functionality to quickly find items that match specific criteria.

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

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.

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.

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.



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.

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 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

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

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 **developper 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:

Chapter 6

Hierarchical Index

6.1 Class Hierarchy

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

Chapter 7

Class Index

7.1 Class List

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

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 AkonadiIface
- 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 BqmInfoIface
- 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 CamItemInfo
- class CamItemSortSettings
- 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 DBInfoIface
- 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 DCameraItemListDrag

  Provides a drag object for a list of camera items.
- class DCategorizedSortFilterProxyModel

  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 chosse a color from a palette.
- class DColorValueSelector
- class DComboBox

- class DConfigDlg

  A dialog base class which can handle multiple pages.
- class DConfigDlgMngr

  The DConfigDlgMngr 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 DetectionBenchmarker
- class DetectionWorker
- class DExpanderBox
- class DExpanderBoxExclusive
- class DFileDialog
- class DFileOperations
- class DFileSelector

  A widget to chosse 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 DMetaInfoIface
- 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 DSelectionItem
- 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 GeoIfaceCluster
- class GeoIfaceGlobalObject

  Global object for geolocation interface to hold items common to all geolocation interface Widget instances.
- class GeoIfaceInternalWidgetInfo

  Class to hold information about map widgets stored in the GeoIfaceGlobalObject.
- class GeoIfaceSharedData
- 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 GPSGeoIfaceModelHelper

- 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 HaarIface
- 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 HistoryImageId
- 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 MaintenanceMngr
- 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 ModelIndexBasedComboBox
- 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 SetupEditorIface
- 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 TileGrouper
- 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(* DItemsListIsLessThanHandler) (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< GeoMouseMode > **GeoMouseModes**
- 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 , **Tfldf** = 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 adjustedEnvironmentForAppImage ()

  If digiKam run into AppImage, 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 &video↩StripHeight)
- 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 returned↩ Profiles, QStringList *const userText)

  NOTE: if needed outside this class, make it a public static method in a namespace.
- void **GeoIface_assert** (const char *const condition, const char *const filename, const int lineNumber)
- GeoCoordinates::PairList **GeoIfaceHelperNormalizeBounds** (const GeoCoordinates::Pair &boundsPair)

  Split bounds crossing the dateline into parts which do not cross the dateline.
- bool **GeoIfaceHelperParseBoundsString** (const QString &boundsString, QPair< GeoCoordinates, GeoCoordinates > *const boundsCoordinates)

  Parses a '((lat1, lon1), (lat2, lon2))' bounds string as returned by the JavaScript parts.
- bool GeoIfaceHelperParseLatLonString (const QString &latLonString, GeoCoordinates *const coordinates)

  Parse a 'lat,lon' string a returned by the JavaScript parts.
- bool **GeoIfaceHelperParseXYStringToPoint** (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 requriedOrder)

  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/.
- 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 **isRunningInAppImageBundle** ()

  Return true if application run in AppImage 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 RGInfo &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)

  qCDebug(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)

  qCDebug(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, dbcoreloader,(QStandardPaths↩ ::locate(QStandardPaths::GenericDataLocation, QLatin1String("digikam/database/dbconfig.xml")), dbcoreconfig↩ _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 **removeAnyInInterval** (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)

  Helpher 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 **AUTOEXPANDDELAY** = 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 GeoIfaceMinMarkerGroupingRadius = 1
- const int **GeoIfaceMinThumbnailGroupingRadius** = 15
- const int **GeoIfaceMinThumbnailSize** = GeoIfaceMinThumbnailGroupingRadius ∗ 2
- static const char ∗ IptcHumanList [ ]
- 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::Frameless↩ WindowHint
- 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 λ 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.

References about DNG: DNG SDK tutorial: www.adobeforums.com/webx/.3bc2944e www.adobeforums.↩ com/webx/.3c054bde DNG review: www.barrypearson.co.uk/articles/dng/index.htm DNG intro: www.↩ adobe.com/digitalimag/pdfs/dng_primer.pdf www.adobe.com/products/dng/pdfs/DNG_primer_manufacturers.pdf DNG Specification: wwwimages.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 DNG SDK reference↩ : www.thomasdideriksen.dk/misc/File%20Formats/dng_sdk_refman.pdf DNG SDK tarball: helpx.adobe.↩ com/photoshop/digital-negative.html::dng_sdk_download DNG users forum: www.adobeforums.com/webx/.3bb5f0ec.

Applications using DNG SDK: DNG4PS2: dng4ps2.chat.ru/index_en.html CORNERFIX: sourceforge.↩ net/projects/cornerfix ADOBE DNG CONVERTER: helpx.adobe.com/photoshop/using/adobe-dng-converter.html DNGCONVERT: github.com/jmue/dngconvert MOVIE2DNG: elphel.svn.sourceforge.net/svnroot/elphel/tools/↩ Movie2DNG RAW2DNG : 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::setIsLess↩ ThanHandler().

8.1.3 Enumeration Type Documentation

8.1.3.1 DetectorNNModel

```
enum Digikam::DetectorNNModel
```

Enumerator

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

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	Album 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

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

8.1.4.4 fastNumberToString()

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

QString::number is 3x slower.

8.1.4.5 GeoIfaceHelperParseLatLonString()

```
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 requried 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) if only chapter and reference are empty, section page is open. (as: 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) else reference at chapter from section page is open. (as: 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

a	Point a
b	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 dysfuntion 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

```
                Ext           Descriptions From
                              www.file-extensions.org
                              en.wikipedia.org/wiki/RAW_file_format
                              filext.com
```

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 GeoIfaceMinMarkerGroupingRadius

```
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.

8.1.5.11 videoStrip16

```
const quint8 Digikam::videoStrip16[16 *16 *3] [static]
```

Initial value:

```
=
{
    "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"
    "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"
    "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"
    "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"
    "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"
    "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\11\11\11\14\14\14"
    "\15\15\15\15\15\15\15\15\15\15\15\15\15\15\15\15\15\15\4\4\4\0\0\0\0\0\0"
    "\0\0\0\0\0\0\0\0\0\0\0\0""888YYYrrr|||\200\200\200\200\200\200\200\200\200"
    "\200\200\200zzzmmm\23\23\23\0\0\0\0\0\0\0\0\0\0\0\0\11\11\11YYY\214\214\214"
    "\257\257\257\276\276\276\302\302\302\302\302\302\302\302\302\301\301\301"
    "\273\273\273\250\250\250@@@\0\0\0\0\0\0\0\0\0\0\0\0\14\14\14qqq\257\257\257"
    "\326\326\326\347\347\347\353\353\353\354\354\354\354\354\354\353\353\353"
    "\344\344\344\317\317\317PPP\0\0\0\0\0\0\0\0\0\0\0\0\15\15\15{{{\274\274\274"
    "\345\345\345\365\365\365\372\372\372\373\373\373\373\373\373\371\371\371"
    "\363\363\363\335\335\335VVV\0\0\0\0\0\0\0\0\0\0\0\0\14\14\14xxx\270\270\270"
    "\340\340\340\361\361\361\365\365\365\366\366\366\366\366\366\365\365\365"
    "\356\356\356\331\331\331UUU\0\0\0\0\0\0\0\0\0\0\0\0\1\1\1ggg\240\240\240"
    "\306\306\306\326\326\326\332\332\332\334\334\334\334\334\334\332\332\332"
    "\324\324\324\277\277\277\"\"\"\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\1\1\1\32\32"
    "\32\40\40\40###$$$$$$$$$$$$###\12\12\12\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"
    "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"
    "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"
    "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"
    "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"
    "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"
}
```

8.1.5.12 videoStrip4

```
const quint8 Digikam::videoStrip4[4 *4 *3] [static]
```

Initial value:

```
=
{
    "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0kkk\207\207\207777\0\0\0\237\237\237\303\303"
    "\303RRR\0\0\0\0\0\0\0\0\0\0\0"
}
```

8.1.5.13 videoStrip8

```
const quint8 Digikam::videoStrip8[8 *8 *3] [static]
```

Initial value:

```
=
{
    "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"
    "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\40\40\40""777::::::444\1\1\1\0\0\0\2"
    "\2\2\205\205\205\320\320\320\333\333\333\333\333\333\313\313\313\32\32\32"
    "\0\0\0\2\2\2\236\236\236\360\360\360\373\373\373\373\373\373\353\353\353"
    "\37\37\37\0\0\0\0\0\0FFFsssyyyyyynnn\2\2\2\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"
    "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"
}
```

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 clockwiseto 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' = m11 x + m12 y y' = m21 x + m22 y

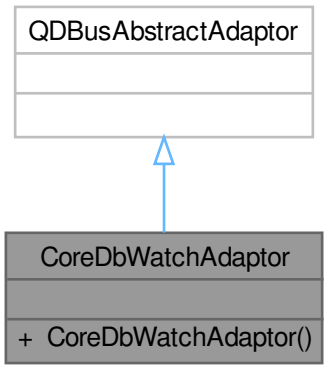
Moreover, all combinations of these rotate/flip operations result in one of the eight matrices defined below. This did not proof that mathematically, but empirically.

Chapter 9

Class Documentation

9.1 CoreDbWatchAdaptor Class Reference

Inheritance diagram for CoreDbWatchAdaptor:



Signals

- void **signalAlbumChangeDBus** (const QString &databaseIdentifier, const QString &applicationIdentifier, const Digikam::AlbumChangeset &changeset)
- void **signalAlbumRootChangeDBus** (const QString &databaseIdentifier, const QString &application↩ Identifier, const Digikam::AlbumRootChangeset &changeset)
- void **signalCollectionImageChangeDBus** (const QString &databaseIdentifier, const QString &application↩ Identifier, const Digikam::CollectionImageChangeset &changeset)
- void **signalImageChangeDBus** (const QString &databaseIdentifier, const QString &applicationIdentifier, const Digikam::ImageChangeset &changeset)
- 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)

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

AlbumTitleRole	Returns the album title. Principally the same as display role, but without any additions.
AlbumTypeRole	Returns the Album::Type of the associated album.
AlbumPointerRole	Returns a pointer to the associated Album object.
AlbumIdRole	Returns the id of the associated Album object.
AlbumGlobalIdRole	Returns the global id (unique across all album types)
AlbumSortRole	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()

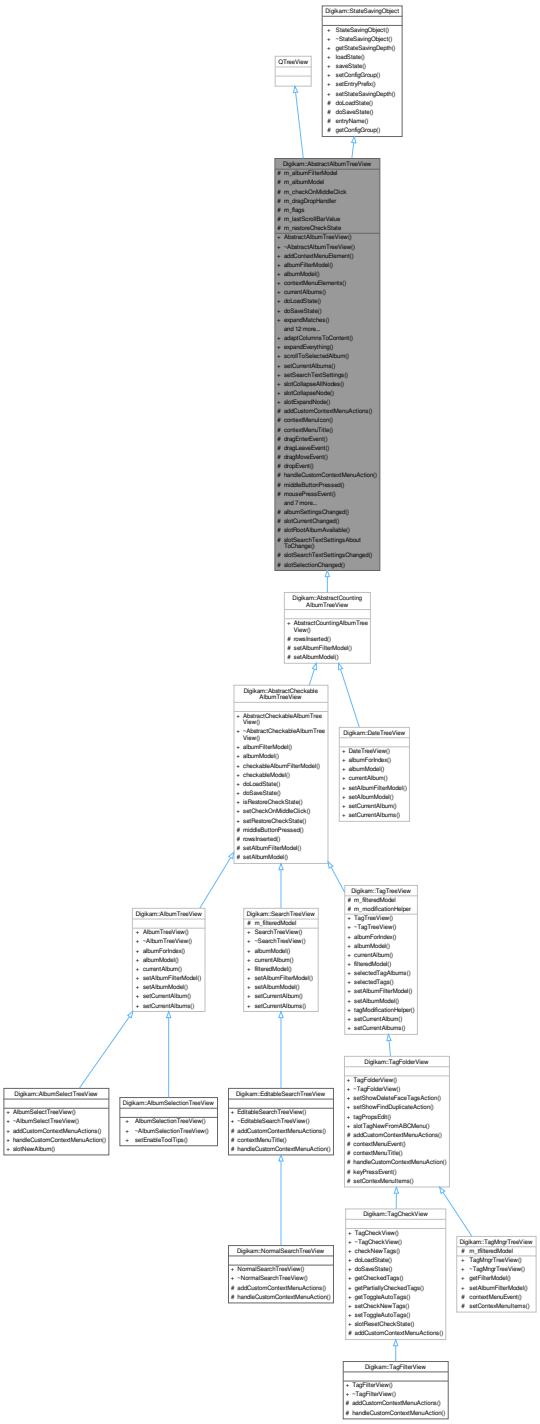
```
QVariant Digikam::AbstractAlbumModel::sortRoleData (
            Album * a ) const  [protected], [virtual]
```

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.

Inheritance diagram for Digikam::AbstractAlbumTreeView:



Classes

- class ContextMenuElement

  Add a context menu element which can add actions to the context menu when the menu is generated.

Public Types

- 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

- 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 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 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 setEnableContextMenu.

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 setShowCount() 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

cmh	helper object to create the context menu
album	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

index	the index to start expanding everything

9.3.4.7 expandMatches()

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

Parameters

index	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

action	the action that was chosen by the user, may be null if none of the custom actions were selected
album	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::EditableSearchTreeVie 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

albums	the albums to select
selectInAlbumManager	the album will be set as current album, if both this parameter is true and setAlbumManagerCurrentAlbum() 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

enable	if true, a context menu can be shown

9.3.4.16 setSelectAlbumOnClick()

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

Parameters

selectOnClick	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

select	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

event	context menu event to react on
albumForEvent	the album at the mouse position or null if there is no album at that position

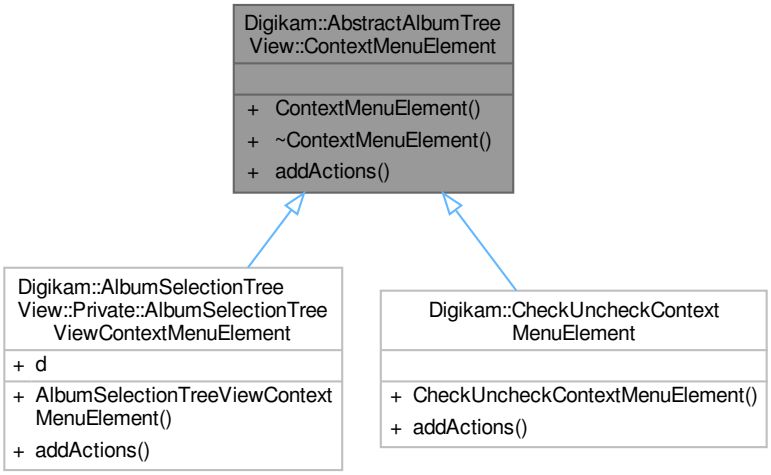
Returns

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

view	The AbstractAlbumTreeView which generates the menu
cmh	helper object to create the context menu
album	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 **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.
- 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.

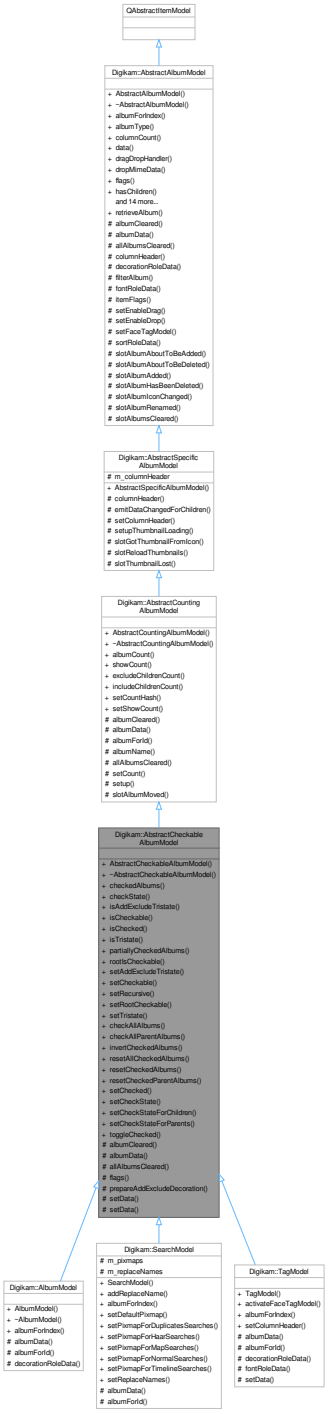
9.5.2.4 setTreeView()

```
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.

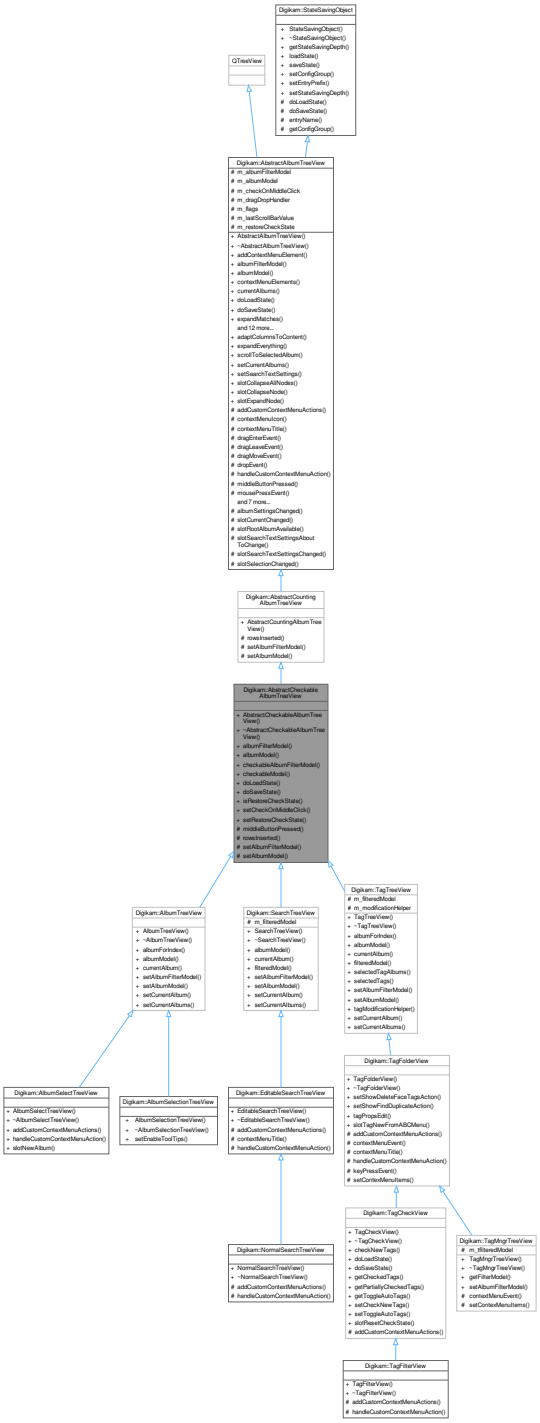
9.6.2.7 setTristate()

```
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 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 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 | 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::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()

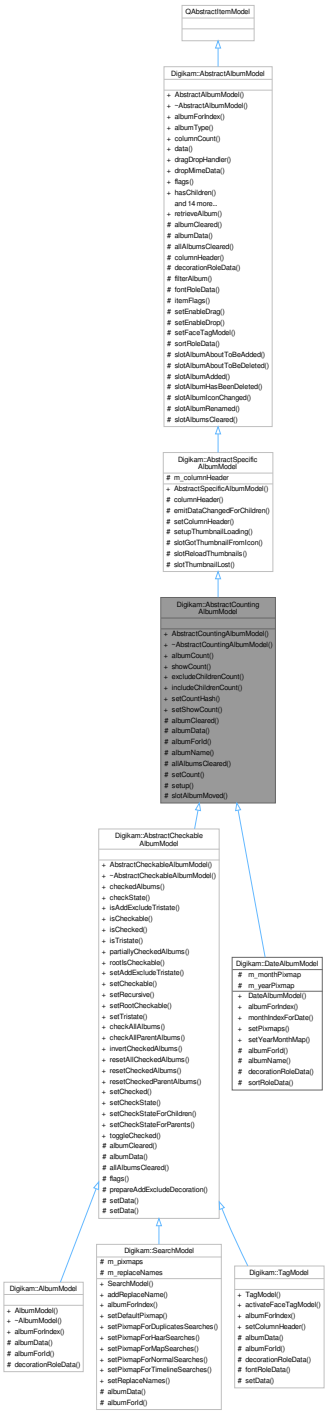
```
void Digikam::AbstractCheckableAlbumTreeView::setRestoreCheckState (
            bool restore )
```

Parameters

restore	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 root↩ Behavior=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.

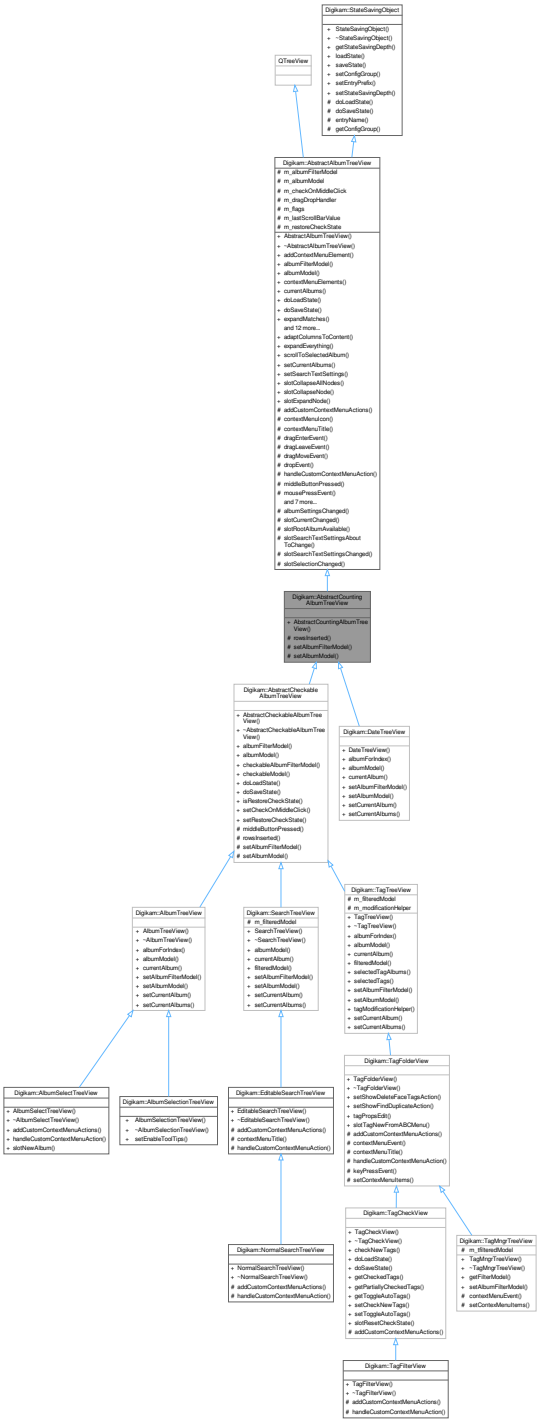
9.8.1.9 setCountHash

```
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 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 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 | 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::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** ()

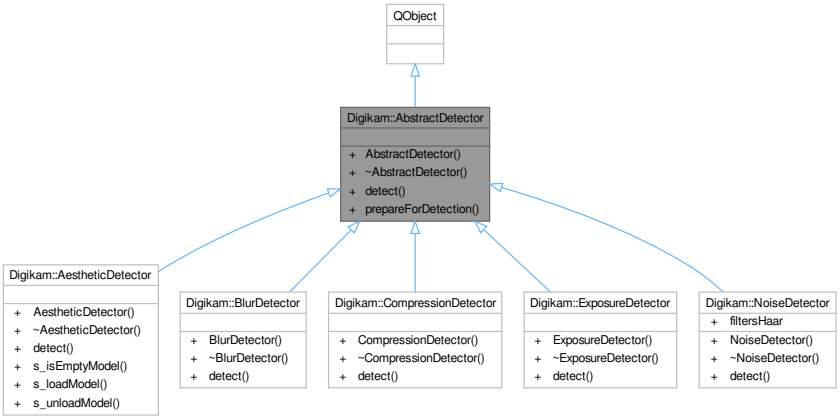
  override if implemented behavior is not as intended
- void **slotSearchTextSettingsAboutToChange** (bool searched, bool willSearch)
- void **slotSearchTextSettingsChanged** (bool wasSearching, bool searching)
- void **slotSelectionChanged** ()

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()

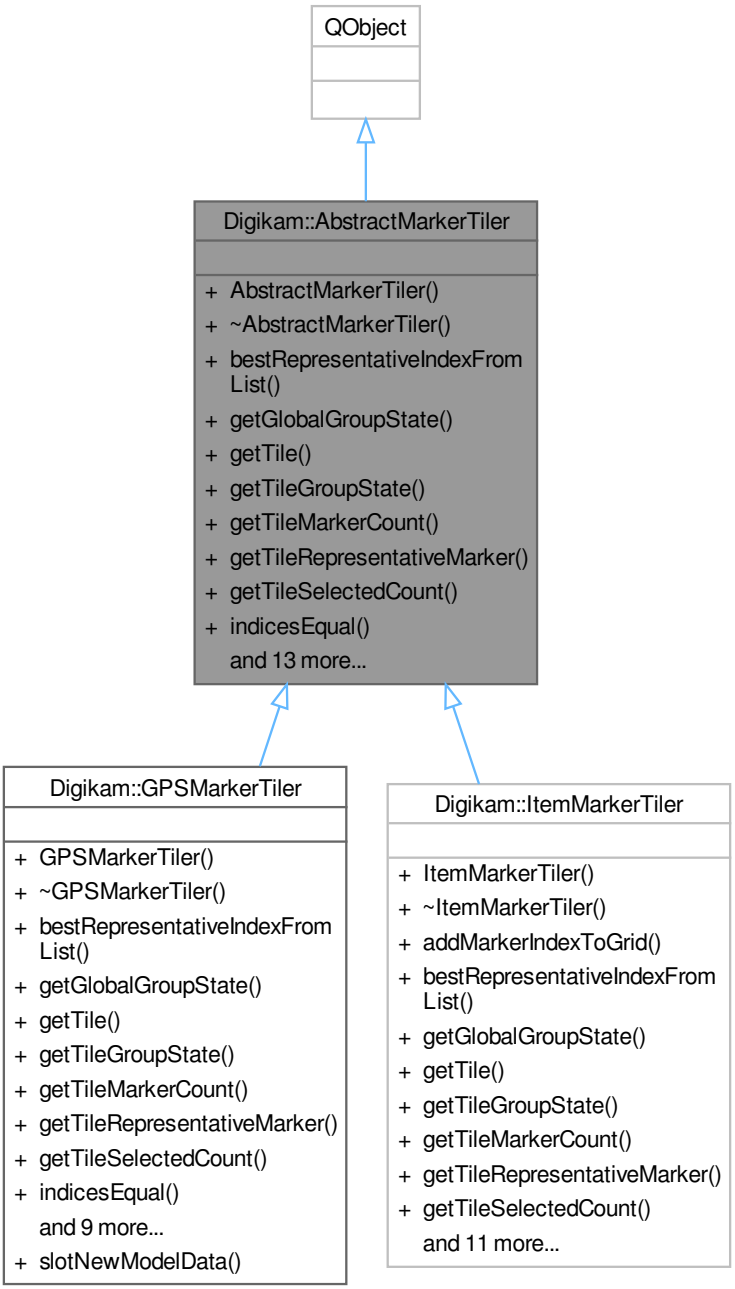
```
QStringList Digikam::AbstractItemDragDropHandler::mimeTypes ( ) const  [virtual]
```

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 QIntList &a, const QIntList &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 **groupSelectionState**
- QVariant **representativeIndex**
- TileIndex::List **tileIndicesList**

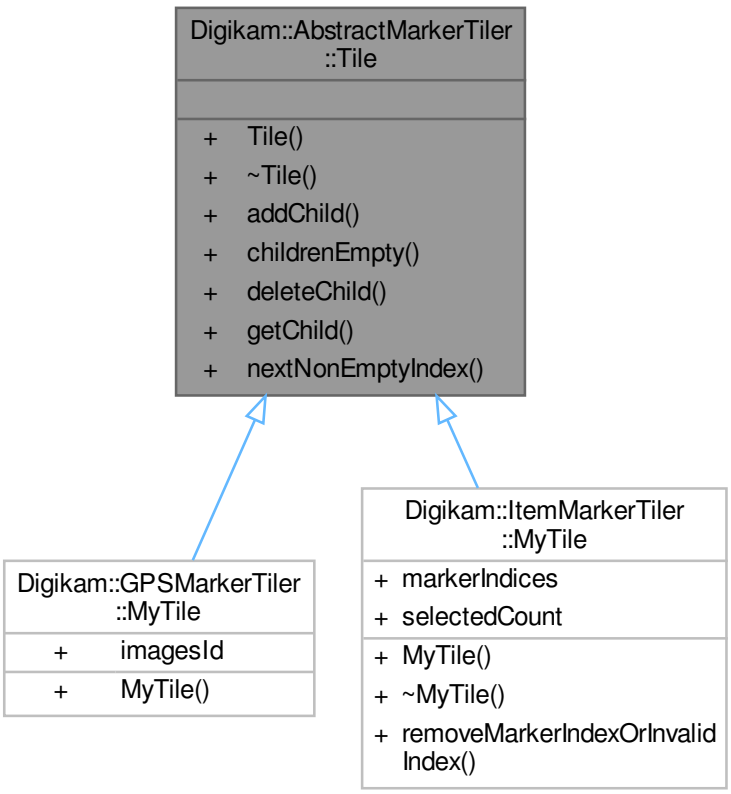
9.14 Digikam::AbstractMarkerTiler::NonEmptyIterator Class Reference

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

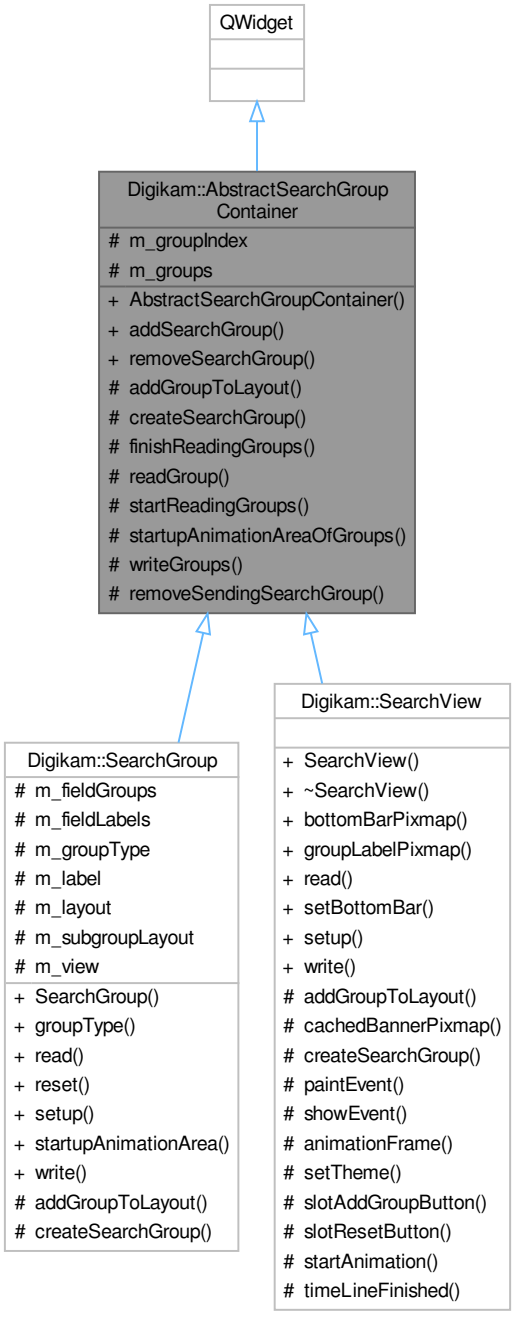
- Tile ∗ **addChild** (const int linearIndex, Tile ∗tilePointer)
- bool **childrenEmpty** () const
- void **deleteChild** (Tile ∗const childTile, const int knownLinearIndex=-1)

  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.

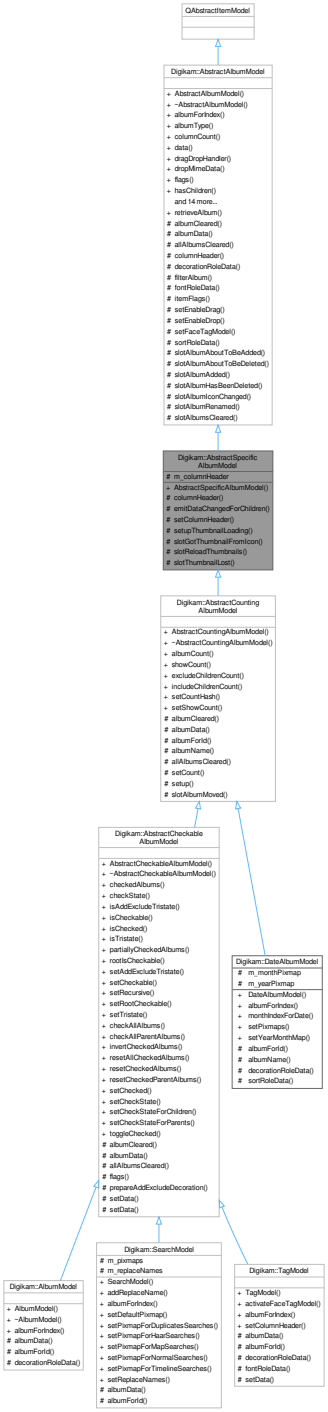
9.16.1.2 createSearchGroup()

```
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.

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.

9.17.1 Member Function Documentation

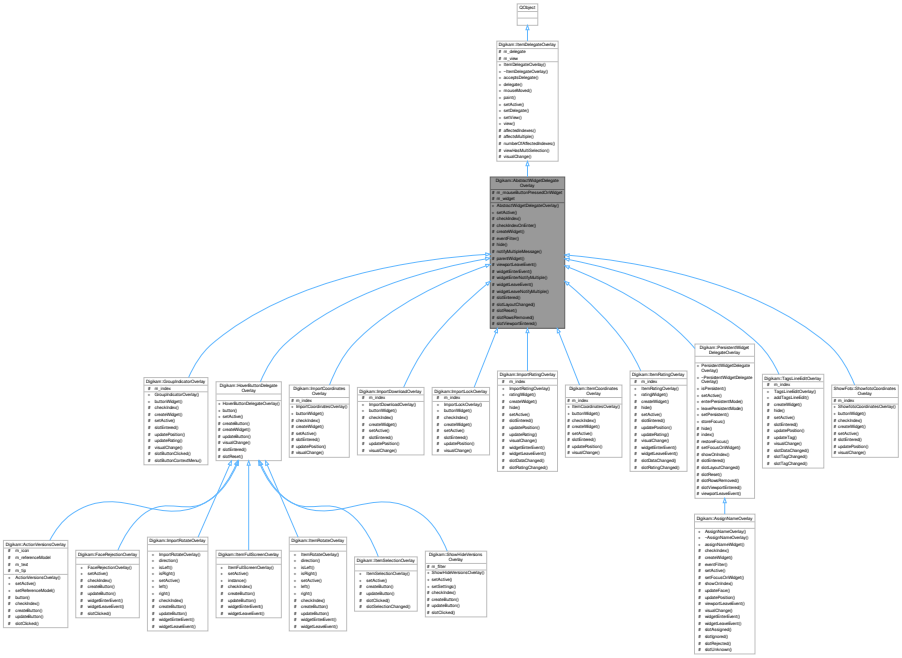
9.17.1.1 columnHeader()

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

Reimplemented from Digikam::AbstractAlbumModel.

9.18 Digikam::AbstractWidgetDelegateOverlay Class Reference

Inheritance diagram for Digikam::AbstractWidgetDelegateOverlay:



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 overlayed 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::ShowHideVersionsOverla 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::ShowfotoCoordinates 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 **rowsInsertedArtifficial** (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

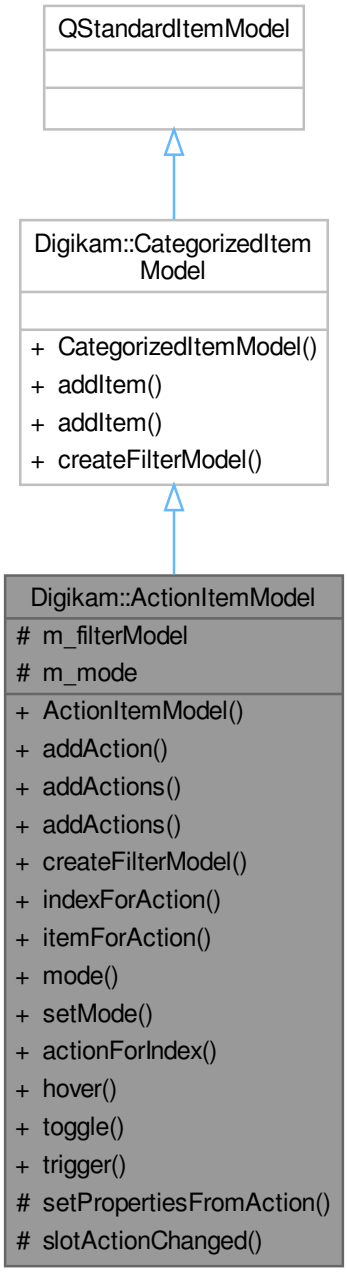
- enum **ActionStatus** {
  None = 0 , **BatchStarted** , **BatchDone** , **BatchFailed** ,
  BatchSkipped , **BatchCanceled** , **TaskDone** , **TaskFailed** ,
  TaskCanceled }

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.

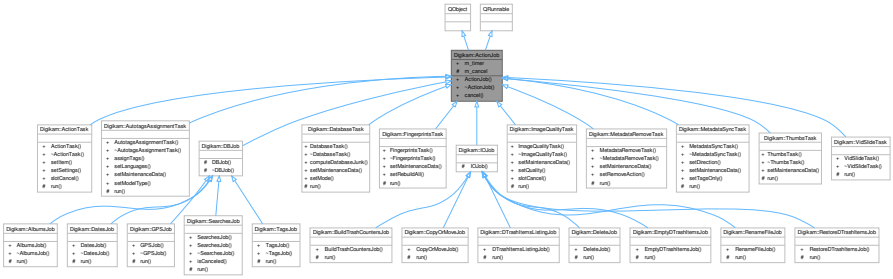
9.21.3.4 itemForAction()

```
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

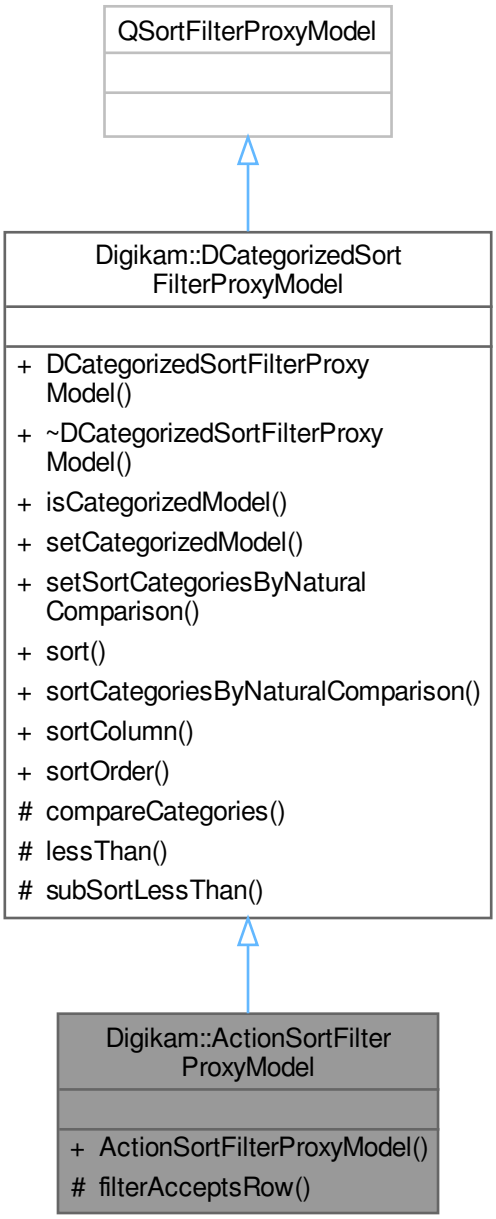
9.22.1.1 ~ActionJob()

```
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.

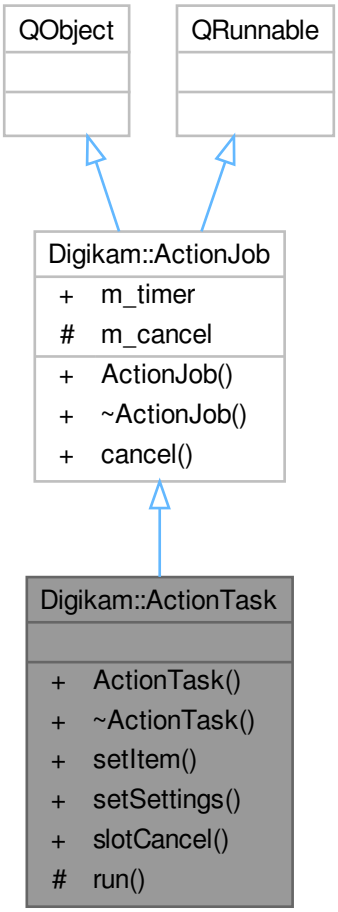
Additional Inherited Members

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.

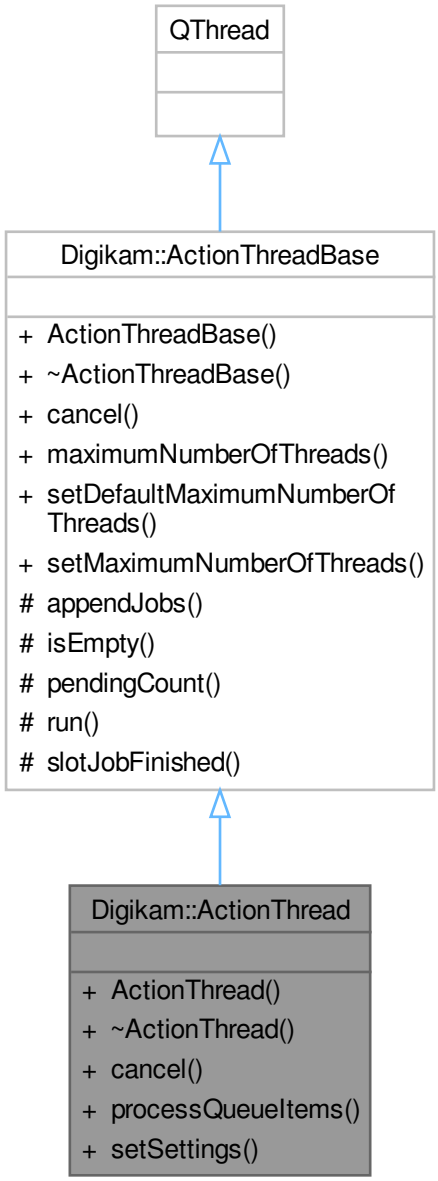
Protected Attributes inherited from Digikam::ActionJob

- 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

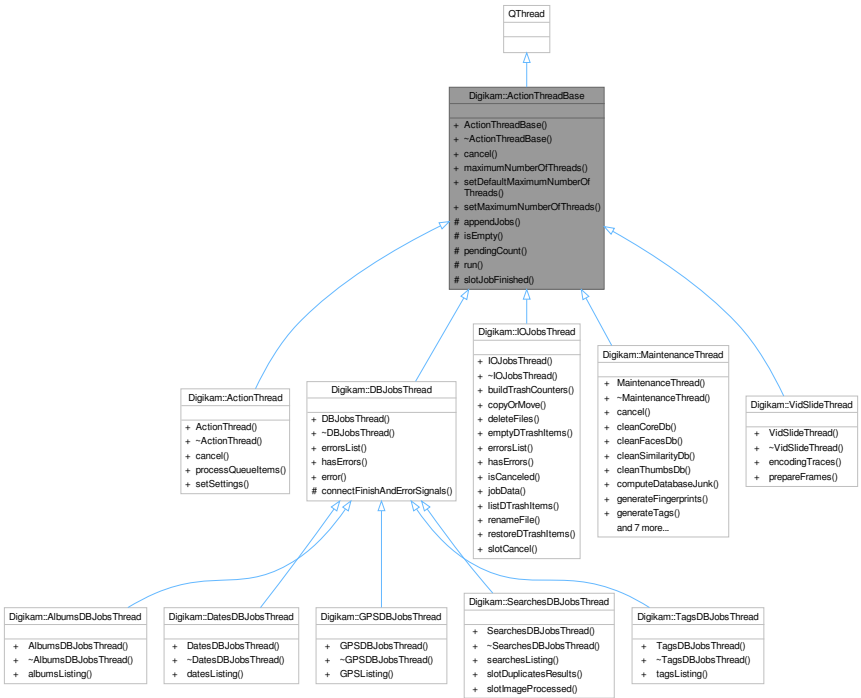
  Return true if list of pending jobs to process is empty.
- int **pendingCount** () const

  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.

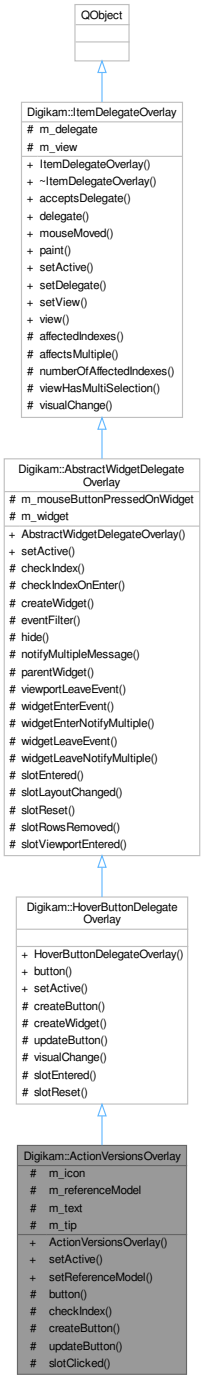
9.26.1.2 setDefaultMaximumNumberOfThreads()

```
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 overlayed 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()

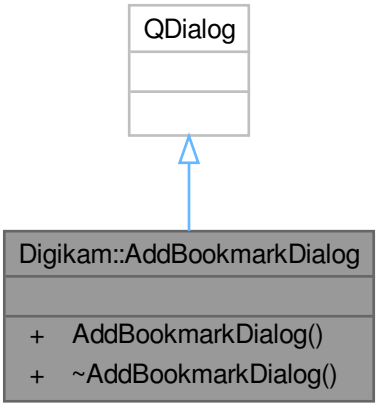
```
void Digikam::ActionVersionsOverlay::updateButton (
            const QModelIndex & index )  [override], [protected], [virtual]
```

Reposition your button here, adjust and store state.

Implements Digikam::HoverButtonDelegateOverlay.

9.28 Digikam::AddBookmarkDialog Class Reference

Inheritance diagram for Digikam::AddBookmarkDialog:



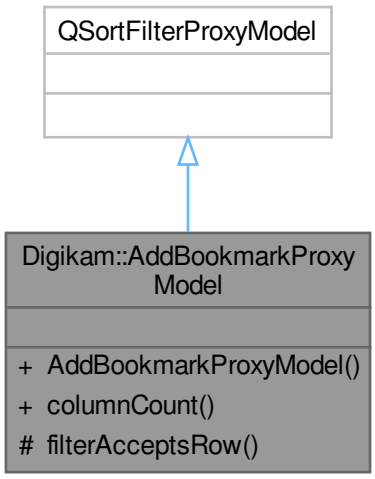
Public Member Functions

- **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 **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 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

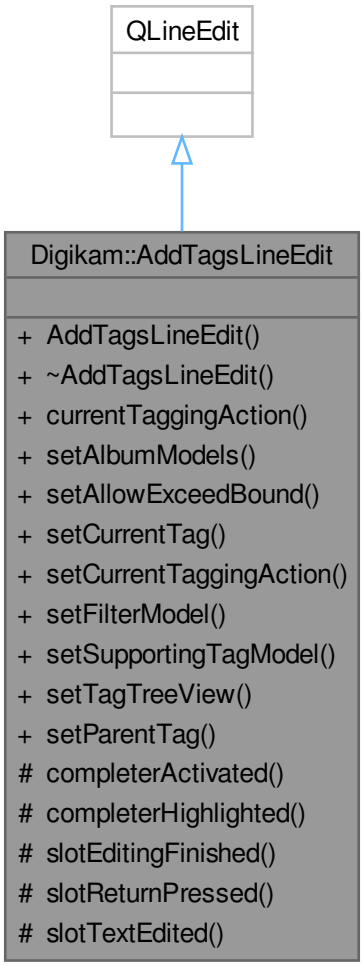
9.30.1.3 taggingActionSelected

```
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.

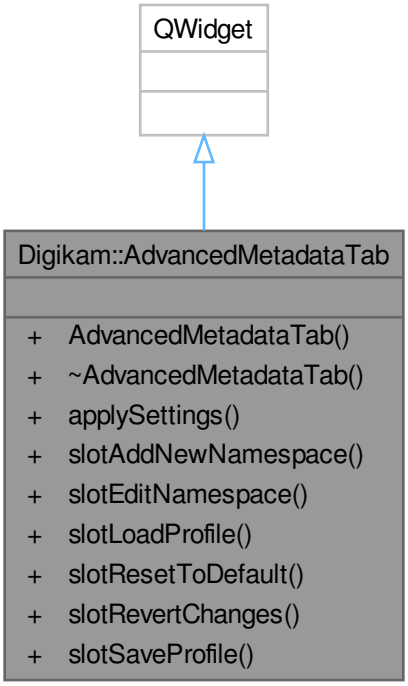
9.31.1.3 taggingActionSelected

```
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

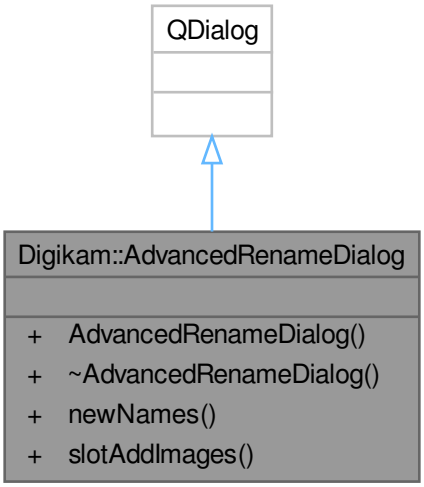
9.32.1.1 AdvancedMetadataTab()

```
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

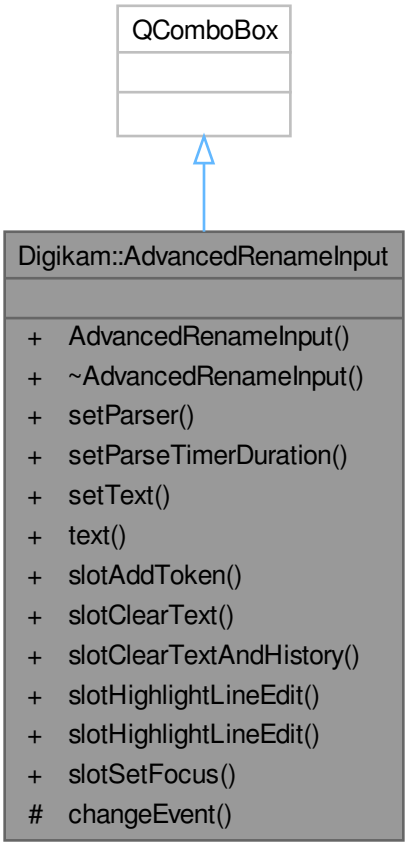
- void **slotAddImages** (const QList< QUrl > &urls)

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

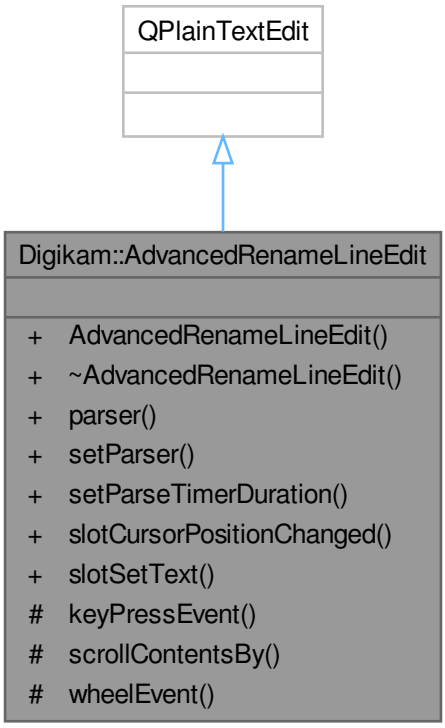
- **AdvancedRenameInput** (QWidget ∗const parent=nullptr)
- void **setParser** (Parser ∗parser)
- void **setParseTimerDuration** (int milliseconds)
- void **setText** (const QString &text)
- QString **text** () const

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

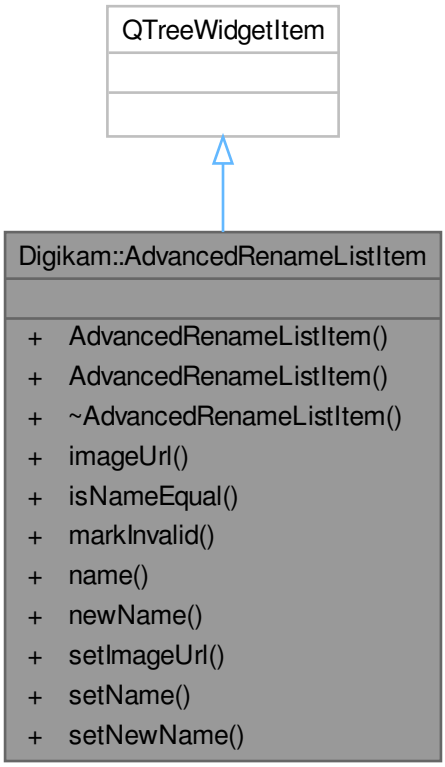
- **AdvancedRenameLineEdit** (QWidget ∗const parent=nullptr)
- Parser ∗ **parser** () const
- void **setParser** (Parser ∗parser)
- void **setParseTimerDuration** (int milliseconds)

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

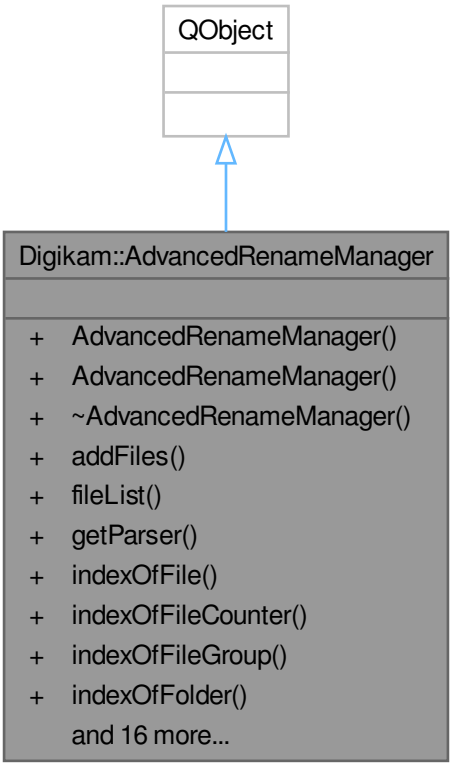
- enum **Column** { **OldName** = 0 , **NewName** }

Public Member Functions

- **AdvancedRenameListItem** (QTreeWidget ∗const view)
- **AdvancedRenameListItem** (QTreeWidget ∗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

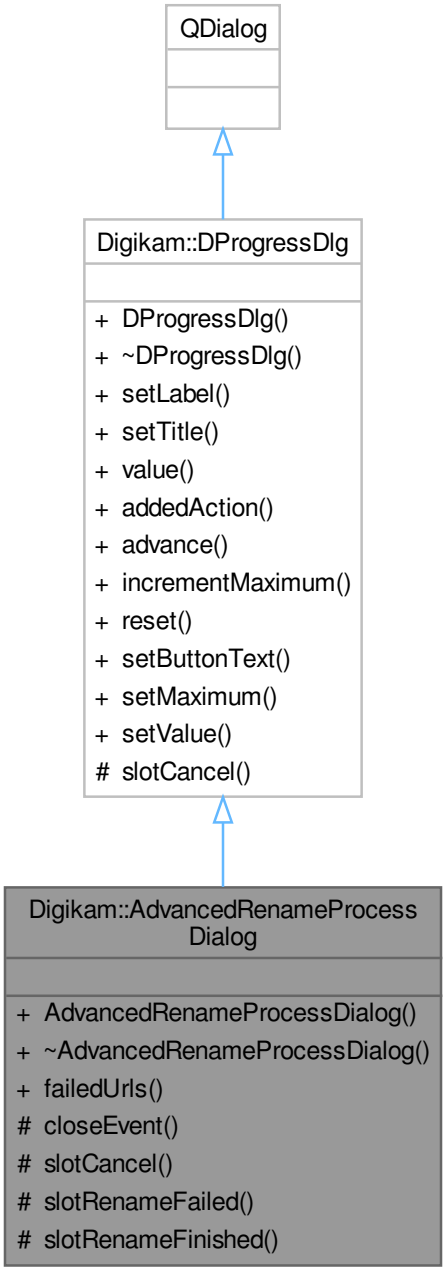
- void **signalSortingChanged** (QList< QUrl >)

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

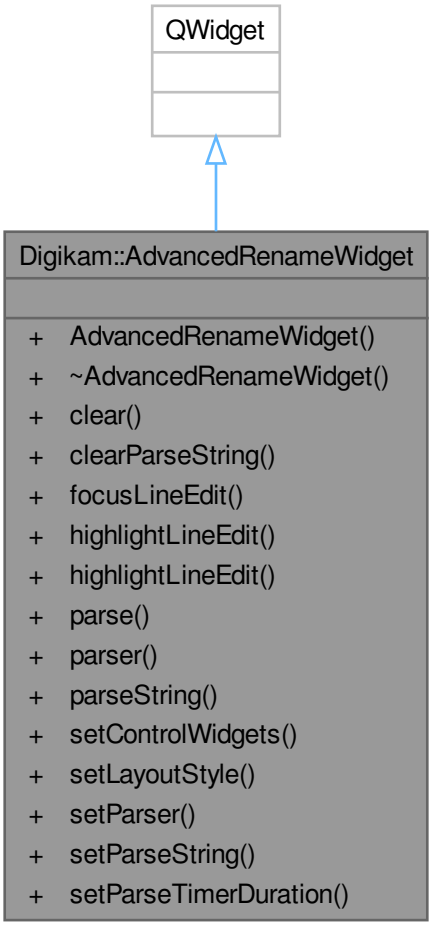
- void **addedAction** (const QPixmap &icon, const QString &text)
- void **advance** (int offset)
- void **incrementMaximum** (int added)
- void **reset** ()
- void **setButtonText** (const QString &text)
- void **setMaximum** (int max)
- void **setValue** (int value)

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

settings	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

mask	a bitmask for setting the control widgets

9.39.1.3 setLayoutStyle()

```
void Digikam::AdvancedRenameWidget::setLayoutStyle (
            LayoutStyle style )
```

Parameters

style	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

parser	a pointer to the new parser instance

9.39.1.5 setParseString()

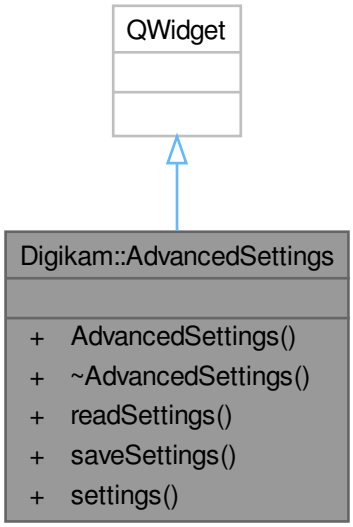
```
void Digikam::AdvancedRenameWidget::setParseString (
            const QString & text )
```

Parameters

text	the new parse string

9.40 Digikam::AdvancedSettings Class Reference

Inheritance diagram for Digikam::AdvancedSettings:



Signals

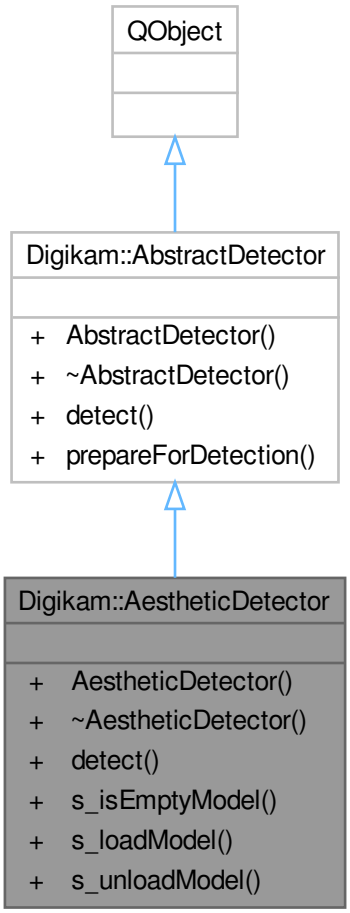
- void **signalDownloadNameChanged** ()

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

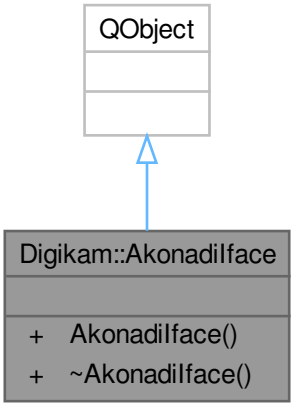
9.41.1.1 detect()

```
float Digikam::AestheticDetector::detect (
            const cv::Mat & image ) const  [override], [virtual]
```

Implements Digikam::AbstractDetector.

9.42 Digikam::AkonadiIface Class Reference

Inheritance diagram for Digikam::AkonadiIface:



Signals

- void **signalContactTriggered** (const QString &)

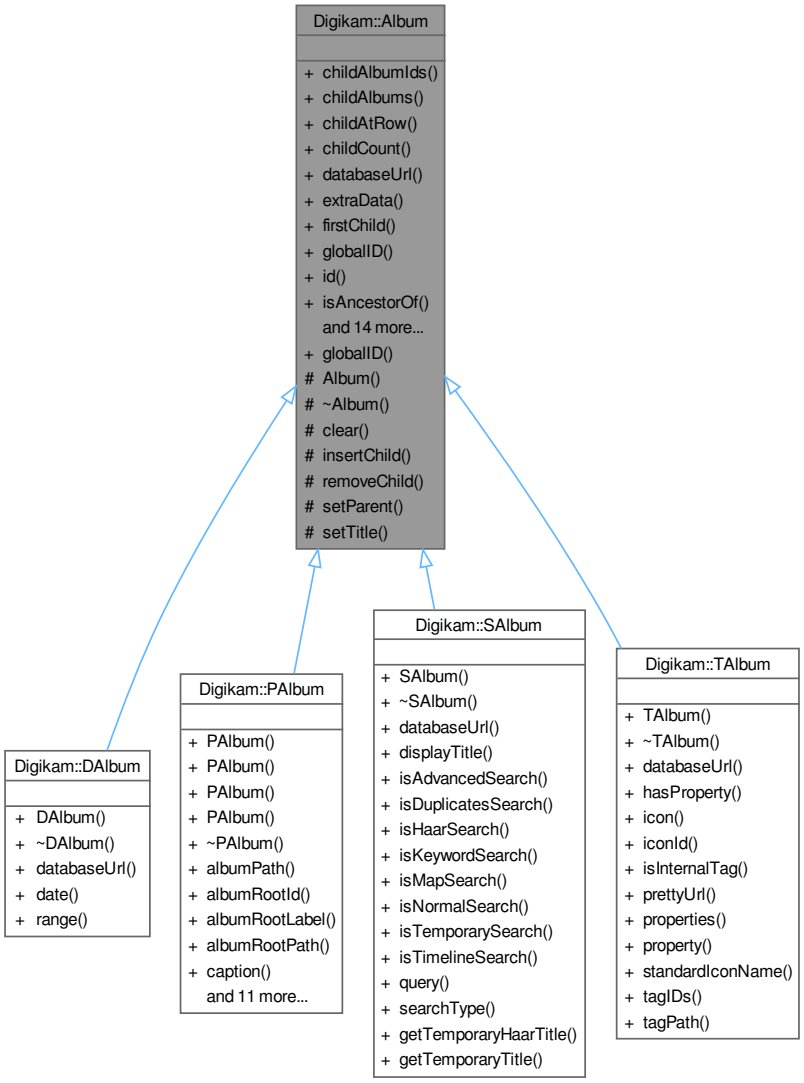
Public Member Functions

- **AkonadiIface** (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** PAlbum
TAG	A tag album type. **See also** TAlbum
DATE	A date album type. **See also** DAlbum
SEARCH	A search album type. **See also** SAlbum
FACE	A faces album type. **See also** FAlbum

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

key	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

type	The type of the album
id	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

album	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

key	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

key	the key of the extra data
value	the value of the extra data

See also

extraData

removeExtraData

9.43.4.22 setUsedByLabelsTree()

```
void Digikam::Album::setUsedByLabelsTree (
            bool isUsed )
```

Parameters

isUsed	=> 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

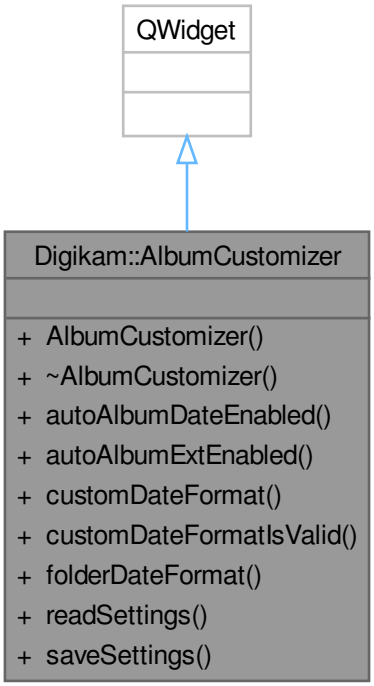
  An AlbumCopyMoveHint describes an existing album and a destination to which this album is expected to be copied, moved or renamed.
- **AlbumCopyMoveHint** (int srcAlbumRootId, int srcAlbum, int dstAlbumRootId, const QString &dstRelative↩ Path)
- int **albumIdSrc** () const
- int **albumRootIdDst** () 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

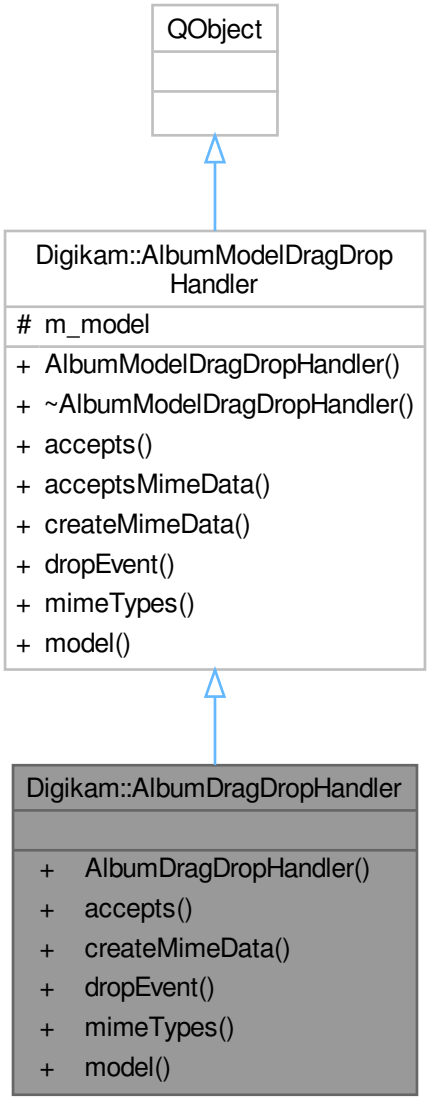
- enum **DateFormatOptions** { **IsoDateFormat** = 0 , **TextDateFormat** , **LocalDateFormat** , **CustomDate↩ Format** }

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 mimeTypes()

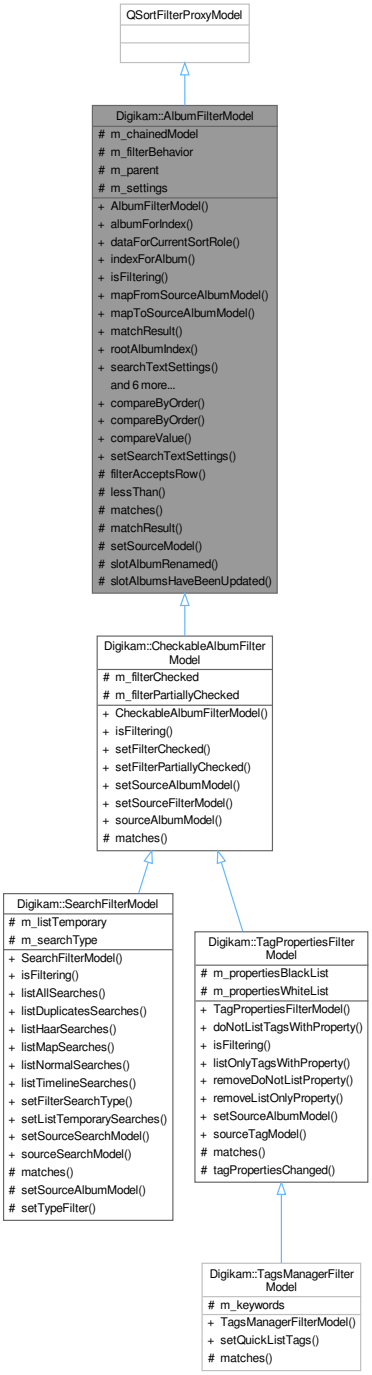
```
QStringList Digikam::AlbumDragDropHandler::mimeTypes ( ) const  [override], [virtual]
```

Called by the default implementation of model's mimeTypes().

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 matched, 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

hasResult	`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

album	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

searched	`true` if filtering by text was enabled before applying the new settings
willSearch	`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

wasSearching	`true` if this is not a new search that
searched	`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

settings	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

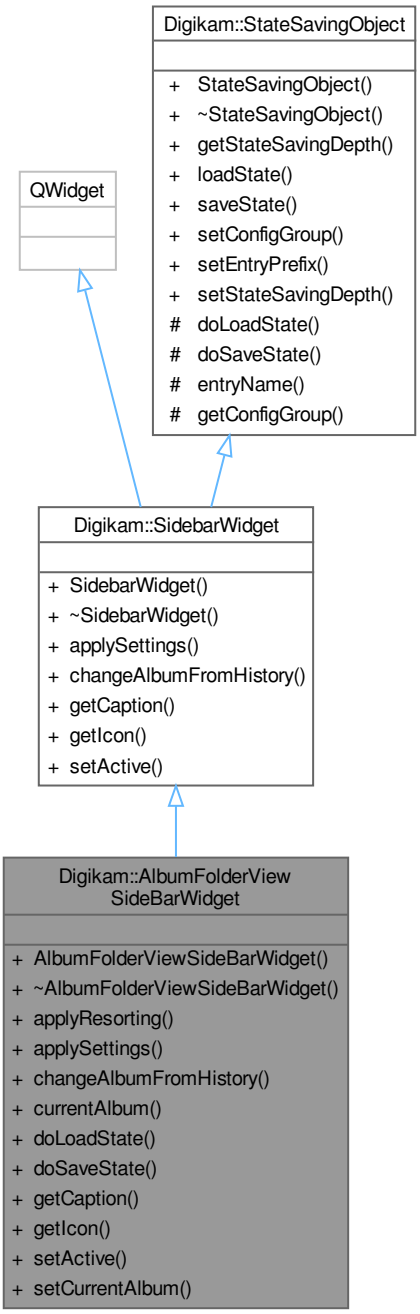
setSourceAlbumModel

Parameters

model	source model

9.49 Digikam::AlbumFolderViewSideBarWidget Class Reference

Inheritance diagram for Digikam::AlbumFolderViewSideBarWidget:



Public Slots

- void **setCurrentAlbum** (PAlbum *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

active	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 **slotClearSelectPAlbum** (const ItemInfo &imageInfo)
- void **slotClearSelectTAlbum** (int id)
- void **slotCurrentChange** (const ItemInfo &info)
- void **slotImageSelected** (const ItemInfoList &selectedImage)

Signals

- void **signalSetCurrent** (qlonglong 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< LabelsTree↩ View::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:

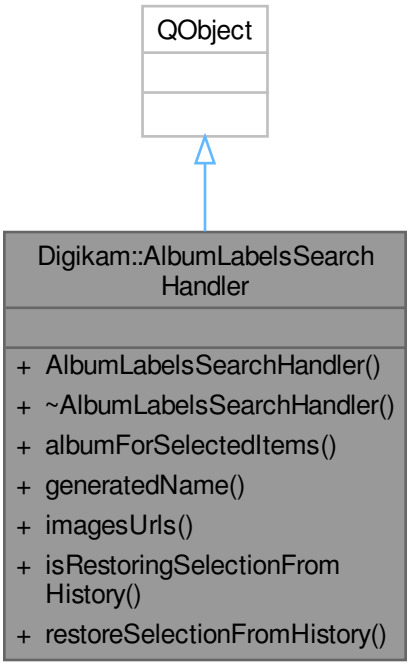
```
AlbumIterator it(album);
while ( it.current() )
{
   qCDebug(DIGIKAM_GENERAL_LOG) << "Album: " << it.current()->title();
   ++it;
}
```

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 )
```

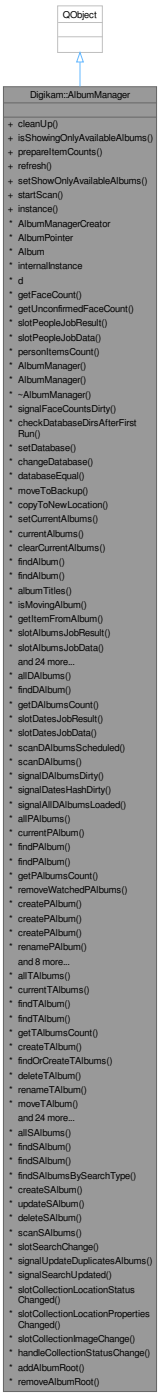
Parameters

neededLabels	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 QSlite Database.
- bool setDatabase (const DbEngineParameters &params, bool priority, const QString &suggestedAlbum↩ Root=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)

  Emittedd 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 showsOnlyAvailableAlbums)

  Emitted when a change is done on available Albums.

Operations on Date Album

- AlbumList allDAlbums () const
- DAlbum * findDAlbum (int id) const
- QMap< YearMonth, int > getDAlbumsCount () const

  Returns the latest count for DAlbums as also emitted via signalDAlbumsDirty.
- void **signalDAlbumsDirty** (const QMap< YearMonth, int > &)
- void **signalDatesHashDirty** (const QHash< QDateTime, int > &)
- void **signalAllDAlbumsLoaded** ()

Operations on Physical Album

- AlbumList allPAlbums () const
- PAlbum * currentPAlbum () const
- PAlbum * findPAlbum (const QUrl &url) const

  Given a complete file url (kde url with file protocol), it will try to find a PAlbum corresponding to it.
- PAlbum * findPAlbum (int id) const
- QHash< int, int > getPAlbumsCount () const

  Returns the latest count for PAlbums as also emitted via signalPAlbumsDirty.
- void **removeWatchedPAlbums** (const PAlbum *const album)
- PAlbum * createPAlbum (PAlbum *parent, const QString &name, const QString &caption, const QDate &date, const QString &category, QString &errMsg)

  Create a new PAlbum 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.

- PAlbum * createPAlbum (const QString &albumRootPath, const QString &name, const QString &caption, const QDate &date, const QString &category, QString &errMsg)

  Overloaded method.
- PAlbum * createPAlbum (const CollectionLocation &location, const QString &name, const QString &caption, const QDate &date, const QString &category, QString &errMsg)

  Overloaded method.
- bool renamePAlbum (PAlbum *album, const QString &newName, QString &errMsg)

  Renames a PAlbum.
- bool updatePAlbumIcon (PAlbum *album, qlonglong iconID, QString &errMsg)

  Update the icon for an album.
- void **signalPAlbumsDirty** (const QHash< int, int > &)
- void **signalEmptyTrash** ()

Operations on Tag Album

- AlbumList allTAlbums () const
- QList< TAlbum * > currentTAlbums () const

  This method is not yet used.
- TAlbum * findTAlbum (int id) const
- TAlbum * findTAlbum (const QString &tagPath) const
- QHash< int, int > getTAlbumsCount () const

  Returns the latest count for TAlbums as also emitted via signalTAlbumsDirty.
- TAlbum * createTAlbum (TAlbum *parent, const QString &name, const QString &iconkde, QString &errMsg)

  Create a new TAlbum with supplied properties as a child of the parent The tag is added to the database.
- AlbumList findOrCreateTAlbums (const QStringList &tagPaths)

  A list of tag paths is supplied.
- bool deleteTAlbum (TAlbum *album, QString &errMsg, QList< qlonglong > *imageIds=nullptr)

  Delete a TAlbum.
- bool renameTAlbum (TAlbum *album, const QString &name, QString &errMsg)

  Renames a TAlbum.
- bool moveTAlbum (TAlbum *album, TAlbum *newParent, QString &errMsg)

  Move a TAlbum to a new parent.
- bool mergeTAlbum (TAlbum *album, TAlbum *destAlbum, bool dialog, QString &errMsg)

  Merge a TAlbum to a TAlbum.
- bool updateTAlbumIcon (TAlbum *album, const QString &iconKDE, qlonglong iconID, QString &errMsg)

  Update the icon for a TAlbum.
- 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 TAlbums 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 **askUserForWriteChangedTAlbumToFiles** (TAlbum *const album)
- void **askUserForWriteChangedTAlbumToFiles** (const QList< qlonglong > &imageIds)
- void **signalTAlbumsDirty** (const QHash< int, int > &)
- void **signalTagPropertiesChanged** (TAlbum *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

location	the collection for the new album
name	the name of the new album
caption	the caption for the new album
date	the date for the new album
category	the category for the new album
errMsg	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

parent	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.
name	the name of the new album
caption	the caption for the new album
date	the date for the new album
category	the category for the new album
errMsg	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

name	name for the new search
type	the type of the search
query	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

parent	the parent album under which to create the new Album
name	the name of the new album
iconkde	the iconkde for the new album (this is a filename which kde iconloader can load up
errMsg	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

album	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 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 or false otherwise

Parameters

album	the TAlbum to delete
errMsg	this will contain the error message describing why the
imageIds	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

type	the type of album
id	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

gid	the global id for the album

9.54.2.19 findDAlbum()

```
DAlbum * Digikam::AlbumManager::findDAlbum (
```

```
            int id ) const
```

Returns

a DAlbum with given ID

Parameters

id	the id for the DAlbum

9.54.2.20 findOrCreateTAlbums()

```
AlbumList Digikam::AlbumManager::findOrCreateTAlbums (
            const QStringList & tagPaths )
```

If no corresponding TAlbum exists, a new one will be created.

Parameters

tagPaths	A list of tag paths

Returns

A list of all TAlbums for the list (already existing or newly created)

9.54.2.21 findPAlbum() [1/2]

```
PAlbum * Digikam::AlbumManager::findPAlbum (
            const QUrl & url ) const
```

Warning

This should not be used, unless really necessary

Returns

PAlbum corresponding to supplied `url`

Parameters

url	the url we need to check

9.54.2.22 findPAlbum() [2/2]

```
PAlbum * Digikam::AlbumManager::findPAlbum (
            int id ) const
```

Returns

a PAlbum with given ID

Parameters

id	the id for the PAlbum

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

name	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

id	the id for the SAlbum

9.54.2.25 findSAlbumsBySearchType()

```
QList< SAlbum * > Digikam::AlbumManager::findSAlbumsBySearchType (
            int searchType ) const
```

Returns

SAlbums with given type, empty list if not found

Parameters

searchType	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

tagPath	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

id	the id for the TAlbum

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

album	The album in which we search the item.
fileName	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

includeInternal	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

album	the Album which should be merged
destAlbum	the Album to which album should be merged
dialog	show dialog to ask the user if he wants to merge
errMsg	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

album	the Album which should be moved
newParent	the Parent Album to which album should be moved
errMsg	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

album	the Album which should be renamed
newName	the new name for the album
errMsg	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

album	the Album which should be renamed
name	the new name for the album
errMsg	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

includeInternal	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

tagIDs	list of tag album IDs
includeInternal	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

leadingSlash	if `true` return tags with a leading slash
includeInternal	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

tagIDs	list of tag album IDs
leadingSlash	if `true` return tags with a leading slash
includeInternal	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

album	the album for which icon should be changed
iconID	the filename of the new icon
errMsg	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

album	the album to update
changedQuery	the new query data of the album
changedName	a new name, or null to keep the current name
type	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

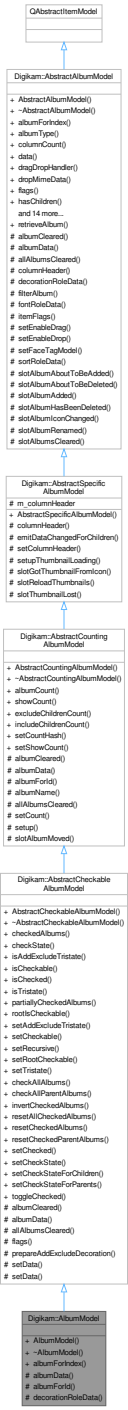
album	the album for which icon should be changed
iconKDE	a simple filename which can be loaded by KIconLoader
iconID	id of the icon image file
errMsg	this will contain the error message describing why the operation failed

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.
- PAlbum ∗ **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 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**

9.55.1 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.

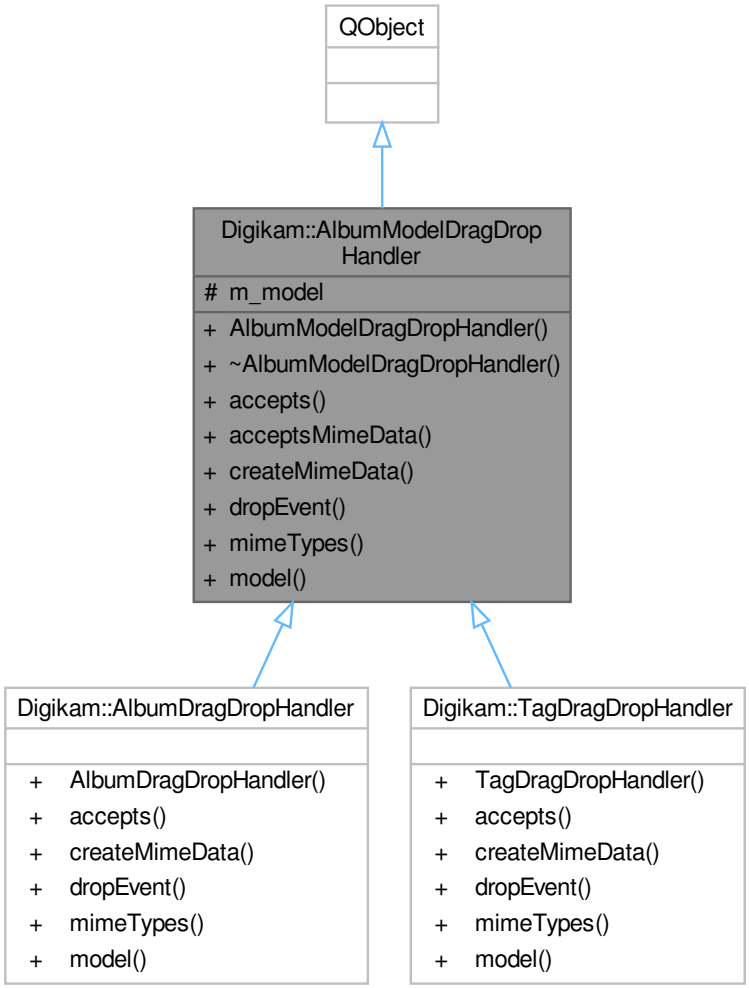
9.55.1.3 decorationRoleData()

```
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 mimeTypes () 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]
```

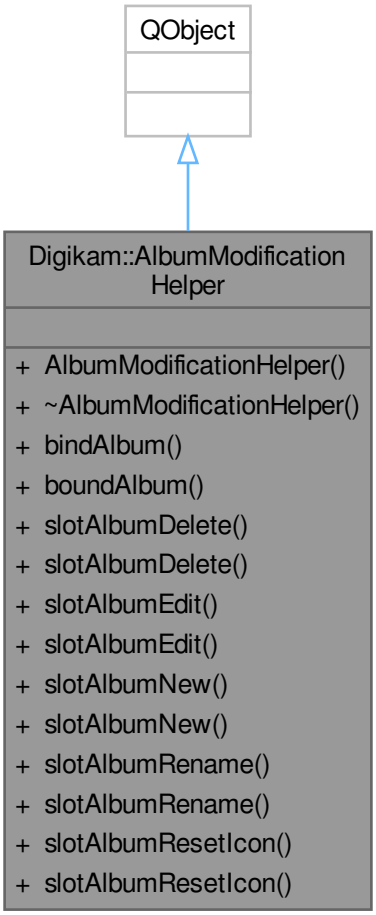
Called by the default implementation of model's mimeTypes().

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

parent	the parent for qt parent child mechanism
dialogParent	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

album	the album to delete

9.57.3.4 slotAlbumEdit

```
void Digikam::AlbumModificationHelper::slotAlbumEdit (
            PAlbum * album )  [slot]
```

Parameters

album	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

parentAlbum	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]
```

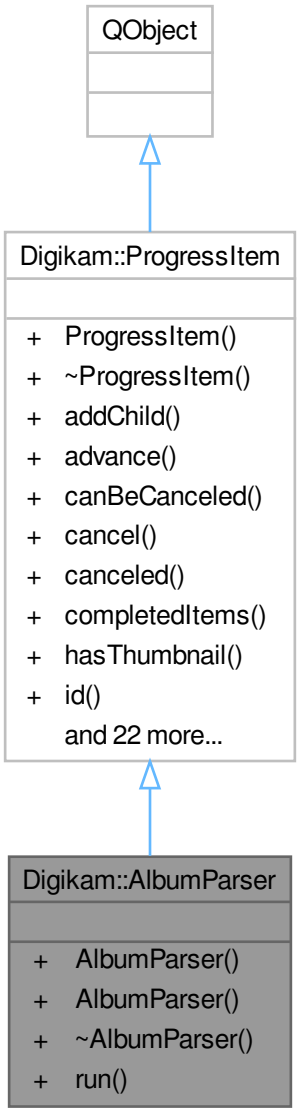
The user will be prompted for a new name.

Parameters

album	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**

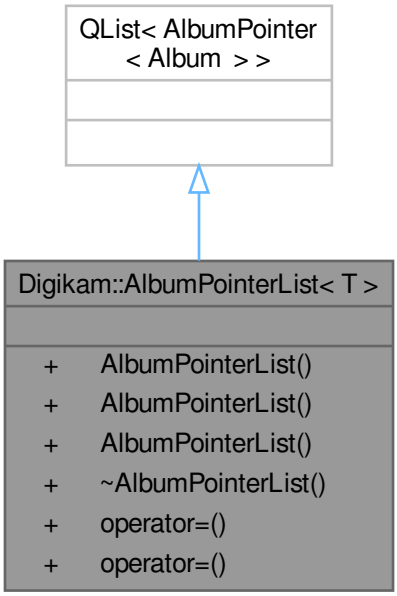
9.59.1 Detailed Description

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

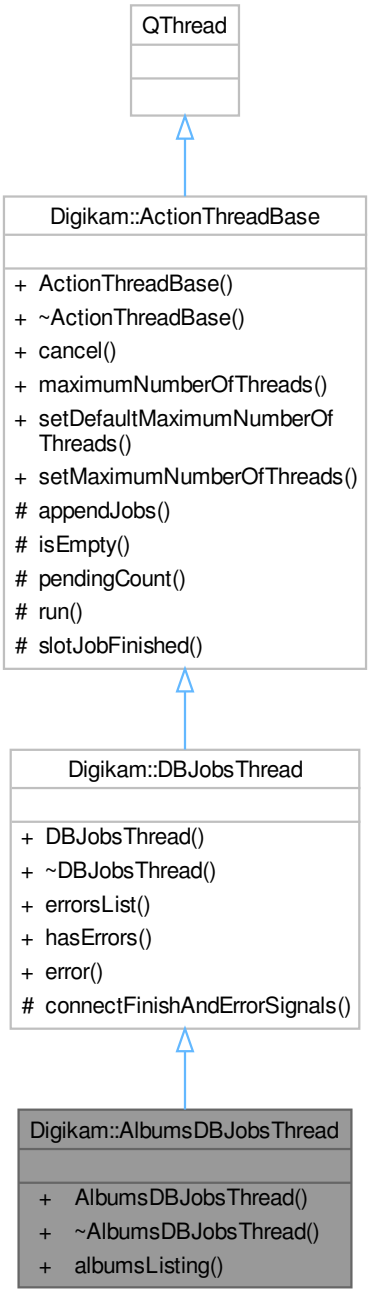
- QString **album** ()
- int **albumRootId** ()
- void **setAlbum** (const QString &album)
- void **setAlbumRootId** (int id)

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()

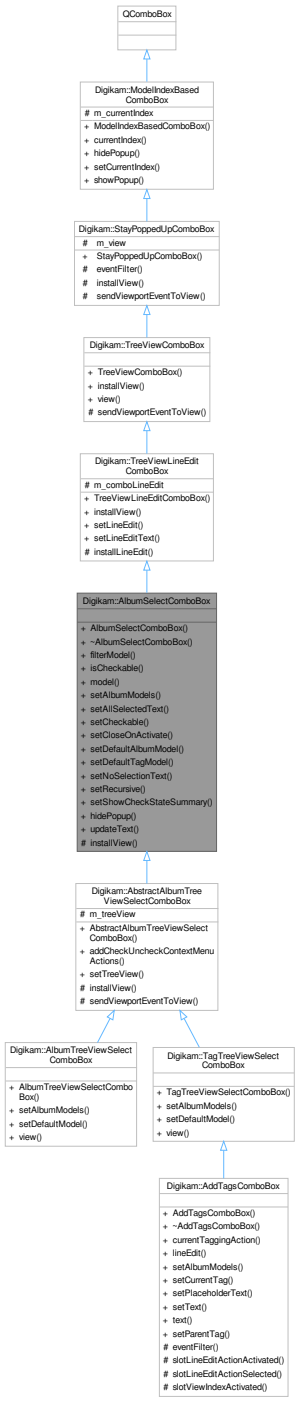
```
void Digikam::AlbumsDBJobsThread::albumsListing (
            const AlbumsDBJobInfo & info )
```

Parameters

info	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 PAlbums 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

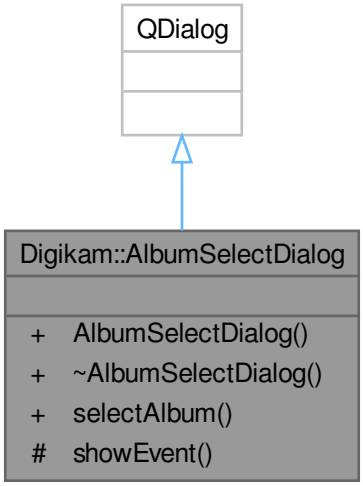
9.66.1.8 updateText

```
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())

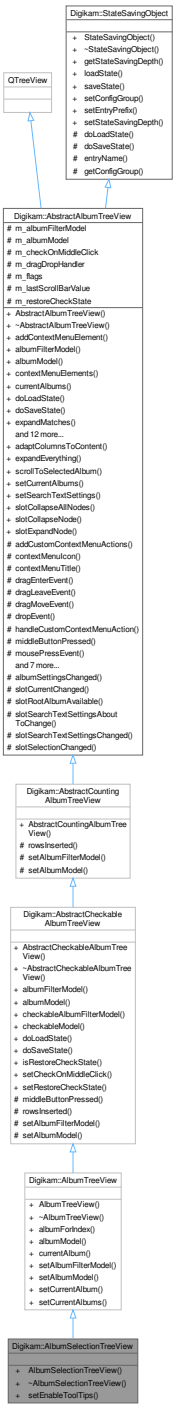
Protected Member Functions

- void **showEvent** (QShowEvent ∗) override

9.68 Digikam::AlbumSelectionTreeView Class Reference

Album tree view used in the left sidebar to select PAlbums 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 **setEnableToolTips** (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 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.

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

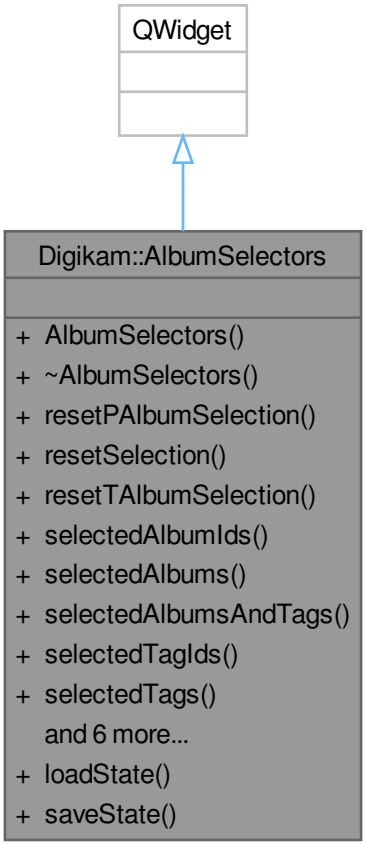
```
void Digikam::AlbumSelectionTreeView::signalFindDuplicates (
            const QList< PAlbum * > & albums )  [signal]
```

Parameters

albums	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.

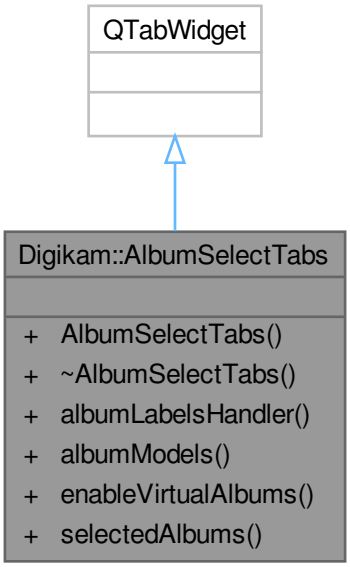
9.69.2.4 setTagSelected()

```
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** ()

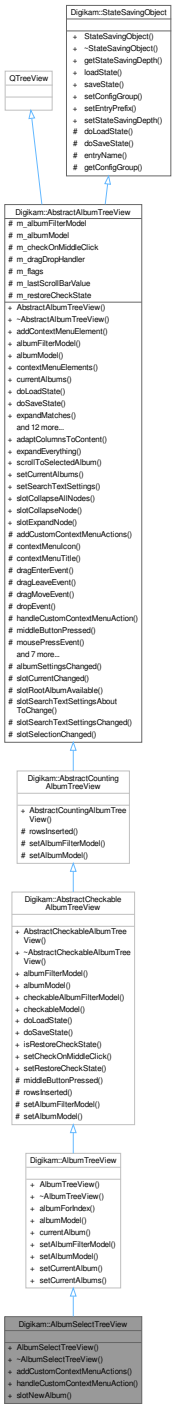
Public Member Functions

- **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 albumModification↩ Helper, 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)
- 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 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.

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.

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

model	album model to work with
albumModificationHelper	helper object for modifying albums
parent	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

cmh	helper object to create the context menu
album	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]
```

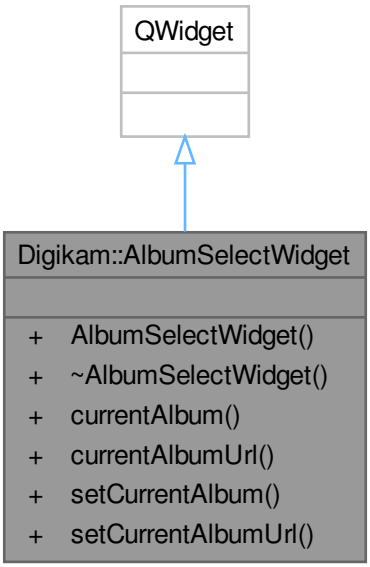
Parameters

action	the action that was chosen by the user, may be null if none of the custom actions were selected
album	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

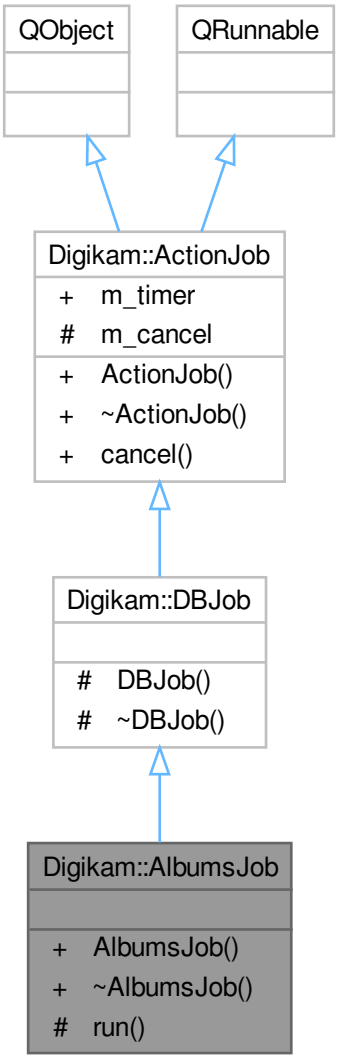
- QStringList **childrenIDs**
- QString **title**
- bool **uploadable** = true

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.

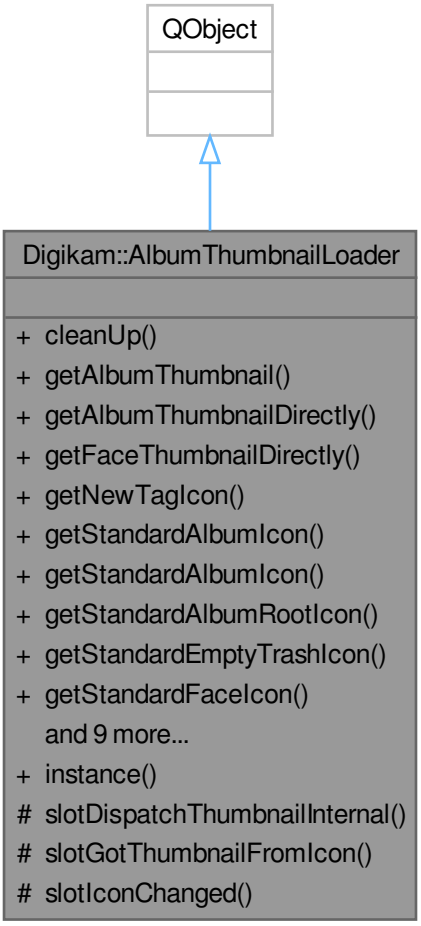
Protected Attributes inherited from Digikam::ActionJob

- 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 (PAlbum ∗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** (PAlbum ∗const album, RelativeSize size=NormalSize)
- QPixmap **getStandardAlbumIcon** (RelativeSize size=NormalSize)
- QPixmap **getStandardAlbumRootIcon** (RelativeSize size=NormalSize)
- QPixmap **getStandardEmptyTrashIcon** (RelativeSize size=NormalSize)
- QPixmap **getStandardFaceIcon** (TAlbum ∗const album, RelativeSize size=NormalSize)
- QPixmap **getStandardFullTrashIcon** (RelativeSize size=NormalSize)
- QPixmap **getStandardOfflineIcon** (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.

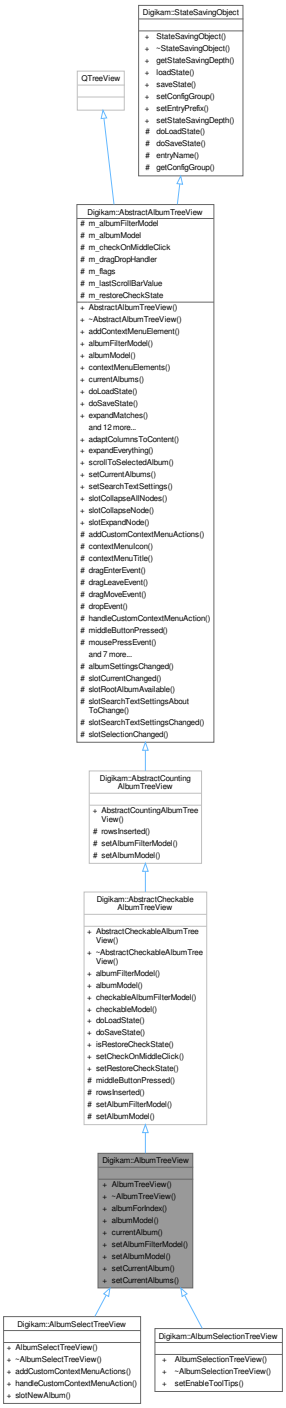
9.76.2.10 signalThumbnail

```
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)
- 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 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.

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.

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

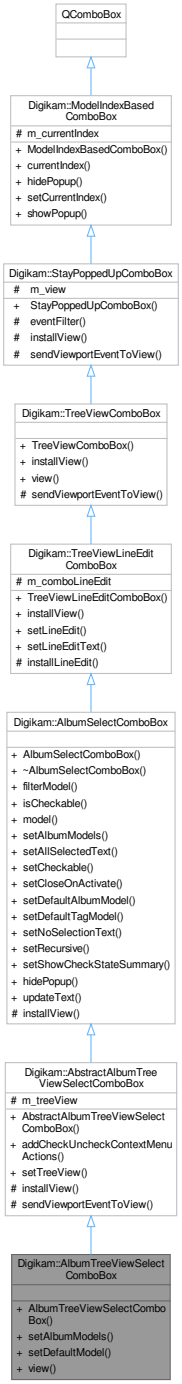
  Returns the config group that must be used for state saving and loading.

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

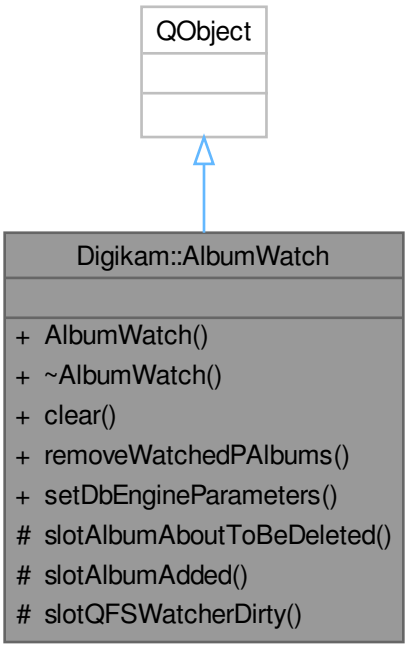
- QAbstractItemView ∗ **m_view** = nullptr

Protected Attributes inherited from Digikam::ModelIndexBasedComboBox

- QPersistentModelIndex **m_currentIndex**

9.79 Digikam::AlbumWatch Class Reference

Inheritance diagram for Digikam::AlbumWatch:



Public Member Functions

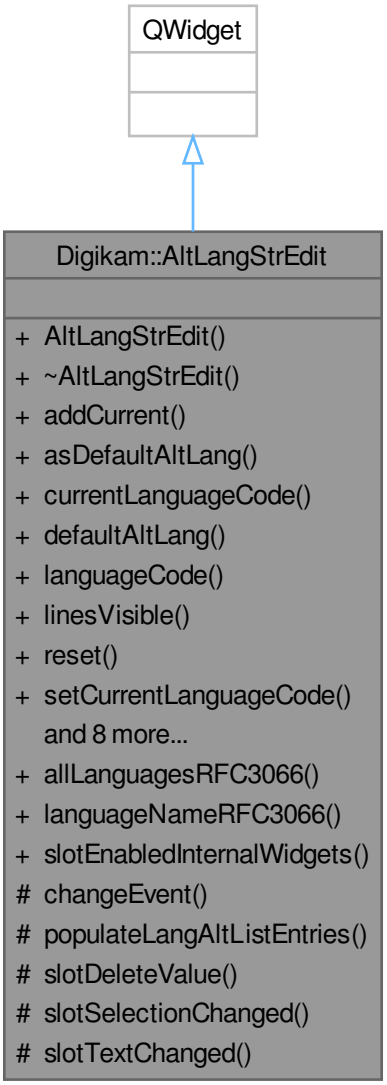
- **AlbumWatch** (AlbumManager *const parent=nullptr)
- void **clear** ()
- void **removeWatchedPAlbums** (const PAlbum *const album)
- void **setDbEngineParameters** (const DbEngineParameters &params)

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 contructor.
- 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 visibile 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 remplaced.

9.80.2.4 setTitleWidget()

```
void Digikam::AltLangStrEdit::setTitleWidget (
            QWidget *const twdg )
```

If a title widget already exists, it's remplaced.

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 **mouseReleaseEvent** (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.

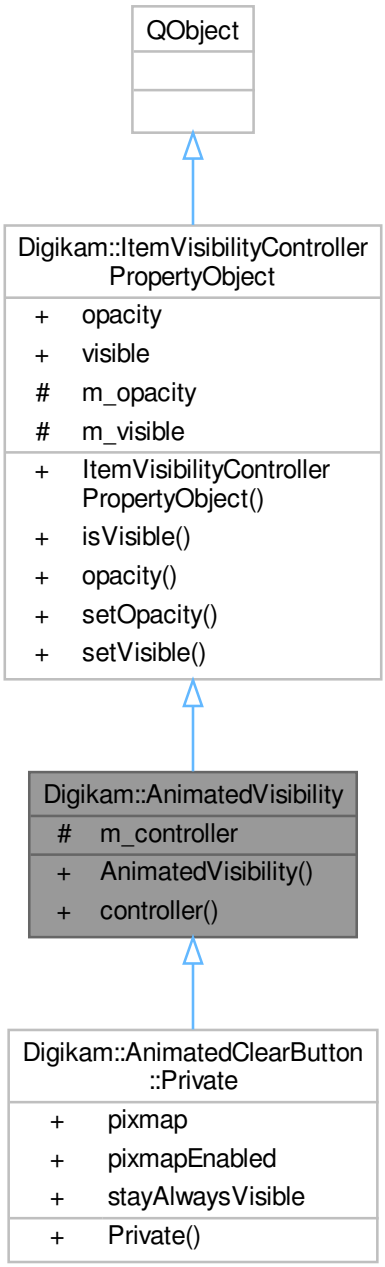
9.81.1.2 stayVisibleWhenAnimatedOut()

```
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** (DImg *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 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** ()

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.

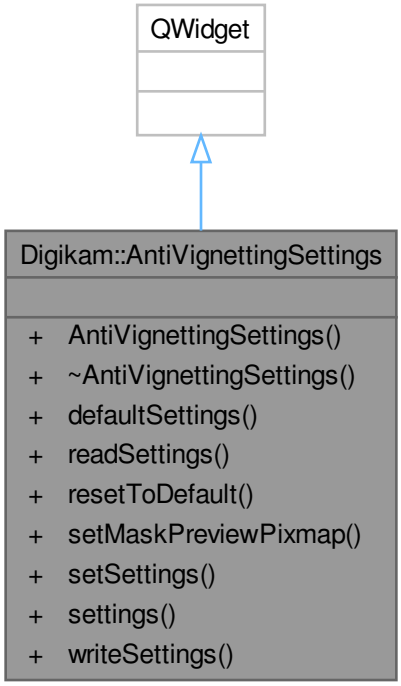
9.84.1.3 readParameters()

```
void Digikam::AntiVignettingFilter::readParameters (
            const FilterAction & action )  [override], [virtual]
```

Implements Digikam::DImgThreadedFilter.

9.85 Digikam::AntiVignettingSettings Class Reference

Inheritance diagram for Digikam::AntiVignettingSettings:



Signals

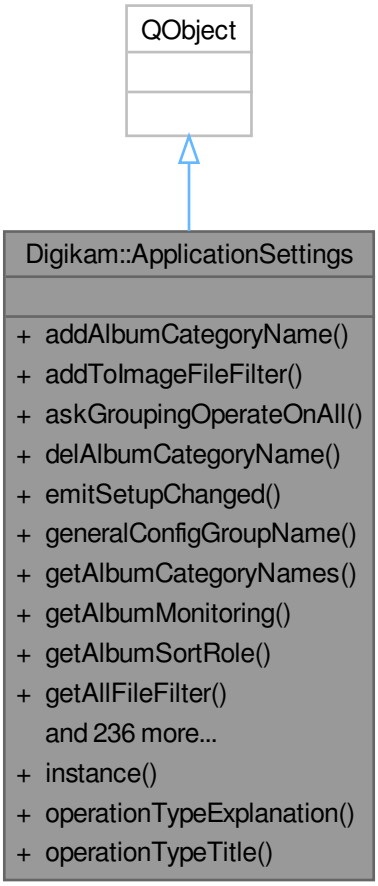
- void **signalSettingsChanged** ()

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
- HaarIface::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** (HaarIface::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

type	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

type	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::getStringComparison↩
Type ( ) const
```

Returns

string comparison type to use.

9.86.2.4 operationTypeExplanation()

```
QString Digikam::ApplicationSettings::operationTypeExplanation (
            OperationType type )  [static]
```

for tooltips)

Parameters

type	Operation to be performed

Returns

Translated operation explanation

9.86.2.5 operationTypeTitle()

```
QString Digikam::ApplicationSettings::operationTypeTitle (
            OperationType type )  [static]
```

Parameters

type	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

dontShowAgainName	the name that identify the message box.

9.86.2.7 saveMsgBoxShouldBeShown()

```
void Digikam::ApplicationSettings::saveMsgBoxShouldBeShown (
            const QString & dontShowAgainName )
```

Parameters

dontShowAgainName	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

type	Operation to be performed
applyAll	Whether to apply to all images or just one, or ask

9.86.2.9 setStringComparisonType()

```
void Digikam::ApplicationSettings::setStringComparisonType (
            ApplicationSettings::StringComparisonType val )
```

Parameters

val	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

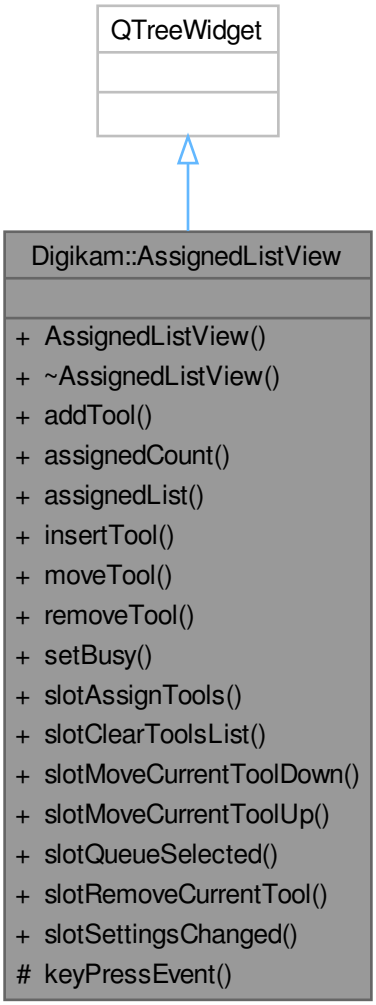
- QString **m_destFileName**
- QUrl **m_itemUrl**
- BatchSetList **m_toolsList**

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 QMultiMap< 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

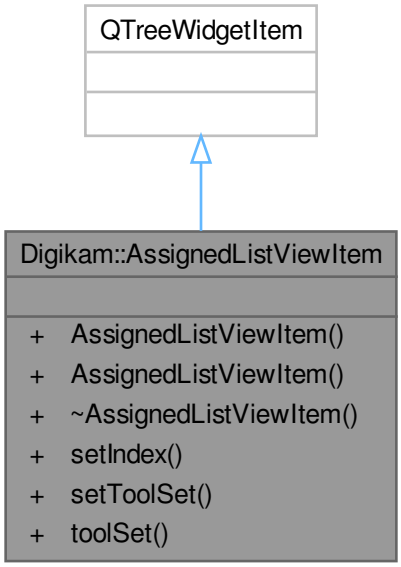
- **AssignedListView** (QWidget ∗const parent)
- AssignedListViewItem ∗ **addTool** (const BatchToolSet &set)
- int **assignedCount** ()
- AssignedBatchTools **assignedList** ()
- AssignedListViewItem ∗ **insertTool** (AssignedListViewItem ∗const preceding, const BatchToolSet &set)
- AssignedListViewItem ∗ **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:

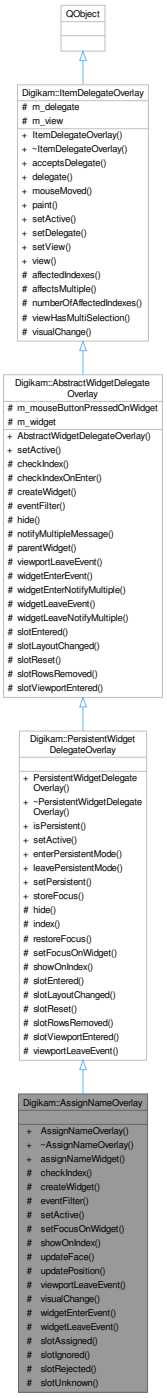


Public Member Functions

- **AssignedListViewItem** (QTreeWidget ∗const parent)
- **AssignedListViewItem** (QTreeWidget ∗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 overlayed 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.

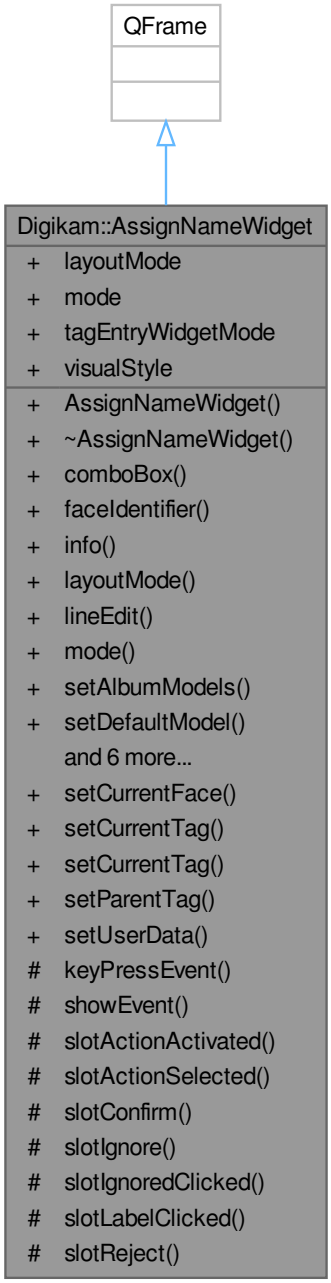
9.90.1.10 widgetLeaveEvent()

```
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** , **AddTagsLine↩ EditMode** }
- enum **VisualStyle** { **InvalidVisualStyle** , **StyledFrame** , **TranslucentDarkRound** , **TranslucentThemed↩ Frameless** }

Public Slots

- void **setCurrentFace** (const FaceTagsIface &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.

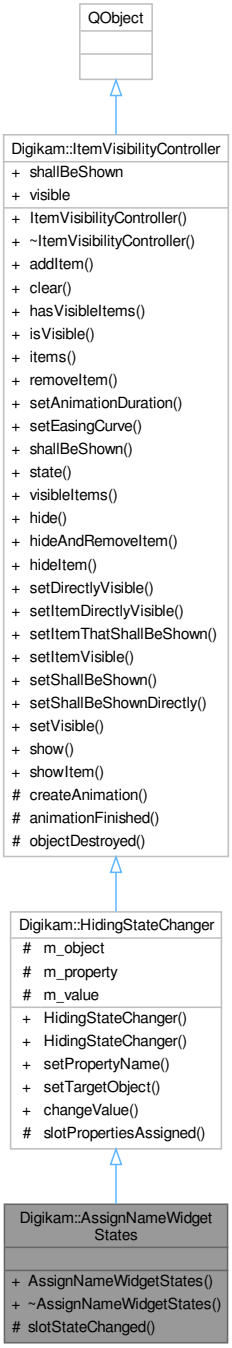
9.91.1.5 setUserData

```
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** (FaceItem ∗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 **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 msecs)
- 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

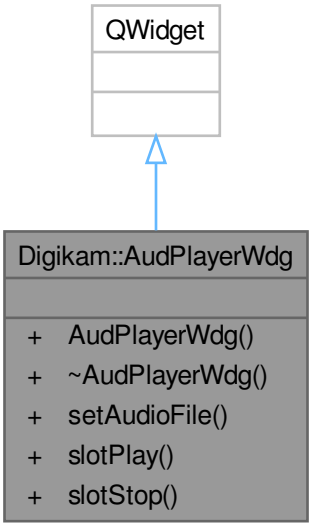
- QObject ∗ **m_object** = nullptr
- QByteArray **m_property**
- QVariant **m_value**

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 ananlyser 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 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.

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 refImage, 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 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** ()

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 refImage, 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 **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

9.96.1 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()

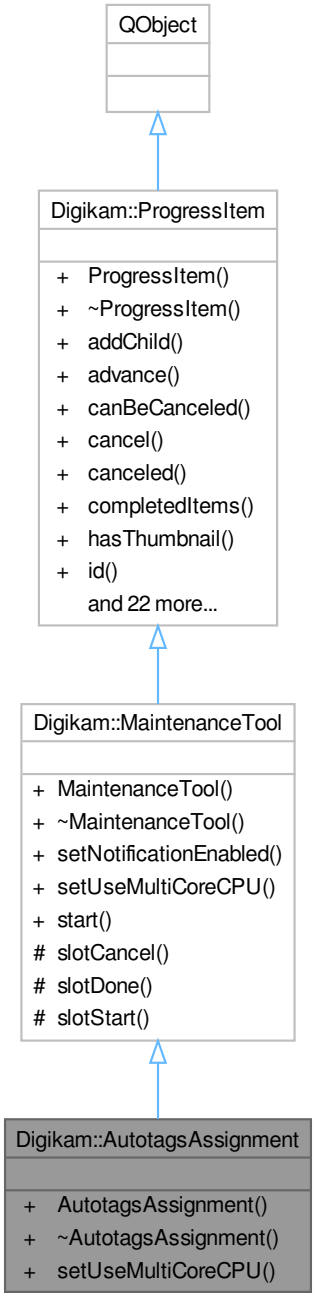
```
QList< QList< QString > > Digikam::AutoTagsAssign::generateTagsList (
            const QList< DImg > & inputImages,
            int batchSize ) const
```

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 elpased 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** ()

9.98.1 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

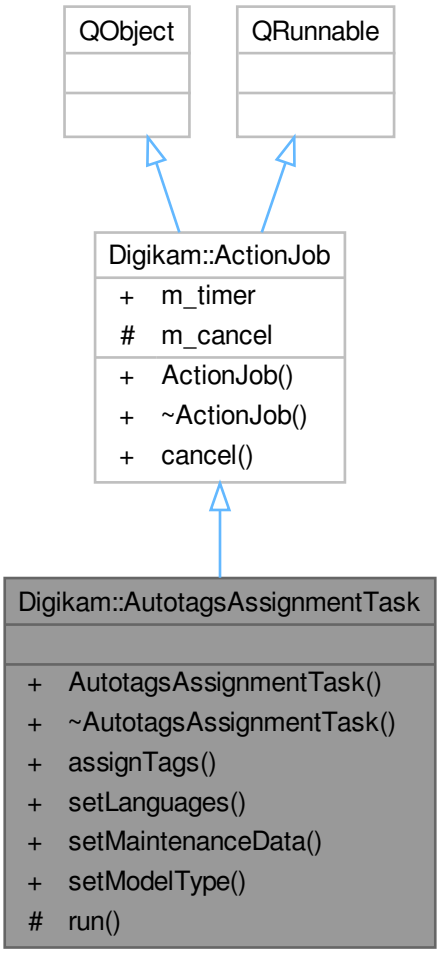
9.98.2.1 setUseMultiCoreCPU()

```
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

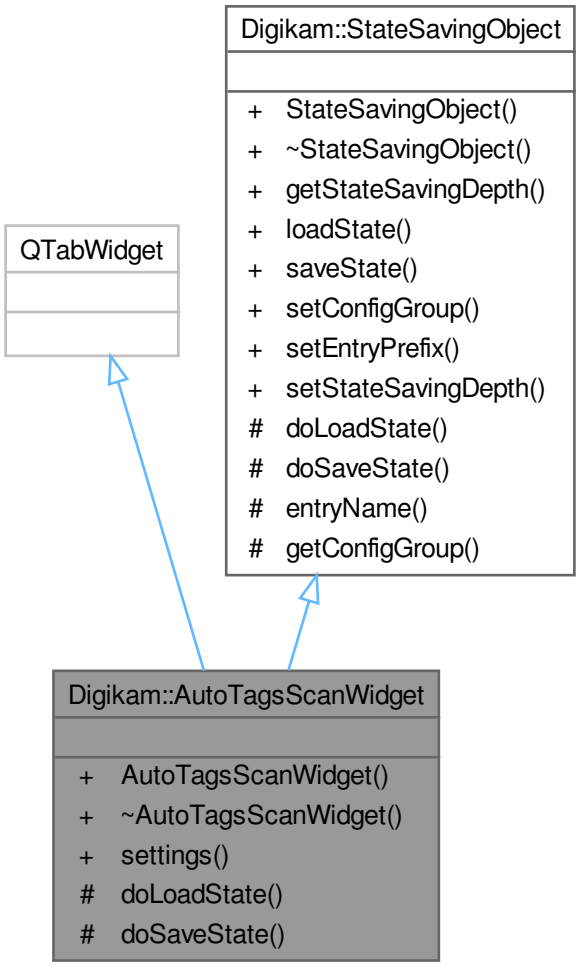
`enum Digikam::AutoTagsScanSettings::ScanMode`

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]
```

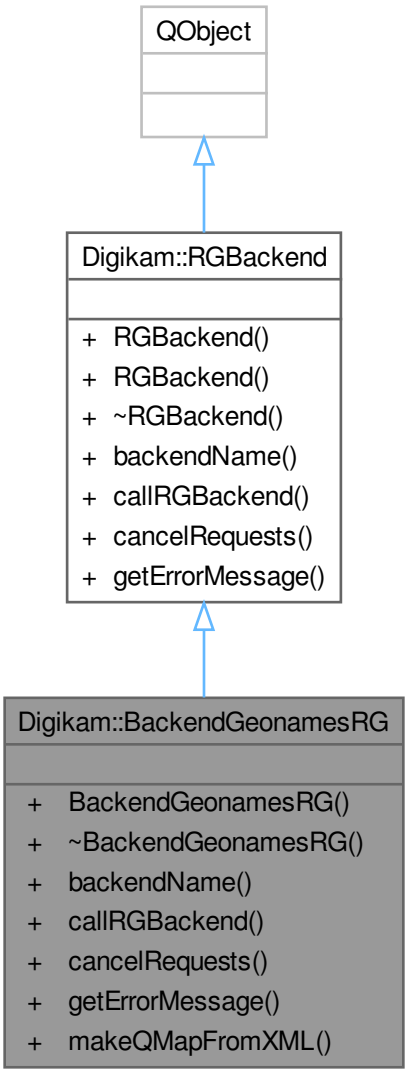
Use getConfigGroup() and entryName() for the implementation.

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

parent	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

rgList	A list containing information needed in reverse geocoding process. At this point, it contains only coordinates.
language	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.

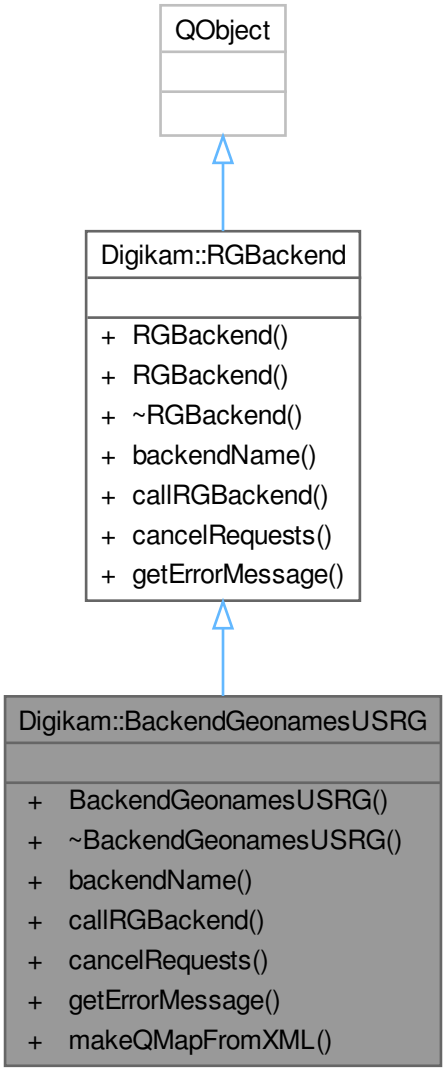
Parameters

xmlData	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 callRGBackend (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

parent	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

rgList	A list containing information needed in reverse geocoding process. At this point, it contains only coordinates.
language	The language in which the data will be returned.

Implements Digikam::RGBackend.

9.103.2.3 cancelRequests()

```
void Digikam::BackendGeonamesUSRG::cancelRequests ( )  [override], [virtual]
```

Implements Digikam::RGBackend.

9.103.2.4 getErrorMessage()

```
QString Digikam::BackendGeonamesUSRG::getErrorMessage ( )  [override], [virtual]
```

Returns

Error message, if any.

Reimplemented from Digikam::RGBackend.

9.103.2.5 makeQMapFromXML()

```
QMap< QString, QString > Digikam::BackendGeonamesUSRG::makeQMapFromXML (
            const QString & xmlData )
```

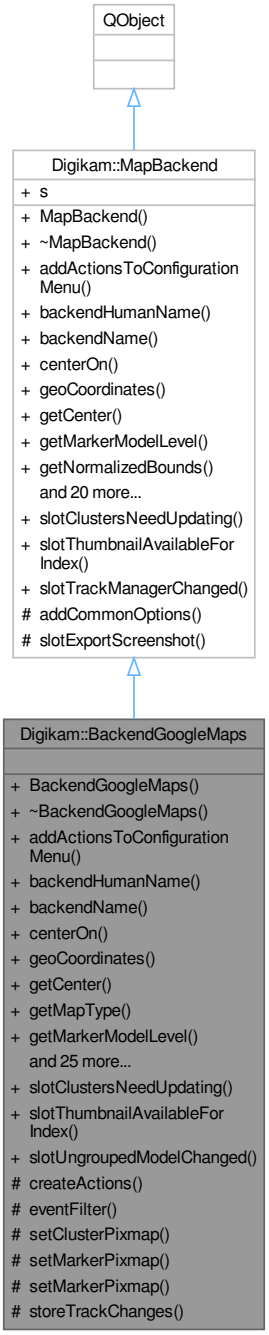
This function translates the XML into a QMap.

Parameters

xmlData	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< GeoIfaceSharedData > &sharedData, QObject *const parent=nullptr)
- ~BackendGoogleMaps () override
- void addActionsToConfigurationMenu (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 (GeoIfaceInternalWidgetInfo *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< GeoIfaceSharedData > &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 &icon↩ Size, 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 QIntList &clusterIndices)
- void **signalClustersMoved** (const QIntList &clusterIndices, const QPair< int, QModelIndex > &snapTarget)
- void **signalMarkersMoved** (const QIntList &markerIndices)
- void **signalSelectionHasBeenMade** (const Digikam::GeoCoordinates::Pair &coordinates)
- void **signalZoomChanged** (const QString &newZoom)

Public Attributes inherited from Digikam::MapBackend

- const QExplicitlySharedDataPointer< GeoIfaceSharedData > **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 (
            GeoIfaceInternalWidgetInfo *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.

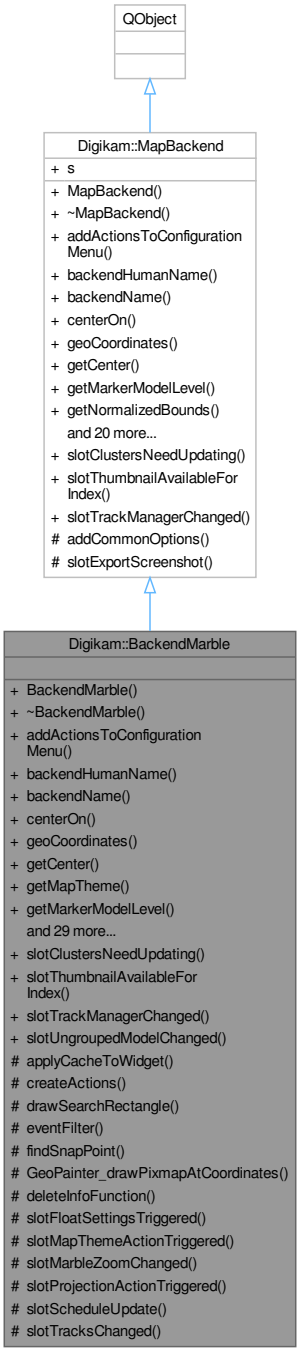
9.104.2.29 zoomOut()

```
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< GeoIfaceSharedData > &sharedData, QObject *const parent=nullptr)
- ~BackendMarble () override
- void addActionsToConfigurationMenu (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 (GeoIfaceInternalWidgetInfo *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< GeoIfaceSharedData > &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 &search↩ Rectangle, const bool isOldRectangle)
- bool eventFilter (QObject ∗object, QEvent ∗event) override
- bool **findSnapPoint** (const QPoint &actualPoint, QPoint ∗const snapPoint, GeoCoordinates ∗const snap↩ Coordinates, 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** (GeoIfaceInternalWidgetInfo ∗const info)

Additional Inherited Members

Signals inherited from Digikam::MapBackend

- void **signalBackendReadyChanged** (const QString &backendName)
- void **signalClustersClicked** (const QIntList &clusterIndices)
- void **signalClustersMoved** (const QIntList &clusterIndices, const QPair< int, QModelIndex > &snapTarget)
- void **signalMarkersMoved** (const QIntList &markerIndices)
- void **signalSelectionHasBeenMade** (const Digikam::GeoCoordinates::Pair &coordinates)
- void **signalZoomChanged** (const QString &newZoom)

Public Attributes inherited from Digikam::MapBackend

- const QExplicitlySharedDataPointer< GeoIfaceSharedData > **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

painter	Marble::GeoPainter on which to draw the pixmap
pixmap	Pixmap to be drawn
coordinates	GeoCoordinates where the image is to be drawn
offsetPoint	Point in the `pixmap` which should be at `coordinates`

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 (
            GeoIfaceInternalWidgetInfo *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.

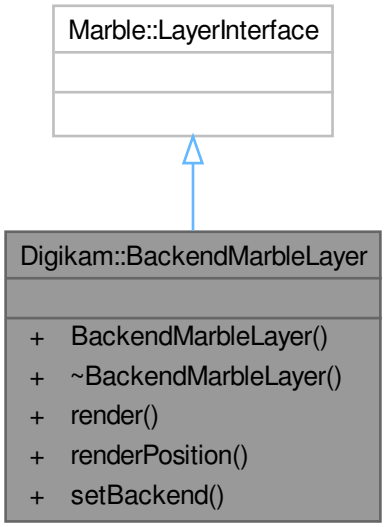
9.105.2.34 zoomOut()

```
void Digikam::BackendMarble::zoomOut ( )  [override], [virtual]
```

Implements Digikam::MapBackend.

9.106 Digikam::BackendMarbleLayer Class Reference

Inheritance diagram for Digikam::BackendMarbleLayer:



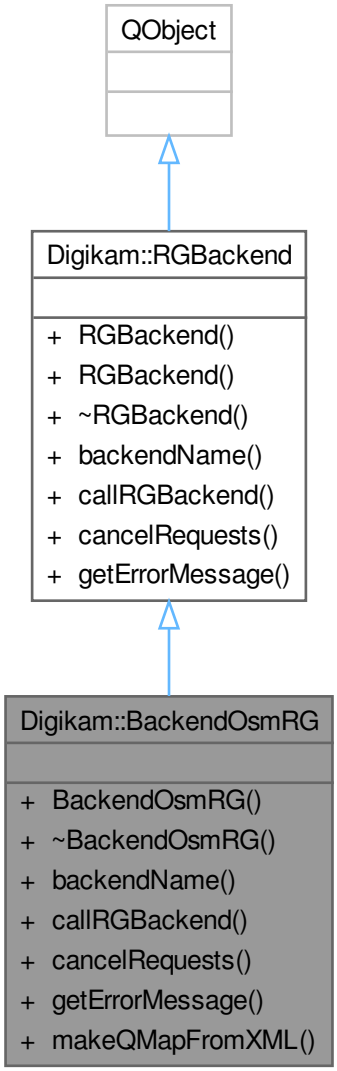
Public Member Functions

- **BackendMarbleLayer** (BackendMarble ∗const pMarbleBackend)
- bool **render** (Marble::GeoPainter ∗painter, Marble::ViewportParams ∗viewport, const QString &renderPos=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 callRGBackend (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

parent	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

rgList	A list containing information needed in reverse geocoding process. At this point, it contains only coordinates.
language	The language in which the data will be returned.

Implements Digikam::RGBackend.

9.107.2.3 cancelRequests()

```
void Digikam::BackendOsmRG::cancelRequests ( )  [override], [virtual]
```

Implements Digikam::RGBackend.

9.107.2.4 getErrorMessage()

```
QString Digikam::BackendOsmRG::getErrorMessage ( )  [override], [virtual]
```

Returns

Error message, if any.

Reimplemented from Digikam::RGBackend.

9.107.2.5 makeQMapFromXML()

```
QMap< QString, QString > Digikam::BackendOsmRG::makeQMapFromXML (
            const QString & xmlData )
```

This function translates the XML into a QMap.

Parameters

xmlData	The returned XML.

9.108 Digikam::BalooInfo Class Reference

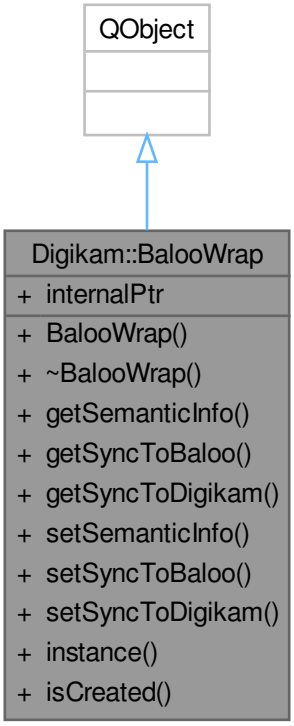
Public Attributes

- 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 &bInfo)

  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

url	- image url

Returns

- container class for tags, comment, rating

9.109.2.2 setSemanticInfo()

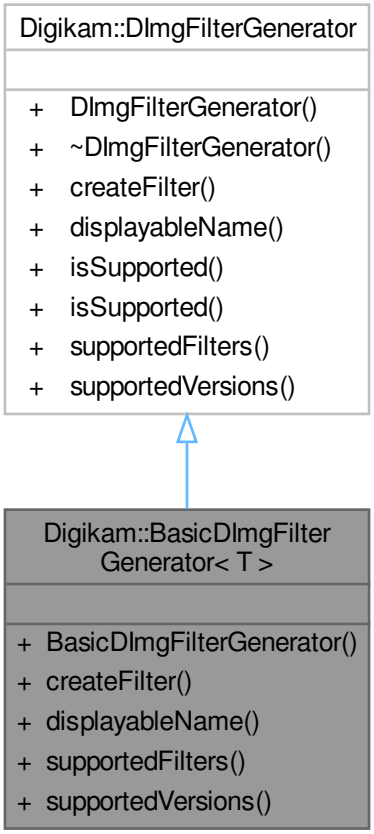
```
void Digikam::BalooWrap::setSemanticInfo (
            const QUrl & url,
            const BalooInfo & bInfo )
```

Parameters

url	- image url
bInfo	- 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 displayableName (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.

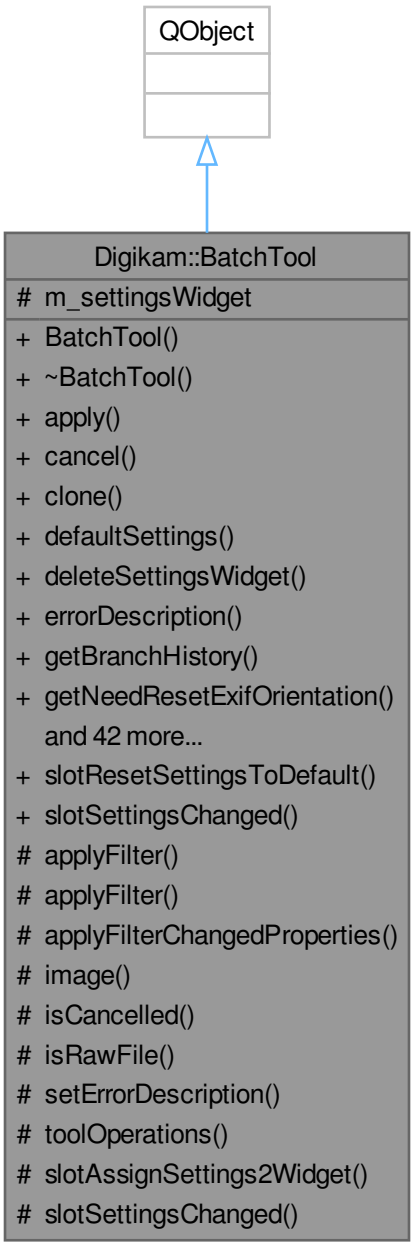
9.110.2.4 supportedVersions()

```
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

BaseTool	digiKam core tools.
CustomTool	List of tools grouped and customized by users.
ColorTool	Tools to manage image colors (Curves, BCG, etc...)
EnhanceTool	Tools to enhance images (NR, sharp, etc...)
TransformTool	Tools to transform images geometry (resize, rotate, flip, etc...)
DecorateTool	Tools to decorate images (Border, watermark, etc...)
FiltersTool	Tools to apply filters and special effects (film grain, BlurFx, etc...)
ConvertTool	Tools to convert images format (PNG, JPEG, TIFF, etc...)
MetadataTool	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

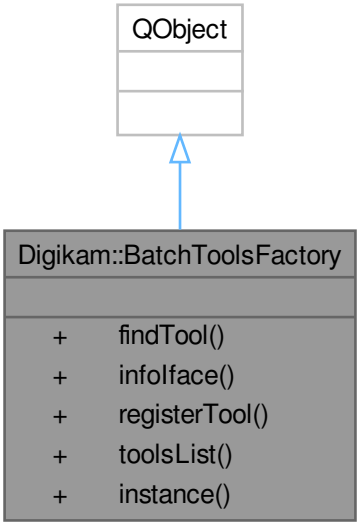
9.112.1.1 operator==()

```
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
- BqmInfoIface * **infoIface** () 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

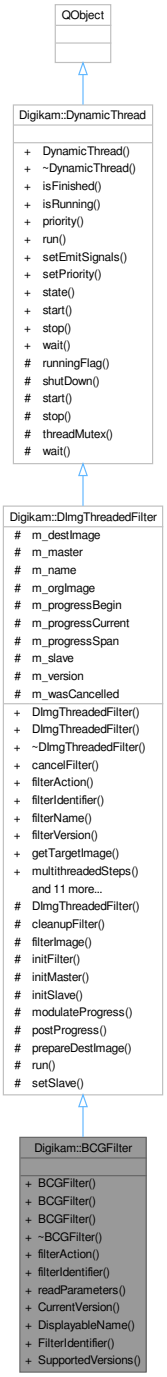
- static BCGContainer **fromFilterAction** (const FilterAction &action, const QString &prefix=QString())

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 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** ()

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.

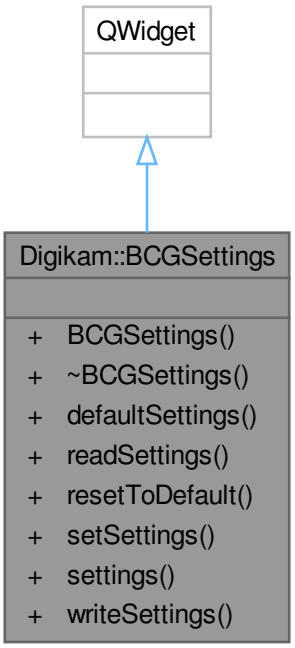
9.115.1.3 readParameters()

```
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 , SQLError , 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 lastInsert↩Id=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 last↩InsertId=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 &field↩ Names, 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.
SQLError	An SQLError 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 & 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()

```
QSqlError 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.

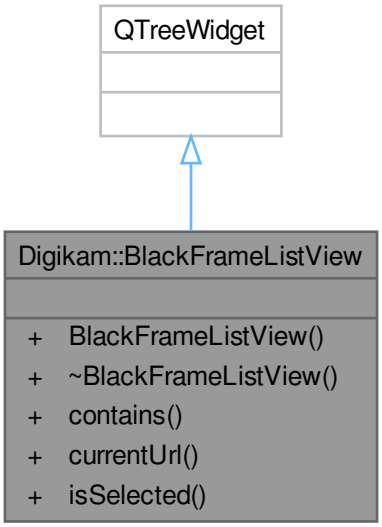
9.118 Digikam::BdEngineBackend::QueryState Class Reference

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

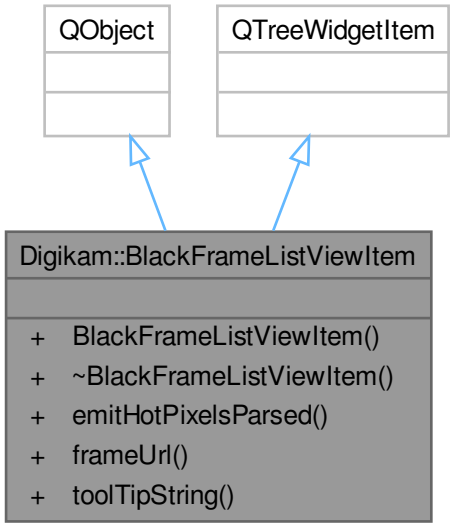
- void **signalBlackFrameRemoved** (const QUrl &)
- void **signalBlackFrameSelected** (const QList< HotPixelProps > &, const QUrl &)
- void **signalClearBlackFrameList** ()

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

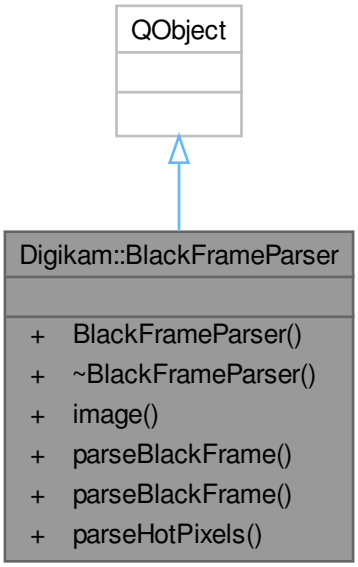
- void **signalHotPixelsParsed** (const QList< HotPixelProps > &, const QUrl &)

Public Member Functions

- **BlackFrameListViewItem** (QTreeWidget *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** (QTreeWidget ∗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 **toolTipIsEmpty** () const
- void **updateToolTip** ()

9.122.1 Member Function Documentation

9.122.1.1 repositionRect()

```
QRect Digikam::BlackFrameToolTip::repositionRect ( )  [override], [protected], [virtual]
```

Implements Digikam::DItemToolTip.

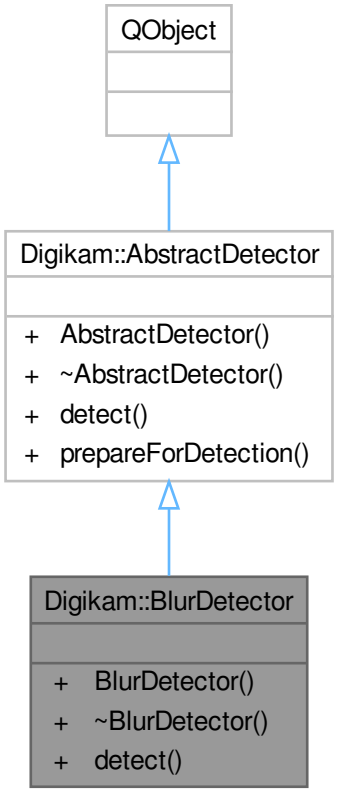
9.122.1.2 tipContents()

```
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 **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

9.124.1 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 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** ()

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.

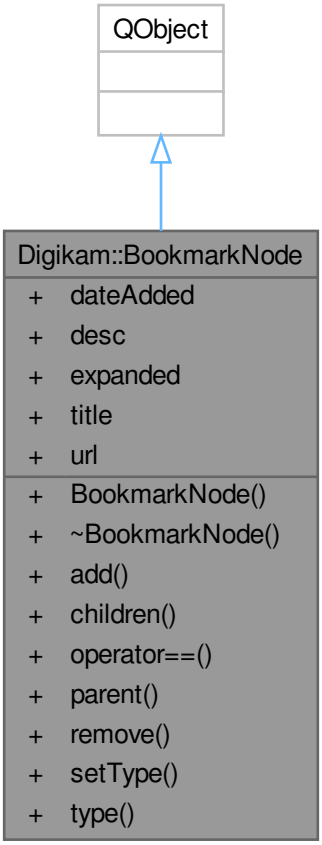
9.125.1.3 readParameters()

```
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

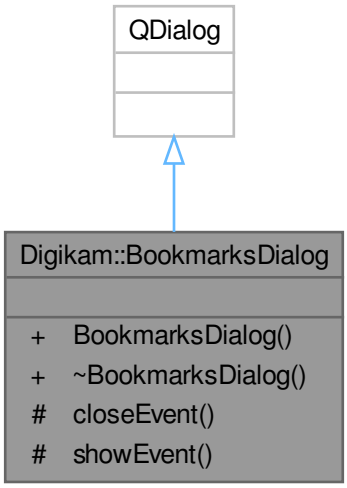
- **BookmarkNode** (Type type=Root, BookmarkNode ∗const parent=nullptr)
- void **add** (BookmarkNode ∗const child, int offset=-1)
- QList< BookmarkNode ∗ > **children** () const
- bool **operator==** (const BookmarkNode &other) const
- BookmarkNode ∗ **parent** () const
- void **remove** (BookmarkNode ∗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)

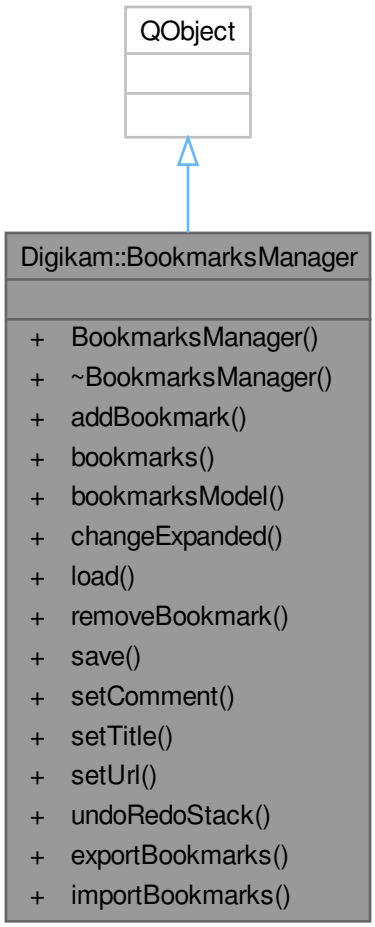
Protected Member Functions

- 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

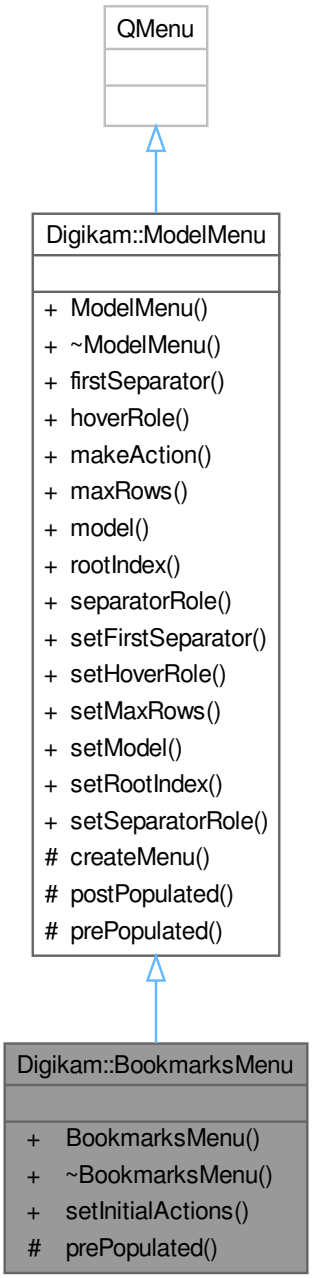
Friends

- 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()

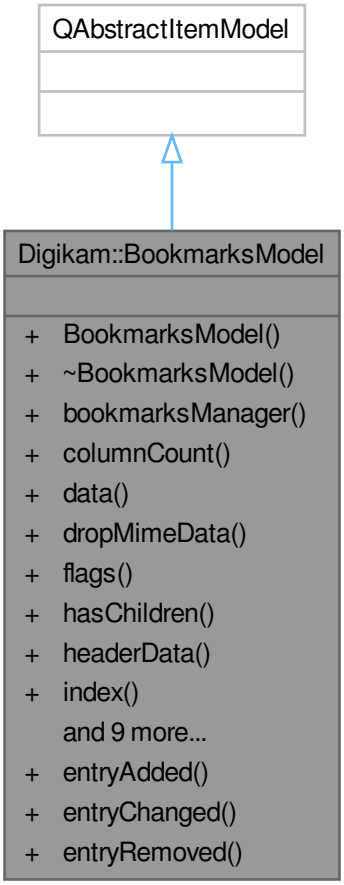
```
bool Digikam::BookmarksMenu::prePopulated ( )  [override], [protected], [virtual]
```

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 ∗orgImage, 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 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** ()

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.

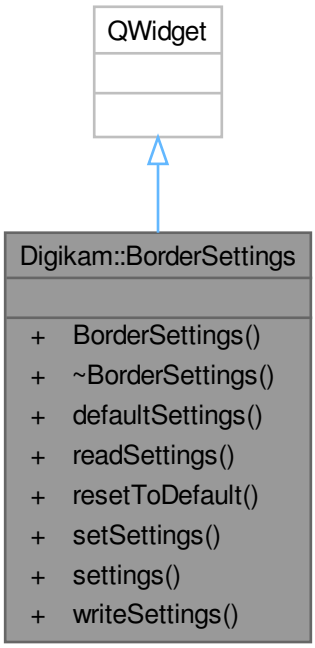
9.132.1.3 readParameters()

```
void Digikam::BorderFilter::readParameters (
            const FilterAction & action )  [override], [virtual]
```

Implements Digikam::DImgThreadedFilter.

9.133 Digikam::BorderSettings Class Reference

Inheritance diagram for Digikam::BorderSettings:



Signals

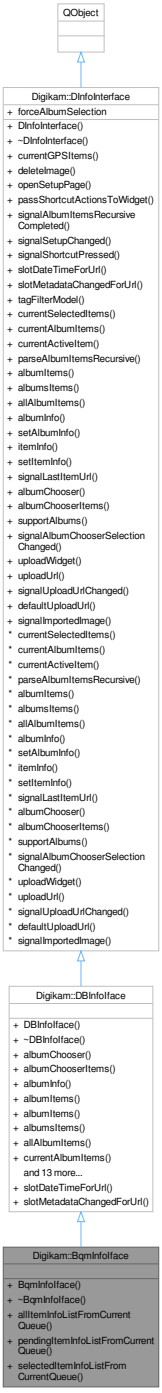
- void **signalSettingsChanged** ()

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::BqmInfoIface Class Reference

Inheritance diagram for Digikam::BqmInfoIface:



Public Member Functions

- **BqmInfoIface** (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::DBInfoIface

- **DBInfoIface** (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 Slots inherited from Digikam::DBInfoIface

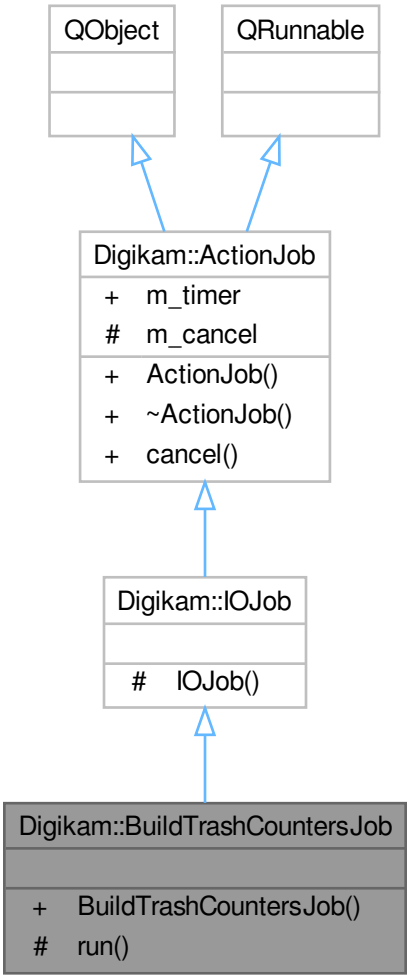
- void **slotDateTimeForUrl** (const QUrl &url, const QDateTime &dt, bool updModDate) override
- void **slotMetadataChangedForUrl** (const QUrl &url) override

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 **signalOneProccessed** (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** , **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 ∗orgImage, 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 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** ()

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.

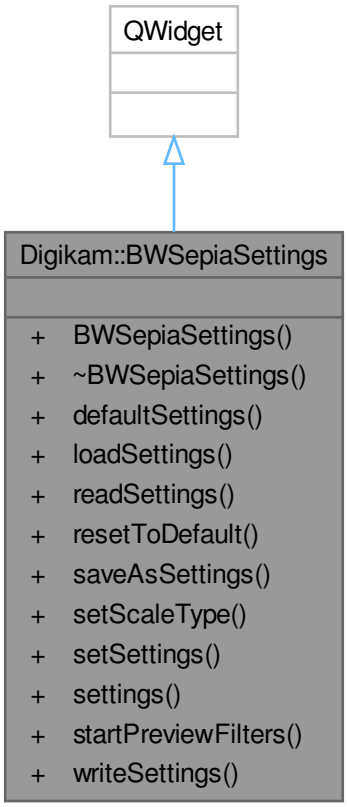
9.137.1.3 readParameters()

```
void Digikam::BWSepiaFilter::readParameters (
            const FilterAction & action )  [override], [virtual]
```

Implements Digikam::DImgThreadedFilter.

9.138 Digikam::BWSepiaSettings Class Reference

Inheritance diagram for Digikam::BWSepiaSettings:



Signals

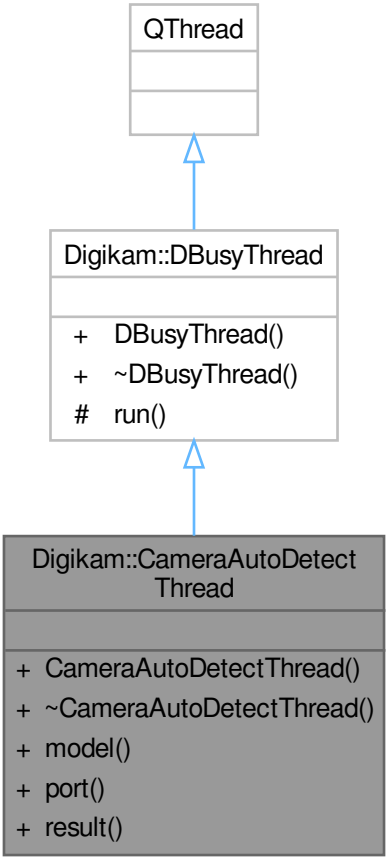
- void **signalSettingsChanged** ()

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** ()

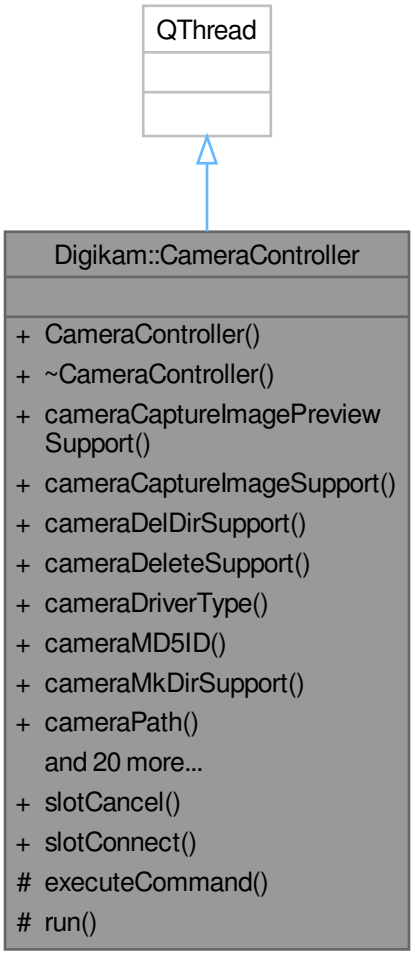
Protected Member Functions inherited from Digikam::DBusyThread

- 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)

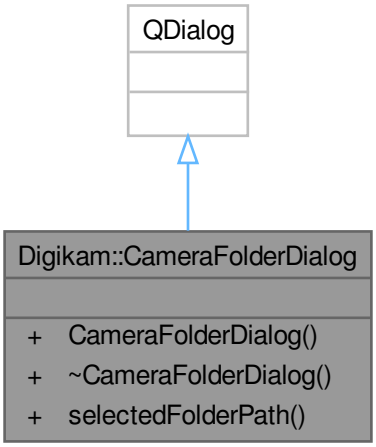
  Get thumbnails for a list of camera items plus advanced information from metadata.
- void **listFiles** (const QString &folder, bool useMetadata)
- void **listFolders** (const QString &folder=QString())
- void **listRootFolder** (bool useMetadata)
- void **lockFile** (const QString &folder, const QString &file, bool lock)
- QIcon **mimeTypeThumbnail** (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:

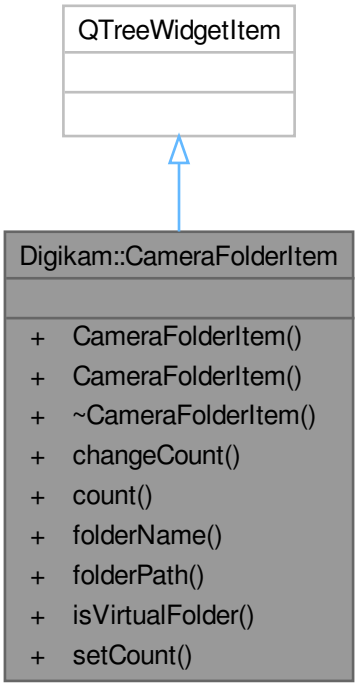


Public Member Functions

- **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:

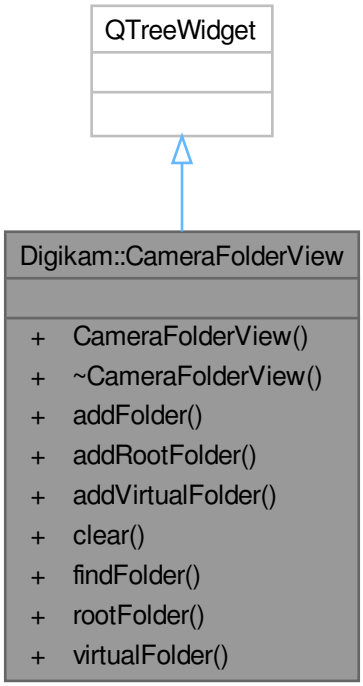


Public Member Functions

- **CameraFolderItem** (QTreeWidget ∗const parent, const QString &name, const QIcon &icon=QIcon::from↩ Theme(QLatin1String("folder")))
- **CameraFolderItem** (QTreeWidgetItem ∗const parent, const QString &folderName, const QString &folder↩ Path, 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

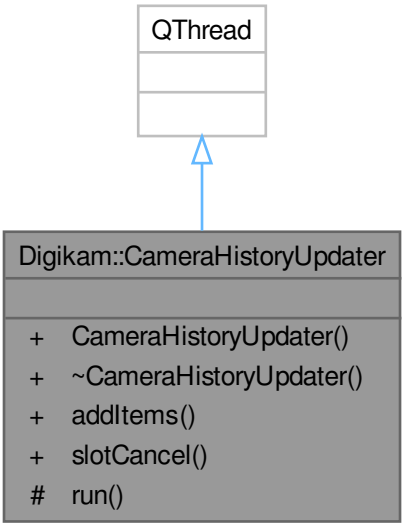
- void **signalCleared** ()
- void **signalFolderChanged** (CameraFolderItem ∗)

Public Member Functions

- **CameraFolderView** (QWidget ∗const parent)
- CameraFolderItem ∗ **addFolder** (const QString &folder, const QString &subFolder, int nbItems, const QIcon &icon=QIcon::fromTheme(QLatin1String("folder")))
- void **addRootFolder** (const QString &folder, int nbItems=-1, const QIcon &icon=QIcon::from↩Theme(QLatin1String("folder")))
- void **addVirtualFolder** (const QString &name, const QIcon &icon=QIcon::fromTheme(QLatin1↩String("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

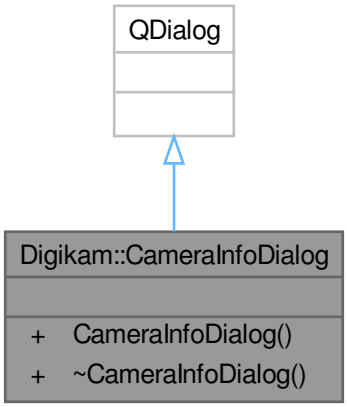
- **CameraHistoryUpdater** (QWidget ∗const parent)
- void **addItems** (const QByteArray &id, CHUpdateItemMap &map)

Protected Member Functions

- void **run** ()

9.145 Digikam::CameraInfoDialog Class Reference

Inheritance diagram for Digikam::CameraInfoDialog:

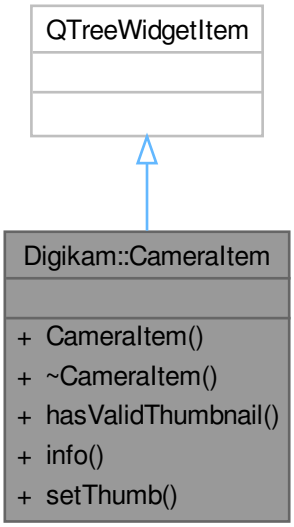


Public Member Functions

- **CameraInfoDialog** (QWidget ∗const parent, const QString &summary, const QString &manual, const QString &about)

9.146 Digikam::CameraItem Class Reference

Inheritance diagram for Digikam::CameraItem:

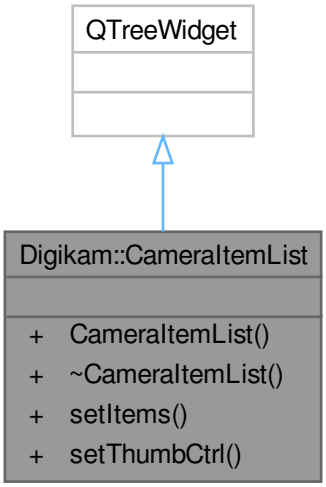


Public Member Functions

- **CameraItem** (QTreeWidget ∗const parent, const CamItemInfo &info)
- bool **hasValidThumbnail** () const
- CamItemInfo **info** () const
- void **setThumb** (const QPixmap &pix, bool hasThumb=true)

9.147 Digikam::CameraItemList Class Reference

Inheritance diagram for Digikam::CameraItemList:

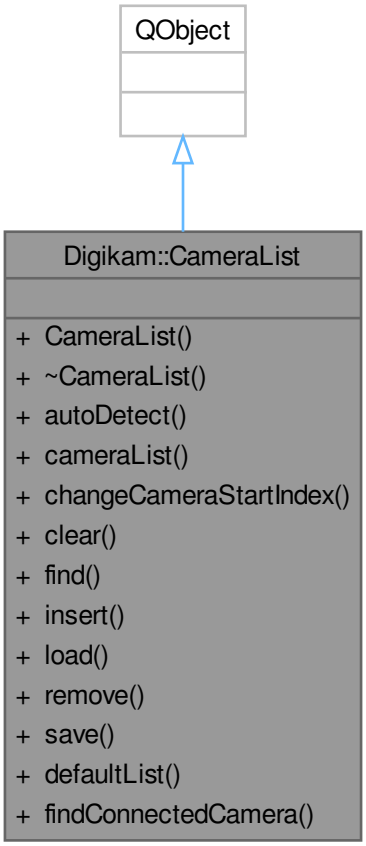


Public Member Functions

- **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

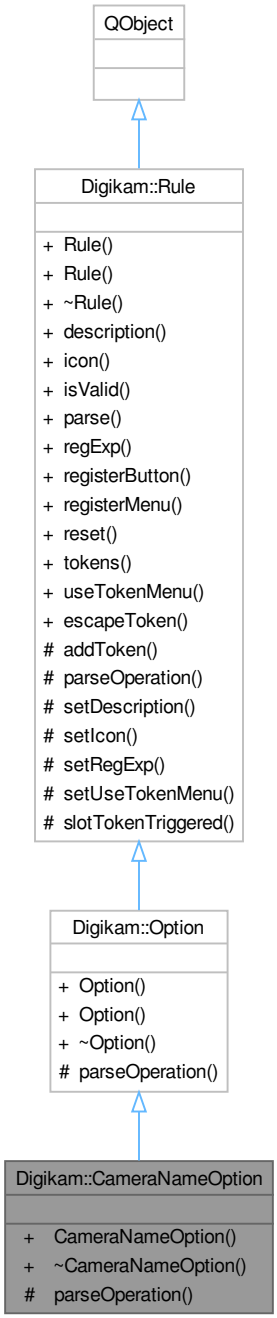
9.150 Digikam::CameraNameHelper Class Reference

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

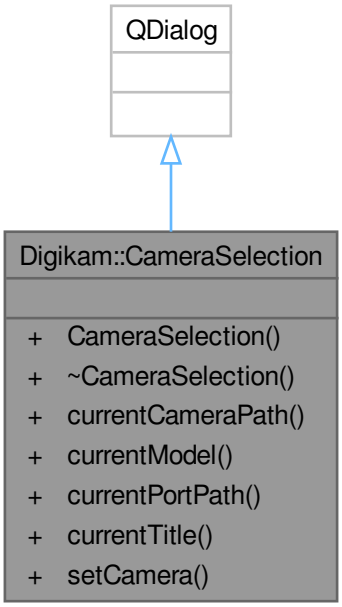
settings	contains settings
match	result of the regular expression match done in Option::parse()

Returns

Implements Digikam::Option.

9.152 Digikam::CameraSelection Class Reference

Inheritance diagram for Digikam::CameraSelection:



Signals

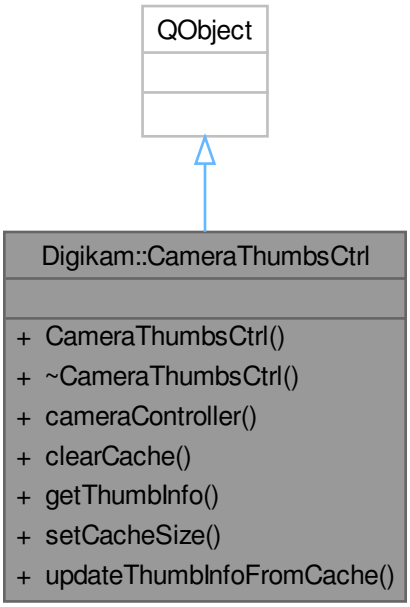
- void **signalOkClicked** (const QString &title, const QString &model, const QString &port, const QString &path)

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.
NewPicture	This is a new item from camera.

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 cam↩ ItemInfo 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 **signalAddedDropedItems** (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 **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 **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 **acceptsMouseClick** (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

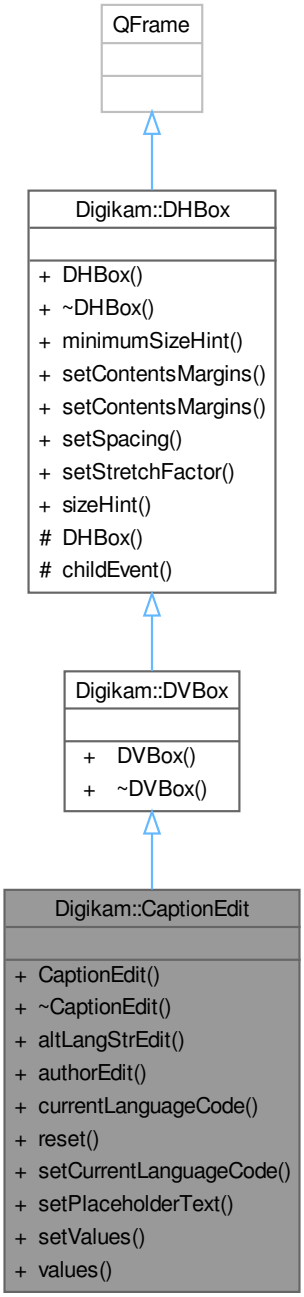
Additional Inherited Members

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

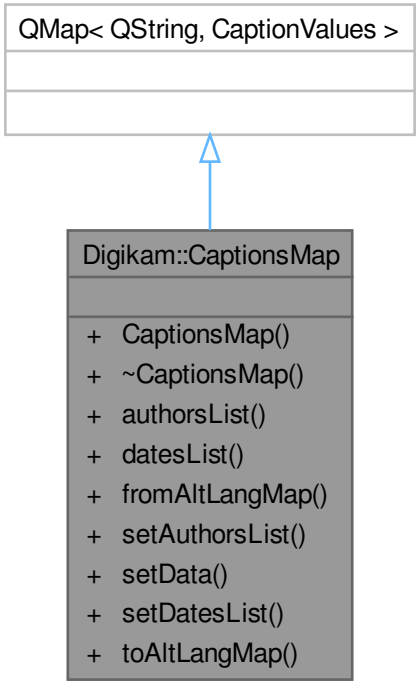
Protected Member Functions inherited from Digikam::DHBox

- **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

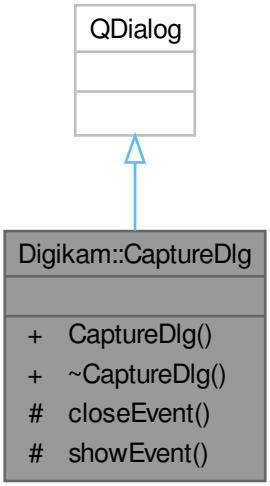
- bool **operator==** (const CaptionValues &val) const

Public Attributes

- QString **author**
- QString **caption**
- QDateTime **date**

9.161 Digikam::CaptureDlg Class Reference

Inheritance diagram for Digikam::CaptureDlg:



Public Member Functions

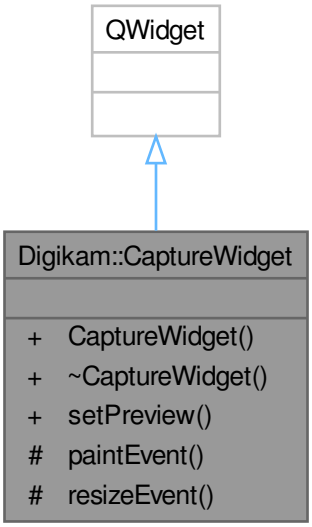
- **CaptureDlg** (QWidget *const parent, CameraController *const controller, const QString &cameraTitle)

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

  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.

9.163.1 Member Function Documentation

9.163.1.1 parseOperation()

```
QString Digikam::CaseModifier::parseOperation (
            ParseSettings & settings,
            const QRegularExpressionMatch & match )  [override], [virtual]
```

Parameters

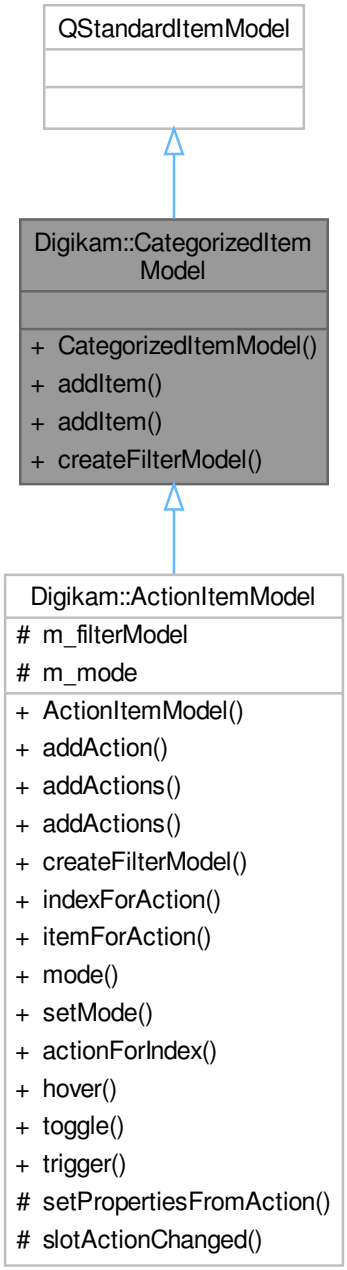
settings	contains settings
match	result of the regular expression match done in Option::parse()

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 &category↩ Sorting=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::CBContainer 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, DImgThreadedFilter *const master, const DImg &orgImage, DImg &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 **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

9.166.1 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.

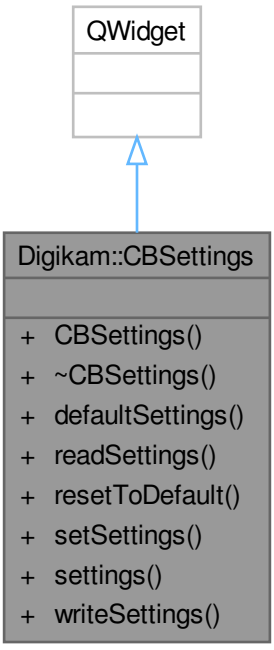
9.166.1.3 readParameters()

```
void Digikam::CBFilter::readParameters (
            const FilterAction & action )  [override], [virtual]
```

Implements Digikam::DImgThreadedFilter.

9.167 Digikam::CBSettings Class Reference

Inheritance diagram for Digikam::CBSettings:



Signals

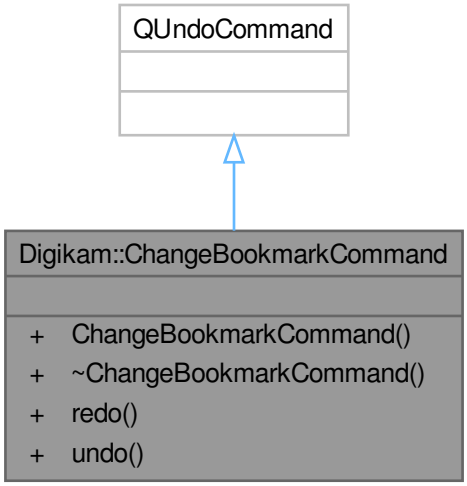
- void **signalSettingsChanged** ()

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

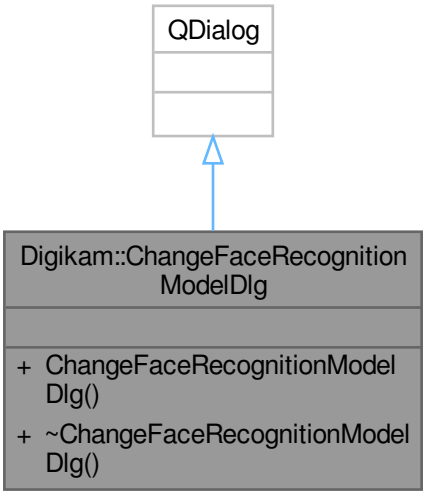
- enum **BookmarkData** { **Url** = 0 , **Title** , **Desc** }

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 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** ()

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()

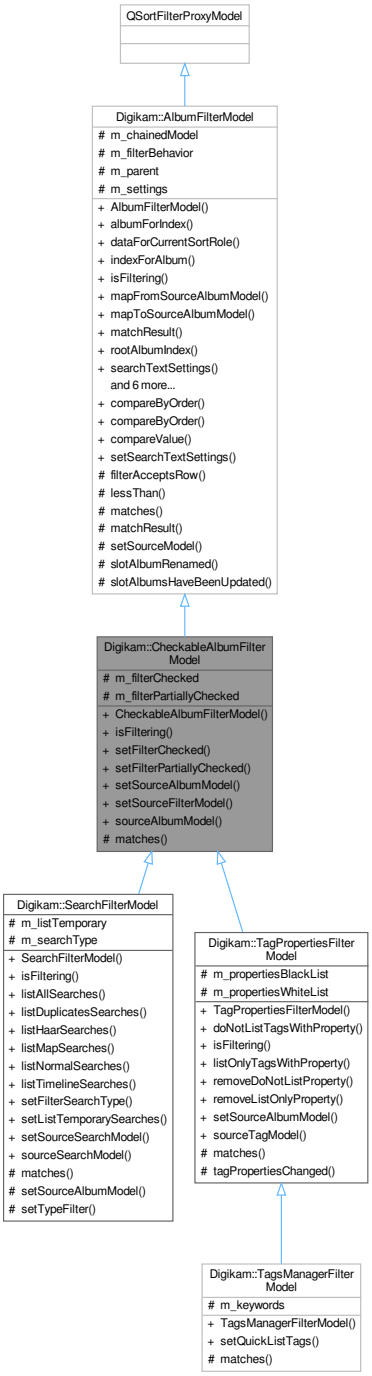
```
void Digikam::CharcoalFilter::readParameters (
            const FilterAction & action )  [override], [virtual]
```

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

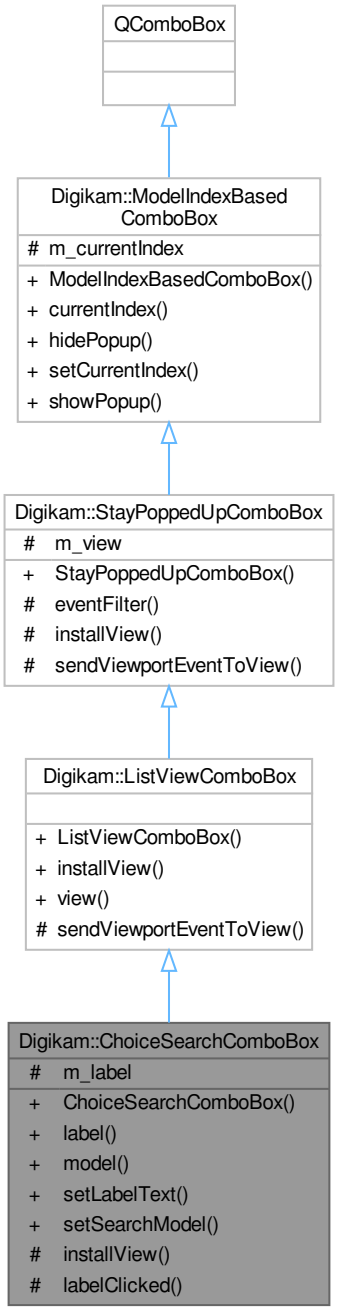
album	the album to tell if it matches the filter criteria or not.

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.

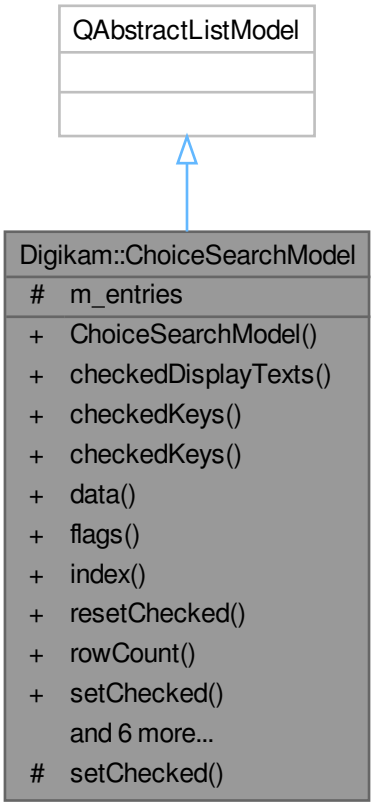
9.172.2.2 setSearchModel()

```
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**

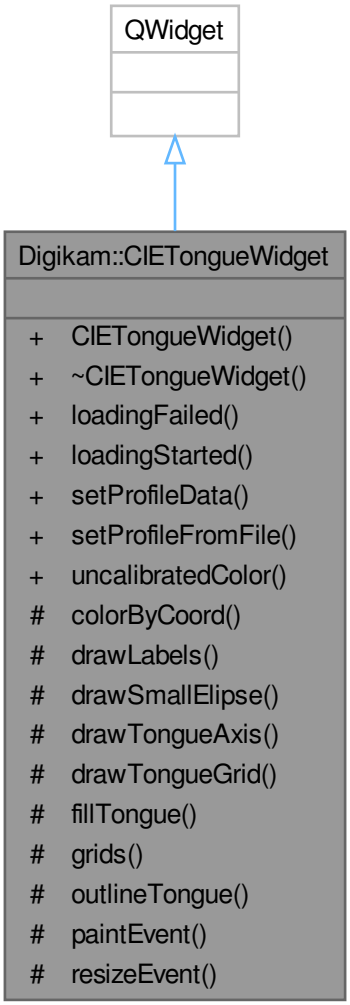
9.174.1 Member Function Documentation

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

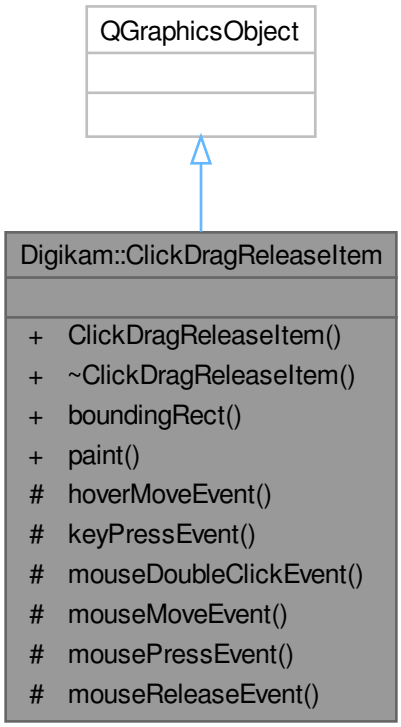
- **CIETongueWidget** (int w, int h, QWidget ∗const parent=nullptr, cmsHPROFILE hMonitor=nullptr)
- void **loadingFailed** ()
- void **loadingStarted** ()
- bool **setProfileData** (const QByteArray &profileData=QByteArray())
- bool **setProfileFromFile** (const QUrl &file=QUrl())
- void **uncalibratedColor** ()

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: mousePress, 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

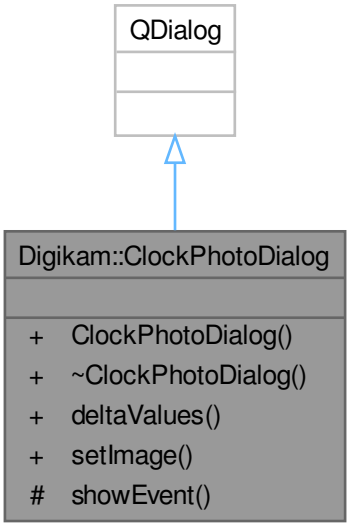
9.176.1.1 started

```
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 Album (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. Album is the source album.
Copied	Images have been copied. This is extra information; an Added changeset is guaranteed to be sent subsequently. Album 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 > &assume↩ Deleted, 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 technical↩ Description, 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.

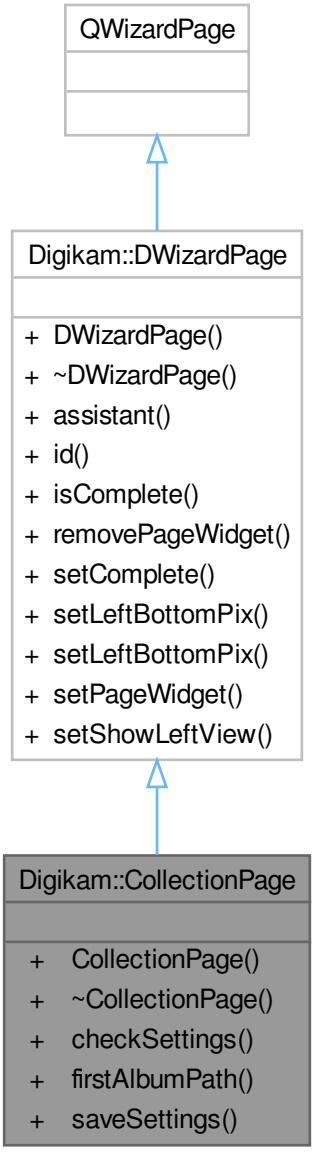
9.181.2.17 setWatchDisabled()

```
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

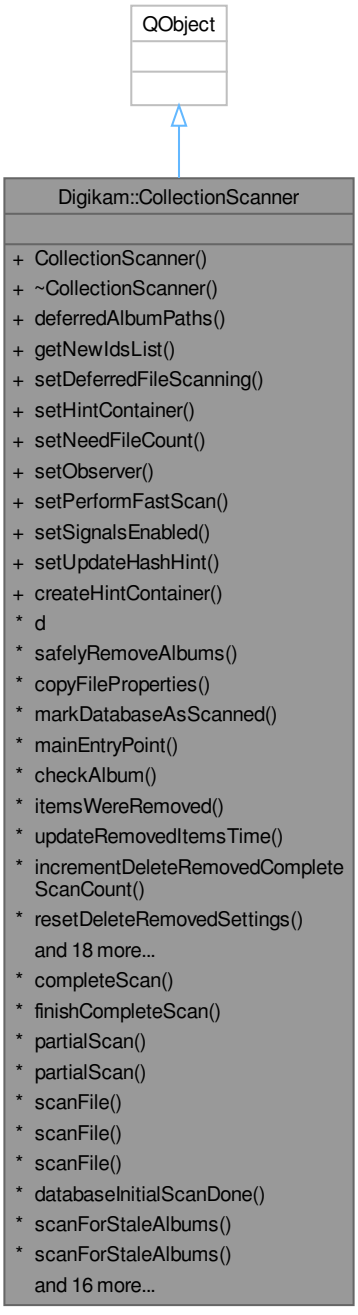
- **CollectionPage** (QWizard ∗const dlg)
- bool **checkSettings** ()
- QString **firstAlbumPath** () const
- void **saveSettings** ()

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]

```
qlonglong 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]

```
qlonglong 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

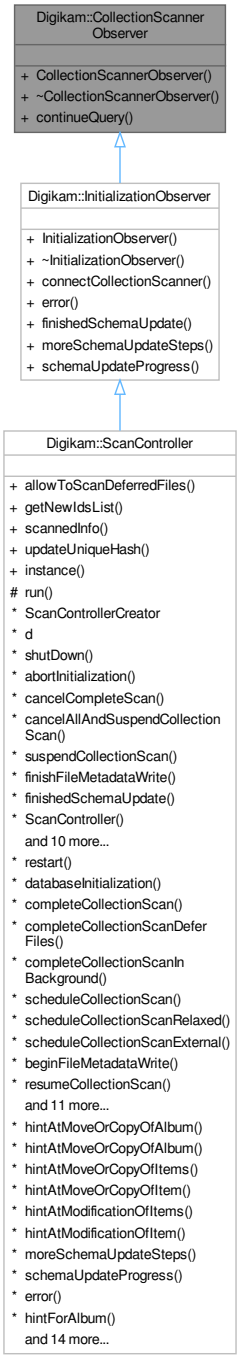
Public Member Functions

- **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:

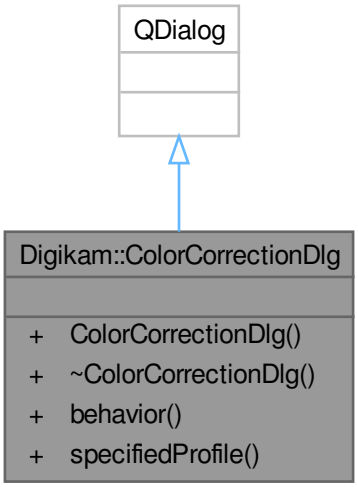


Public Member Functions

- 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.

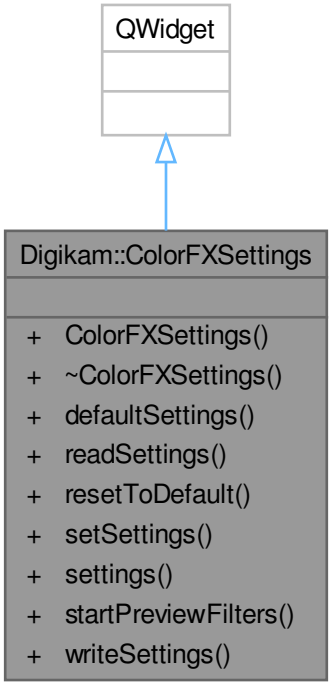
9.188.1.3 readParameters()

```
void Digikam::ColorFXFilter::readParameters (
            const FilterAction & action )  [override], [virtual]
```

Implements Digikam::DImgThreadedFilter.

9.189 Digikam::ColorFXSettings Class Reference

Inheritance diagram for Digikam::ColorFXSettings:



Signals

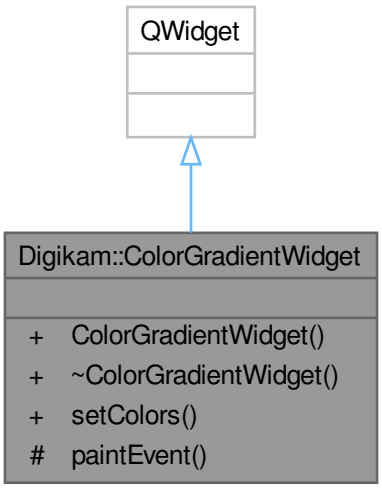
- void **signalSettingsChanged** ()

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

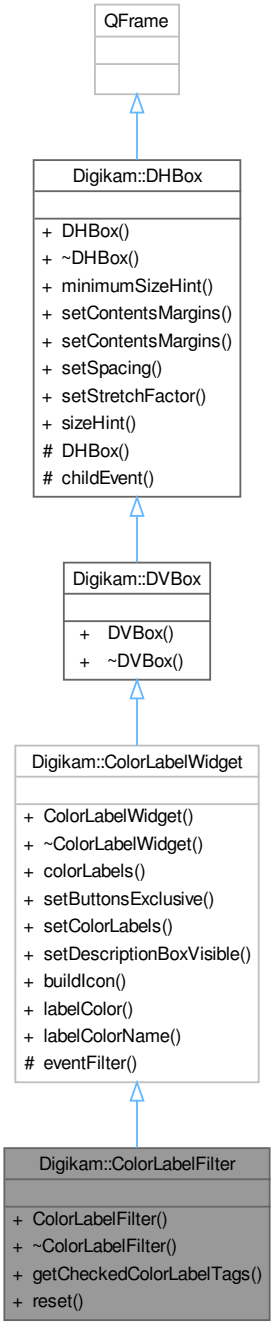
- **ColorGradientWidget** (Qt::Orientation orientation, int size, QWidget ∗const parent=nullptr)
- void **setColors** (const QColor &col1, const QColor &col2)

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 **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::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

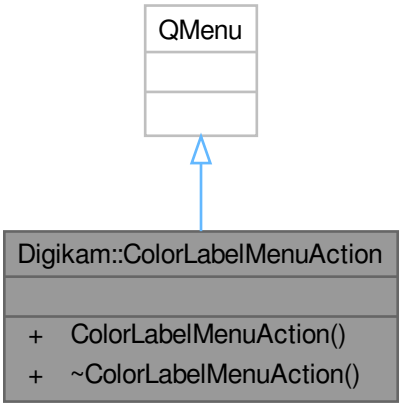
- bool **eventFilter** (QObject ∗obj, QEvent ∗ev) override

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

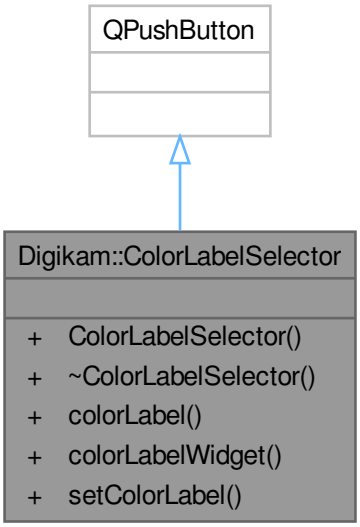
- void **signalColorLabelChanged** (int)

Public Member Functions

- **ColorLabelMenuAction** (QMenu ∗const parent=nullptr)

9.193 Digikam::ColorLabelSelector Class Reference

Inheritance diagram for Digikam::ColorLabelSelector:



Signals

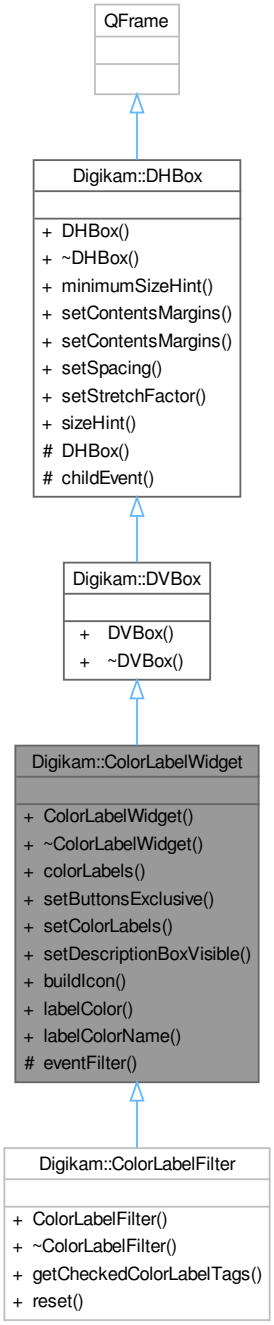
- void **signalColorLabelChanged** (int)

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.

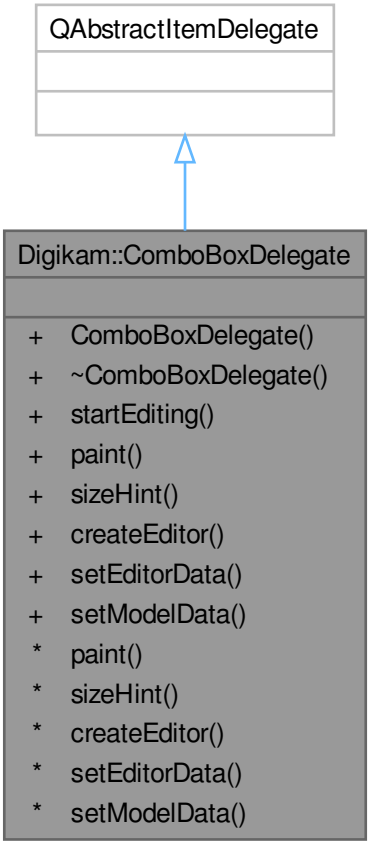
9.194.1.2 setColorLabels()

```
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

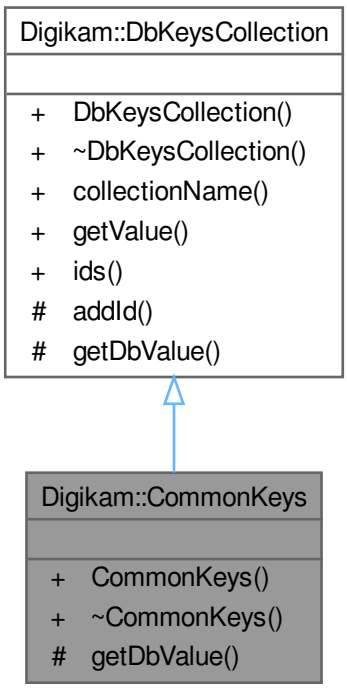
- bool **isNull** () const

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

key	the key representing the value in the database
settings	the ParseSettings object holding all relevant information about the image.

Returns

the value of the given database key

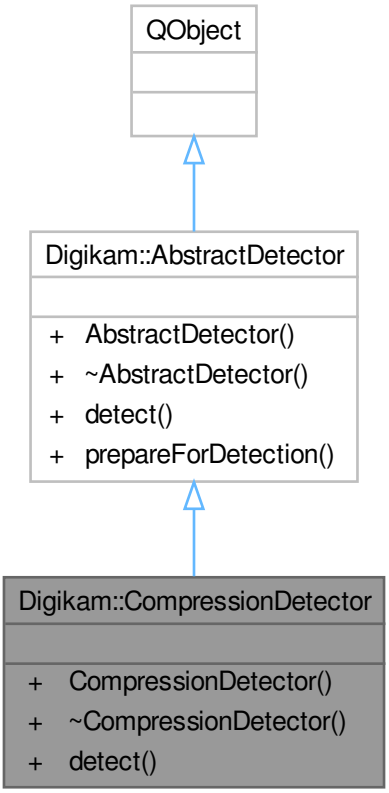
See also

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 **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

9.200.1 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()

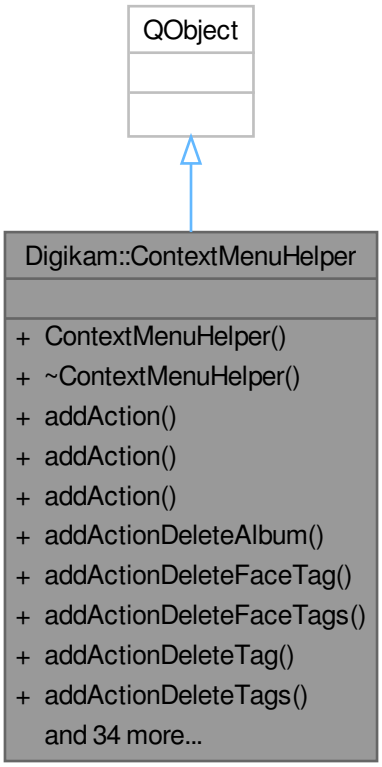
```
void Digikam::ContentAwareFilter::readParameters (
            const FilterAction & action )  [override], [virtual]
```

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 > **imageIds**

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, PAlbum *const album)
- void **addActionDeleteFaceTag** (TagModificationHelper *const helper, TAlbum *const tag)

  Add action to delete tags from people sidebar.
- void **addActionDeleteFaceTags** (TagModificationHelper *const helper, const QList< TAlbum * > &tags)
- void **addActionDeleteTag** (TagModificationHelper *const helper, TAlbum *const tag)
- void **addActionDeleteTags** (TagModificationHelper *const helper, const QList< TAlbum * > &tags)
- void **addActionEditAlbum** (AlbumModificationHelper *const helper, PAlbum *const album)
- void **addActionEditTag** (TagModificationHelper *const helper, TAlbum *const tag)
- void addActionNewAlbum (AlbumModificationHelper *const helper, PAlbum *const parentAlbum=nullptr)

  Add actions to add, remove or edit a tag.
- void addActionNewTag (TagModificationHelper *const helper, TAlbum *const parentTag=nullptr)

  Add actions to add, remove or edit a tag.
- void **addActionRenameAlbum** (AlbumModificationHelper *const helper, PAlbum *const album)
- void **addActionResetAlbumIcon** (AlbumModificationHelper *const helper, PAlbum *const album)
- void **addActionTagsToFaceTags** (TagModificationHelper *const helper, const QList< TAlbum * > &tags)
- void **addActionTagToFaceTag** (TagModificationHelper *const helper, TAlbum *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 adressbook 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

parent	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

name	the name of the action in the actionCollection
addDisabled	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

action	the action to add
addDisabled	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

action	the action to add
recv	the receiver of the triggered action
slot	the slot to connect the triggered action to
addDisabled	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

ids	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

ids	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 recv,
            const char *const slot )
```

Parameters

recv	the receiver of the triggered action
slot	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

ids	the list of selected items
lightTable	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

ids	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

ids	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

selectedItems	the list of selected items

9.201.3.16 addStandardActionCopy()

```
void Digikam::ContextMenuHelper::addStandardActionCopy (
            QObject *const recv,
            const char *const slot )
```

Parameters

recv	the receiver of the triggered action
slot	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

recv	the receiver of the triggered action
slot	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

recv	the receiver of the triggered action
slot	the slot to connect the triggered action to
quantity	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

recv	the receiver of the triggered action
slot	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

ids	the selected items in the current view
album	the current album the AlbumIconView is displaying

9.201.3.22 addSubMenu()

```
void Digikam::ContextMenuHelper::addSubMenu (
            QMenu * subMenu )
```

Parameters

subMenu	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

pos	position of the triggered action in the registered menu
at	the action that should be at the position pos

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.

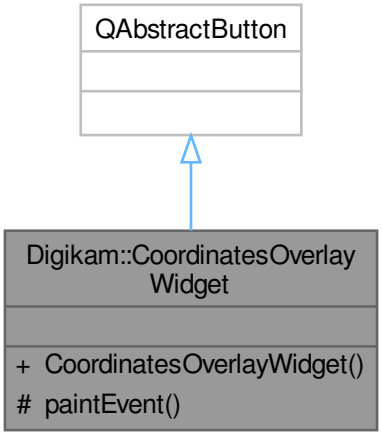
9.201.3.25 setItemFilterModel()

```
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

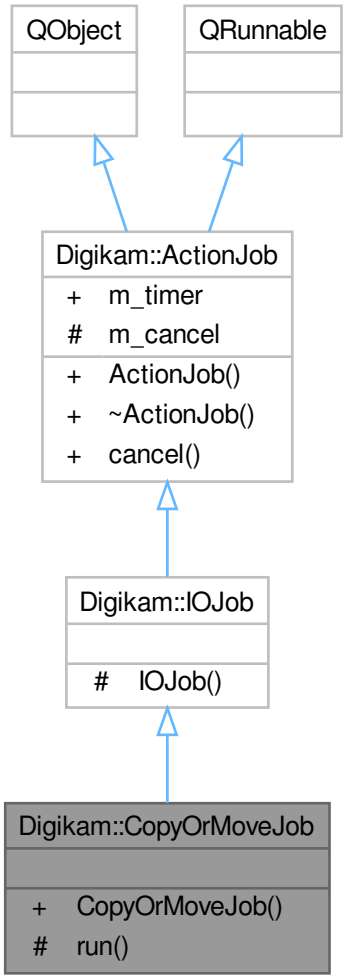
- **CoordinatesOverlayWidget** (QWidget *const parent=nullptr)

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 **signalOneProccessed** (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 (qlonglong 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** (qlonglong subjectId, qlonglong objectId, DatabaseRelation::Type type)

  Adds an image relation entry.
- void **addImageRelations** (const QList< qlonglong > &subjectIds, const QList< qlonglong > &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 (qlonglong imageId, int tagId, const QString &property, const QString &value)

  Adds a tag property.
- qlonglong addItem (int albumID, const QString &name, DatabaseItem::Status status, DatabaseItem::↩ Category category, const QDateTime &modificationDate, qlonglong fileSize, const QString &uniqueHash) const

  Put a new item in the database or replace an existing one.
- void addItemInformation (qlonglong imageID, const QVariantList &infos, DatabaseFields::ItemInformation fields=DatabaseFields::ItemInformationAll)

  Add (or replace) the ItemInformation of the specified item.
- void addItemPosition (qlonglong 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 (qlonglong 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, qlonglong 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, Database↩ Fields::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 dstId)

  Copies all entries in the ImageProperties table.
- void **copyImageTags** (qlonglong srcId, qlonglong dstId)

  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 QDate↩ Time &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 metadataFields=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 getImagesFields (qlonglong imageID, DatabaseFields::Images imagesFields) const

  Returns the requested fields from the Images table.
- QVector< QList< qlonglong > > **getImagesRelatedFrom** (const QList< qlonglong > &subjectIds, DatabaseRelation::Type type=DatabaseRelation::UndefinedType) const
- QList< qlonglong > getImagesRelatedFrom (qlonglong subjectId, DatabaseRelation::Type type=DatabaseRelation::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 > > **getImagesRelatingTo** (const QList< qlonglong > &objectIds, DatabaseRelation::Type type=DatabaseRelation::UndefinedType) const
- QList< qlonglong > getImagesRelatingTo (qlonglong objectId, DatabaseRelation::Type type=DatabaseRelation::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 > **getImagesWithImageTagProperty** (int tagId, const QString &property) const

  Returns all image ids that are associated to the tag with the given property.
- QList< qlonglong > **getImagesWithProperty** (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=Database↩ Fields::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=Database↩ Fields::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 *audioFilter↩ String)

  Returns the user-configurable filter settings.
- void **getUserIgnoreDirectoryFilterSettings** (QString *ignoreDirectoryFilterString)

- QVariantList getVideoMetadata (qlonglong imageID, DatabaseFields::VideoMetadata metadataFields=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** (qlonglong 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** (qlonglong imageId, const QString &history)

  Changes (adds or updates) the image history.
- void **setItemManualOrder** (qlonglong imageID, qlonglong value)

  Updates the manualOrder field for the item.
- void **setItemModificationDate** (qlonglong imageID, const QDateTime &modificationDate)

  Updates the modification date field for the item.
- void setItemStatus (qlonglong 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, qlonglong 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 (qlonglong imageID, DatabaseItem::Category category, const QDateTime &modification↩ Date, qlonglong 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::ItemInformation 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< qlonglong > > **getRelatedImages** (QList< qlonglong > ids, bool fromOrTo, Database↩ Relation::Type type, bool boolean) const
- QList< qlonglong > **getRelatedImages** (qlonglong 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

albumRootId	id of the album root of the new album
relativePath	url of the album
caption	the album caption
date	the date for the album
collection	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

type	The type of the album root
identifier	The album root identifier
specificPath	The path specific to volume
label	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

albumID	the albumID of the item
name	the name of the item
tagID	the tagID for the tag

9.205.1.9 addItemTag() [2/2]

```
void Digikam::CoreDB::addItemTag (
            qlonglong imageID,
            int tagID,
            bool newTag = false )
```

Parameters

imageID	the ID of the item
tagID	the tagID for the tag
newTag	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

type	search type
name	name of the search
query	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

parentTagID	the id of the tag which will become the new tags parent
name	the name of the tag
iconKDE	the name of the icon file (this is filename which kde iconloader can load up)
iconID	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 changeImageComment()

```
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) of src and dest are identical.

Parameters

srcAlbumID	the id of the source album
dstAlbumID	the id of the destination album
srcName	the name of the source file
dstName	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

albumID	the id of the album

9.205.1.24 deleteAlbumRoot()

```
void Digikam::CoreDB::deleteAlbumRoot (
            int rootId )
```

Parameters

rootId	the id of the album root

9.205.1.25 deleteItem() [1/2]

```
void Digikam::CoreDB::deleteItem (
            int albumID,
            const QString & file )
```

Parameters

albumID	The id of the album.
file	The filename of the file to delete.

9.205.1.26 deleteItem() [2/2]

```
void Digikam::CoreDB::deleteItem (
            qlonglong imageId )
```

This method can only be used if the album of the image is null!

Parameters

imageId	The id of the image.

9.205.1.27 deleteObsoleteItem()

```
void Digikam::CoreDB::deleteObsoleteItem (
            qlonglong imageId )
```

Parameters

imageId	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

searchID	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

tagID	the id of the tag

9.205.1.31 findImageId()

```
qlonglong Digikam::CoreDB::findImageId (
            int albumID,
            const QString & name,
            DatabaseItem::Status status,
            DatabaseItem::Category category,
            qlonglong fileSize,
            const QString & uniqueHash ) const
```

Parameters

albumID	the albumID of the item (-1 means null)
name	the name of the item
status	the status of the item
category	the category of the item
fileSize	the file size
uniqueHash	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,
            qlonglong 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

albumRootId	id of the album root of the album
relativePath	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

albumID	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

albumRootId	id of the album root of the album
relativePath	The relative path for which you want the albumID relative to the album root
create	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

albumID	the id of the album to calculate

Returns

the date.

9.205.1.37 getAlbumLowestDate()

```
QDate Digikam::CoreDB::getAlbumLowestDate (
            int albumID ) const
```

Parameters

albumID	the id of the album to calculate

Returns

the date.

9.205.1.38 getAlbumModificationDate()

```
QDateTime Digikam::CoreDB::getAlbumModificationDate (
            int albumID ) const
```

Parameters

albumID	the id of the album

9.205.1.39 getAlbumModificationMap()

```
QMap< QString, QDateTime > Digikam::CoreDB::getAlbumModificationMap (
            int albumRootId ) const
```

Parameters

albumRootId	id of the album root of the album

9.205.1.40 getAlbumRelativePath()

```
QString Digikam::CoreDB::getAlbumRelativePath (
            int albumID ) const
```

Parameters

albumID	the id of the album

Returns

the url of the album

9.205.1.41 getAlbumRootId()

```
int Digikam::CoreDB::getAlbumRootId (
            int albumID ) const
```

Parameters

albumID	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 getDirtyOrMissingFaceImageUrls()

```
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

albumID	the albumID of the item
name	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

status	The status.

Returns

The ids of the images that have the given status.

9.205.1.51 getImageIds() [2/4]

```
QList< qlonglong > Digikam::CoreDB::getImageIds (
            DatabaseItem::Status status,
            DatabaseItem::Category category ) const
```

Parameters

status	The status.
category	The category.

Returns

The ids of the images that have the given status.

9.205.1.52 getImageIds() [3/4]

```
QList< qlonglong > Digikam::CoreDB::getImageIds (
            int albumID,
            const QString & name,
            DatabaseItem::Status status ) const
```

Parameters

albumID	the albumID of the item (-1 means NULL)
name	the name of the item
status	the status of the item

Returns

the ImageIds for the item, or an empty list if there are no matching entries.

9.205.1.53 getImageIds() [4/4]

```
QList< qlonglong > Digikam::CoreDB::getImageIds (
            int albumID,
            DatabaseItem::Status status,
            bool scanned = true ) const
```

Parameters

albumID	the albumID of the item (-1 means NULL)
status	the status of the item
scanned	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

imageID	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

imageIDList	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

albumID	The albumId in which we search the item.
fileName	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

albumID	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

albumID	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

tagID	the id of the tag
recursive	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

imageID	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

albumID	The albumID for which you want all items.
recursive	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

imageID	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

imageID	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

albumID	the id of the album
order	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

tagID	the id of the tag
recursive	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

albumID	the id of the album

9.205.1.76 getNumberOfItemsInAlbum()

```
int Digikam::CoreDB::getNumberOfItemsInAlbum (
            int albumID ) const
```

Parameters

albumID	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

keyword	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

imageIDList	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

srcAlbumID	the id of the source album
dstAlbumID	the id of the destination album
srcName	the name of the source file
dstName	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

imageID	the ID of the item
currentTagIds	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

itemIDs	a list of item IDs to be marked
albumIDs	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

albumID	The id of the album
ids_forInformation	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

itemIDs	a list of item IDs to be marked
albumIDs	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

imageID	the ID of the item
tagID	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

albumID	the id of the album
caption	the new caption for the album

9.205.1.103 setAlbumCategory()

```
void Digikam::CoreDB::setAlbumCategory (
            int albumID,
            const QString & category )
```

Parameters

albumID	the id of the album
category	the new category for the album

9.205.1.104 setAlbumDate()

```
void Digikam::CoreDB::setAlbumDate (
            int albumID,
            const QDate & date )
```

Parameters

albumID	the id of the album
date	the date for the album

9.205.1.105 setAlbumIcon()

```
void Digikam::CoreDB::setAlbumIcon (
            int albumID,
            qlonglong iconID )
```

Parameters

albumID	the id of the album
iconID	the id of the icon file

9.205.1.106 setAlbumModificationDate()

```
void Digikam::CoreDB::setAlbumModificationDate (
            int albumID,
            const QDateTime & modificationDate )
```

Parameters

albumID	the id of the album
modificationDate	the modification date time for the album

9.205.1.107 setAlbumRootLabel()

```
void Digikam::CoreDB::setAlbumRootLabel (
            int rootId,
            const QString & newLabel )
```

Parameters

rootId	the id of the album root
newLabel	new label for the album root

9.205.1.108 setAlbumRootPath()

```
void Digikam::CoreDB::setAlbumRootPath (
            int rootId,
            const QString & newPath )
```

Parameters

rootId	the id of the album root
newPath	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

imageID	The imageID of the image
comment	The comment string
type	The type of the comment
language	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.
author	Optional information about the author who wrote the comment. If not supported by the source, pass a null string.
date	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

keyword	The keyword
value	The value

9.205.1.114 setTagIcon()

```
void Digikam::CoreDB::setTagIcon (
            int tagID,
            const QString & iconKDE,
            qlonglong iconID )
```

Parameters

tagID	the id of the tag
iconKDE	the filename for the kde icon file
iconID	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

tagID	the id of the tag
name	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

tagID	the id of the tag
newParentTagID	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

imageID	the image that has been modified
category	the image category that has been modified
modificationDate	the image time-stamp that has been modified
fileSize	the image file size that has been modified
uniqueHash	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

searchID	the id of the search
type	type of the search
name	name of the search
query	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 DatabaseAcccess. 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

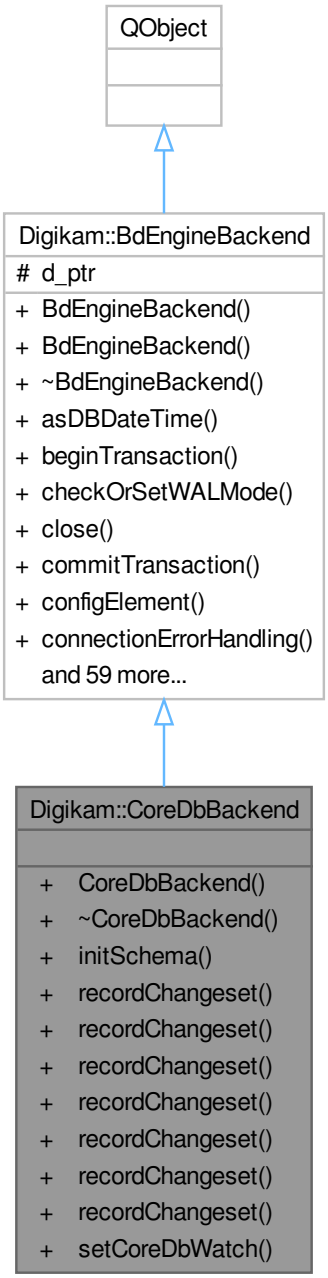
9.207.1.1 CoreDbAccessUnlock()

```
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, 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 &field↩Names, 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

9.208.1 Member Function Documentation

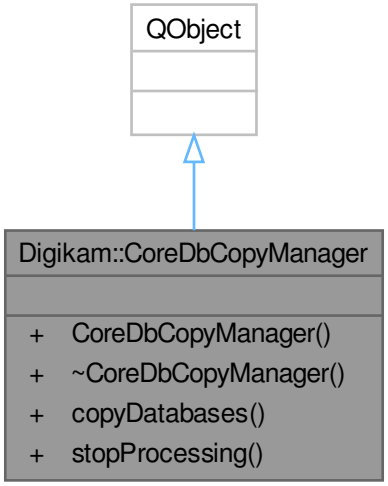
9.208.1.1 initSchema()

```
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 &to↩ DBParameters)

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 file↩ Size, 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 msecs)

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 &dbActionName)
- 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)

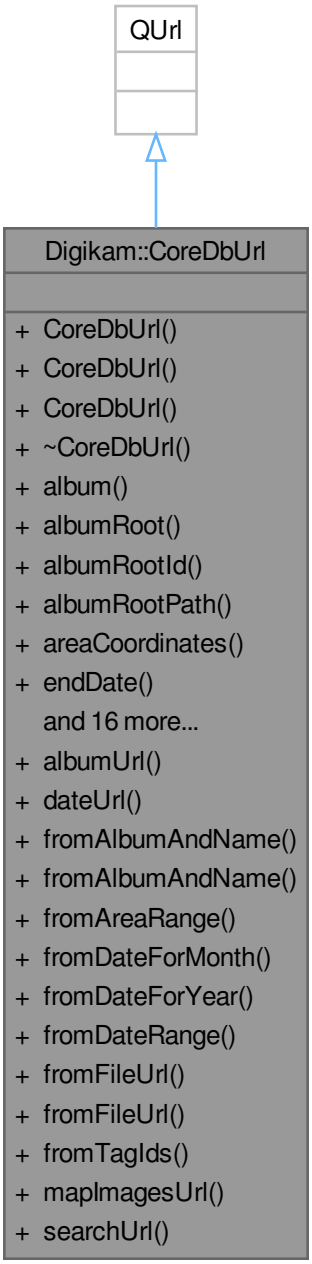
  Use an existing CoreDbAccess object, which must live as long as this object exists.

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::parameter

  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)

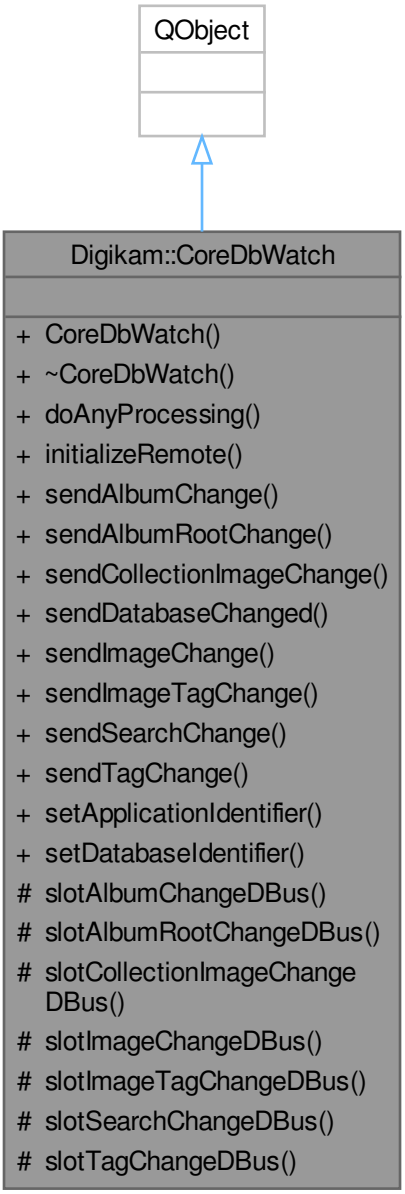
9.216.1.15 tagId()

```
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 &application↩ Identifier, const Digikam::AlbumRootChangeset &changeset)
- void **signalCollectionImageChangeDBus** (const QString &databaseIdentifier, const QString &application↩ Identifier, 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 &application↩ Identifier, 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.

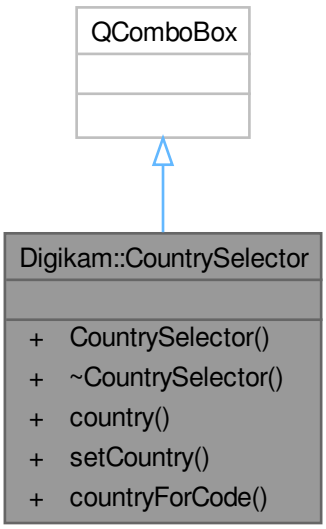
9.217.1.2 imageChange

```
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

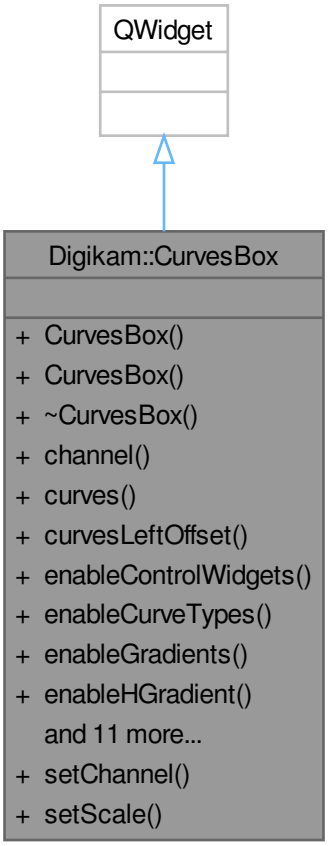
- **CountrySelector** (QWidget ∗const parent)
- bool **country** (QString &countryCode, QString &countryName) const
- void **setCountry** (const QString &countryCode)

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

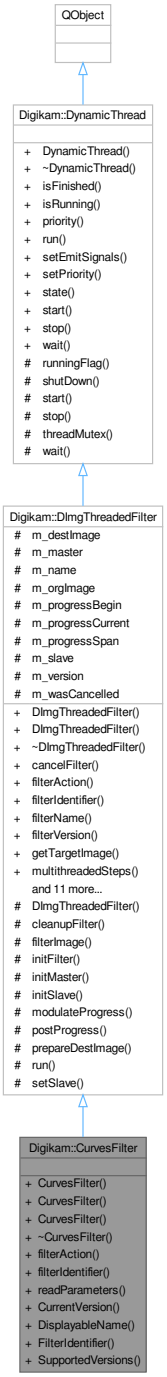
9.220.3.1 curvesType

```
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 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** ()

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.

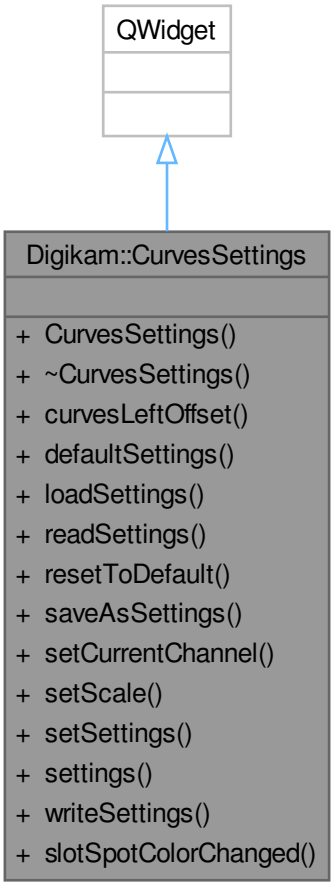
9.221.1.3 readParameters()

```
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

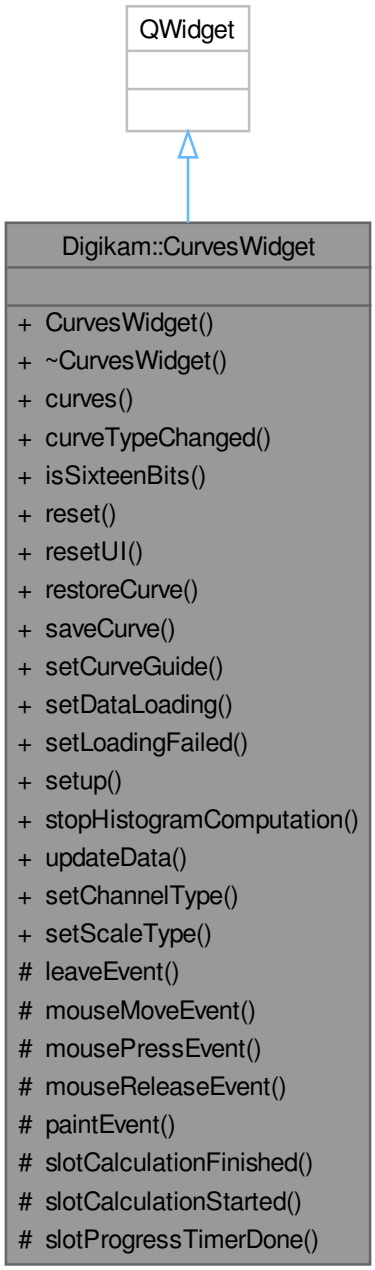
- void **signalChannelReset** (int)
- void **signalPickerChanged** (int)
- void **signalSettingsChanged** ()
- void **signalSpotColorChanged** ()

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 DColor &color)
- void **setDataLoading** ()
- void **setLoadingFailed** ()
- void **setup** (int w, int h, bool readOnly)
- void **stopHistogramComputation** ()

  Stop current histogram computations.
- void updateData (const DImg &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

group	the group to restore the curve from
prefix	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

group	the group to save the curve to
prefix	the prefix prepended to the point numbers in the config

9.223.1.3 updateData()

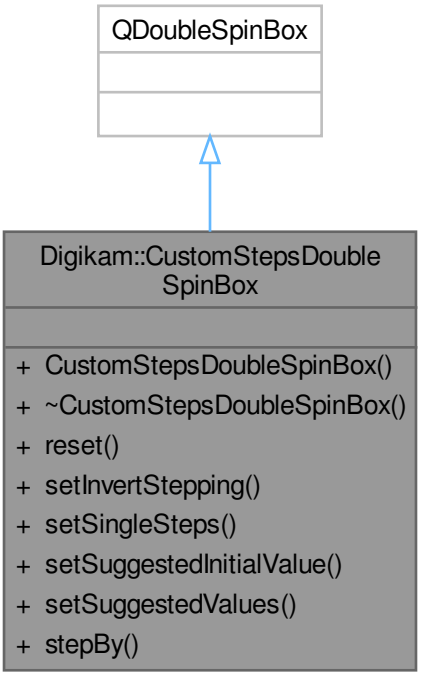
```
void Digikam::CurvesWidget::updateData (
            const DImg & img )
```

Parameters

img	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

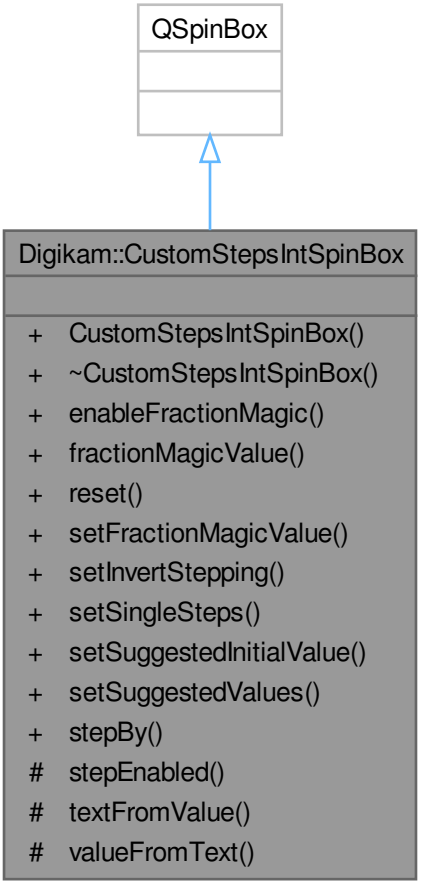
9.224.1.1 setSuggestedValues()

```
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

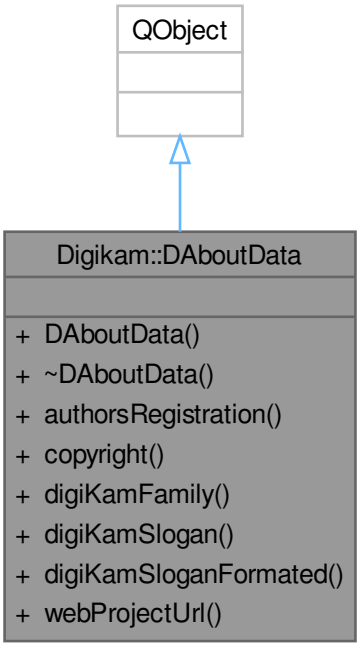
9.225.1.1 setSuggestedValues()

```
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

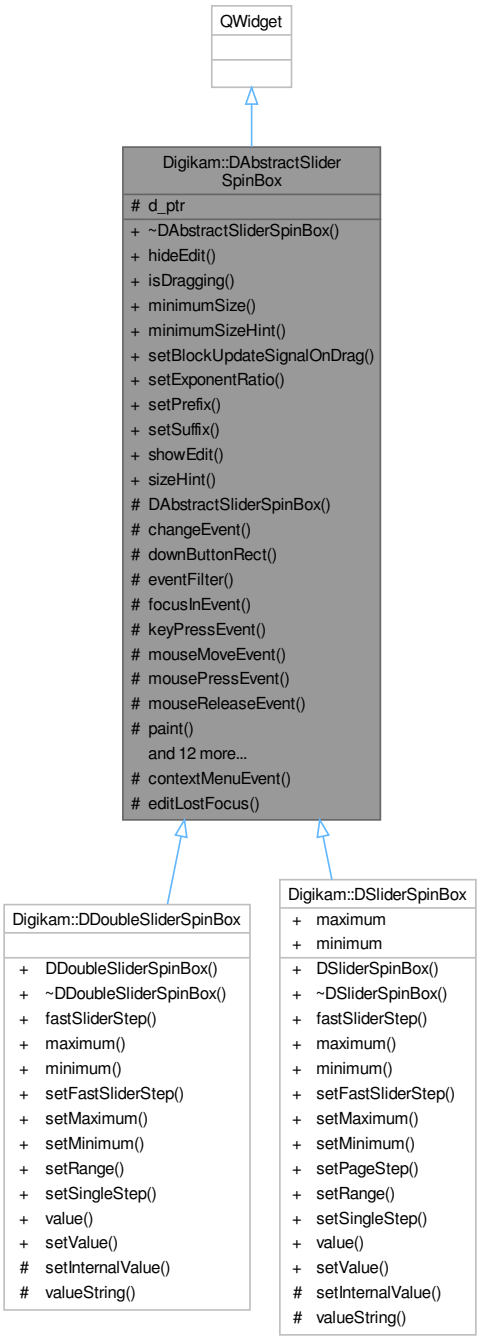
- **DAboutData** (DXmlGuiWindow ∗const parent)

Static Public Member Functions

- static void **authorsRegistration** (KAboutData &aboutData)
- static const QString **copyright** ()
- static const QString **digiKamFamily** ()
- static const QString **digiKamSlogan** ()
- static const QString **digiKamSloganFormated** ()
- 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]
```

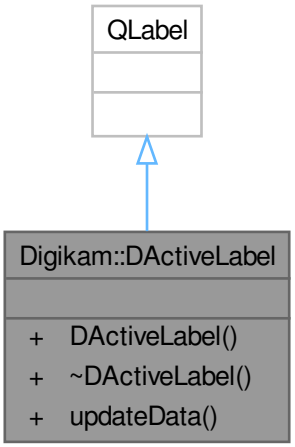
Inheriting classes should respect blockUpdateSignal so that, in specific cases, we have a performance improvement. See setIgnoreMouseMoveEvents.

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:



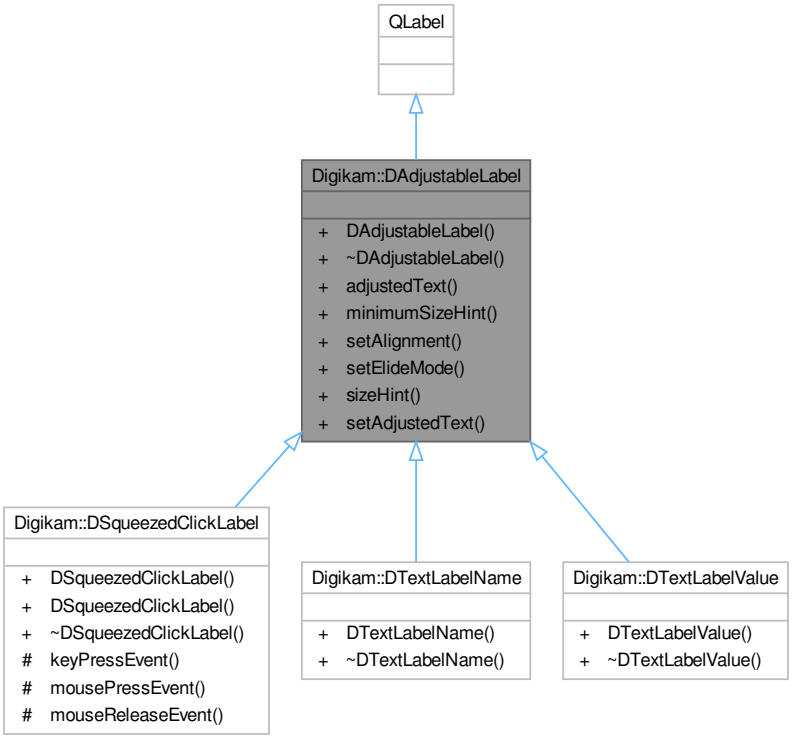
Public Member Functions

- **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::DAlbum Class Reference

A Date Album representation.

Inheritance diagram for Digikam::DAlbum:

Public Types

- enum **Range** { **Month** = 0 , **Year** }

Public Types inherited from Digikam::Album

- enum Type {
  PHYSICAL = 0 , TAG , DATE , SEARCH ,
  FACE }

Public Member Functions

- **DAlbum** (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

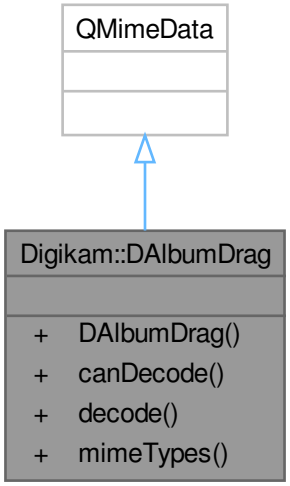
the kde url of the album

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 **mimeTypes** ()

9.231.1 Detailed Description

When an album is moved through drag'n'drop an object of this class is created.

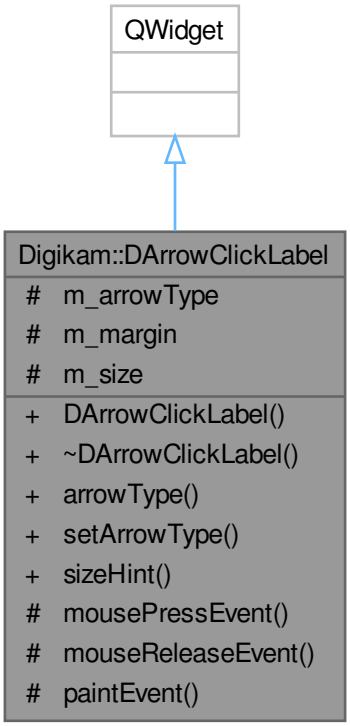
9.232 Digikam::DAlbumInfo Class Reference

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

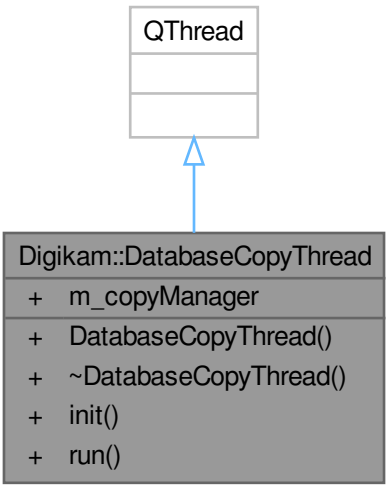
- void **mousePressEvent** (QMouseEvent ∗event) override
- void **mouseReleaseEvent** (QMouseEvent ∗event) override
- void **paintEvent** (QPaintEvent ∗event) override

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 &to↩ DatabaseSettingsWidget)
- 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

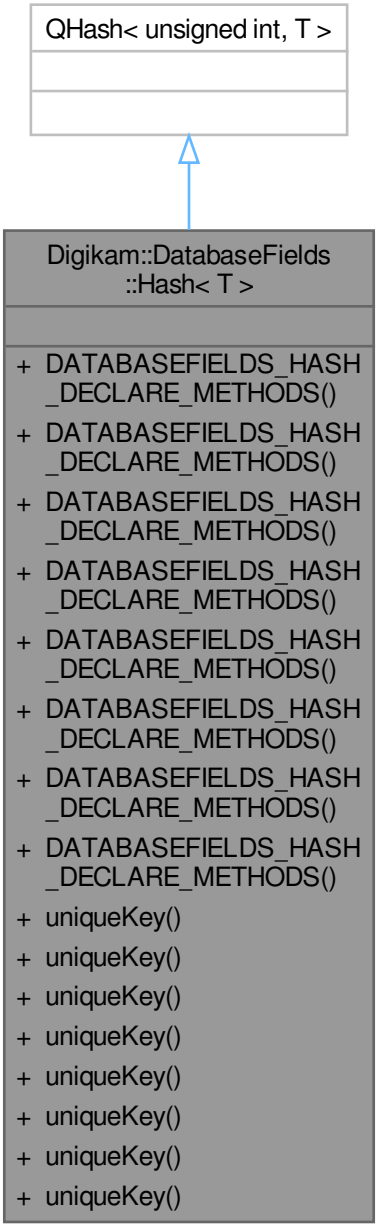
- **DatabaseFieldsEnumIteratorSetOnly** (const FieldName setValues)
- bool **atEnd** () const
- FieldName **operator*** () const
- void **operator++** ()

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

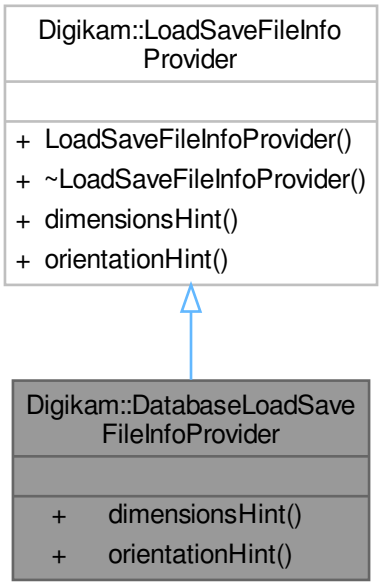
This class provides a set of all DatabaseFields enums, without resorting to a QSet.

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()

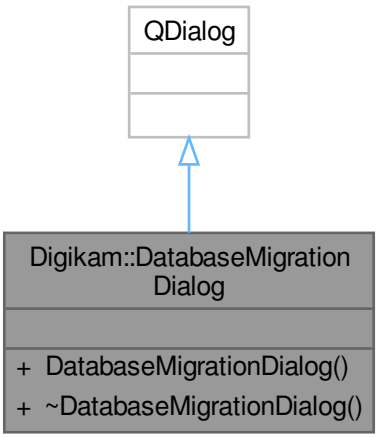
```
int Digikam::DatabaseLoadSaveFileInfoProvider::orientationHint (
            const QString & path )  [override], [virtual]
```

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:

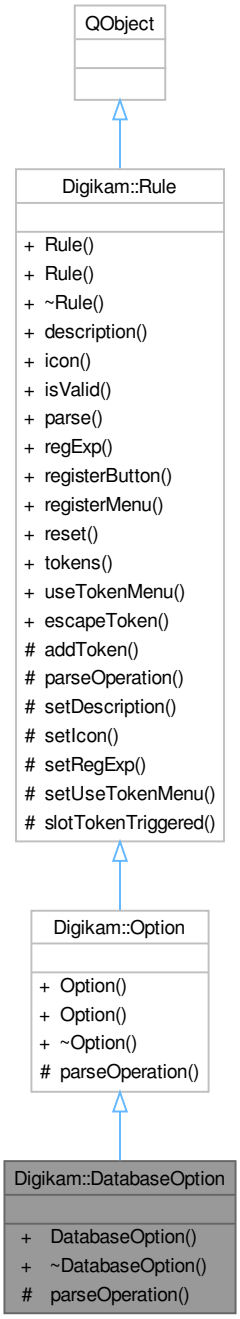


Public Member Functions

- **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

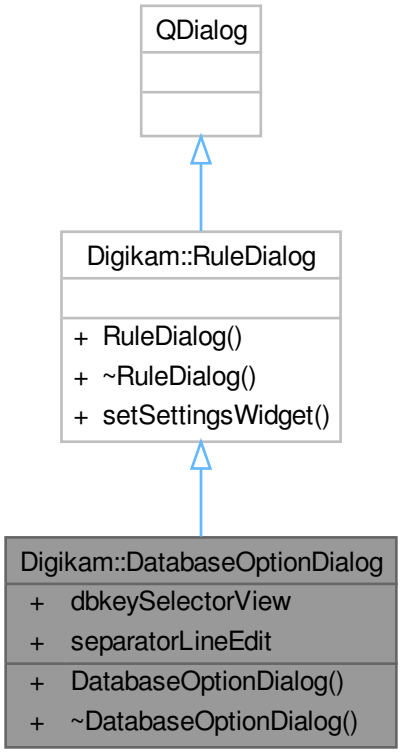
settings	contains settings
match	result of the regular expression match done in Option::parse()

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

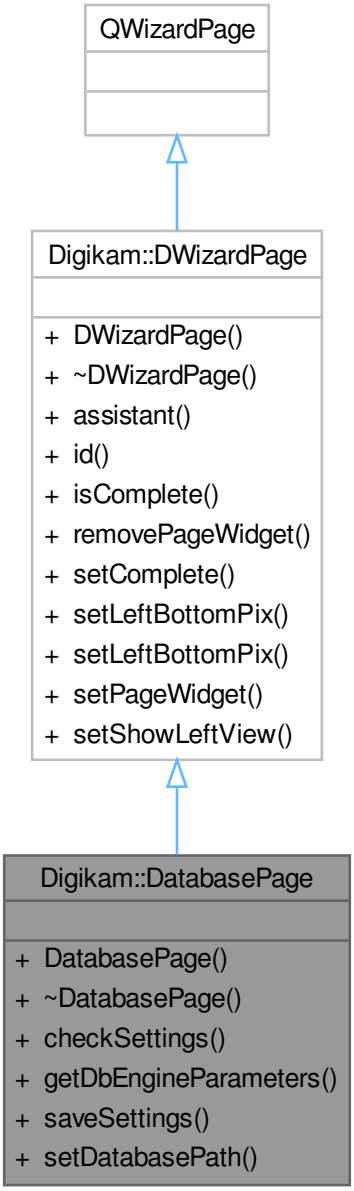
- **RuleDialog** (Rule ∗const parent)
- void **setSettingsWidget** (QWidget ∗const settingsWidget)

Public Attributes

- DbKeySelectorView ∗ **dbkeySelectorView** = nullptr
- QLineEdit ∗ **separatorLineEdit** = nullptr

9.244 Digikam::DatabasePage Class Reference

Inheritance diagram for Digikam::DatabasePage:



Public Member Functions

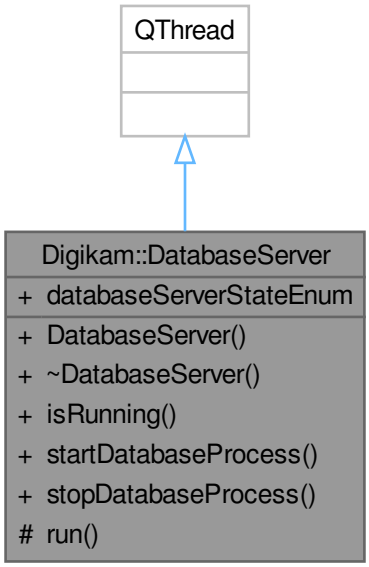
- **DatabasePage** (QWizard ∗const dlg)
- bool **checkSettings** ()
- DbEngineParameters **getDbEngineParameters** () const
- void **saveSettings** ()
- void **setDatabasePath** (const QString &path)

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::instanc
- 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

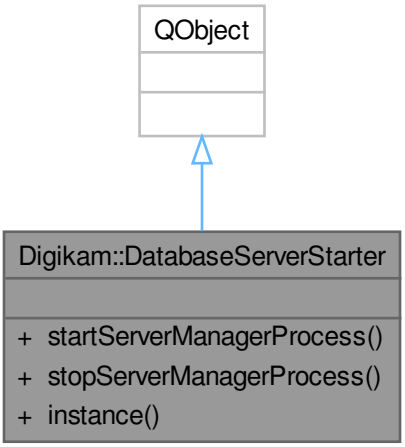
```
enum Digikam::DatabaseServerError::DatabaseServerErrorEnum
```

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

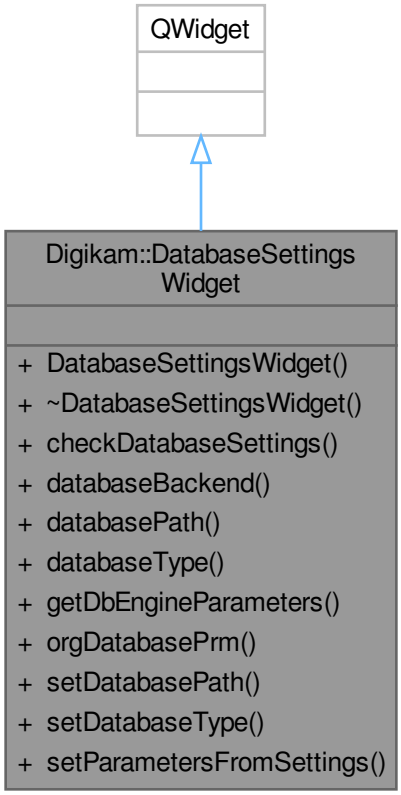
9.247.1.1 instance()

```
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

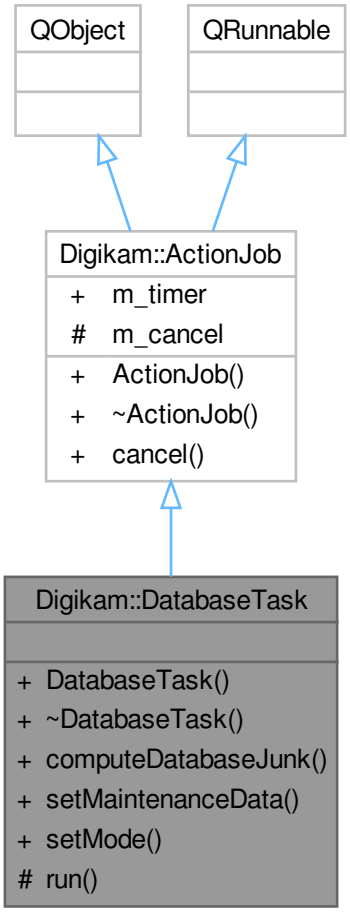
9.248.1.1 checkDatabaseSettings()

```
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.

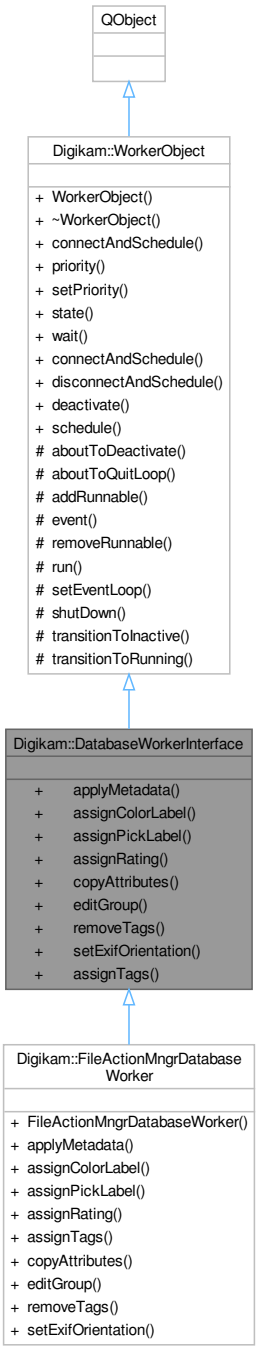
Protected Attributes inherited from Digikam::ActionJob

- 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 ()

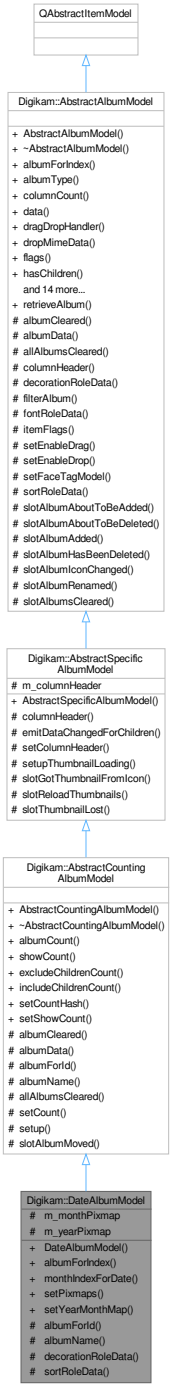
  Called from within thread's event loop to quit processing.
- void **addRunnable** (WorkerObjectRunnable ∗loop)
- bool **event** (QEvent ∗e) override
- void **removeRunnable** (WorkerObjectRunnable ∗loop)
- void **run** ()
- void **setEventLoop** (QEventLoop ∗loop)
- void shutDown ()

  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 root↩ Behavior=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

parent	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 DAlbum.

Parameters

date	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

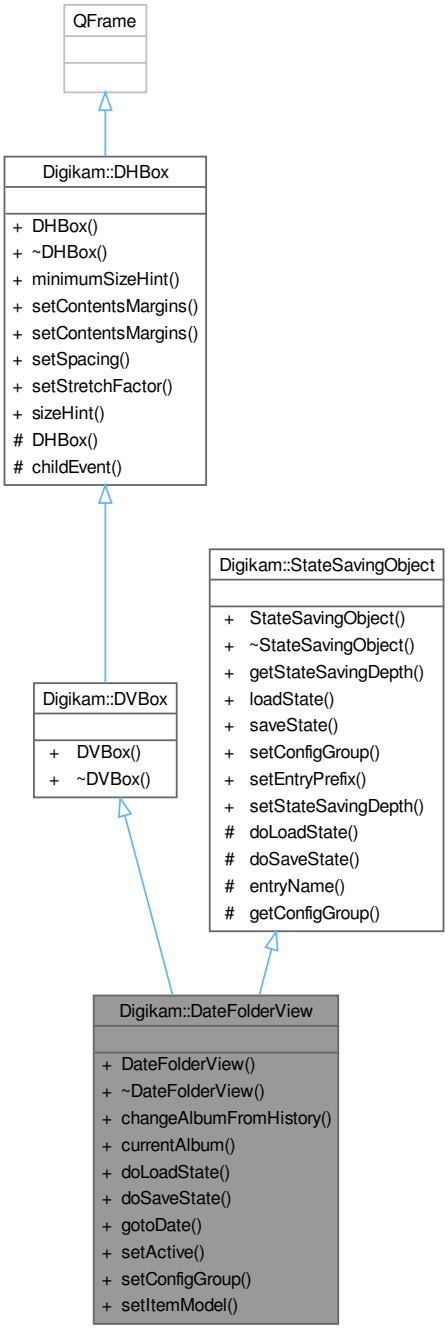
9.252.2.5 sortRoleData()

```
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< DAlbum > **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.

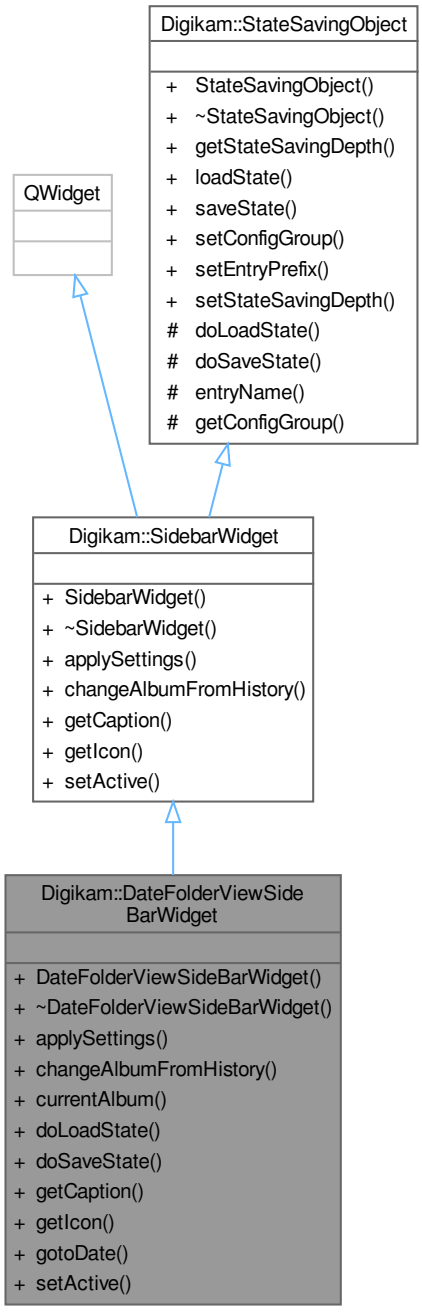
Parameters

group	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< DAlbum > **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

active	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

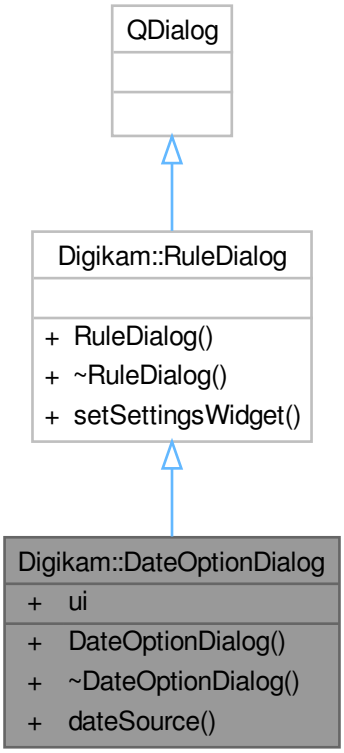
settings	contains settings
match	result of the regular expression match done in Option::parse()

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

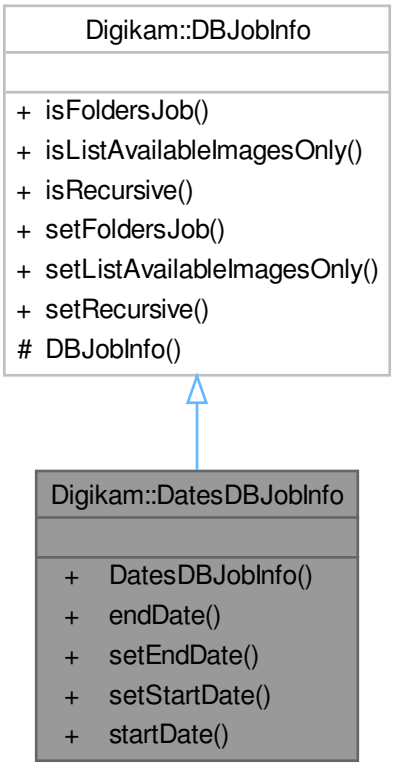
- **RuleDialog** (Rule ∗const parent)
- void **setSettingsWidget** (QWidget ∗const settingsWidget)

Public Attributes

- Ui::DateOptionDialogWidget ∗const **ui** = nullptr

9.258 Digikam::DatesDBJobInfo Class Reference

Inheritance diagram for Digikam::DatesDBJobInfo:



Public Member Functions

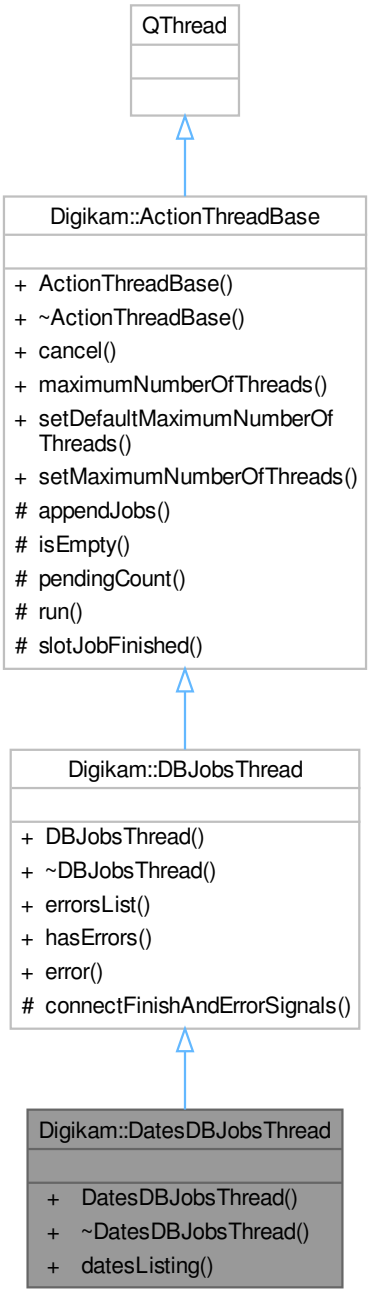
- QDate **endDate** () const
- void **setEndDate** (const QDate &date)
- void **setStartDate** (const QDate &date)
- QDate **startDate** () const

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()

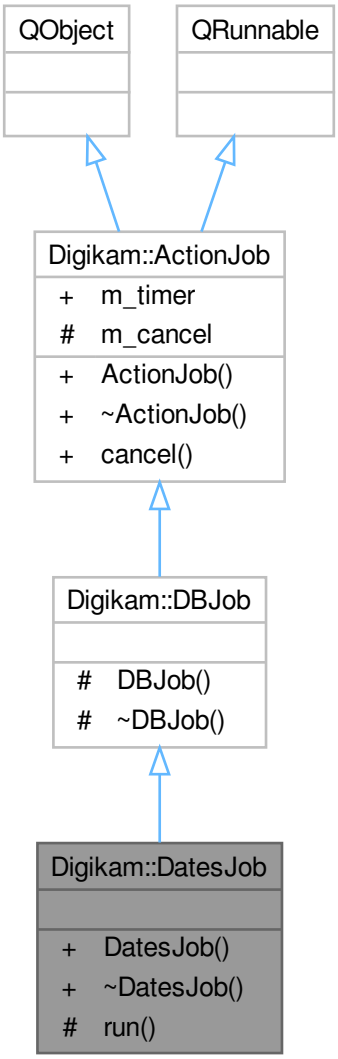
```
void Digikam::DatesDBJobsThread::datesListing (
            const DatesDBJobInfo & info )
```

Parameters

info	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.

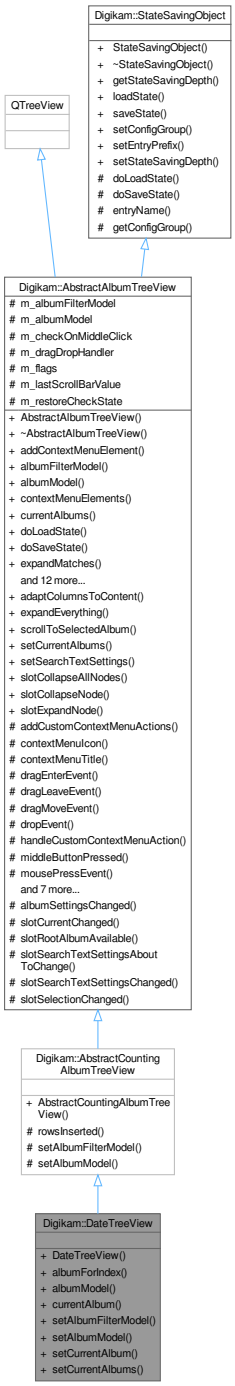
Protected Attributes inherited from Digikam::ActionJob

- 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 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.

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.

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

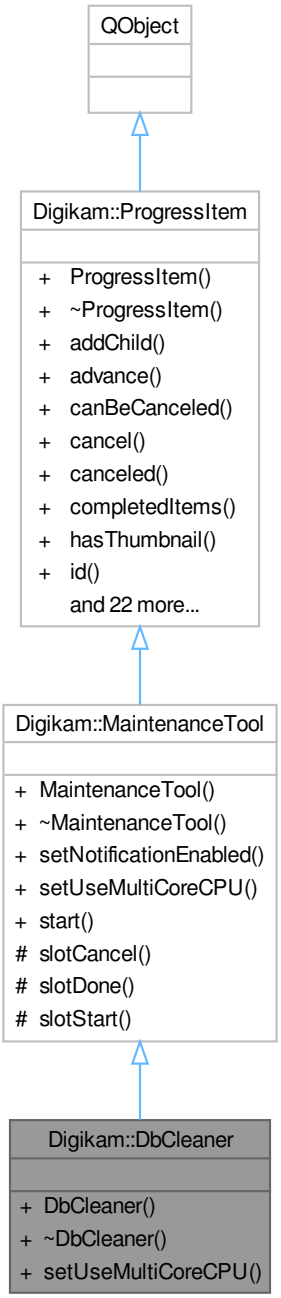
  Returns the config group that must be used for state saving and loading.

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 elpased 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

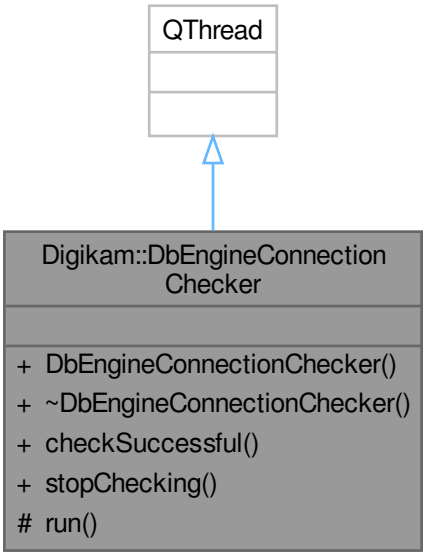
- **DbEngineConfigSettingsLoader** (const QString &filepath, int xmlVersion)
- bool **readConfig** (const QString &filepath, int xmlVersion)
- DbEngineConfigSettings **readDatabase** (const QDomElement &databaseElement)
- void **readDBActions** (const QDomElement &sqlStatementElements, DbEngineConfigSettings &config↩ Element)

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

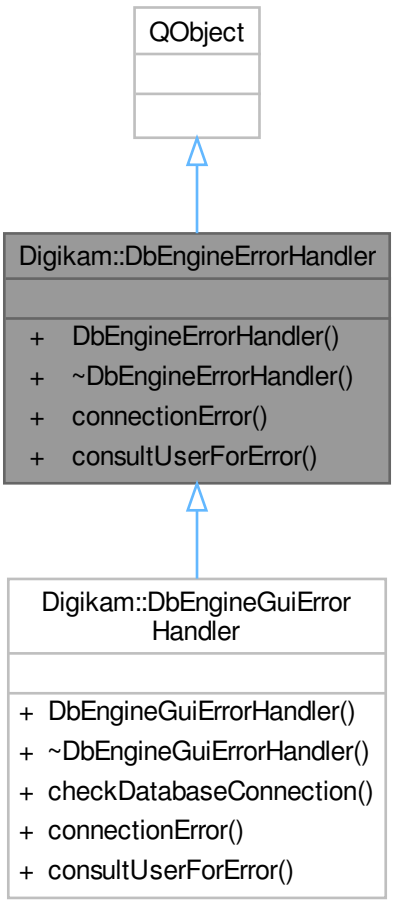
9.271 Digikam::DbEngineErrorAnswer Class Reference

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 connectionError↩ ContinueQueries() or connectionErrorAbortQueries(). The method is guaranteed to be invoked in the UI thread.

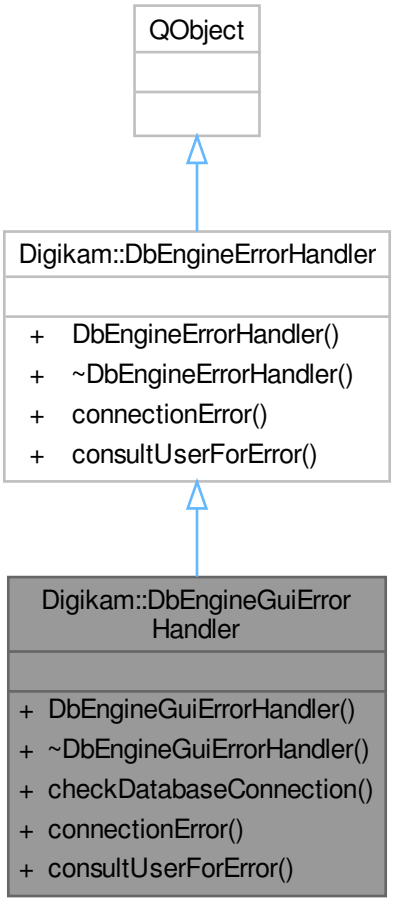
9.272.1.2 consultUserForError

```
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 connection↩ ErrorAbortQueries(). 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.

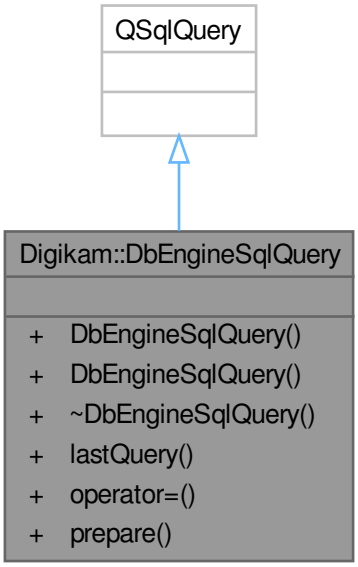
9.275.2.4 SQLiteDatabaseType()

```
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:

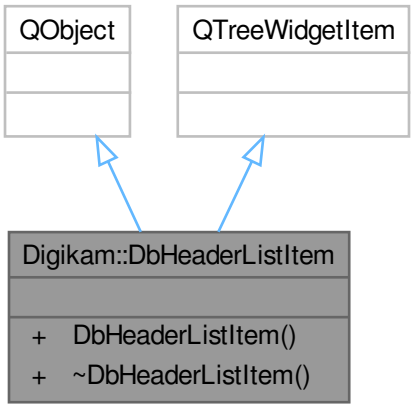


Public Member Functions

- **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:

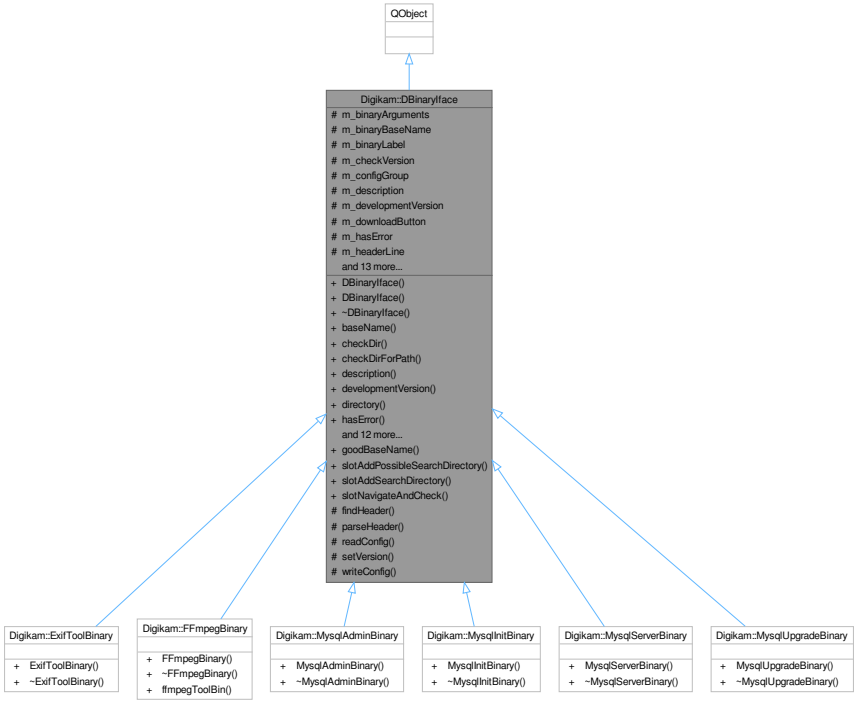


Public Member Functions

- **DbHeaderListItem** (QTreeWidget *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** ()

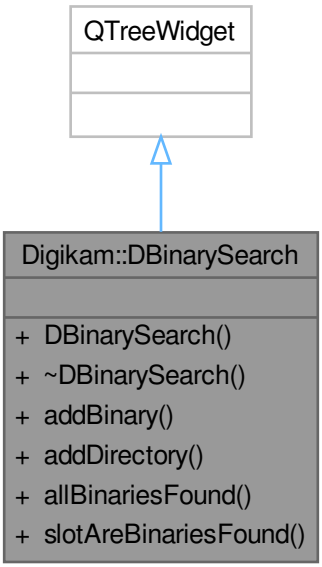
Protected Attributes

- 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

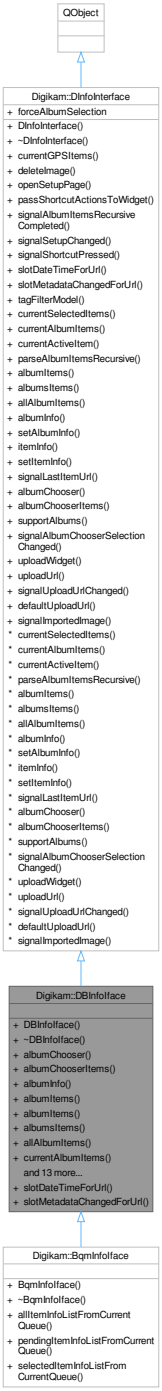
- void **signalAddDirectory** (const QString &dir)
- void **signalAddPossibleDirectory** (const QString &dir)
- void **signalBinariesFound** (bool)

Public Member Functions

- **DBinarySearch** (QWidget ∗const parent)
- void **addBinary** (DBinaryIface &binary)
- void **addDirectory** (const QString &dir)
- bool **allBinariesFound** ()

9.280 Digikam::DBInfoIface Class Reference

Inheritance diagram for Digikam::DBInfoIface:



Public Slots

- void **slotDateTimeForUrl** (const QUrl &url, const QDateTime &dt, bool updModDate) override
- void **slotMetadataChangedForUrl** (const QUrl &url) override

Public Member Functions

- **DBInfoIface** (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.

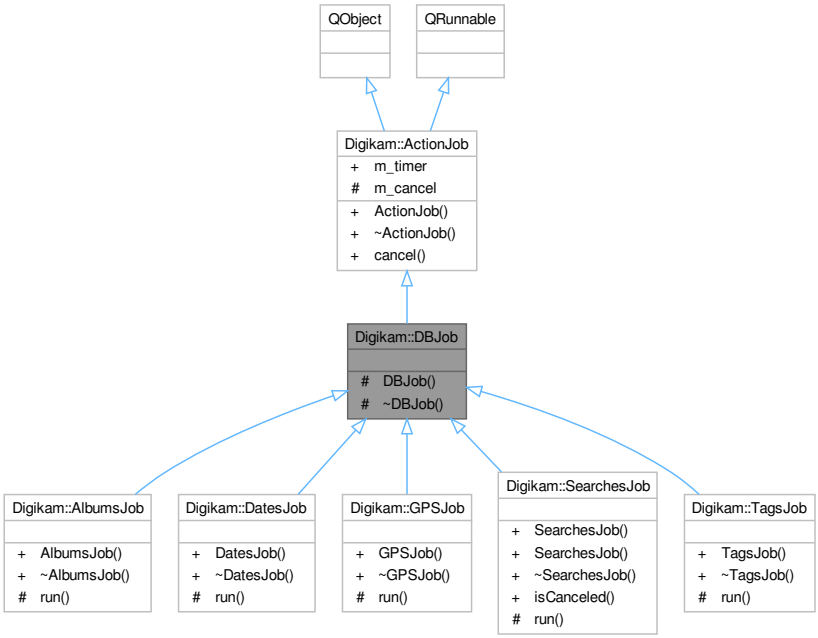
9.280.1.20 uploadWidget()

```
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.

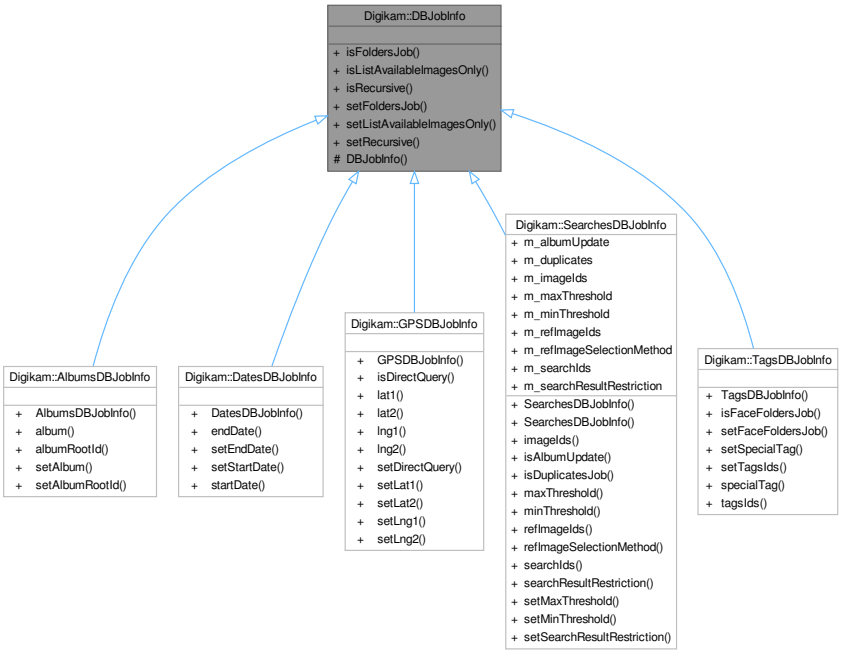
Protected Attributes inherited from Digikam::ActionJob

- 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:

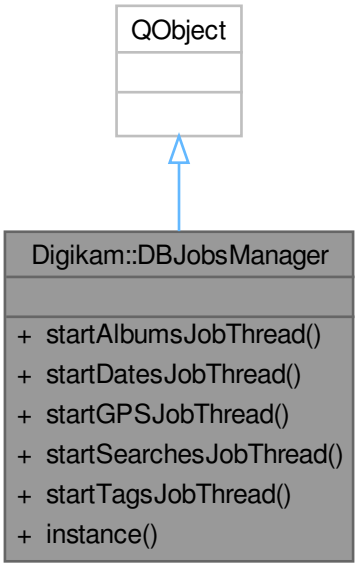


Public Member Functions

- 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

jInfo	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

jInfo	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

jInfo	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

jInfo	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

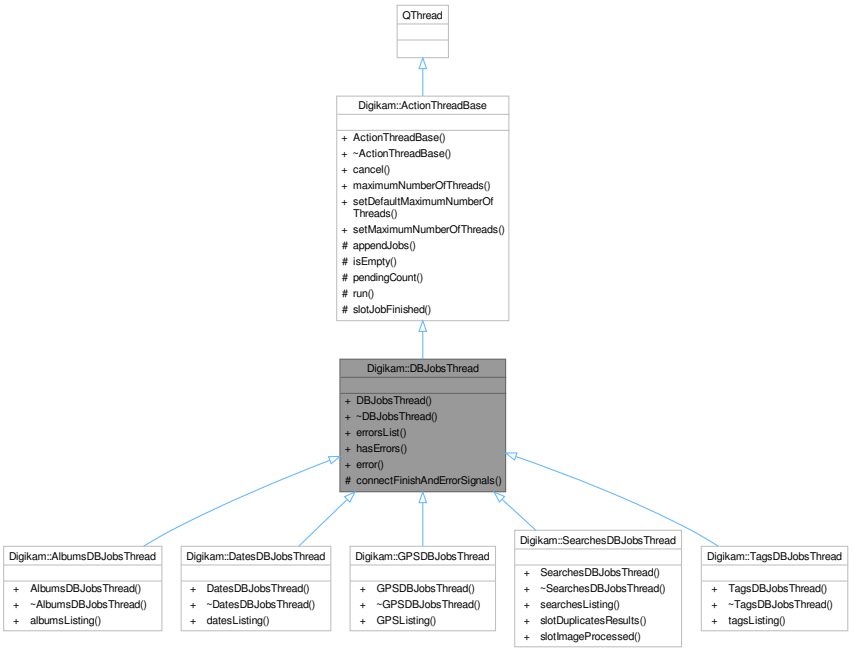
jInfo	holds job info about the DB job

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

j	Job that wanted to be connected

9.284.1.2 error

```
void Digikam::DBJobsThread::error (
            const QString & errString )  [slot]
```

Parameters

errString	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 ( )
```

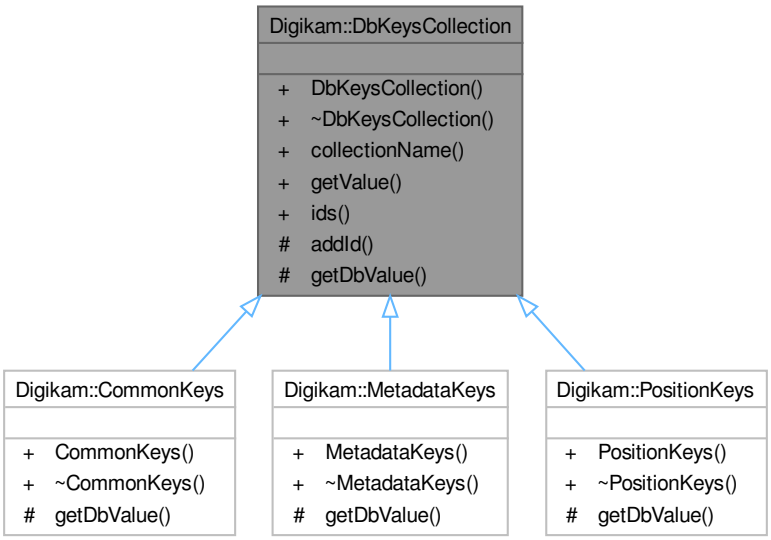
Returns

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

n	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

id	the id of the database key
description	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

key	the key representing the value in the database
settings	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

key	the key representing the value in the database
settings	the ParseSettings object holding all relevant information about the image.

Returns

the value of the given database key

9.285.3.5 ids()

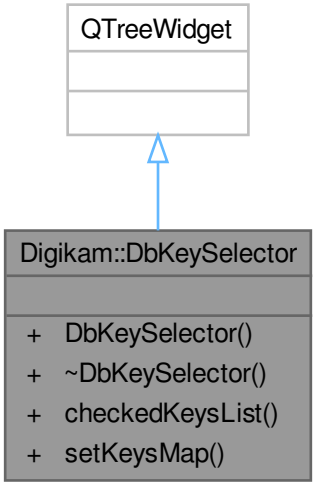
```
DbKeyIdsMap Digikam::DbKeysCollection::ids ( ) const
```

Returns

a map of all associated ids and their description

9.286 Digikam::DbKeySelector Class Reference

Inheritance diagram for Digikam::DbKeySelector:

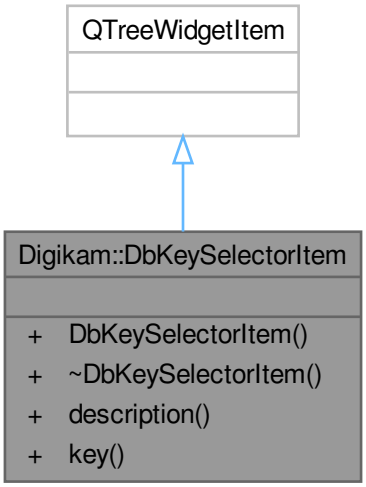


Public Member Functions

- **DbKeySelector** (QWidget ∗const parent)
- QStringList **checkedKeysList** ()
- void **setKeysMap** (const DbOptionKeysMap &map)

9.287 Digikam::DbKeySelectorItem Class Reference

Inheritance diagram for Digikam::DbKeySelectorItem:

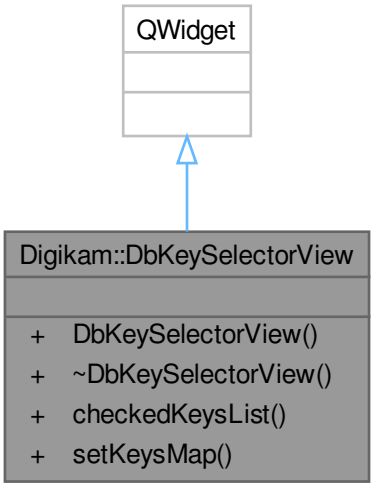


Public Member Functions

- **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:

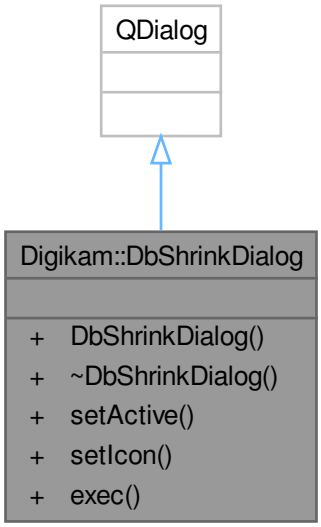


Public Member Functions

- **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

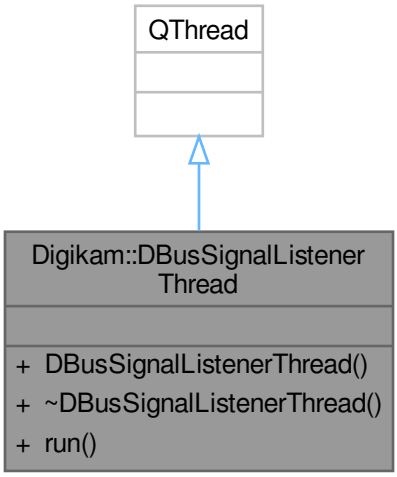
- **DBStatDlg** (QWidget ∗const parent)

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 ∗ **tabView** () const

9.291 Digikam::DBusSignalListenerThread Class Reference

Inheritance diagram for Digikam::DBusSignalListenerThread:

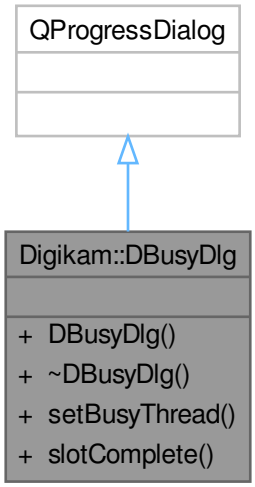


Public Member Functions

- **DBusSignalListenerThread** (CoreDbWatch ∗const qq, CoreDbWatch::Private ∗const dd)
- void **run** () override

9.292 Digikam::DBusyDlg Class Reference

Inheritance diagram for Digikam::DBusyDlg:



Public Slots

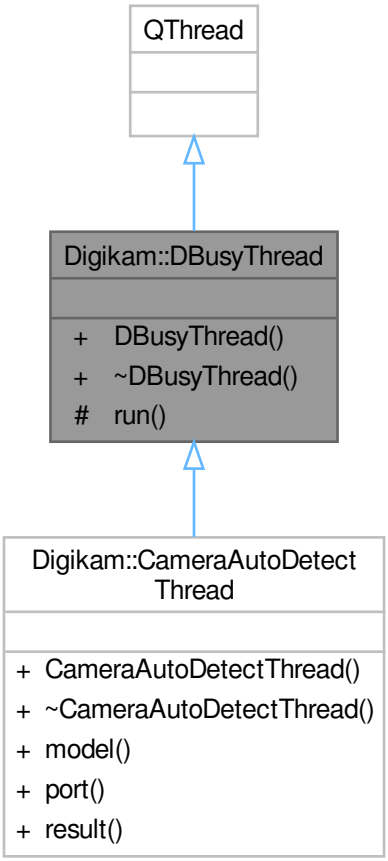
- void **slotComplete** ()

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

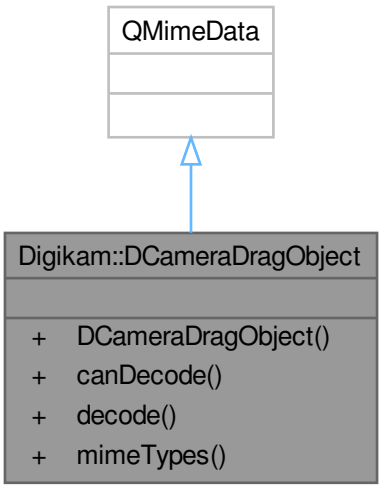
- void **run** () override

  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** ()

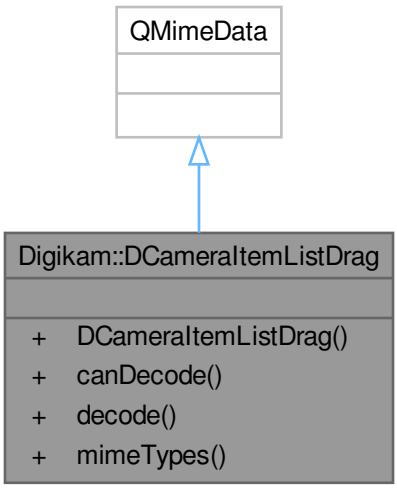
9.294.1 Detailed Description

When a camera object is moved through drag'n'drop an object of this class is created.

9.295 Digikam::DCameraItemListDrag Class Reference

Provides a drag object for a list of camera items.

Inheritance diagram for Digikam::DCameraItemListDrag:



Public Member Functions

- **DCameraItemListDrag** (const QStringList &cameraItemPaths)

Static Public Member Functions

- static bool **canDecode** (const QMimeData ∗e)
- static bool **decode** (const QMimeData ∗e, QStringList &cameraItemPaths)
- static QStringList **mimeTypes** ()

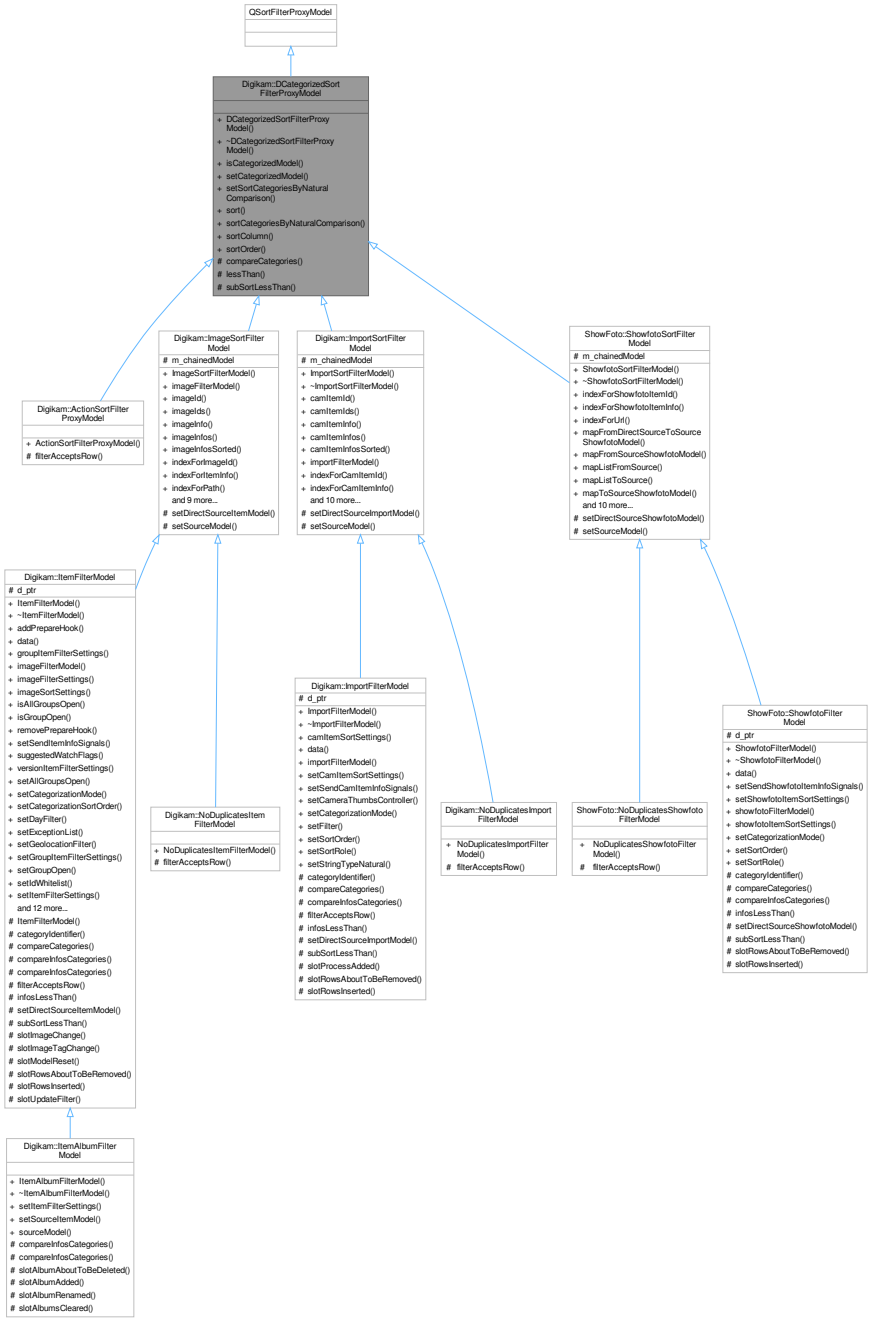
9.295.1 Detailed Description

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. **Note** 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 qlonglong (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 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

categorizedModel	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

sortCategoriesByNaturalComparison	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

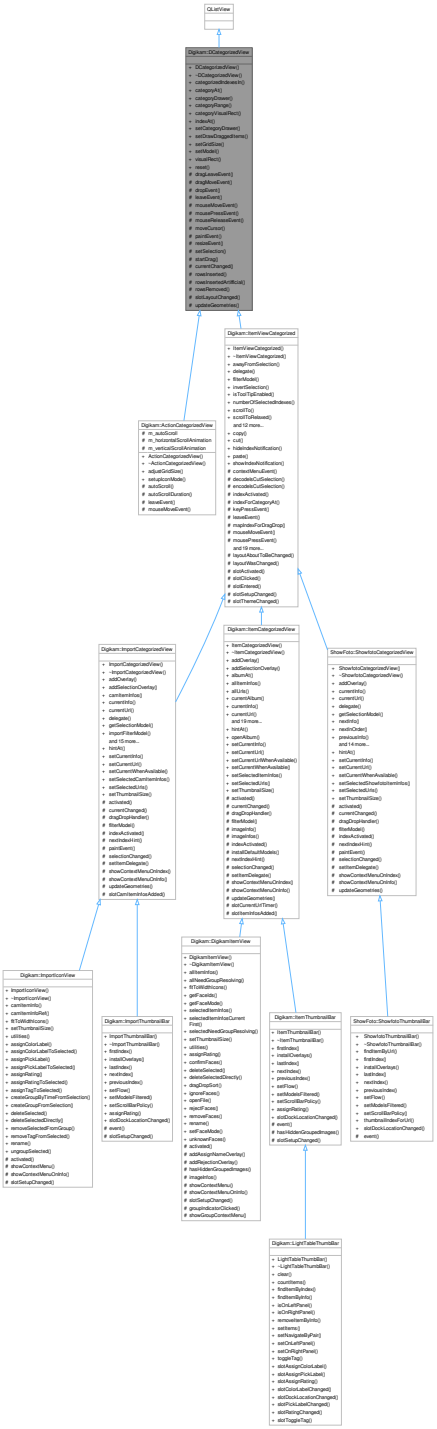
Returns true if the item `left` is less than the item `right` when sorting.

Reimplemented in Digikam::ItemFilterModel, ShowFoto::ShowfotoFilterModel, and Digikam::ImportFilterModel.

9.297 Digikam::DCategorizedView Class Reference

Item view for listing items.

Inheritance diagram for Digikam::DCategorizedView:



Public Slots

- void **reset** () override

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 **rowsInsertedArtifficial** (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

rect	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.

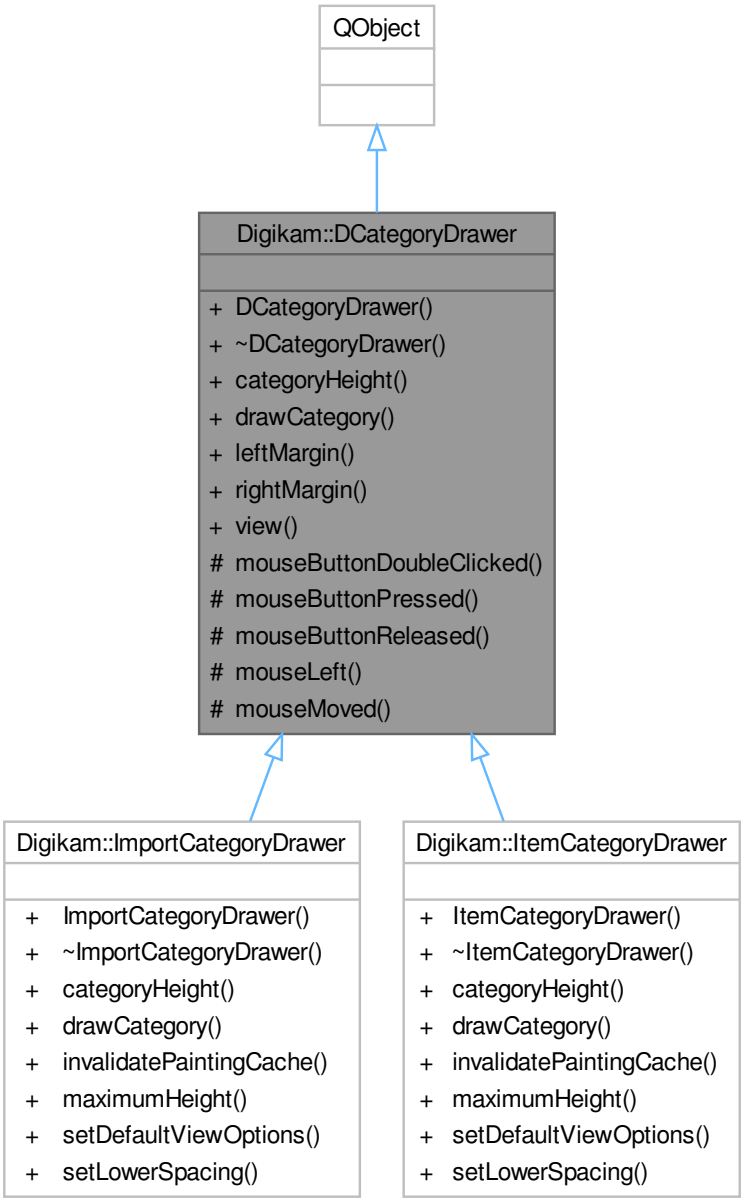
Parameters

drawDraggedItems	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

index	with the given
sortRole	sorting role
option	painter style options
painter	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

index	The representative index of the block of items.
blockRect	The rect occupied by the block of items.
event	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

index	The representative index of the block of items.
blockRect	The rect occupied by the block of items.
event	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

index	The representative index of the block of items.
blockRect	The rect occupied by the block of items.
event	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

index	The representative index of the block of items.
blockRect	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

index	The representative index of the block of items.
blockRect	The rect occupied by the block of items.
event	The mouse event.

9.298.2.10 rightMargin()

```
int Digikam::DCategoryDrawer::rightMargin ( ) const  [virtual]
```

Note

0 by default

9.298.2.11 view()

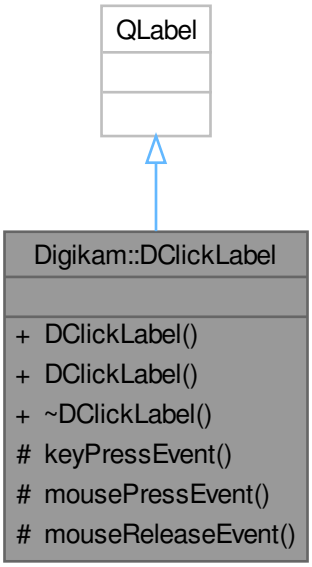
```
DCategorizedView * Digikam::DCategoryDrawer::view ( ) const
```

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 of 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()

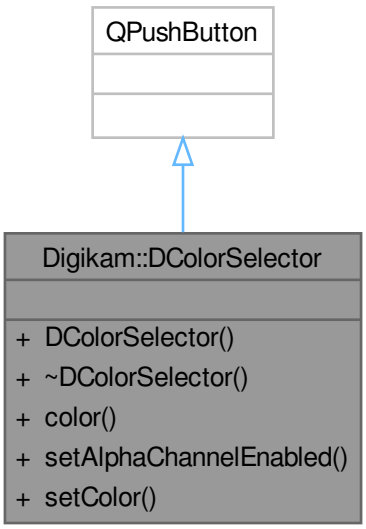
```
DColorComposer * Digikam::DColorComposer::getComposer (
            DColorComposer::CompositingOperation rule )  [static]
```

The object needs to be deleted by the caller.

9.302 Digikam::DColorSelector Class Reference

A widget to chosse a color from a palette.

Inheritance diagram for Digikam::DColorSelector:



Signals

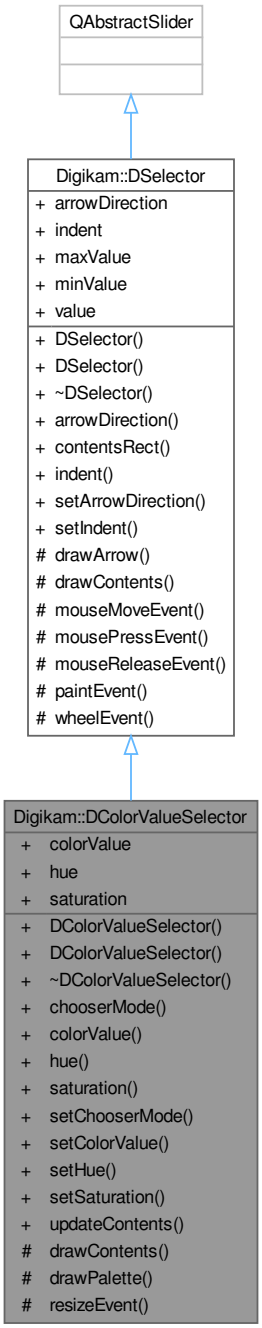
- void **signalColorSelected** (const QColor &)

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

chooserMode	Sets the chooser mode (one of the DColorChooserMode 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

colorValue	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

hue	Sets the hue value (0-359)

9.303.1.9 setSaturation()

```
void Digikam::DColorValueSelector::setSaturation (
            int saturation )
```

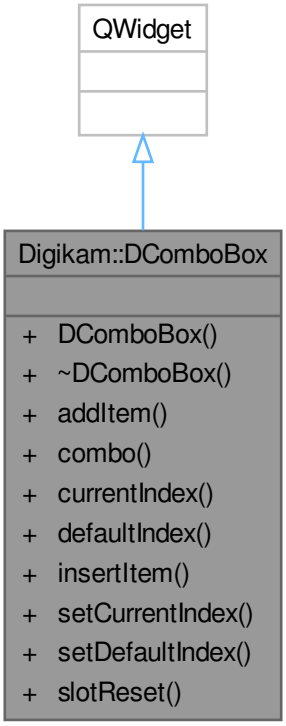
Doesn't automatically update the widget; you have to call updateContents manually.

Parameters

saturation	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** ()

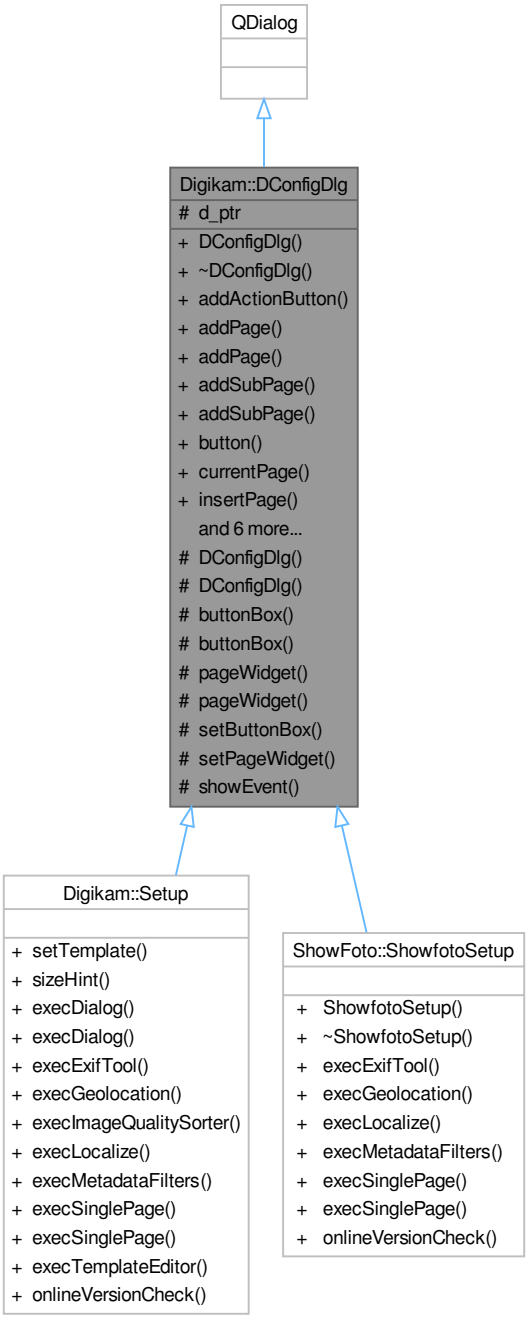
Public Member Functions

- **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 &dd, 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

widget	The DConfigDlgWdg object will be reparented to this object, so you can create it without parent and you are not allowed to delete it.
parent	The widget parent instance
flags	The window flags

9.305.4 Member Function Documentation

9.305.4.1 addPage() [1/2]

```
void Digikam::DConfigDlg::addPage (
            DConfigDlgWdgItem *const item )
```

Parameters

item	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

widget	The widget of the page.
name	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

parent	The new page will be insert as child of this

See also

DConfigDlgWdgItem.

Parameters

item	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

parent	The new page will be insert as child of this

See also

DConfigDlgWdgItem.

Parameters

widget	The widget of the page.
name	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

current	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

before	The new page will be insert before this

See also

DConfigDlgWdgItem on the same level in hierarchy.

Parameters

item	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

before	The new page will be insert before this

See also

DConfigDlgWdgItem on the same level in hierarchy.

Parameters

widget	The widget of the page.
name	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

page	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

box	The QDialogButtonBox 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

widget	The DConfigDlgWdg 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

- DConfigDlgMngr (QWidget ∗const parent, KConfigSkeleton ∗const conf)

  Constructor.
- ∼**DConfigDlgMngr** () 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

parent	Dialog widget to manage
conf	Object that contains settings

9.306.3 Member Function Documentation

9.306.3.1 addWidget()

```
void Digikam::DConfigDlgMngr::addWidget (
            QWidget *const widget )
```

Parameters

widget	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

trackChanges	- 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

widget	- Parent of the children to look at.
trackChanges	- 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

widget	- The widget group (pass in via addWidget()) 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

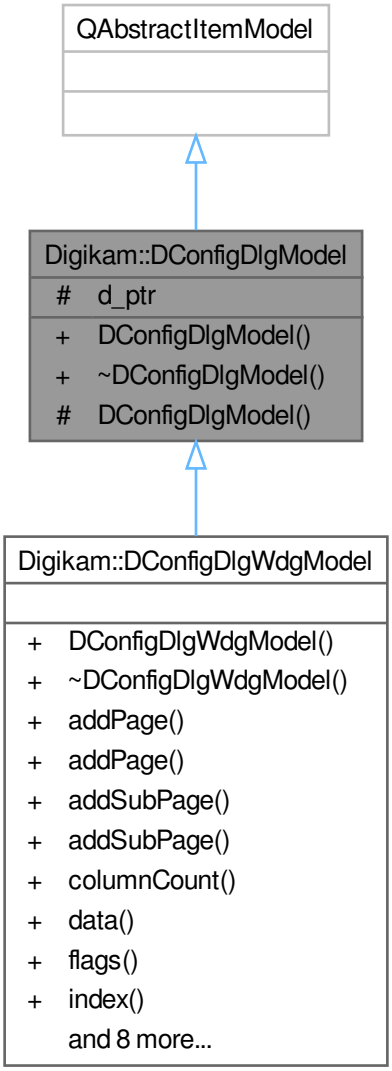
```
void Digikam::DConfigDlgMngr::widgetModified ( )  [signal]
```

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`

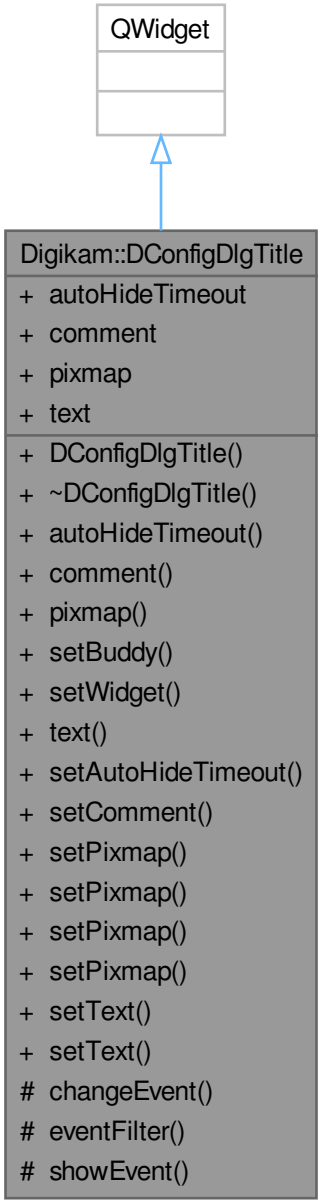
Enumerator

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 msecs)

  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=PlainMessage)
- 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

PlainMessage	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

parent	.

9.308.4 Member Function Documentation

9.308.4.1 autoHideTimeout()

```
int Digikam::DConfigDlgTitle::autoHideTimeout ( ) const
```

Returns

timeout value in msecs

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 msecs )  [slot]
```

Parameters

msecs	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

buddy	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

comment	Text displayed beneath the main title as a comment. It can either be plain text or rich text.
type	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

icon	The pixmap to display in the header. The pixmap is by default right, but
alignment	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

pixmap	Pixmap displayed in the header. The pixmap is by default right, but
alignment	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

icon	name of the icon to display in the header. The pixmap is by default right, but
alignment	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

type	The message type to display as pixmap in the header. The message is by default right, but
alignment	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

text	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.
type	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

text	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.
alignment	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

widget	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

index	to be the current page and emits the

See also

currentPageChanged signal.

9.309.3.4 setItemDelegate()

```
void Digikam::DConfigDlgView::setItemDelegate (
            QAbstractItemDelegate * delegate )
```

Parameters

delegate	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()

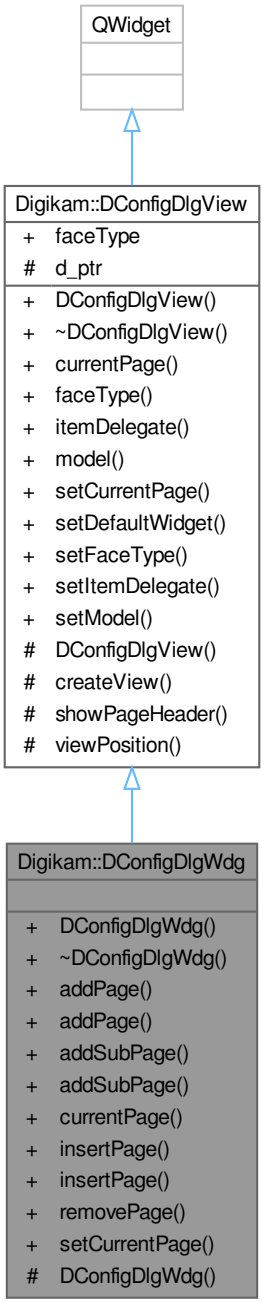
```
Qt::Alignment Digikam::DConfigDlgView::viewPosition ( ) const  [protected], [virtual]
```

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

parent	The parent widget.

9.310.3 Member Function Documentation

9.310.3.1 addPage() [1/2]

```
void Digikam::DConfigDlgWdg::addPage (
            DConfigDlgWdgItem * item )
```

Parameters

item	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

widget	The widget of the page.
name	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

parent	The new page will be insert as child of this

See also

DConfigDlgWdgItem.

Parameters

item	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

parent	The new page will be insert as child of this

See also

DConfigDlgWdgItem.

Parameters

widget	The widget of the page.
name	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

current	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

before	The new page will be insert before this

See also

DConfigDlgWdgItem on the same level in hierarchy.

Parameters

item	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

before	The new page will be insert before this

See also

DConfigDlgWdgItem on the same level in hierarchy.

Parameters

widget	The widget of the page.
name	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

page	The page which is removed

9.310.3.10 pageToggled

```
void Digikam::DConfigDlgWdg::pageToggled (
            DConfigDlgWdgItem * page,
            bool checked )  [signal]
```

Parameters

checked	is true when the
page	is checked, or false if the
page	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 )
```

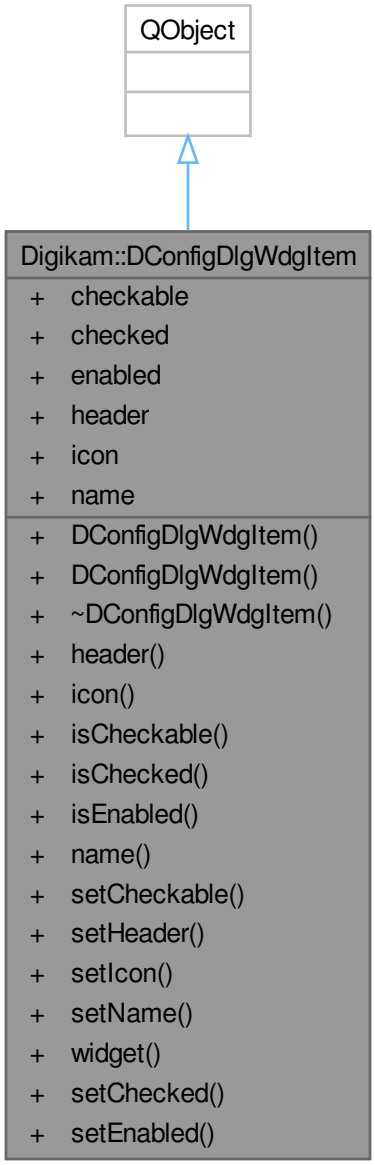
See also

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

widget	The widget that is shown as page in the DConfigDlgWdg.

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

widget	The widget that is shown as page in the DConfigDlgWdg.
name	The localized string that is show in the navigation view of the DConfigDlgWdg.

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

checkable	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

header	Header of the page widget item.

9.311.2.3 setIcon()

```
void Digikam::DConfigDlgWdgItem::setIcon (
            const QIcon & icon )
```

Parameters

icon	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

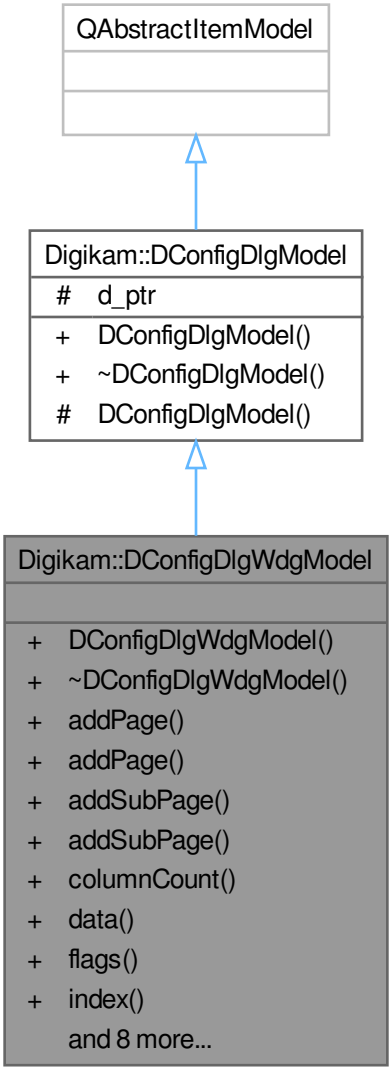
```
bool Digikam::DConfigDlgWdgItem::enabled  [read], [write]
```

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

parent	The parent object.

9.312.3 Member Function Documentation

9.312.3.1 addPage() [1/2]

```
void Digikam::DConfigDlgWdgModel::addPage (
            DConfigDlgWdgItem * item )
```

Parameters

item	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

widget	The widget of the page.
name	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

parent	The new page will be insert as child of this

See also

DConfigDlgWdgItem.

Parameters

item	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

parent	The new page will be insert as child of this

See also

DConfigDlgWdgItem.

Parameters

widget	The widget of the page.
name	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

before	The new page will be insert before this

See also

DConfigDlgWdgItem on the same level in hierarchy.

Parameters

item	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

before	The new page will be insert before this

See also

DConfigDlgWdgItem on the same level in hierarchy.

Parameters

widget	The widget of the page.
name	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]
```

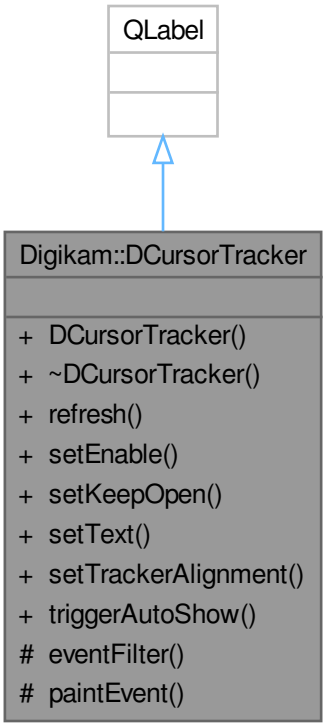
Parameters

checked	is true when the
page	is checked, or false if the
page	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 **setEnable** (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

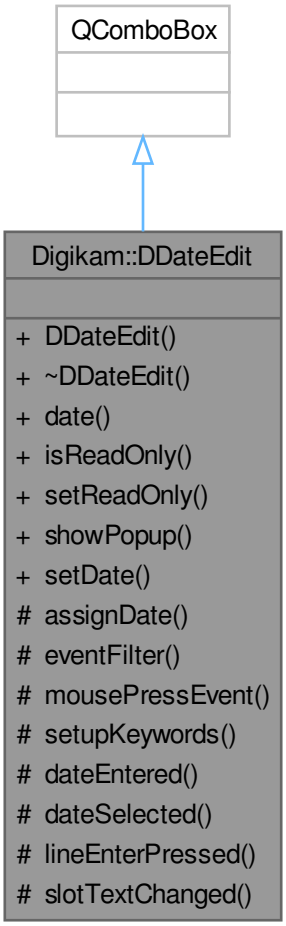
9.313.1 Detailed Description

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

date	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

date	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

readOnly	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()

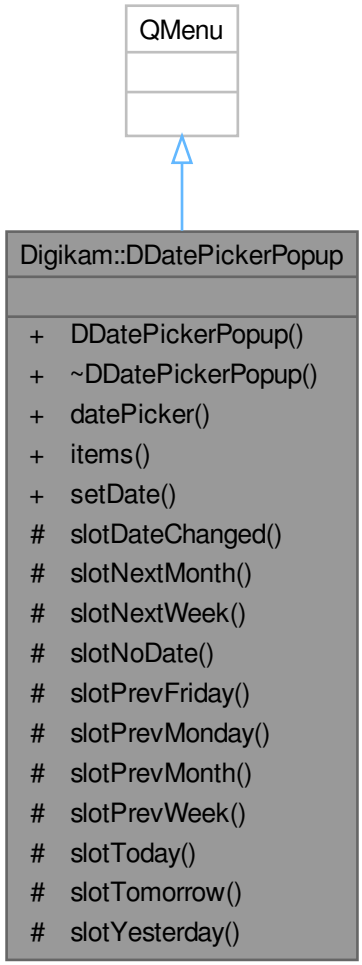
```
QSize Digikam::DDatePicker::sizeHint ( ) const  [override]
```

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

items	List of all desirable items, separated with a bitwise OR.
date	Initial date of datepicker-widget.
parent	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
```

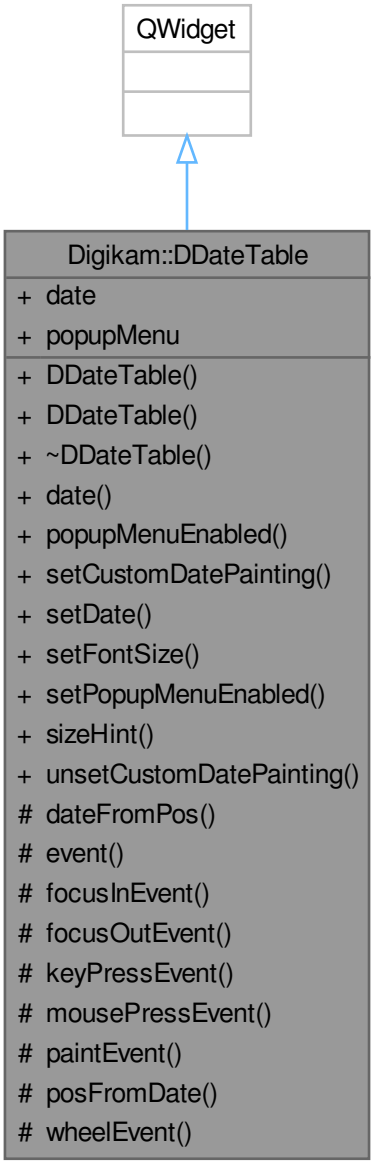
Returns

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=No↩ BgMode, 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

cur	The current date
old	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()

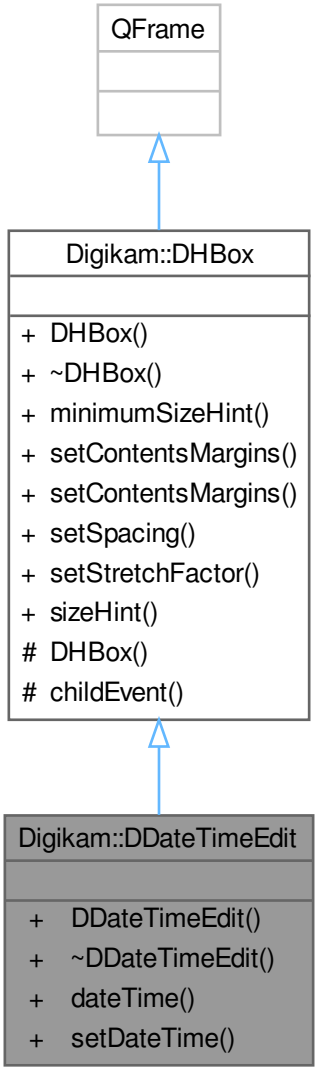
```
QSize Digikam::DDateTable::sizeHint ( ) const  [override]
```

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

parent	the parent widget
name	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

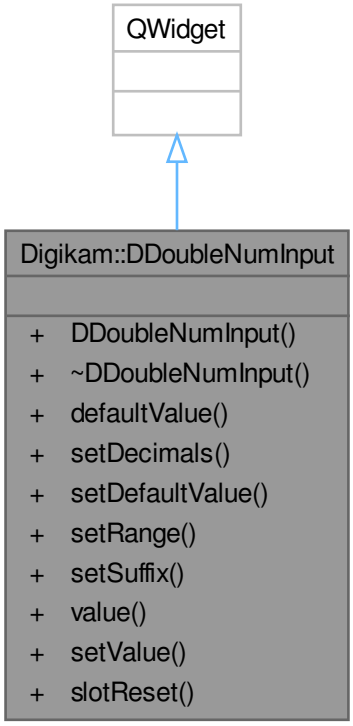
9.318.2.2 dateTimeChanged

```
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

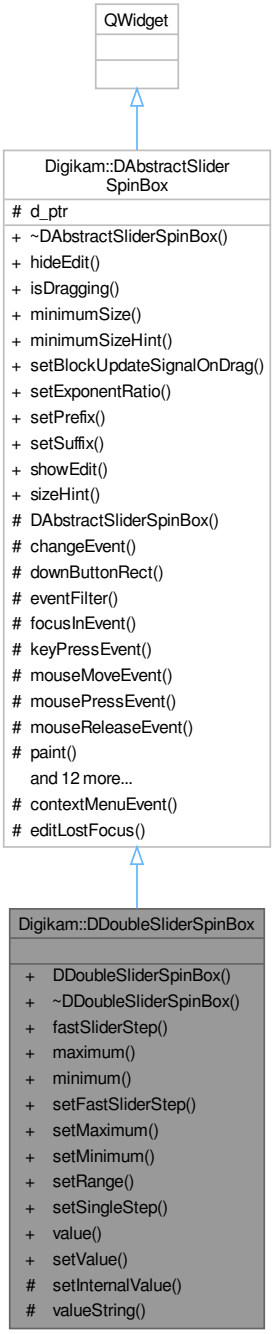
- void **reset** ()
- void **valueChanged** (double)

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.

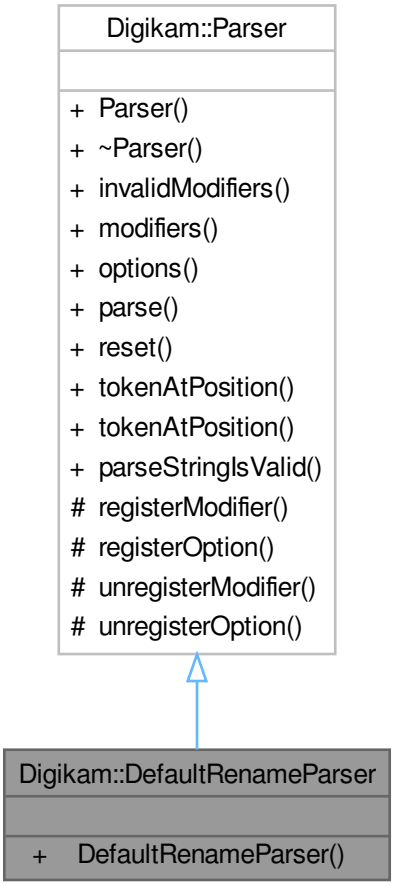
9.320.1.2 valueString()

```
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 parseStringIsValid (const QString &str)

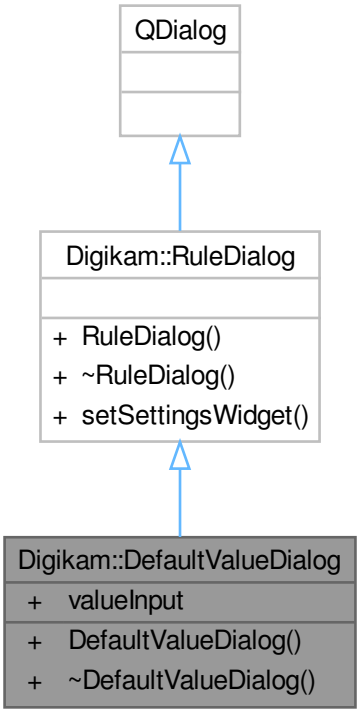
  check if the given parse string is valid

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

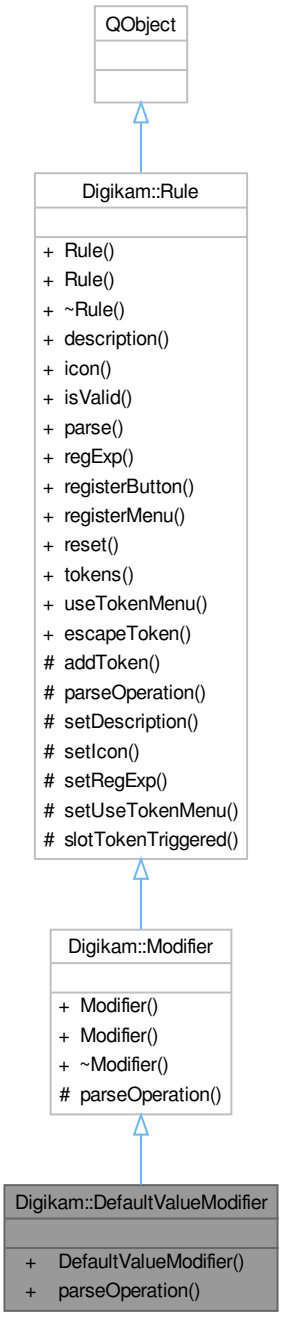
- **RuleDialog** (Rule ∗const parent)
- void **setSettingsWidget** (QWidget ∗const settingsWidget)

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

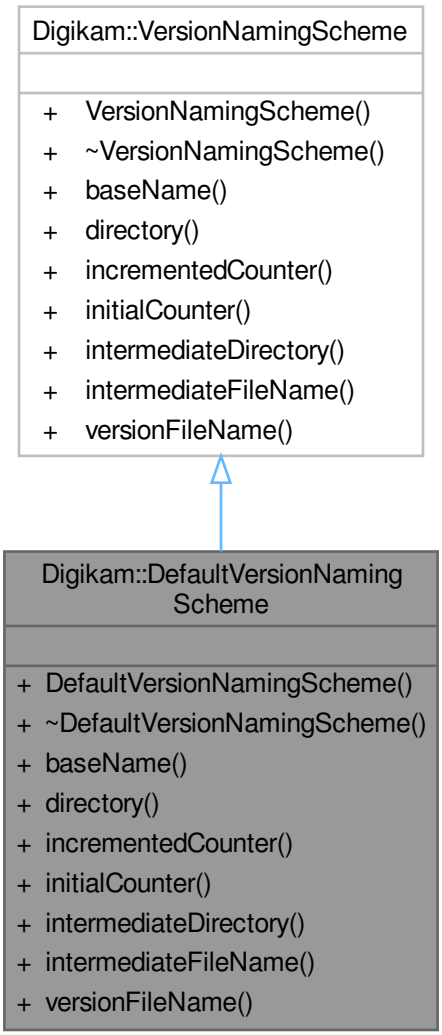
settings	contains settings
match	result of the regular expression match done in Option::parse()

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()

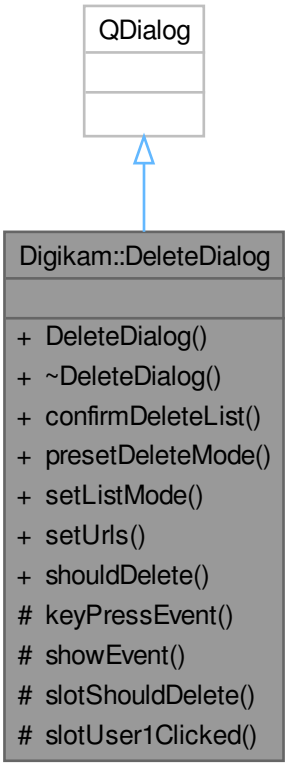
```
QString Digikam::DefaultVersionNamingScheme::versionFileName (
            const QString & path,
            const QString & baseName,
            const QVariant & counter )  [override], [virtual]
```

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

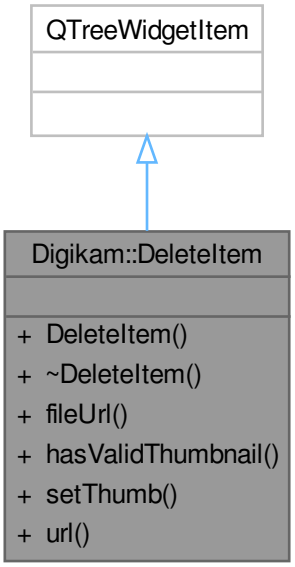
- void **slotShouldDelete** (bool)
- void **slotUser1Clicked** ()

Protected Member Functions

- void **keyPressEvent** (QKeyEvent ∗) override
- void **showEvent** (QShowEvent ∗) override

9.326 Digikam::DeleteItem Class Reference

Inheritance diagram for Digikam::DeleteItem:

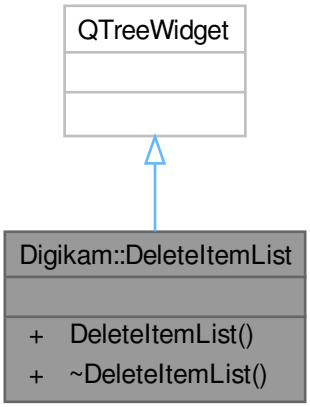


Public Member Functions

- **DeleteItem** (QTreeWidget ∗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::DeleteItemList Class Reference

Inheritance diagram for Digikam::DeleteItemList:

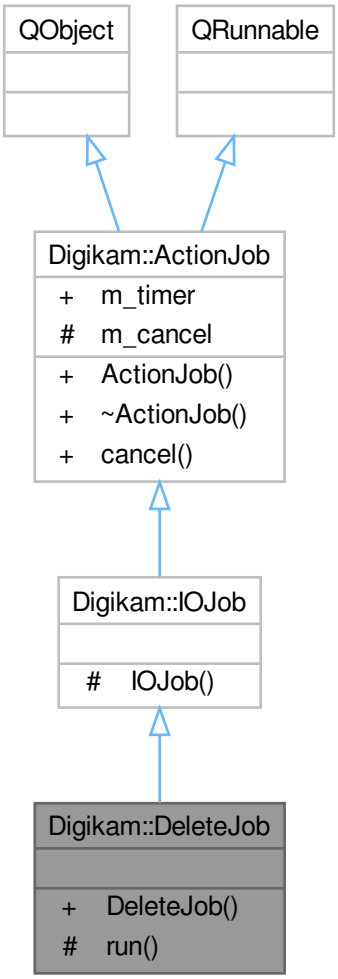


Public Member Functions

- **DeleteItemList** (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 **signalOneProccessed** (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.

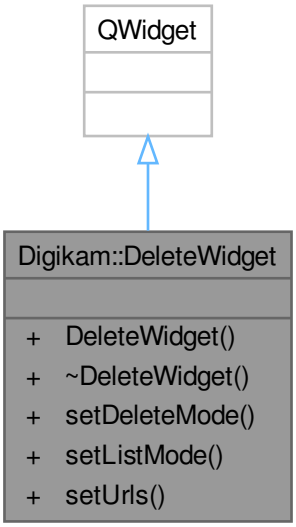
Protected Attributes inherited from Digikam::ActionJob

- 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

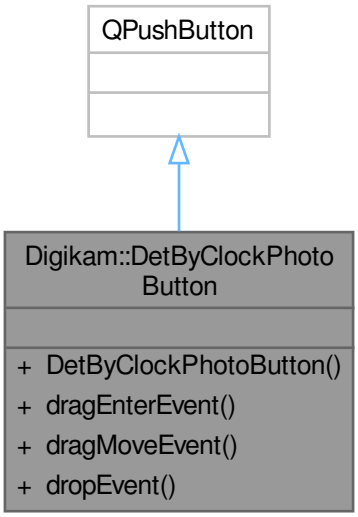
  Check if at least one option is selected.

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

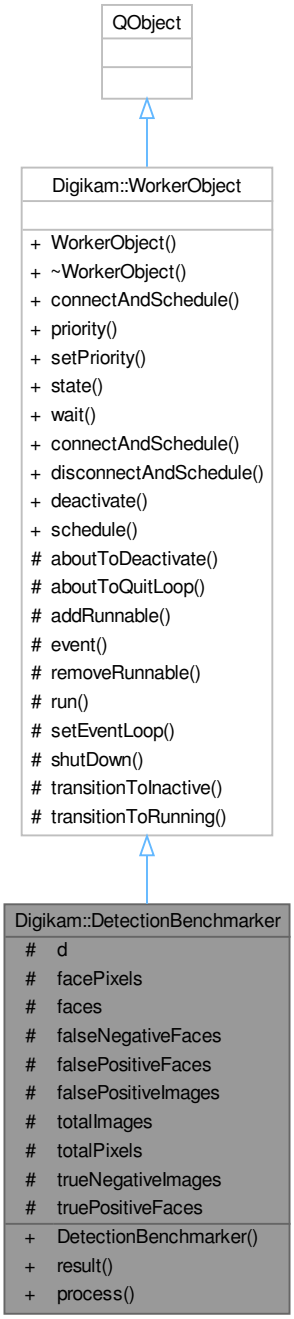
- void **signalClockPhotoDropped** (const QUrl &)

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

- **DetectionBenchmarker** (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 ()

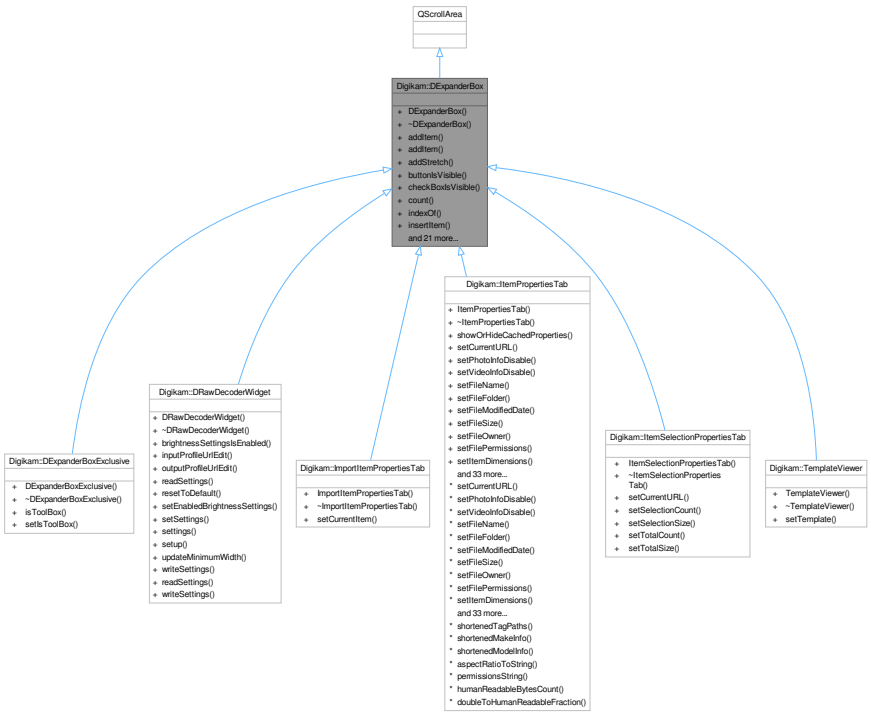
  Called from deactivate(), typically from a different thread than the worker thread, possibly the UI thread.
- virtual void aboutToQuitLoop ()

  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 expand↩ Bydefault)
- 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).

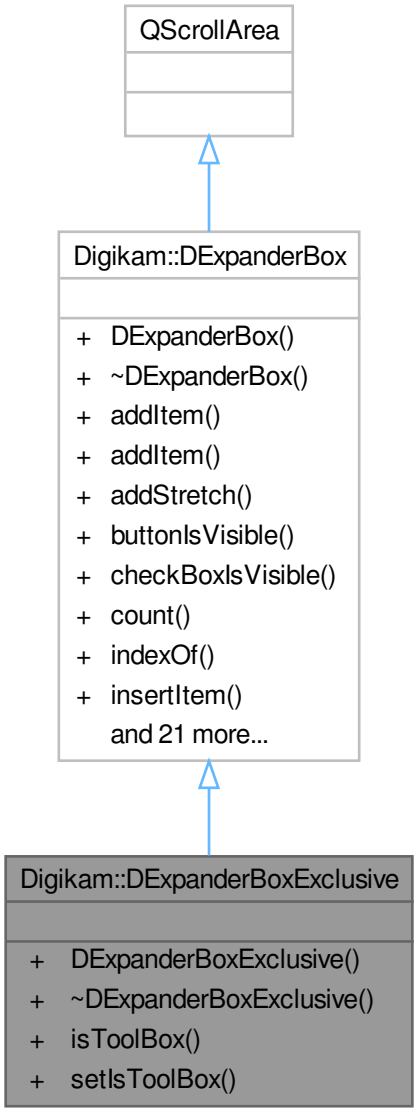
9.334.1.2 insertItem()

```
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 expand↩ Bydefault)
- 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)

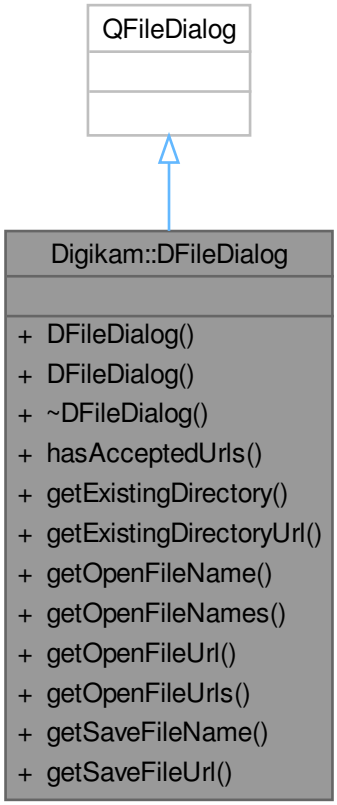
Additional Inherited Members

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=QString↩List())
- 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 &itemId=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()

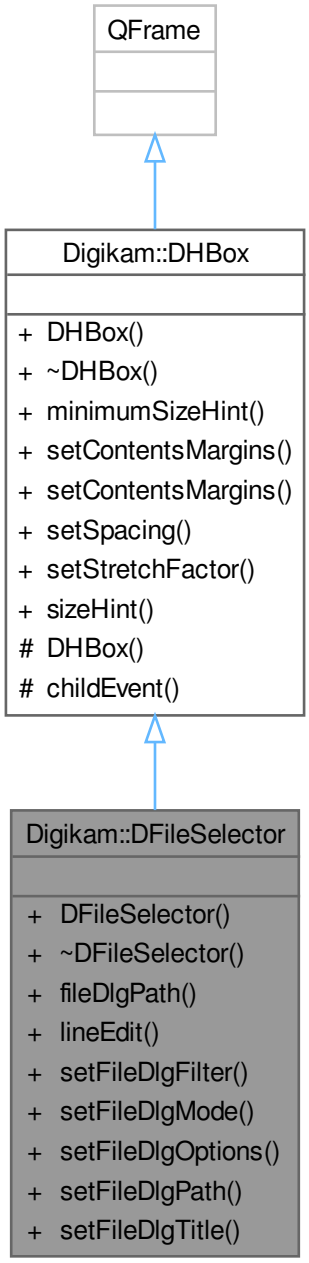
```
bool Digikam::DFileOperations::setModificationTime (
            const QString & srcFile,
            const QDateTime & dateTime )  [static]
```

Keep access time from source file.

9.338 Digikam::DFileSelector Class Reference

A widget to chosse 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

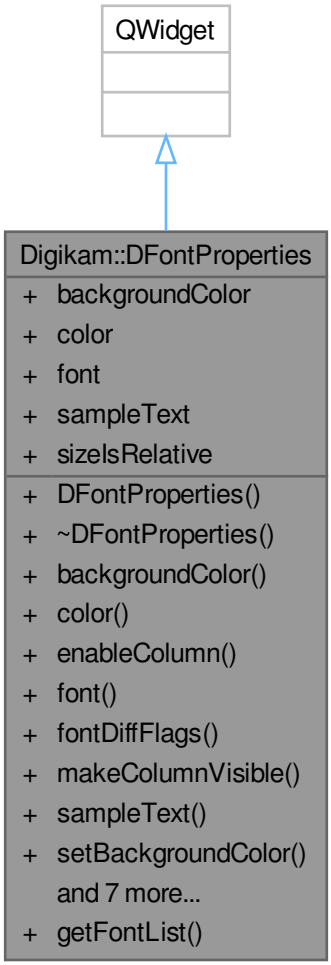
- **DHBox** (bool vertical, QWidget ∗const parent)
- void **childEvent** (QChildEvent ∗e) override

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 sizeIsRelativeState=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 setSizeIsRelative (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 sizeIsRelative () 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 **sizeIsRelative**

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

parent	The parent widget.
flags	Defines how the font chooser is displayed.

See also

DisplayFlags

Parameters

fontList	A list of fonts to display, in XLFD format.
visibleListSize	The minimum number of visible entries in the fontlists.
sizeIsRelativeState	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 ∗sizeIsRelativeState, user choice may be retrieved by calling sizeIsRelative().

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

column	Specify the columns. An or'ed combination of `FamilyList`, `StyleList` and `SizeList` is possible.
state	If `false` 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

list	The list is returned here.
fontListCriteria	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

column	Specify the columns. An or'ed combination of `FamilyList`, `StyleList` and `SizeList` is possible.
state	If `false` 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

font	The font to select.
onlyFixed	Readjust the font list to display only fixed width fonts if `true`, or vice-versa.

9.339.3.10 setSampleBoxVisible()

```
void Digikam::DFontProperties::setSampleBoxVisible (
            bool visible )
```

Parameters

visible	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

text	The new sample text. The current will be removed.

9.339.3.12 setSizeIsRelative()

```
void Digikam::DFontProperties::setSizeIsRelative (
            Qt::CheckState relative )
```

Note

If parameter sizeIsRelative was not set in the constructor of the widget this setting will be ignored.

9.339.3.13 sizeIsRelative()

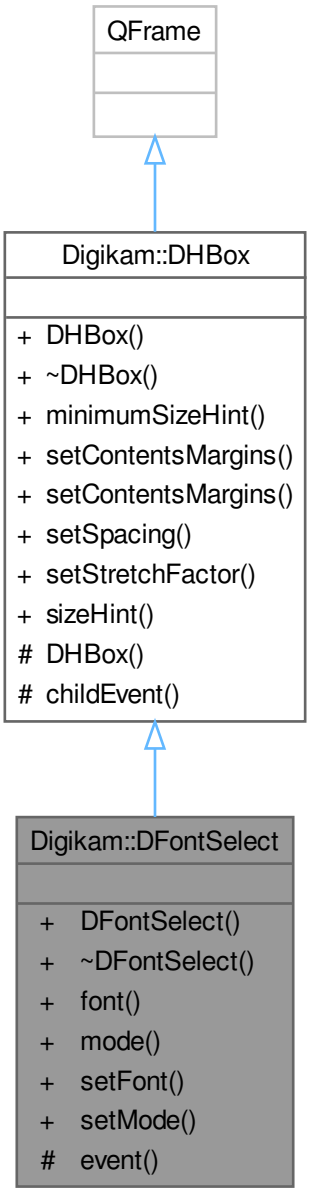
```
Qt::CheckState Digikam::DFontProperties::sizeIsRelative ( ) const
```

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

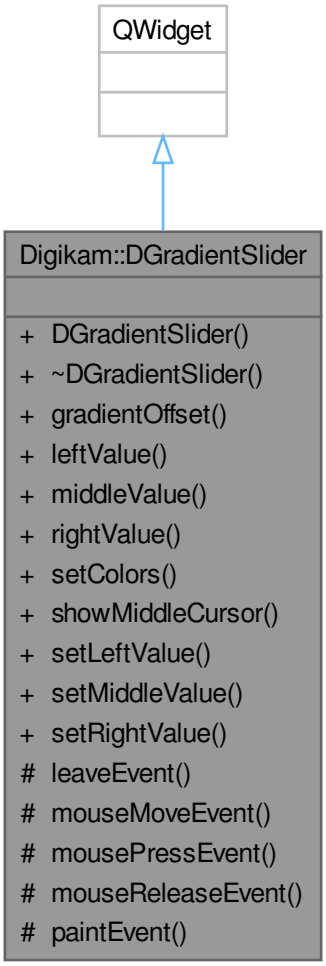
- bool **event** (QEvent ∗e) override

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)

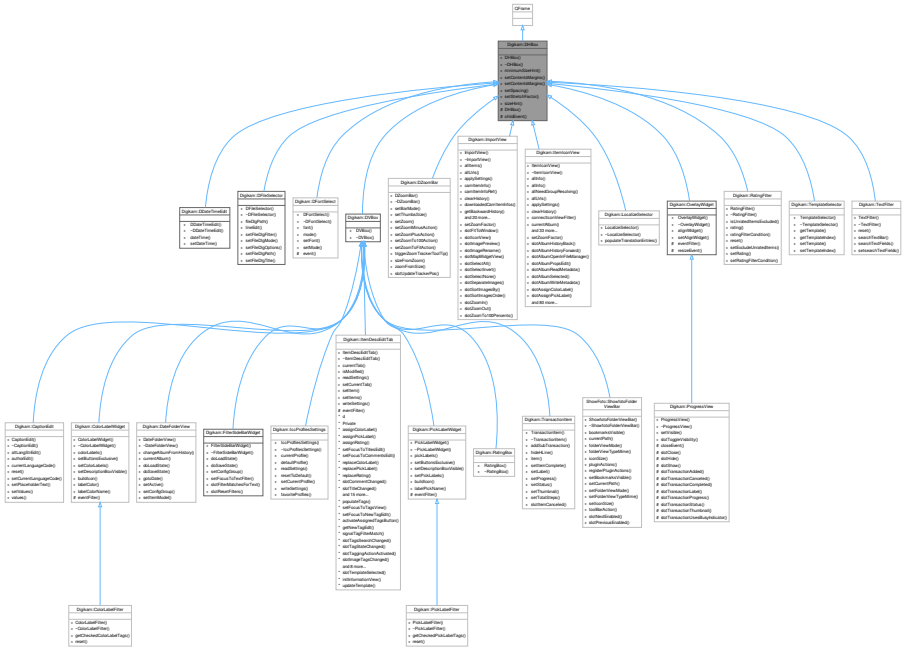
Protected Member Functions

- 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

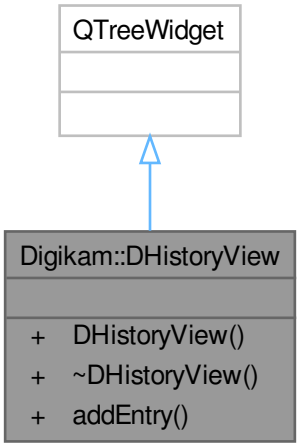
- **DHBox** (QWidget ∗const parent=nullptr)
- QSize **minimumSizeHint** () const override
- void **setContentsMargins** (const QMargins &margins)
- void **setContentsMargins** (int left, int top, int right, int bottom)
- void **setSpacing** (int space)
- void **setStretchFactor** (QWidget ∗const widget, 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

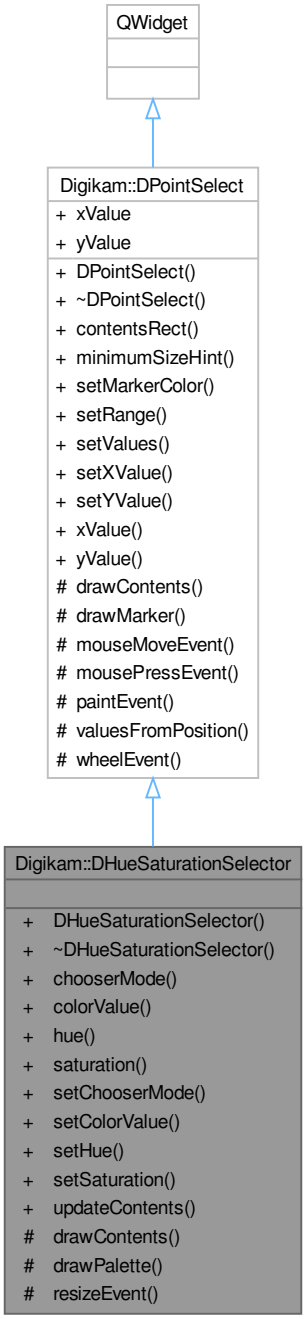
- void **signalEntryClicked** (const QVariant &metadata)

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

chooserMode	The chooser mode as defined in DColorChooserMode

9.344.1.7 setColorValue()

```
void Digikam::DHueSaturationSelector::setColorValue (
            int color )
```

Parameters

color	The color value (0-255)

9.344.1.8 setHue()

```
void Digikam::DHueSaturationSelector::setHue (
            int hue )
```

Parameters

hue	The hue value (0-360)

9.344.1.9 setSaturation()

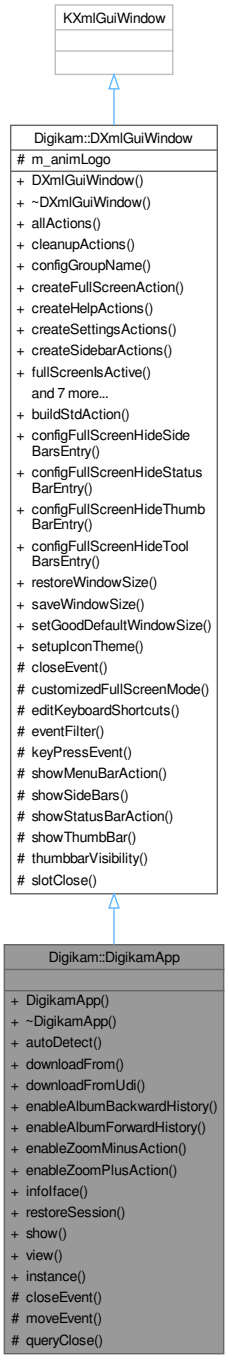
```
void Digikam::DHueSaturationSelector::setSaturation (
            int saturation )
```

Parameters

saturation	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 * infoIface (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 **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 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

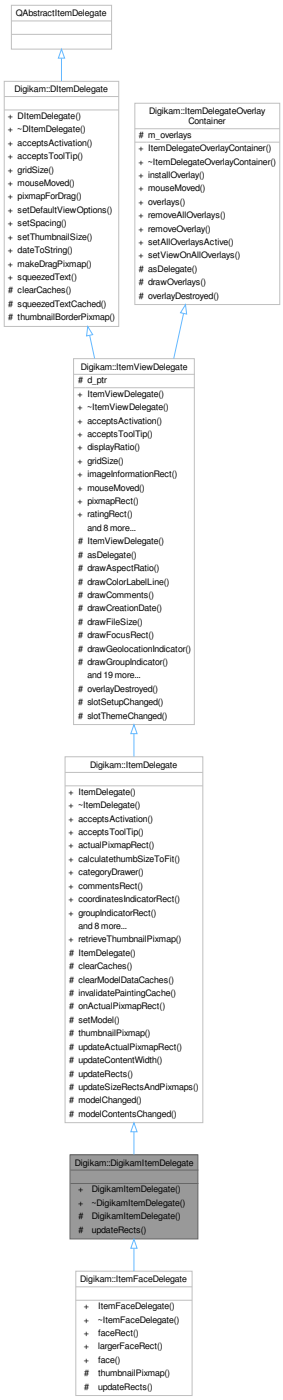
9.345.1.1 infoIface()

```
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 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 **drawPanelSideIcon** (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 **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::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::Delete↩ UseTrash)
- 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** (qlonglong 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 **rowsInsertedArtifficial** (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 **decodeIsCutSelection** (const QMimeData ∗mimeData)
- void **encodeIsCutSelection** (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 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 **decodeIsCutSelection** (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 **encodeIsCutSelection** (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 HistoryImageId: 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< HistoryImageId > **allReferredImages** () const
- void **appendReferredImage** (const HistoryImageId &id)
- void clearReferredImages ()

  Edit referred images.
- HistoryImageId **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** (HistoryImageId::Type type) const
- bool **hasReferredImages** () const
- void **insertReferredImage** (int entryIndex, const HistoryImageId &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 HistoryImageId &imageId)

  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
- HistoryImageId **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< HistoryImageId > & **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 toand 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 addCurrentUniqueImageId (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 bitBltImage (const DImg *const src, int dx, int dy)

  Copy a region of pixels from a source image to this image.
- void **bitBltImage** (const DImg *const src, int sx, int sy, int dx, int dy)
- void **bitBltImage** (const DImg *const src, int sx, int sy, int w, int h, int dx, int dy)
- void **bitBltImage** (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 loadICCData, bool loadUniqueHash, bool load↩ History, 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 loadICCData=true, bool load↩ UniqueHash=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 copyData=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::IgnoreAspectRatio) 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 bitBltImage 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 bitBltImage()

```
void Digikam::DImg::bitBltImage (
            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 dcraw 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 loadICCData = true,
            bool loadUniqueHash = true,
            bool loadImageHistory = true )
```

If loadMetadata is true, the metadata will be available with getComments, getExif, getIptc, getXmp . If loadICCData 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 two 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 setHistory↩ BranchForLastSteps(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 **displayableName** () const

  Returns a displayableName for this filter.
- FilterAction filterAction () const

  Returns the FilterAction describing this filter.
- QString **filterIcon** () const
- QString **i18nDisplayableName** () 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 **i18nDisplayableName** (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.

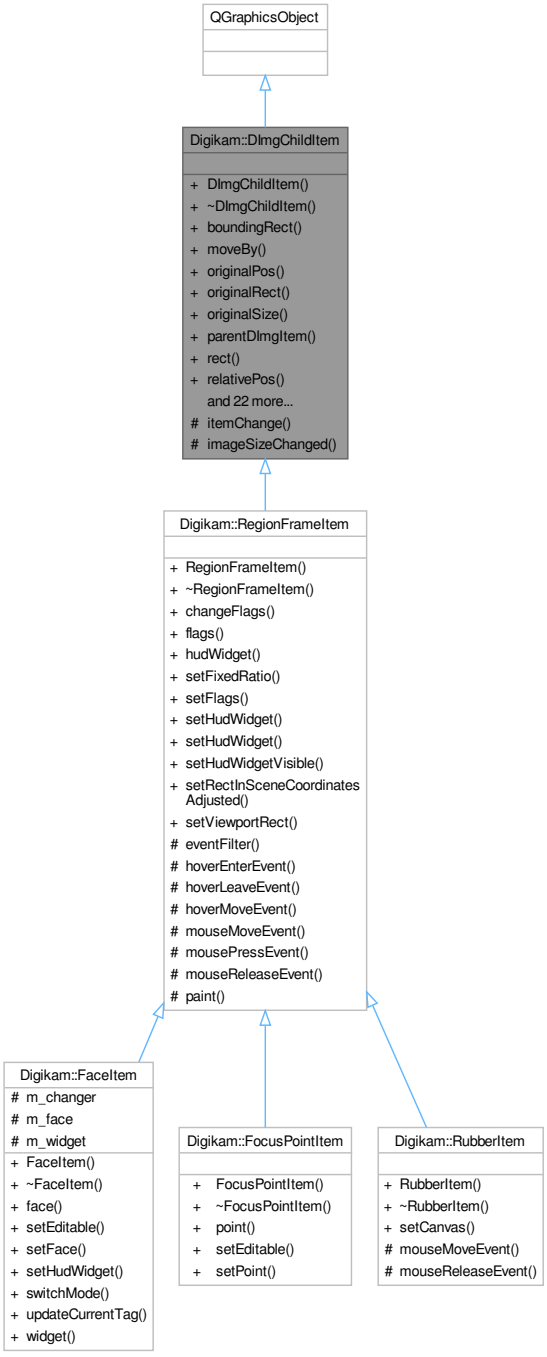
9.351.3.2 reverseFilter()

```
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.

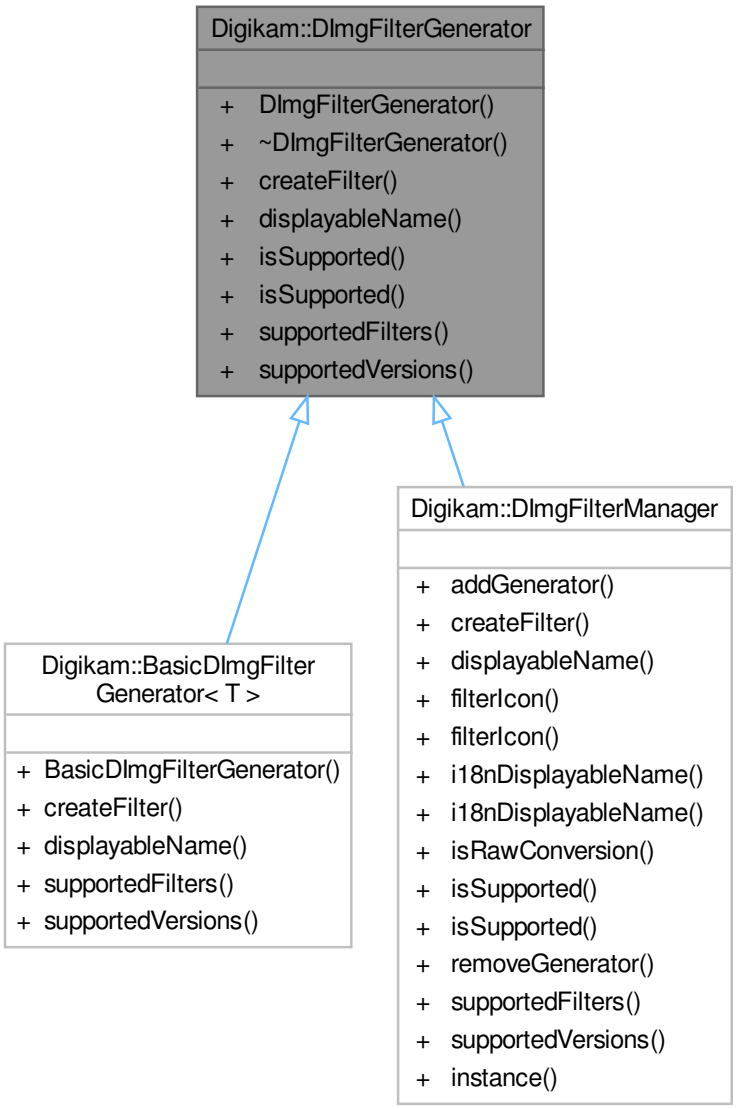
9.352.2.9 setRelativePos()

```
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 displayableName (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.

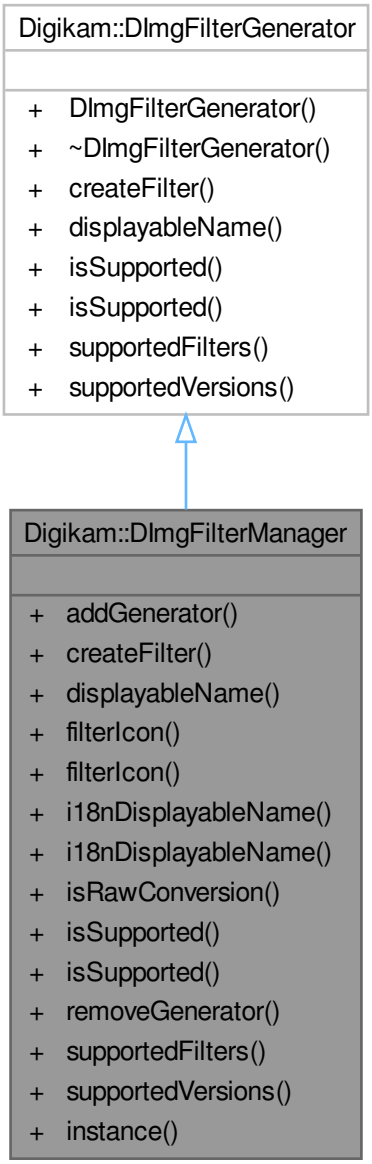
9.353.1.5 supportedVersions()

```
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 displayableName (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 , LoadICCData = 4 , LoadImageData = 8 ,
  LoadUniqueHash = 16 , LoadImageHistory = 32 , LoadPreview = 64 , LoadAll = LoadItemInfo | LoadMetadata | LoadICCData | 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 **imageGetEmbbededText** (const QString &key) const
- bool **imageHasAlpha** () const
- unsigned int & **imageHeight** ()
- quint64 **imageNumBytes** () const
- void **imageSetAttribute** (const QString &key, const QVariant &value)
- void **imageSetEmbbededText** (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

LoadItemInfo	Load image information without image data. Image info as width and height
LoadMetadata	Image metadata.
LoadICCData	Image color profile.
LoadImageData	Full image data.
LoadUniqueHash	Image unique hash.
LoadImageHistory	Image version history.
LoadPreview	Special mode to load reduced image data. Load embedded preview image instead full size image
LoadAll	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).

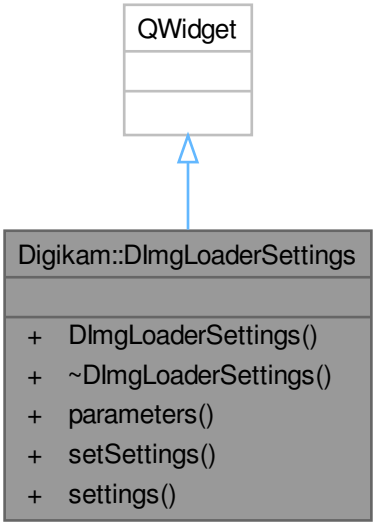
9.356.1.2 progressInfo()

```
virtual void Digikam::DImgLoaderObserver::progressInfo (
            float progress )  [inline], [virtual]
```

Reimplemented in Digikam::IccTransformFilter.

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

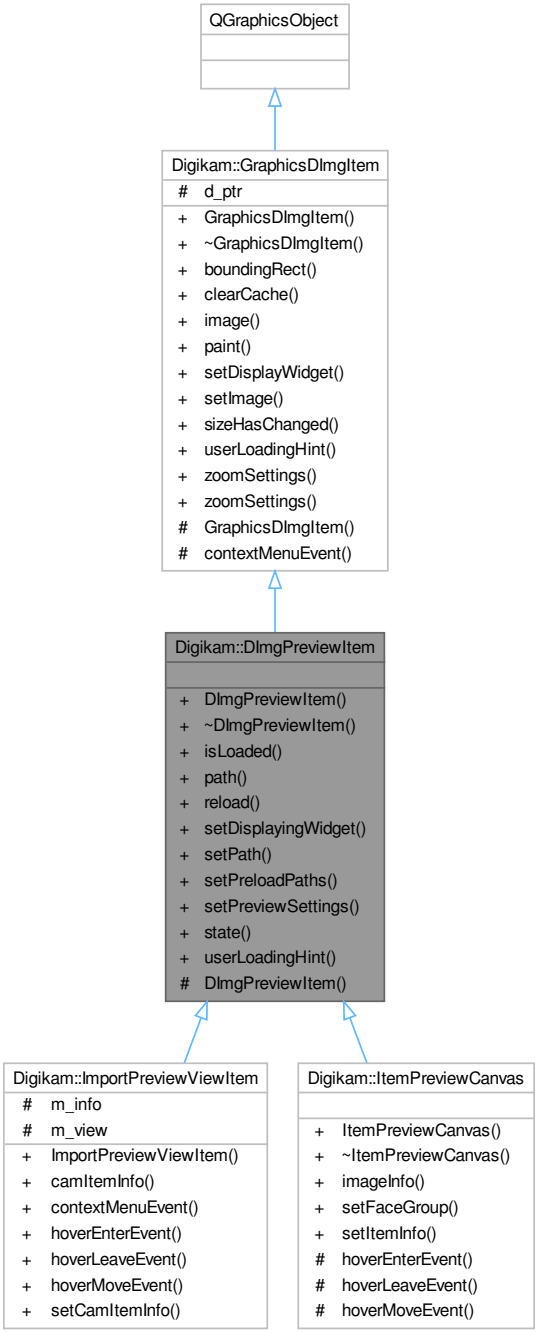
  Return the list of supported parameter names.
- virtual void **setSettings** (const DImgLoaderPrms &set)=0

  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 **isLoaded** () 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 ananlyser 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 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.

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()

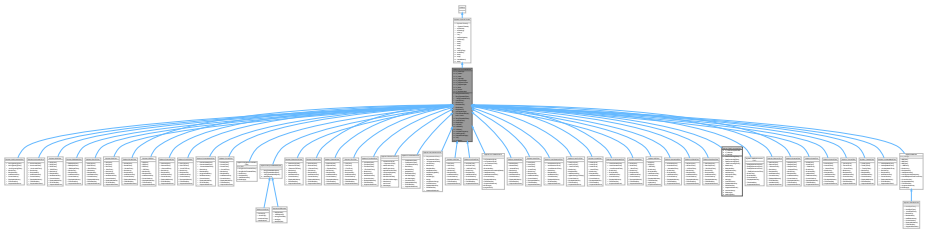
```
virtual void Digikam::DImgThreadedAnalyser::startAnalyse ( )  [protected], [pure virtual]
```

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 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

- 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 multithreading, 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 dest↩ Image. 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

success	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::IccTransformFilter.

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 **displayableName** () 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 **setDisplayableName** (const QString &displayableName)
- 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

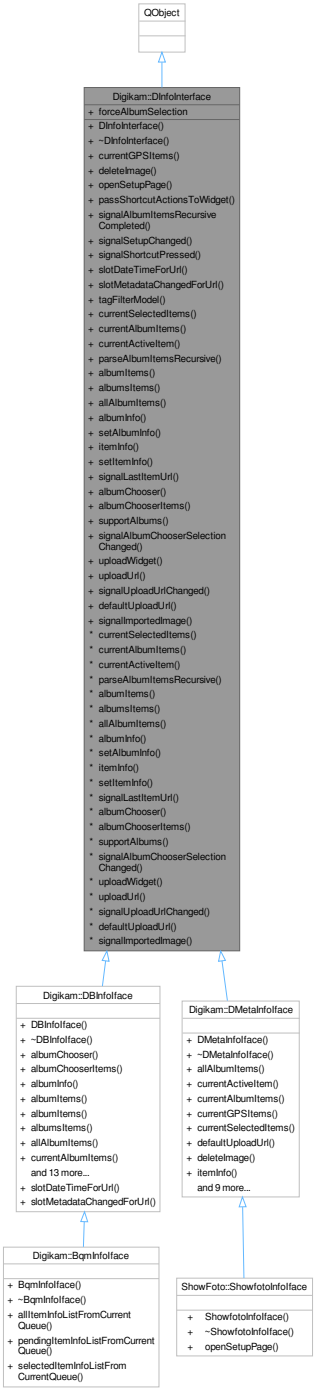
- enum Category {
  ReproducibleFilter = 0 , ComplexFilter = 1 , DocumentedHistory = 2 , **CategoryFirst** = ReproducibleFilter ,
  CategoryLast = DocumentedHistory }
- enum Flag { ExplicitBranch = 1 << 0 }
- typedef QFlags< Flag > **Flags**

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 > **DAlbumIDs**

  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::DBInfoIface.

9.362.1.2 currentSelectedItems()

```
QList< QUrl > Digikam::DInfoInterface::currentSelectedItems ( ) const  [virtual]
```

Reimplemented in Digikam::DBInfoIface, and Digikam::DMetaInfoIface.

9.362.1.3 defaultUploadUrl()

```
QUrl Digikam::DInfoInterface::defaultUploadUrl ( ) const  [virtual]
```

Reimplemented in Digikam::DBInfoIface, and Digikam::DMetaInfoIface.

9.362.1.4 deleteImage()

```
void Digikam::DInfoInterface::deleteImage (
            const QUrl & url )  [virtual]
```

Reimplemented in Digikam::DBInfoIface, and Digikam::DMetaInfoIface.

9.362.1.5 openSetupPage()

```
void Digikam::DInfoInterface::openSetupPage (
            SetupPage page )  [virtual]
```

Reimplemented in Digikam::DBInfoIface, and ShowFoto::ShowfotoInfoIface.

9.362.1.6 passShortcutActionsToWidget()

```
QMap< QString, QString > Digikam::DInfoInterface::passShortcutActionsToWidget (
            QWidget * const  ) const  [virtual]
```

Reimplemented in Digikam::DBInfoIface.

9.362.1.7 slotDateTimeForUrl()

```
void Digikam::DInfoInterface::slotDateTimeForUrl (
            const QUrl & url,
            const QDateTime & dt,
            bool updModDate )  [virtual]
```

Reimplemented in Digikam::DMetaInfoIface.

9.362.1.8 slotMetadataChangedForUrl()

```
void Digikam::DInfoInterface::slotMetadataChangedForUrl (
            const QUrl & url )  [virtual]
```

Reimplemented in Digikam::DMetaInfoIface.

9.362.1.9 tagFilterModel()

```
QAbstractItemModel * Digikam::DInfoInterface::tagFilterModel ( )  [virtual]
```

Reimplemented in Digikam::DBInfoIface.

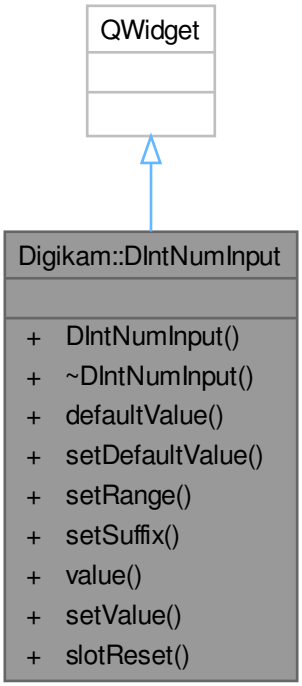
9.362.1.10 uploadWidget()

```
QWidget * Digikam::DInfoInterface::uploadWidget (
            QWidget *const parent ) const  [virtual]
```

Reimplemented in Digikam::DBInfoIface, and Digikam::DMetaInfoIface.

9.363 Digikam::DIntNumInput Class Reference

Inheritance diagram for Digikam::DIntNumInput:



Public Slots

- void **setValue** (int d)
- void **slotReset** ()

Signals

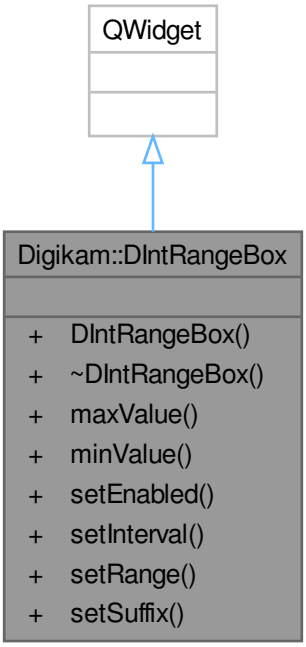
- void **reset** ()
- void **valueChanged** (int)

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

enabled	If the interval boxes should be enabled.

9.364.1.4 setInterval()

```
void Digikam::DIntRangeBox::setInterval (
            int min,
            int max )
```

Parameters

min	The minimum value of the interval.
max	The maximum value of the interval.

9.364.1.5 setRange()

```
void Digikam::DIntRangeBox::setRange (
            int min,
            int max )
```

Parameters

min	the lowest value to which the interval can be expanded.
max	the highest value to which the interval can be expanded.

9.364.1.6 setSuffix()

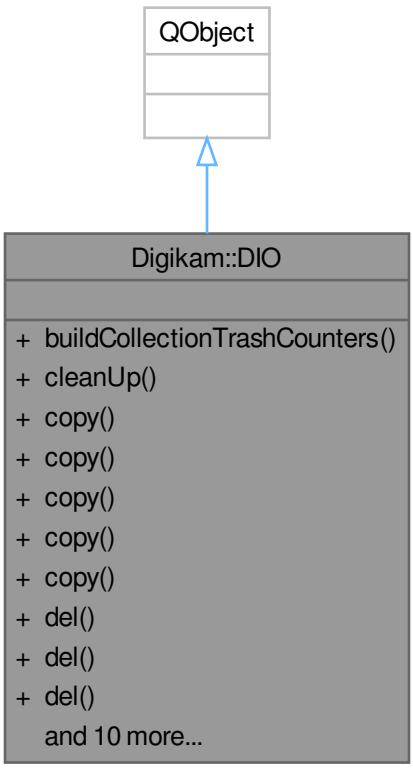
```
void Digikam::DIntRangeBox::setSuffix (
            const QString & suffix )
```

Parameters

suffix	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

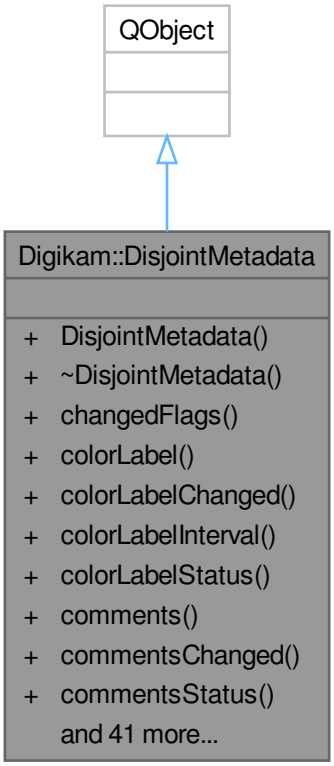
settings	contains settings
match	result of the regular expression match done in Option::parse()

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::
- void **setDataFields** (const DisjointMetadataDataFields &data)
- void **setDateTime** (const QDateTime &dateTime, DisjointMetadataDataFields::Status status=DisjointMetadataDataFields::Meta

  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::Metad
- 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::MetadataAv
- 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. • metadata fields changed • 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.
PartialWrite	Write only the changed parts. Metadata fields which cannot be changed from MetadataHub (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::Metadata -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::Metadata 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::Metadata -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::Metadata -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 (Disjoint↩ Metadata::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

MetadataInvalid	Not yet filled with any value.
MetadataAvailable	Only one data set has been added, or a common value is available.
MetadataDisjoint	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** ,

CilindricalHV , **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 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** ()

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::ShowfotoItemViewDe
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()

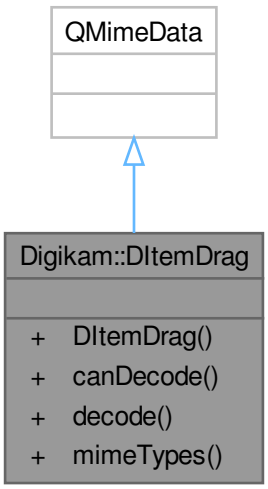
```
virtual void Digikam::DItemDelegate::setThumbnailSize (
            const ThumbnailSize & thumbSize )  [pure virtual]
```

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 > &image↩ IDs)

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∗ hostIface; // The host application interface instance.

DInfoInterface::DInfoMap info = hostIface->itemInfo(itemUrl); // First stage is to get the information map from host application. DItemInfo item(info); // Second stage, is to create the DIntenInfo 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.

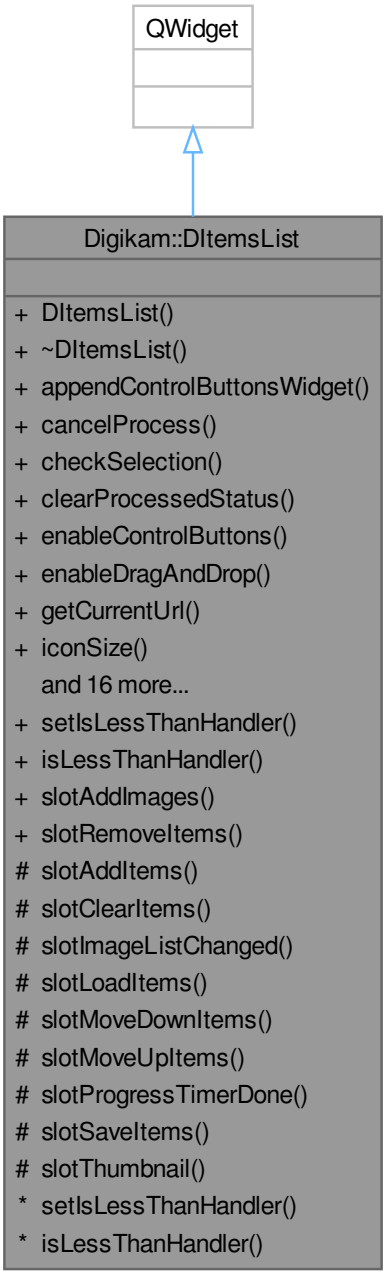
// WRITE INFO TO HOST ---—

QUrl itemUrl; // The item url that you want to retrieve information. DInfoInterface∗ hostIface; // The host application interface instance.

DItemInfo item; // Create the DIntenInfo 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. hostIface->set↩ ItemInfo(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)
- QBoxLayout * 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().

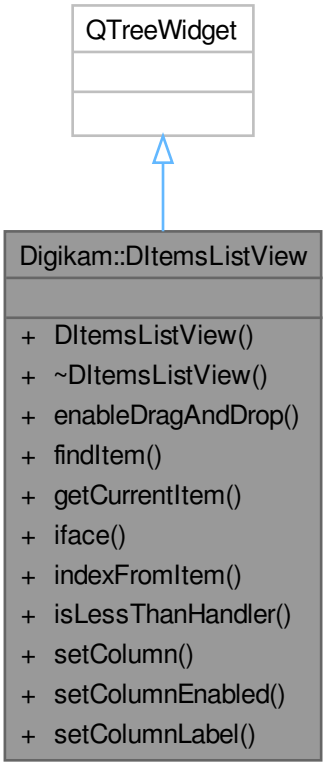
9.373.1.3 setIsLessThanHandler()

```
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 **signalAddedDropedItems** (const QList< QUrl > &)
- void **signalContextMenuRequested** ()
- void **signalItemClicked** (QTreeWidgetItem *)

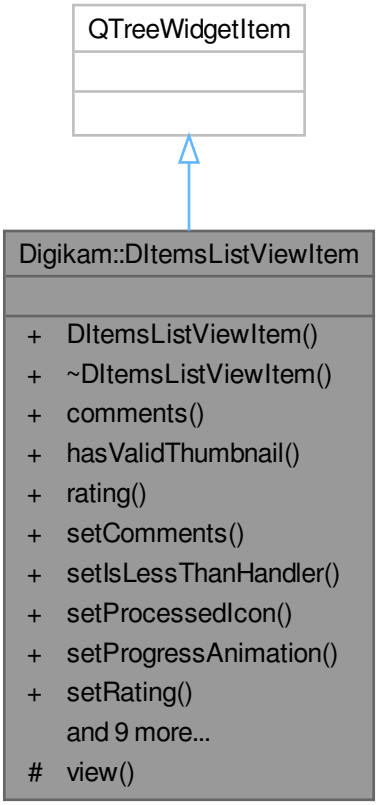
Public Member Functions

- **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

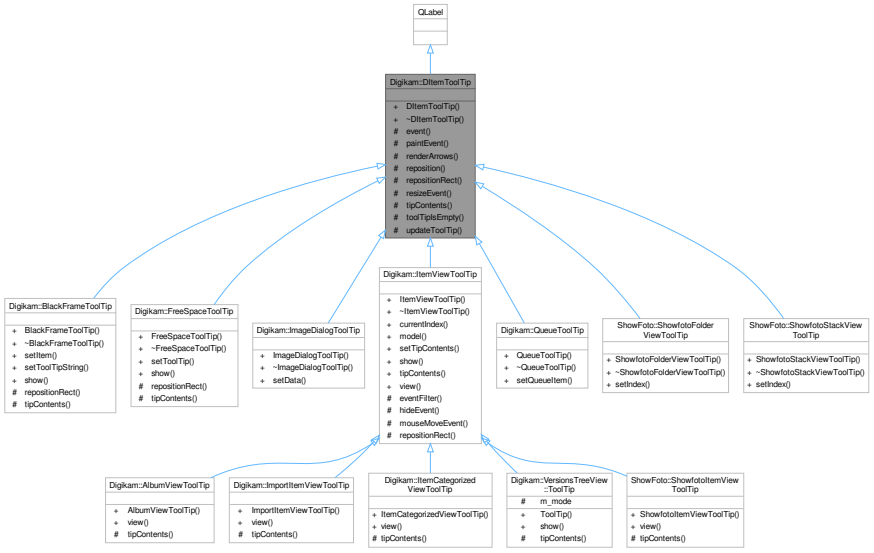
9.375.1.1 updateItemWidgets()

```
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 **toolTipIsEmpty** () const
- void **updateToolTip** ()

9.376.1 Member Function Documentation

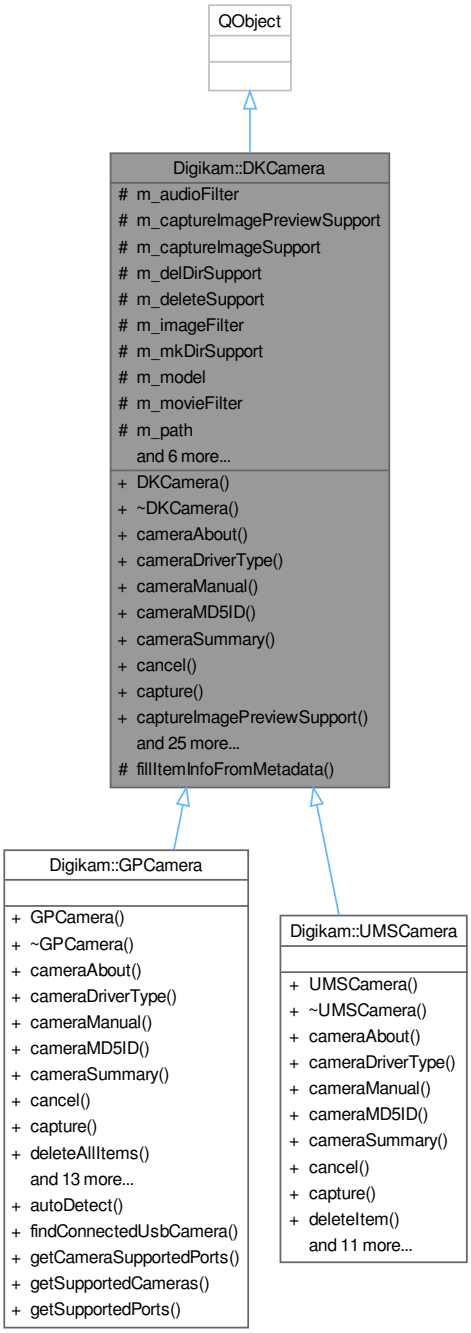
9.376.1.1 tipContents()

```
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 **deleteItem** (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.

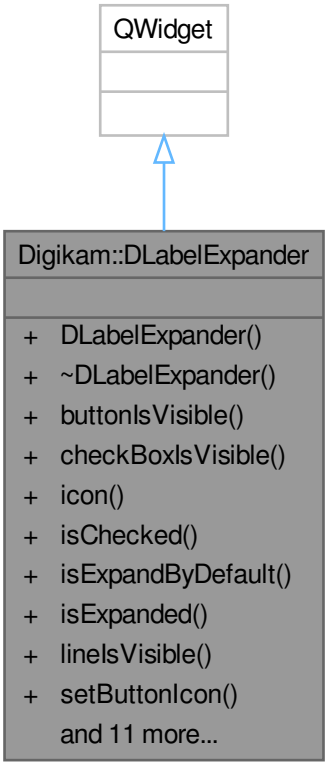
9.377.1.4 getPreview()

```
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

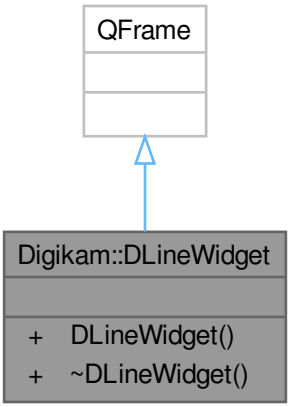
- **DLabelExpander** (QWidget ∗const parent=nullptr)
- bool **buttonIsVisible** () const
- bool **checkBoxIsVisible** () const
- QIcon **icon** () const
- bool **isChecked** () const
- bool **isExpandByDefault** () const
- bool **isExpanded** () const
- bool **lineIsVisible** () const
- void **setButtonIcon** (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:

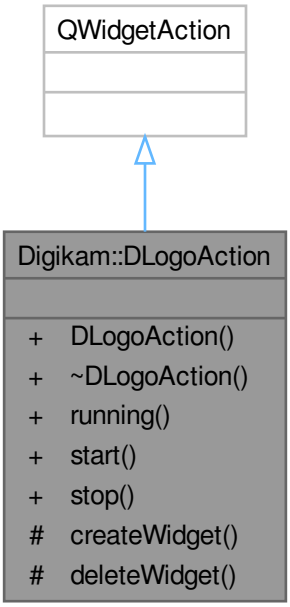


Public Member Functions

- **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

dontShowAgainName	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

dontShowAgainName	the name that identify the message box. If empty, this method does nothing.
value	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.

9.382 Digikam::DMetadata Class Reference

Inheritance diagram for Digikam::DMetadata:

Public Types

- typedef QMap< QString, QString > **CountryCodeMap**

- 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 **getItemUniqueId** () 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 **removeIptcTags** (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 DMetadataSettingsContainer &settings=DMetadataSettings::instance()->settings()) const
- bool **setItemComments** (const CaptionsMap &comments, const DMetadataSettingsContainer &settings=DMetadataSettings::in->settings()) const
- bool setItemFacesMap (const QMultiMap< 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 DMetadataSettingsContainer &settings=DMetadataSettings::instance()->settings()) const
- bool **setItemRating** (int rating, const DMetadataSettingsContainer &settings=DMetadataSettings::instance()->settings()) const
- bool **setItemTagsPath** (const QStringList &tagsPath, const DMetadataSettingsContainer &settings=DMetadataSettings::instanc->settings()) const
- bool **setItemTitles** (const CaptionsMap &title, const DMetadataSettingsContainer &settings=DMetadataSettings::instance()->settings()) const
- bool **setItemUniqueId** (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 setDateTimeDigitized=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 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'.

- 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 from "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 (
            QMultiMap< 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 metadatainfo.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 QMultiMap< QString, QVariant > & facesPath,
            bool write,
            const QSize & size = QSize() ) const
```

Parameters

facesPath	The face map to register in metadata based on tags stored in Picassa/Metadatagroup
write	If true all faces will be written, else update mode search if at least a face tag exist and write if true.
size	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.

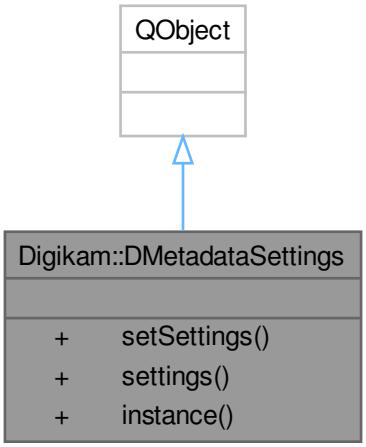
9.382.2.21 valueToString()

```
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::DMetadataSettings Class Reference

Inheritance diagram for Digikam::DMetadataSettings:



Signals

- void **signalDMetadataSettingsChanged** (const DMetadataSettingsContainer &current, const DMetadataSettingsContainer &previous)
- void **signalSettingsChanged** ()

Public Member Functions

- void **setSettings** (const DMetadataSettingsContainer &settings)

  Sets the current Metadata settings and writes them to config.
- DMetadataSettingsContainer **settings** () const

  Returns the current Metadata settings.

Static Public Member Functions

- static DMetadataSettings * instance ()

  Global container for Metadata settings.

Friends

- class **DMetadataSettingsCreator**

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::DMetaInfoIface Class Reference

Inheritance diagram for Digikam::DMetaInfoIface:

Public Member Functions

- **DMetaInfoIface** (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 > **albumsItems** (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()

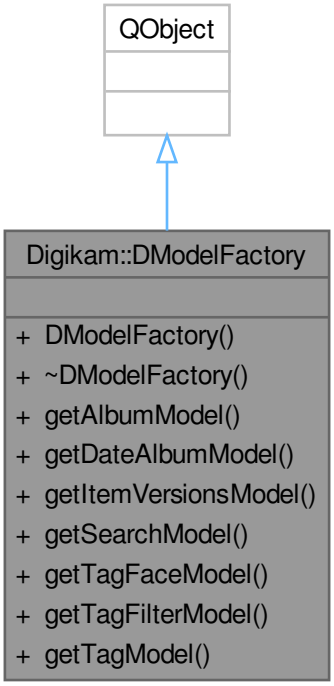
```
QWidget * Digikam::DMetaInfoIface::uploadWidget (
            QWidget *const parent ) const  [override], [virtual]
```

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

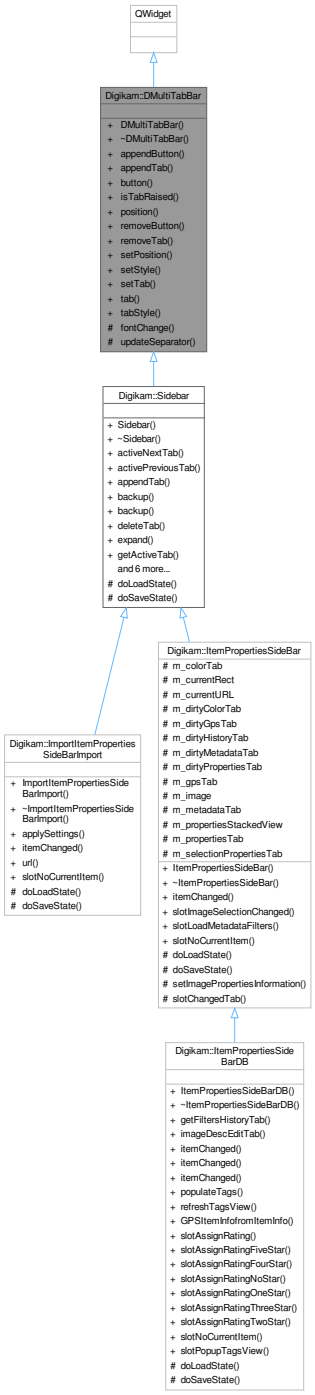
Author

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

pic	a icon for the button
id	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.
popup	A popup menu which should be displayed if the button is clicked
not_used_yet	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

pic	a icon for the tab
id	an arbitrary ID which can be used later on to identify the tab
text	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

pos	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

id	The ID of the tab to manipulate
state	true == activated/raised, false == not active

9.387.2.6 tabStyle()

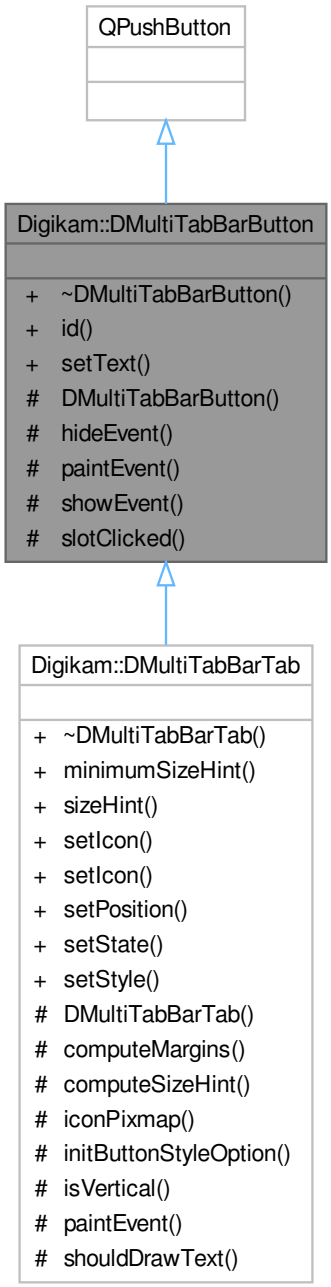
```
DMultiTabBar::TextStyle Digikam::DMultiTabBar::tabStyle ( ) const
```

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

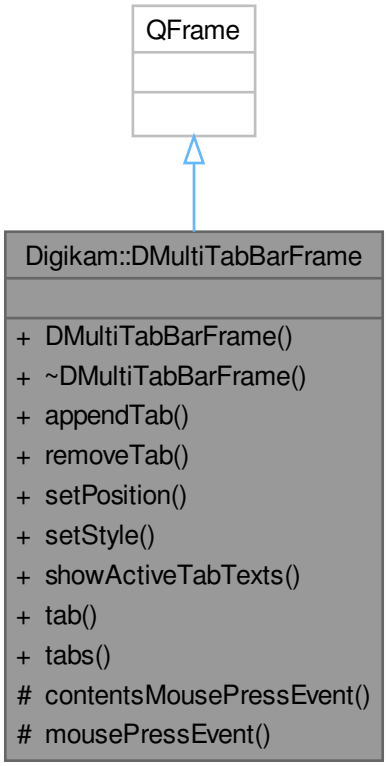
```
void Digikam::DMultiTabBarButton::signalClicked (
            int id )  [signal]
```

Parameters

id	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 **contentsMousePressEvent** (QMouseEvent ∗)

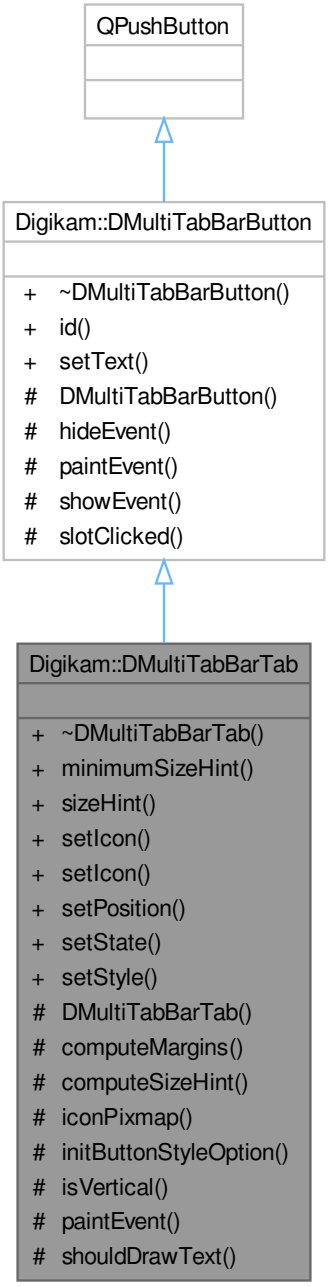
  Reimplemented from QScrollView in order to ignore all mouseEvents on the viewport, so that the parent can handle them.
- void **mousePressEvent** (QMouseEvent ∗) override

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

state	true==active false==not active

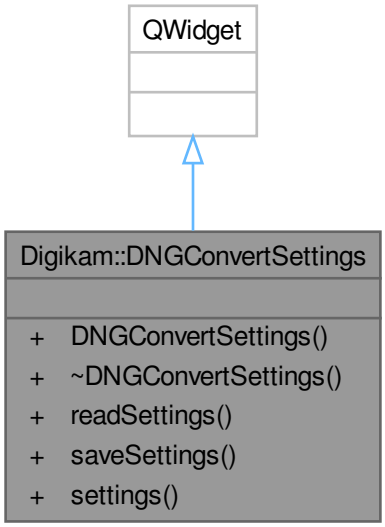
9.390.1.3 setStyle

```
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

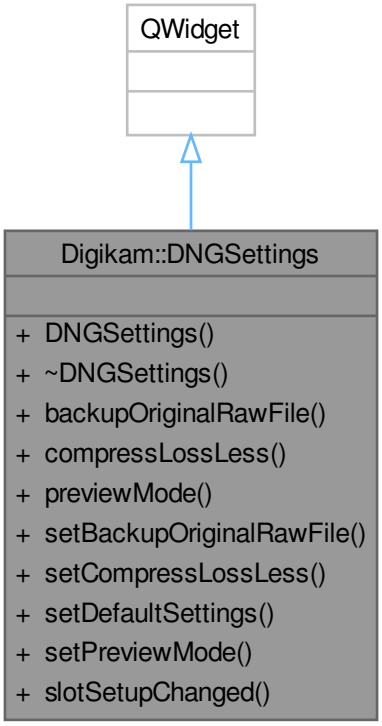
- void **signalDownloadNameChanged** ()

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

PROCESS_CONTINUE	Current stages is done.
PROCESS_COMPLETE	All stages done.
PROCESS_FAILED	A failure happen while processing.
PROCESS_CANCELED	User has canceled processing.
FILE_NOT_SUPPORTED	Raw file format is not supported by converter.
DNG_SDK_INTERNAL_ERROR	Adobe DNG SDK has generated an error while processing.

9.393.1.2 JPEGPreview

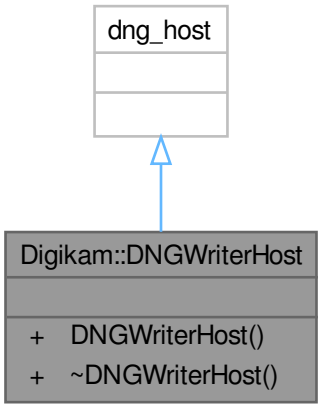
enum Digikam::DNGWriter::JPEGPreview

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:

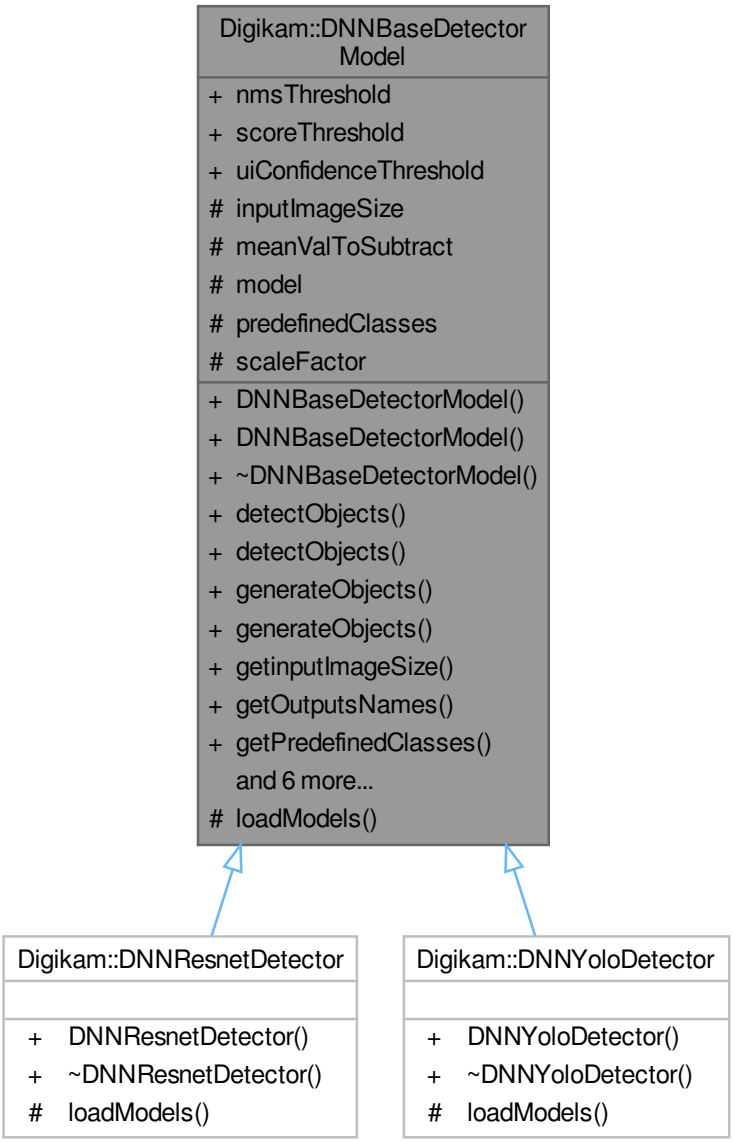


Public Member Functions

- **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 QHash< QString, QVector< QRect > > **postprocess** (const cv::Mat &inputImage, const cv::Mat &out) const =0
- QList< QHash< QString, QVector< QRect > > > **postprocess** (const std::vector< cv::Mat > &input↩ BatchImages, 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

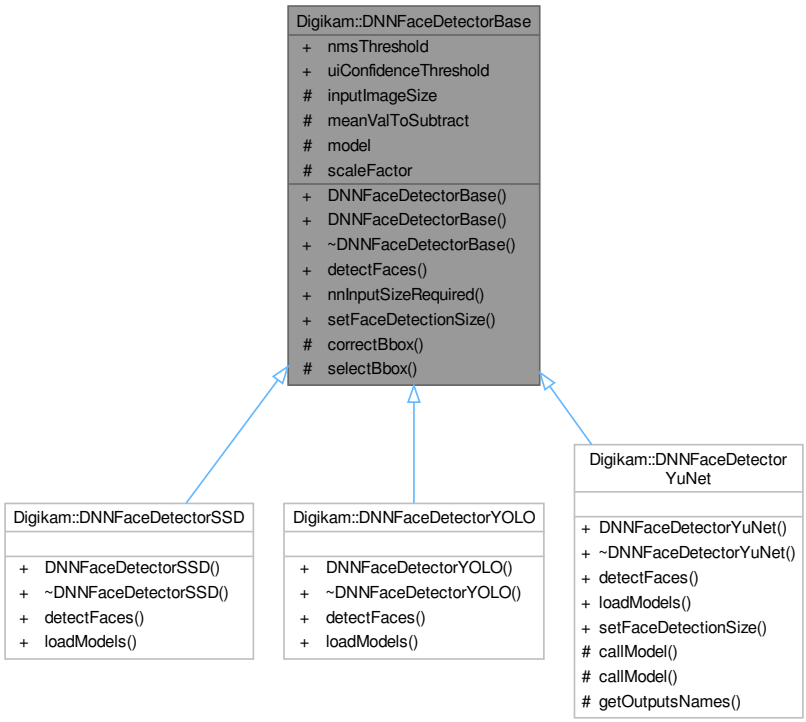
9.395.1.1 uiConfidenceThreshold

```
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()

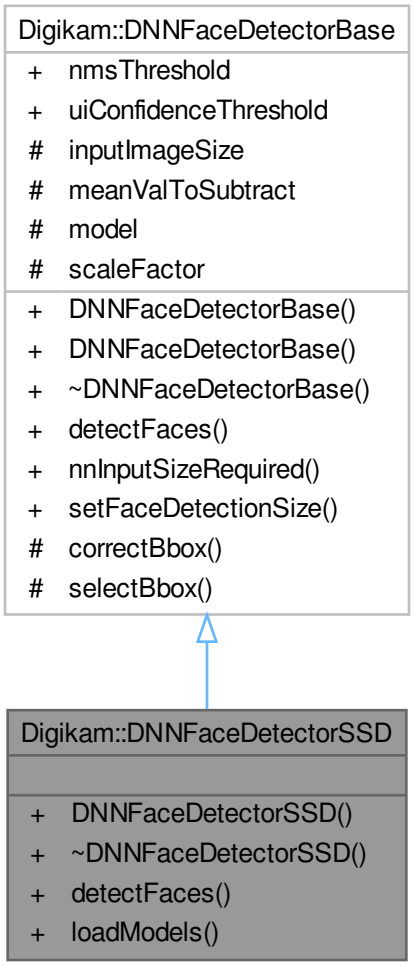
```
void Digikam::DNNFaceDetectorBase::selectBbox (
            const cv::Size & paddedSize,
            float confidence,
            int left,
            int right,
            int top,
            int bottom,
            std::vector< float > & goodConfidences,
            std::vector< cv::Rect > & 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

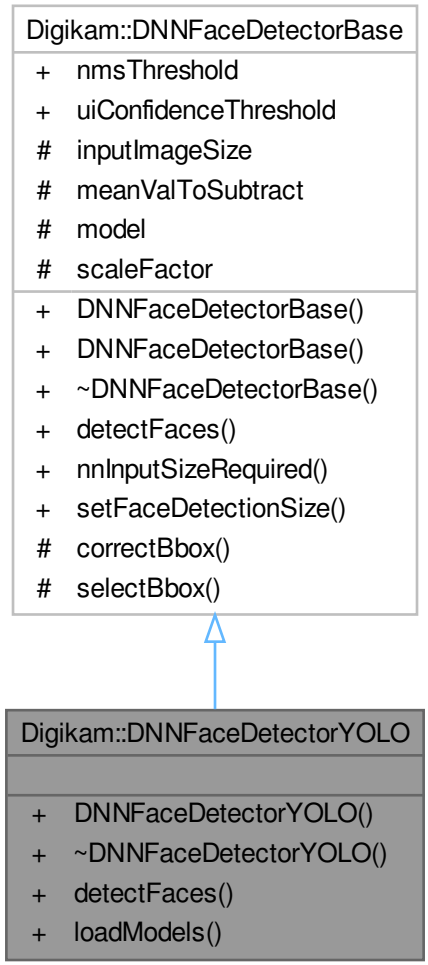
9.397.1.1 detectFaces()

```
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

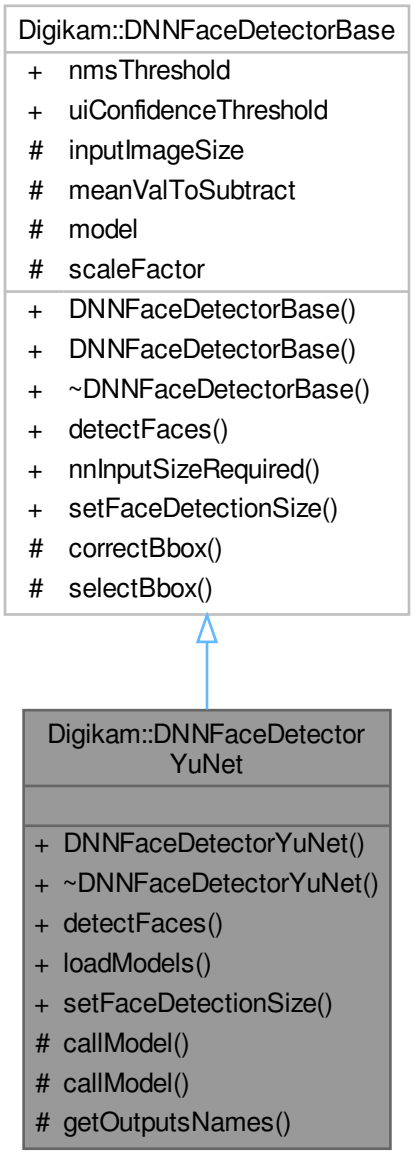
9.398.1.1 detectFaces()

```
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.

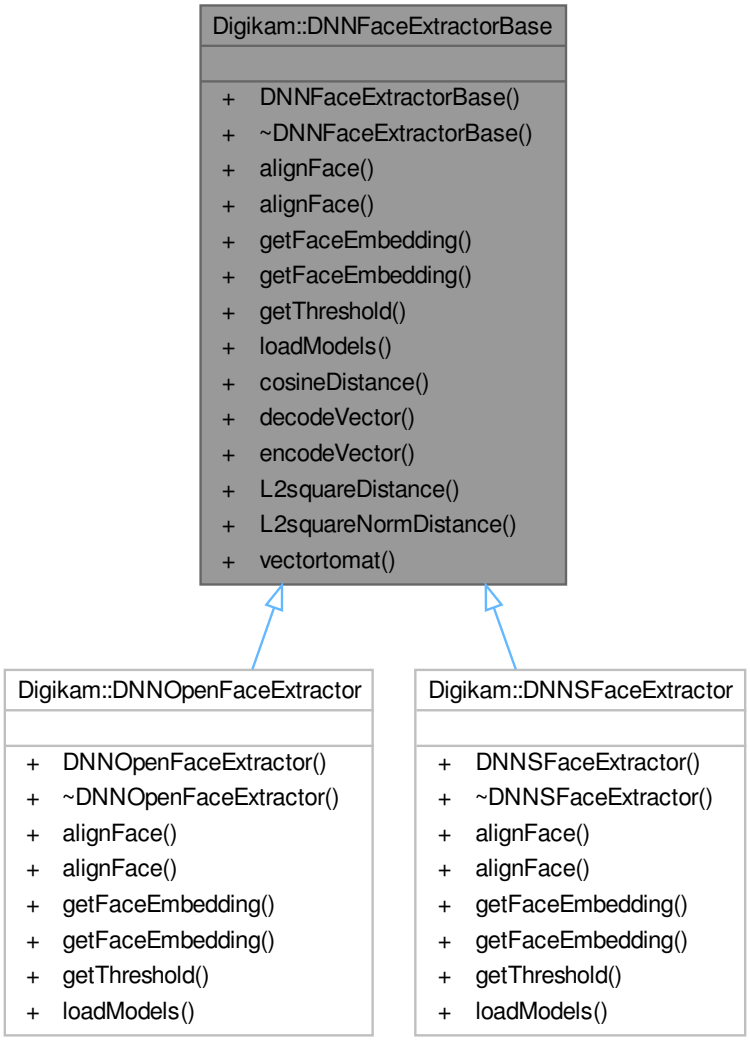
9.399.1.2 setFaceDetectionSize()

```
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::Mat **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.

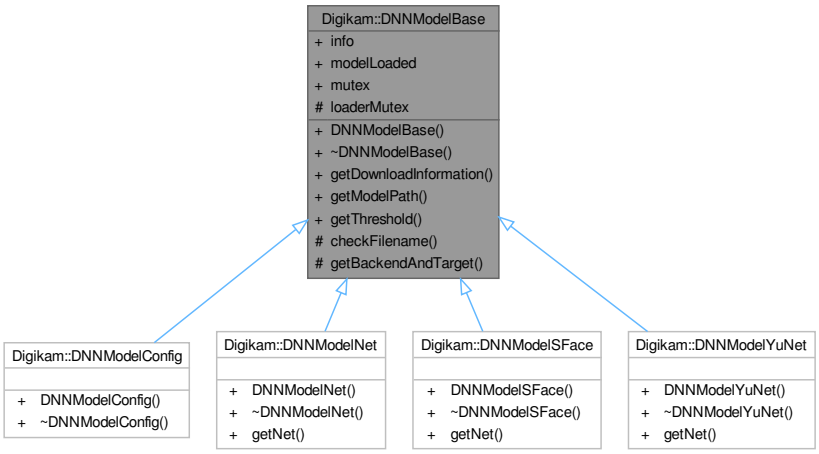
9.400.1.2 loadModels()

```
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

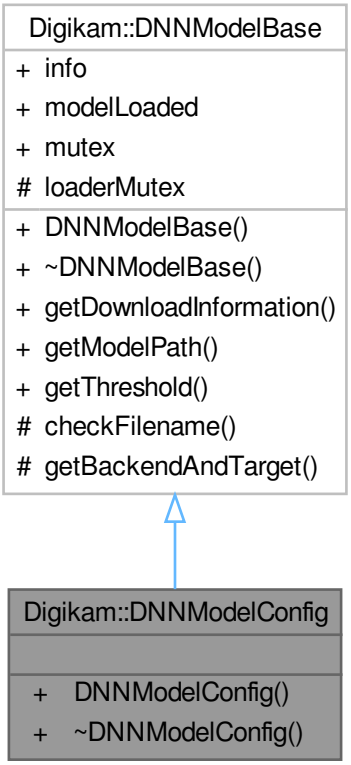
9.401.1.1 getThreshold()

```
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 DNNModel↩ UsageList &_usage, const QVersionNumber &_minVersion, const QString &_downloadPath, const QString &_sha256, const qint64 &_fileSize, int _defaultThreshold, int _minUsableThreshold, int _maxUsable↩ Threshold, 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 loder 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**

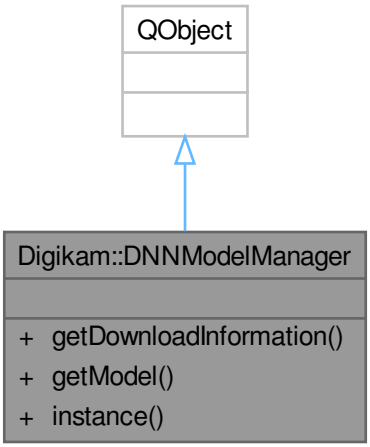
  Minimum version of digiKam needed to use this model.
- QString **sha256**

  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.

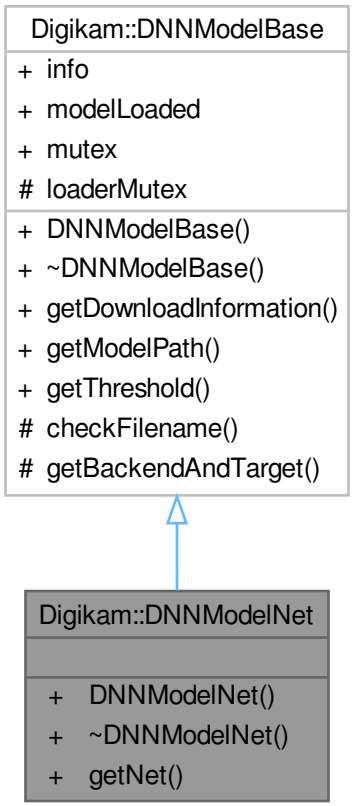
9.404.1.2 instance()

```
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

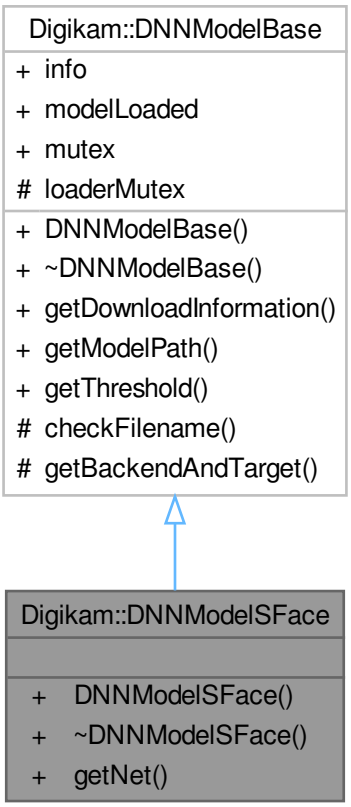
- bool **checkFilename** () const
- const QPair< int, int > **getBackendAndTarget** () const

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

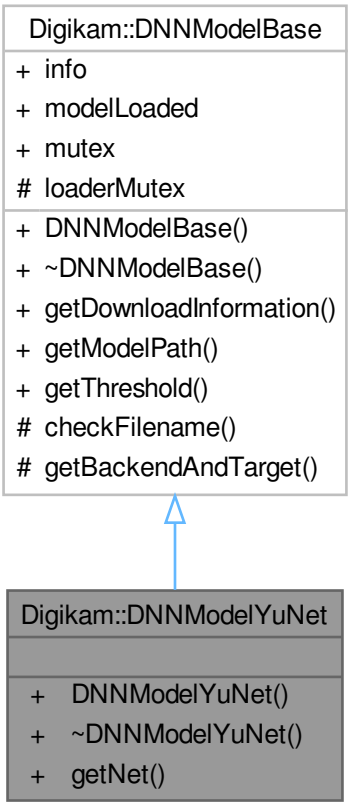
- bool **checkFilename** () const
- const QPair< int, int > **getBackendAndTarget** () const

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

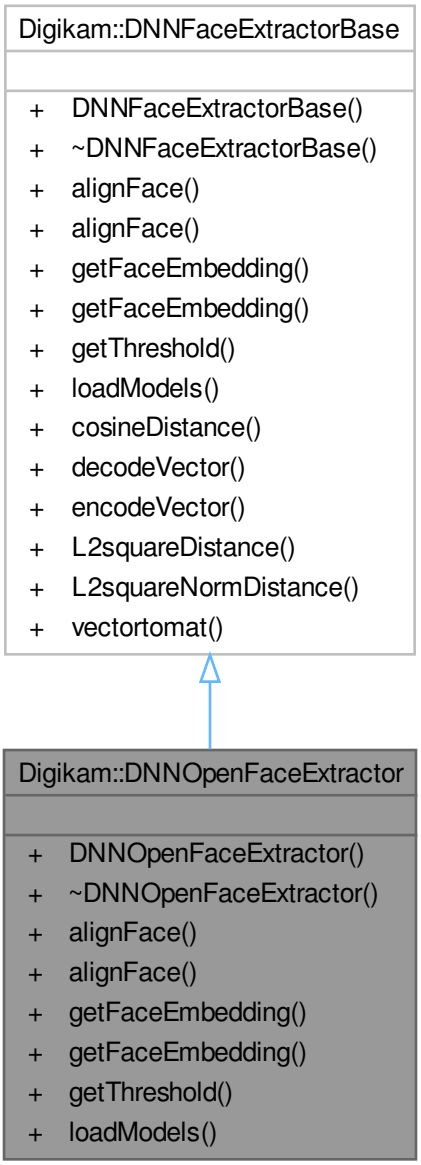
- bool **checkFilename** () const
- const QPair< int, int > **getBackendAndTarget** () const

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.

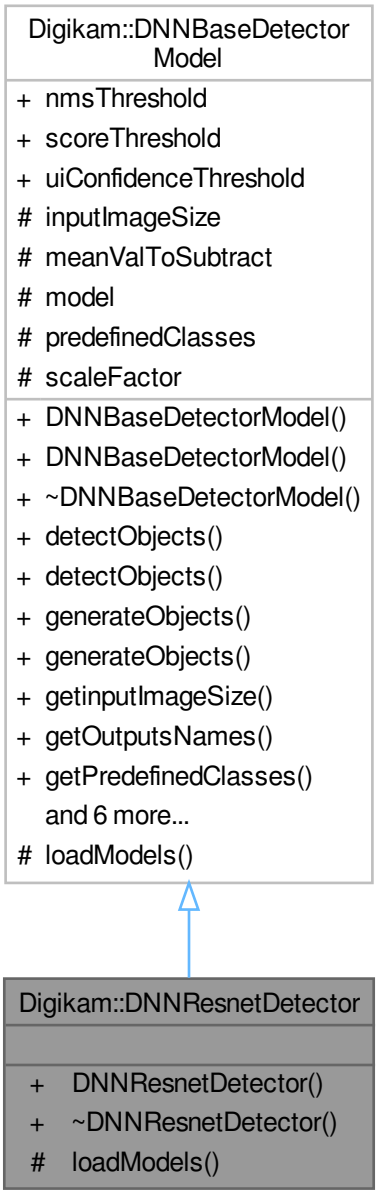
9.408.1.6 loadModels()

```
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 > &input↩ BatchImages, 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

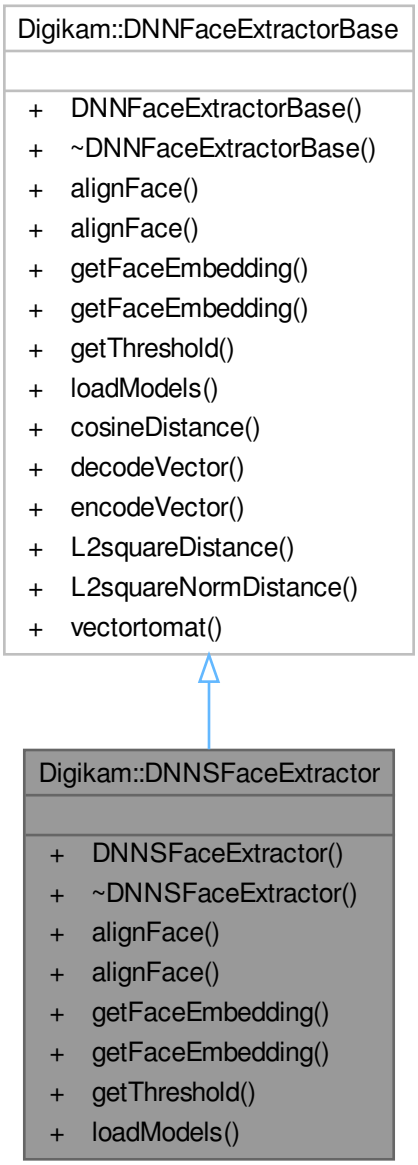
9.409.1.1 loadModels()

```
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 &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.

9.410.1 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.

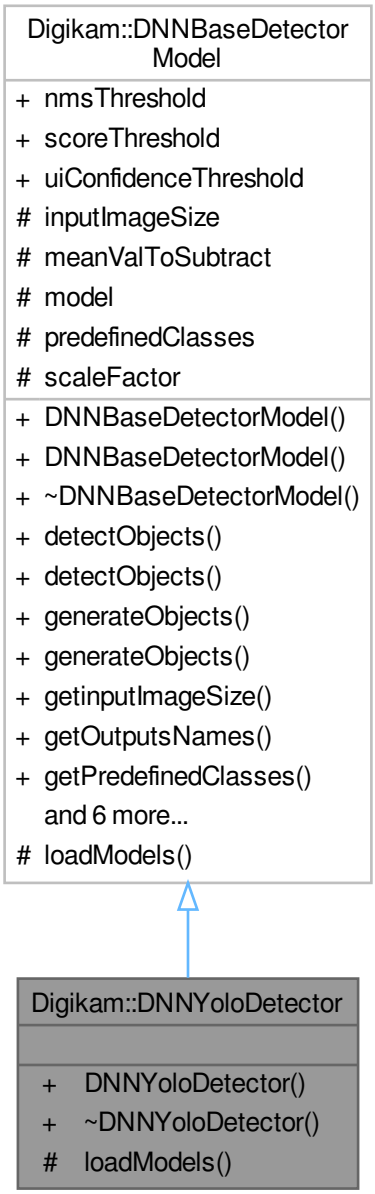
9.410.1.6 loadModels()

```
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 > &input↩BatchImages, 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 QSystem↩ TrayIcon.
- 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

caption	The window caption (title) on the popup
text	The text for the popup
icon	The icon to use for the popup
parent	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 *)

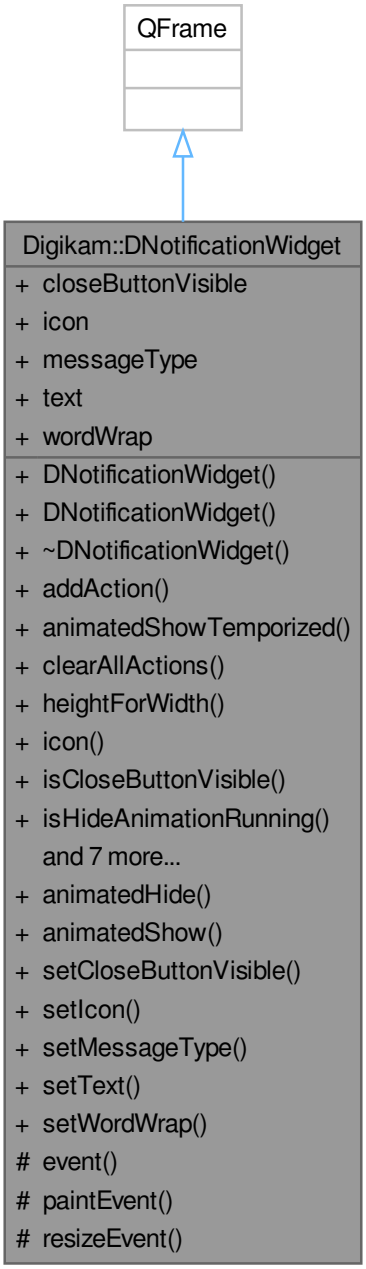
setView(const QString&, const QString&)

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

action	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

width	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

contents	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

contents	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

action	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

text	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

wordWrap	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.

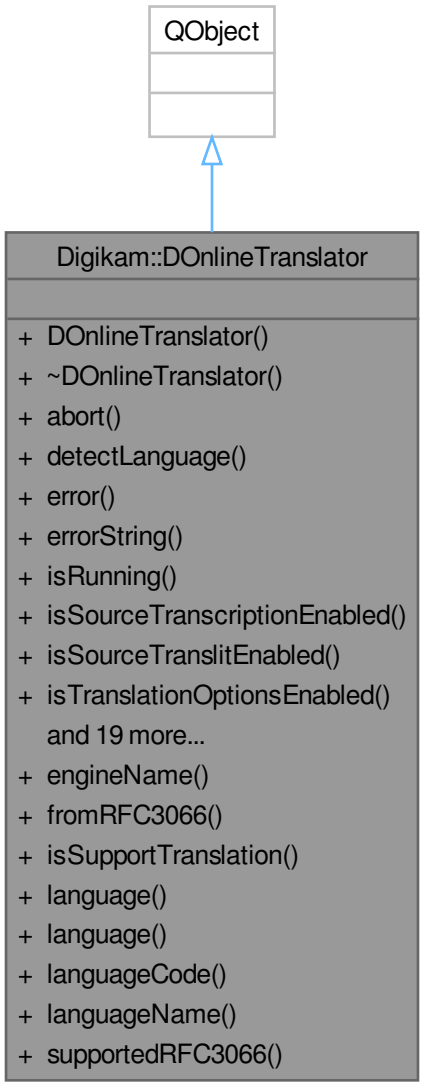
See also

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** , **Papiamento** ,
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

parent	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

text	the text for language detection
engine	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

engine	the engine to use
lang	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

locale	the locale to use

Returns

language

9.414.3.11 language() [2/2]

```
DOnlineTranslator::Language Digikam::DOnlineTranslator::language (
            const QString & langCode )  [static]
```

Parameters

langCode	code

Returns

language

9.414.3.12 languageCode()

```
QString Digikam::DOnlineTranslator::languageCode (
            Language lang )  [static]
```

Parameters

lang	language

Returns

language code

9.414.3.13 languageName()

```
QString Digikam::DOnlineTranslator::languageName (
            Language lang )  [static]
```

Parameters

lang	language

Returns

language name

9.414.3.14 setEngineApiKey()

```
void Digikam::DOnlineTranslator::setEngineApiKey (
            Engine engine,
            const QByteArray & apiKey )
```

Affects only LibreTranslate.

Parameters

engine	the engine to use
apiKey	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

engine	the engine to use
url	engine url

9.414.3.16 setSourceTranscriptionEnabled()

```
void Digikam::DOnlineTranslator::setSourceTranscriptionEnabled (
```

```
            bool enable )
```

Parameters

enable	whether to enable source transcription

9.414.3.17 setSourceTranslitEnabled()

```
void Digikam::DOnlineTranslator::setSourceTranslitEnabled (
            bool enable )
```

Parameters

enable	whether to enable source transliteration

9.414.3.18 setTranslationOptionsEnabled()

```
void Digikam::DOnlineTranslator::setTranslationOptionsEnabled (
            bool enable )
```

Parameters

enable	whether to enable translation options

See also

DOnlineTranslatorOption

9.414.3.19 setTranslationTranslitEnabled()

```
void Digikam::DOnlineTranslator::setTranslationTranslitEnabled (
            bool enable )
```

Parameters

enable	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

text	the text to translate
engine	online engine to use
translationLang	language to translation
sourceLang	language of the passed text
uiLang	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

- QJsonObject 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]
```

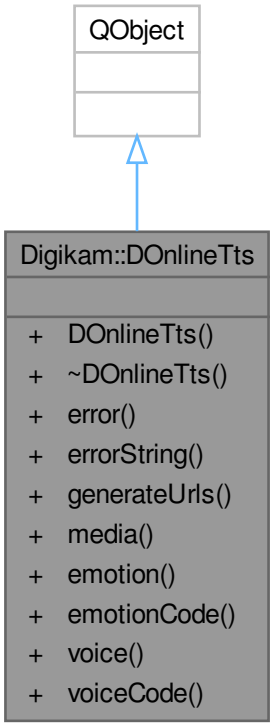
Returns

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

parent	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

emotionCode	emotion code

Returns

corresponding emotion

9.416.4.2 emotionCode()

```
QString Digikam::DOnlineTts::emotionCode (
            Emotion emotion )  [static]
```

Used only by Yandex.

Parameters

emotion	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

text	the text to speak
engine	online translation engine
lang	text language
voice	the voice to use (used only by Yandex)
emotion	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

voiceCode	voice code

Returns

corresponding voice

9.416.4.8 voiceCode()

```
QString Digikam::DOnlineTts::voiceCode (
            Voice voice )  [static]
```

Parameters

voice	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()

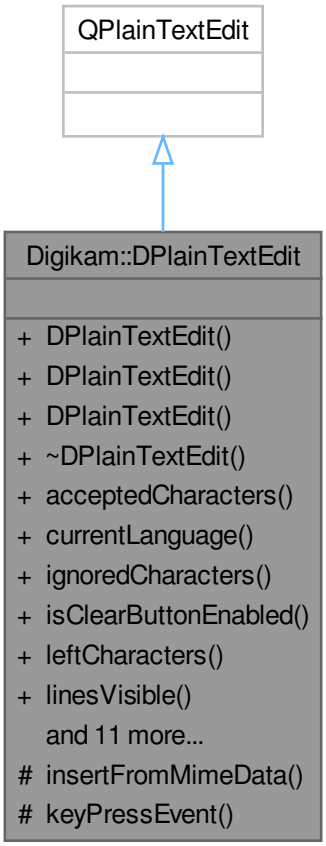
```
void Digikam::DPixelsAliasFilter::pixelAntiAliasing16 (
            unsigned short *const data,
            int Width,
            int Height,
            double X,
            double Y,
            unsigned short *const A,
            unsigned short *const R,
            unsigned short *const G,
            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 ()

  Emmited 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 QLine↩ Edit. 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 depedencies 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()

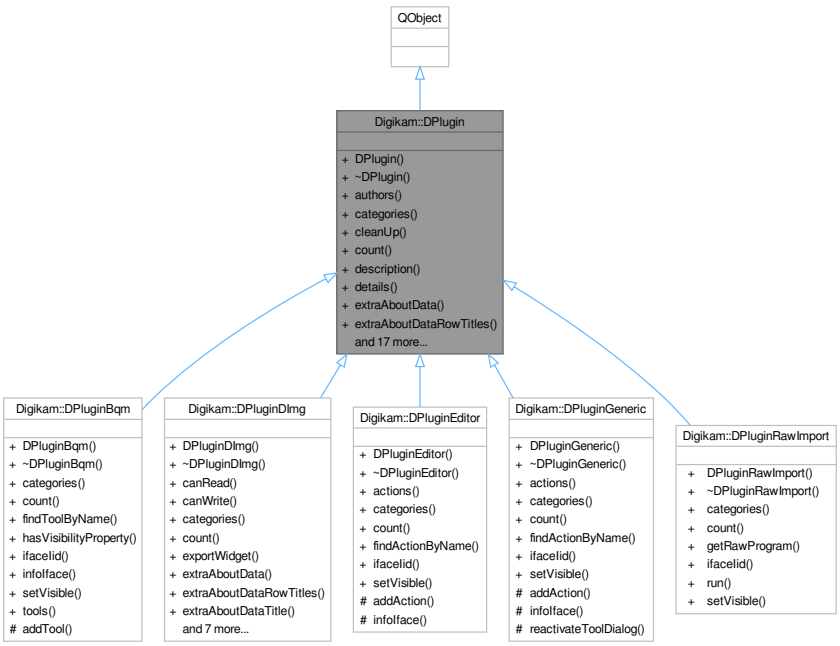
```
QString Digikam::DPlainTextEdit::text ( ) const
```

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 ifaceIid () 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.

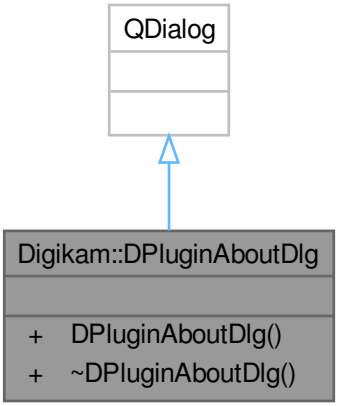
9.421.1.20 version()

```
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:

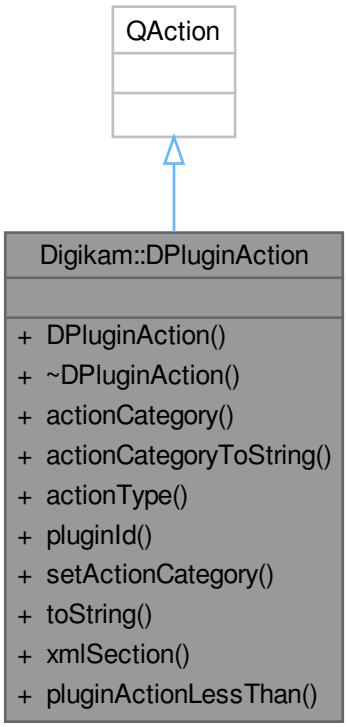


Public Member Functions

- **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 adjustement 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 &_nane, 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

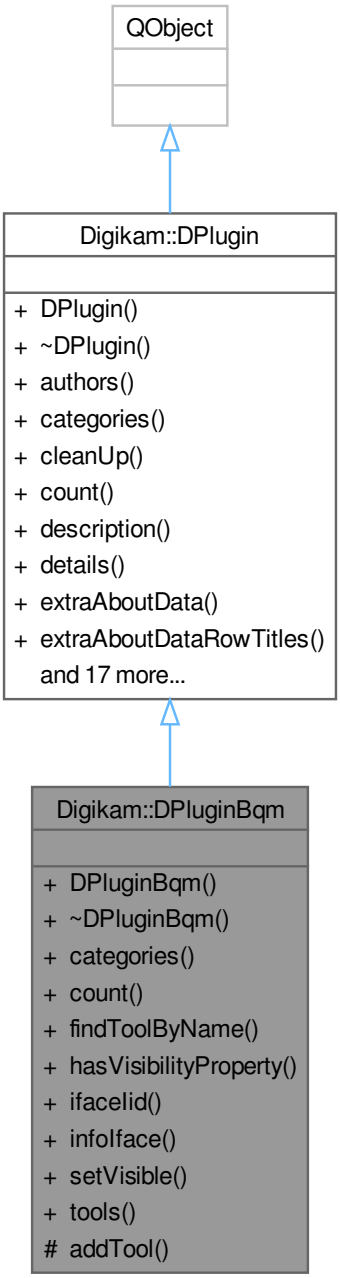
9.424.1.1 toString()

```
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 ifaceIid () const override

  Return the plugin interface identifier.
- BqmInfoIface ∗ **infoIface** () 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 ifaceIid()

```
QString Digikam::DPluginBqm::ifaceIid ( ) const  [override], [virtual]
```

Implements Digikam::DPlugin.

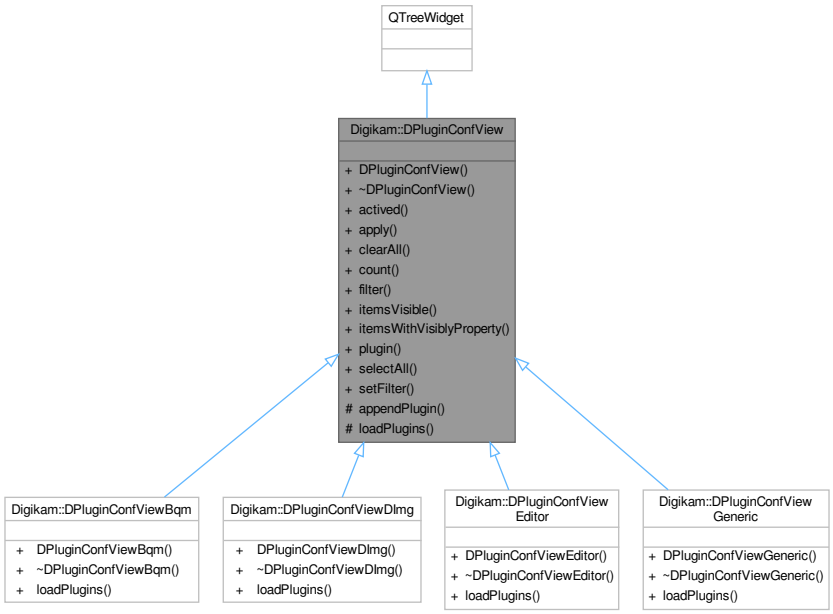
9.425.1.5 setVisible()

```
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 **actived** () 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 **actived** () 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

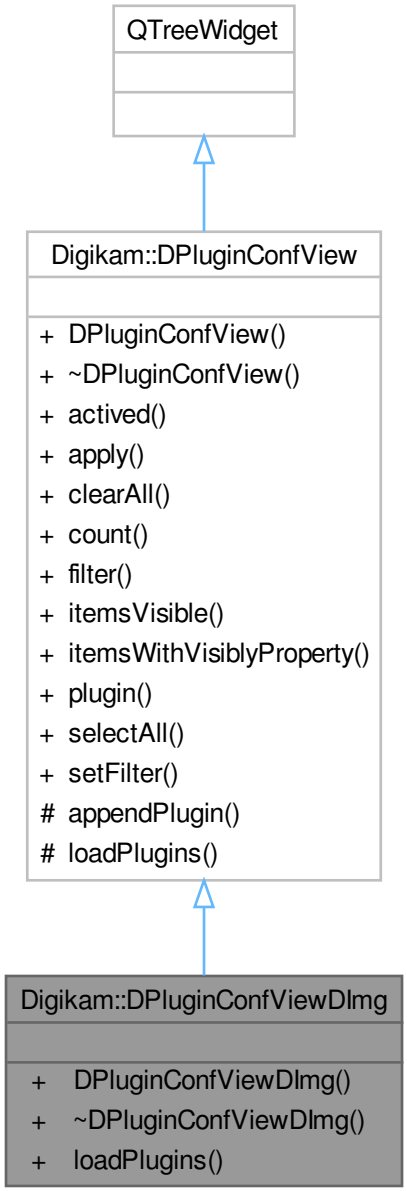
9.427.1.1 loadPlugins()

```
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 **actived** () 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 **actived** () 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 **actived** () 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

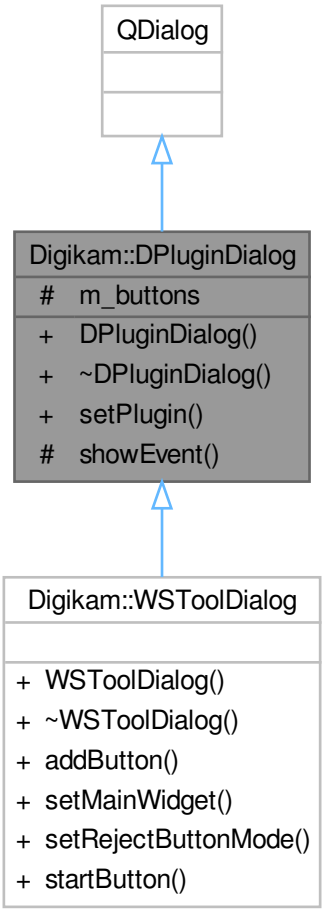
9.430.1.1 loadPlugins()

```
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

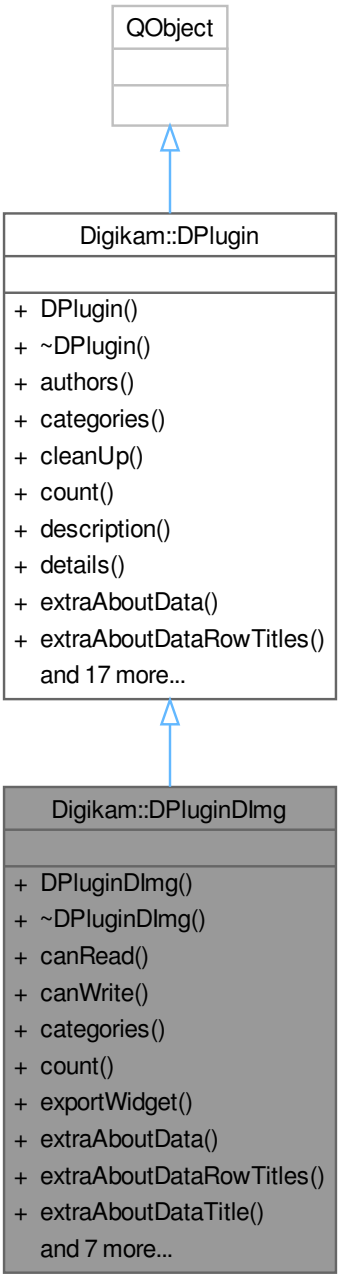
- void **showEvent** (QShowEvent ∗) override

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 ifaceIid () 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 ifaceIid()

```
QString Digikam::DPluginDImg::ifaceIid ( ) 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.

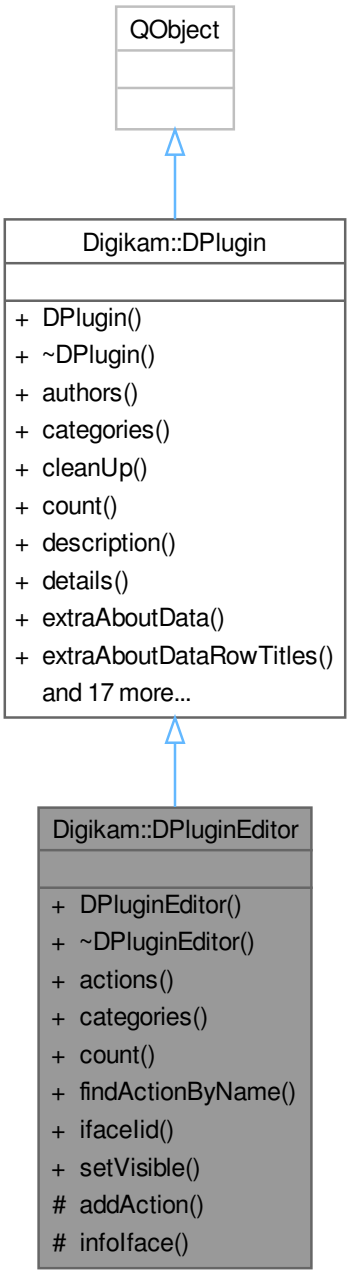
9.432.1.13 typeMimes()

```
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 ifaceIid () 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 ∗ **infoIface** (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 ifaceIid()

```
QString Digikam::DPluginEditor::ifaceIid ( ) const  [inline], [override], [virtual]
```

Implements Digikam::DPlugin.

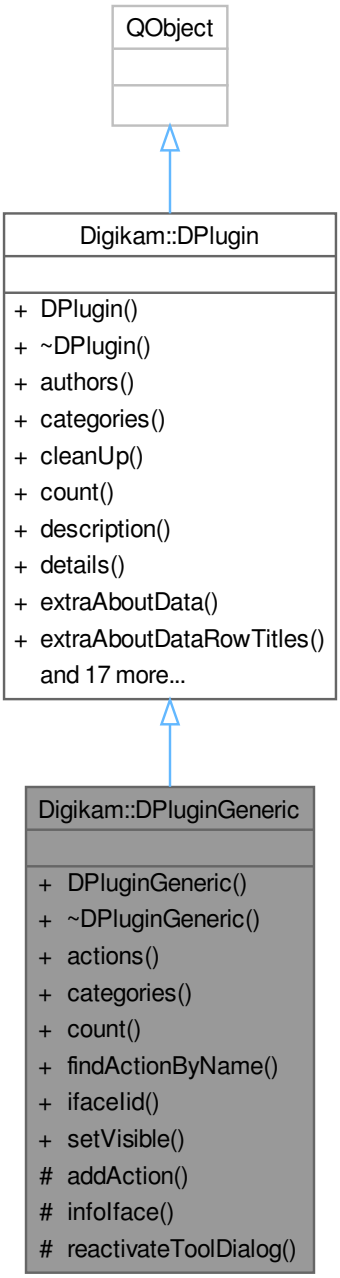
9.433.1.4 setVisible()

```
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 ifaceIid () 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 ∗ **infoIface** (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::ifaceIid ( ) const  [inline], [override], [virtual]
```

Implements Digikam::DPlugin.

9.434.1.4 setVisible()

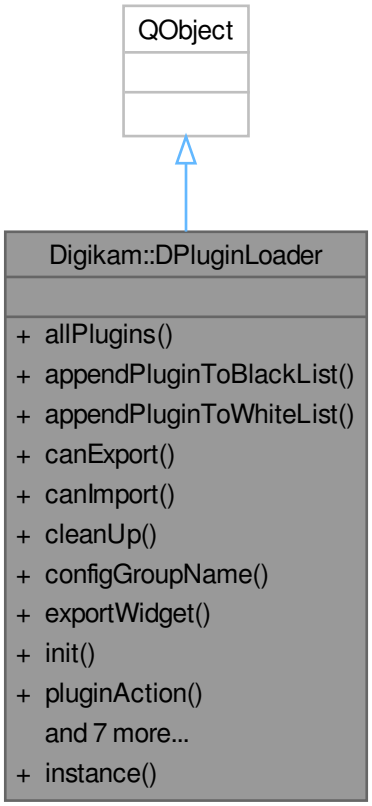
```
void Digikam::DPluginGeneric::setVisible (
            bool b )  [override], [virtual]
```

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

filename	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

filename	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.

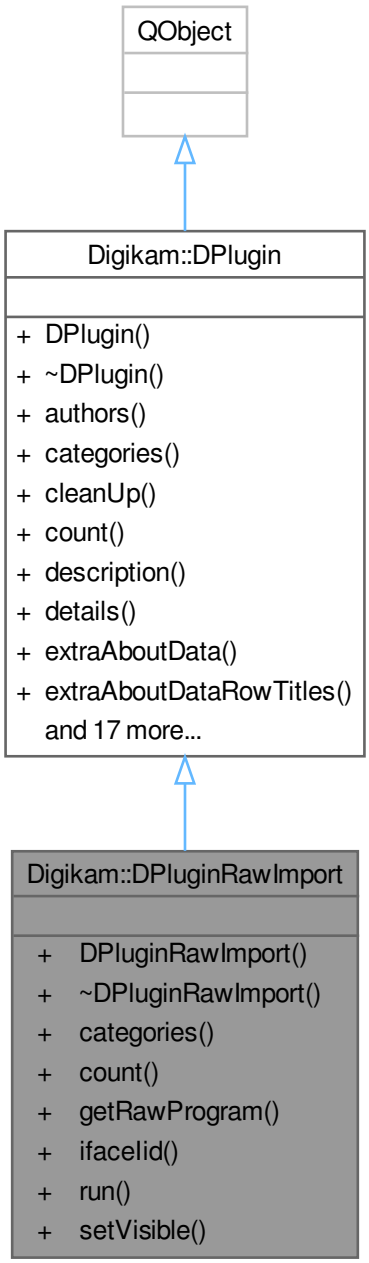
9.435.2.10 pluginsActions() [2/2]

```
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 ifaceIid () 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::ifaceIid ( ) const  [inline], [override], [virtual]
```

Implements Digikam::DPlugin.

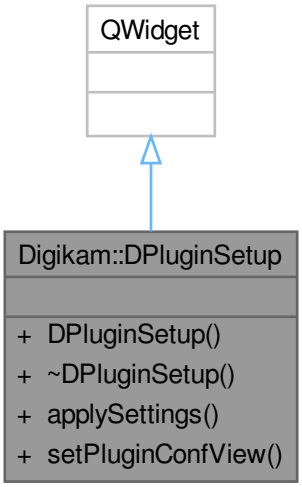
9.436.1.4 setVisible()

```
void Digikam::DPluginRawImport::setVisible (
            bool  )  [inline], [override], [virtual]
```

Implements Digikam::DPlugin.

9.437 Digikam::DPluginSetup Class Reference

Inheritance diagram for Digikam::DPluginSetup:

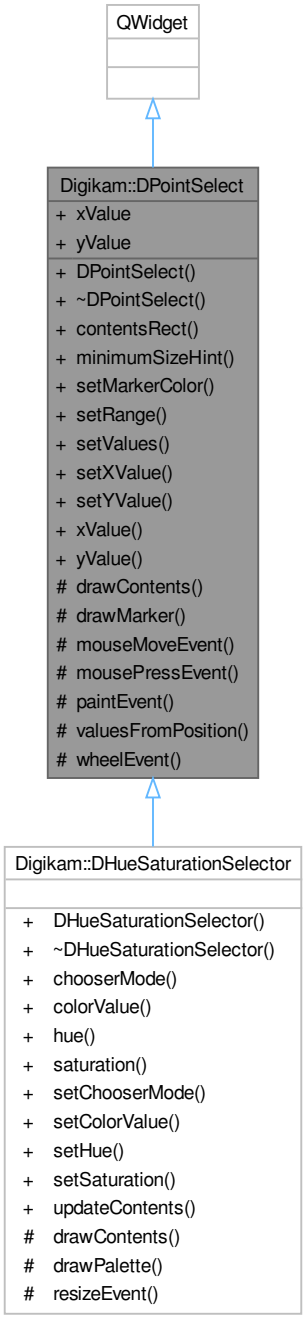


Public Member Functions

- **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

col	the color

9.438.1.4 setValues()

```
void Digikam::DPointSelect::setValues (
            int xPos,
            int yPos )
```

Parameters

xPos	the horizontal value
yPos	the vertical value

9.438.1.5 setXValue()

```
void Digikam::DPointSelect::setXValue (
            int xPos )
```

Parameters

xPos	the horizontal value

9.438.1.6 setYValue()

```
void Digikam::DPointSelect::setYValue (
            int yPos )
```

Parameters

yPos	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()

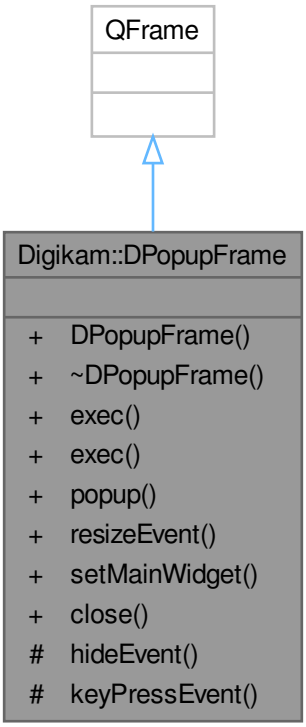
```
int Digikam::DPointSelect::yValue ( ) const
```

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.

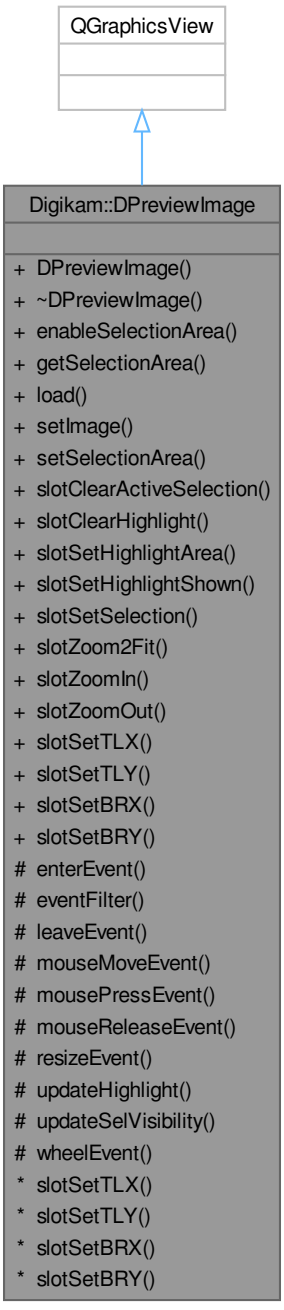
9.439.2.3 setMainWidget()

```
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 itselfes. 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

rectangle	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

tl_x	is the x coordinate of the top left corner 0=0 1=image with.
tl_y	is the y coordinate of the top left corner 0=0 1=image height.
br_x	is the x coordinate of the bottom right corner 0=0 1=image with.
br_y	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

percentage	is the percentage of the highlighted area that is shown.
highLightColor	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

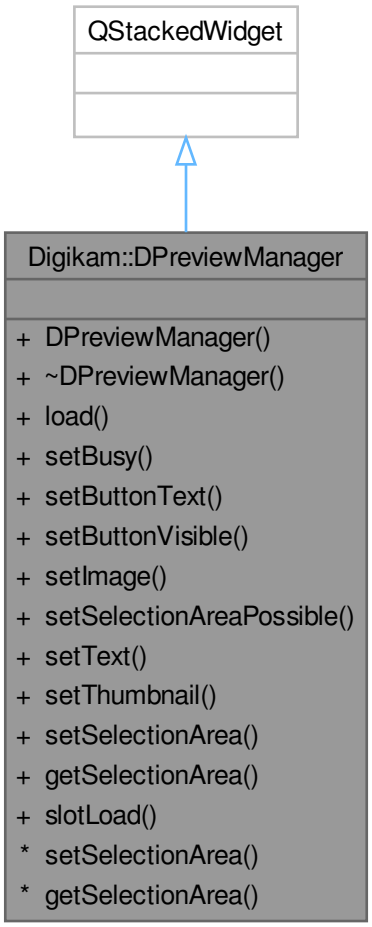
all parameters must be in the range 0.0 -> 1.0.

Parameters

tl↩ _x	is the x coordinate of the top left corner 0=0 1=image with.
tl↩ _y	is the y coordinate of the top left corner 0=0 1=image height.
br↩ _x	is the x coordinate of the bottom right corner 0=0 1=image with.
br↩ _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()

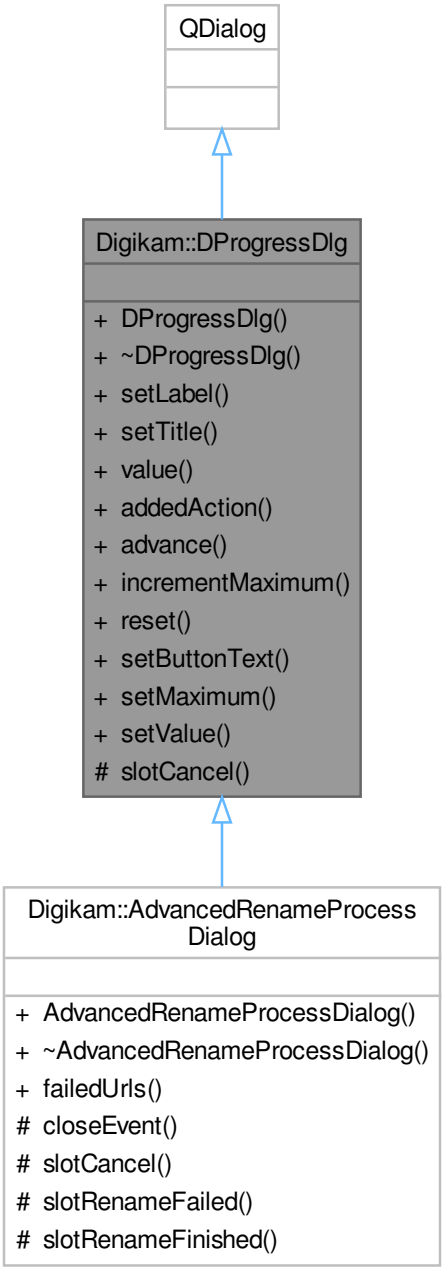
```
void Digikam::DPreviewManager::setSelectionArea (
            const QRectF & rectangle )
```

Parameters

rectangle	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

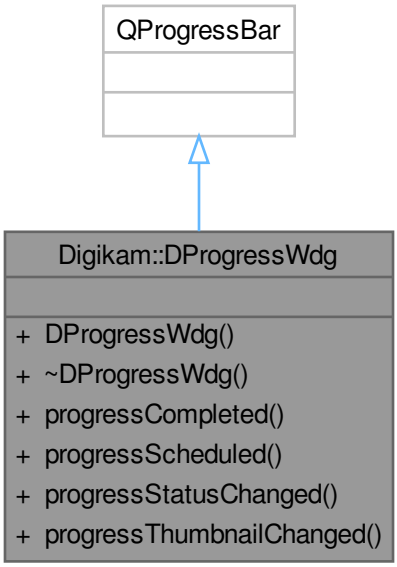
- **DProgressDlg** (QWidget ∗const parent=nullptr, const QString &caption=QString())
- void **setLabel** (const QString &text)
- void **setTitle** (const QString &text)
- int **value** () const

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 progresssThumbnailChanged().

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.

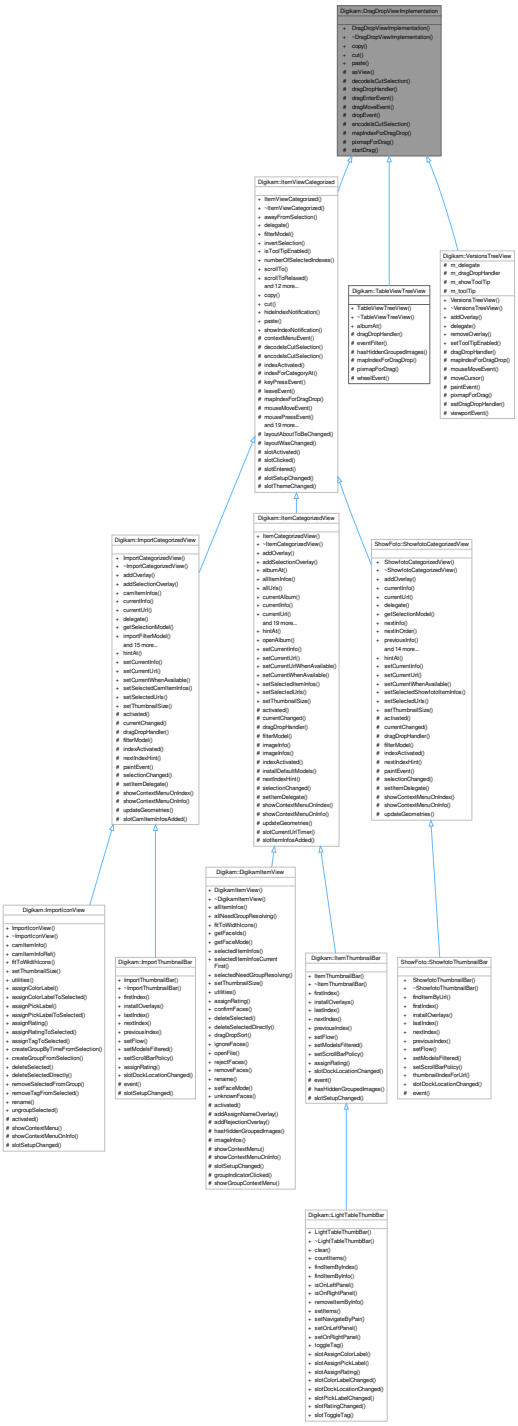
9.444.2.3 supportedDropActions()

```
Qt::DropActions Digikam::DragDropModelImplementation::supportedDropActions ( ) const
```

All functionality is redirected to the handler. dropMimeData() always returns false, leaving implementation to the view.

9.445 Digikam::DragDropViewImplementation Class Reference

Inheritance diagram for Digikam::DragDropViewImplementation:



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 **decodeIsCutSelection** (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 **encodeIsCutSelection** (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()

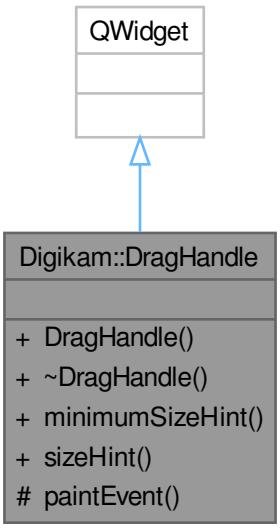
```
virtual QPixmap Digikam::DragDropViewImplementation::pixmapForDrag (
            const QList< QModelIndex > & indexes ) const  [protected], [pure virtual]
```

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

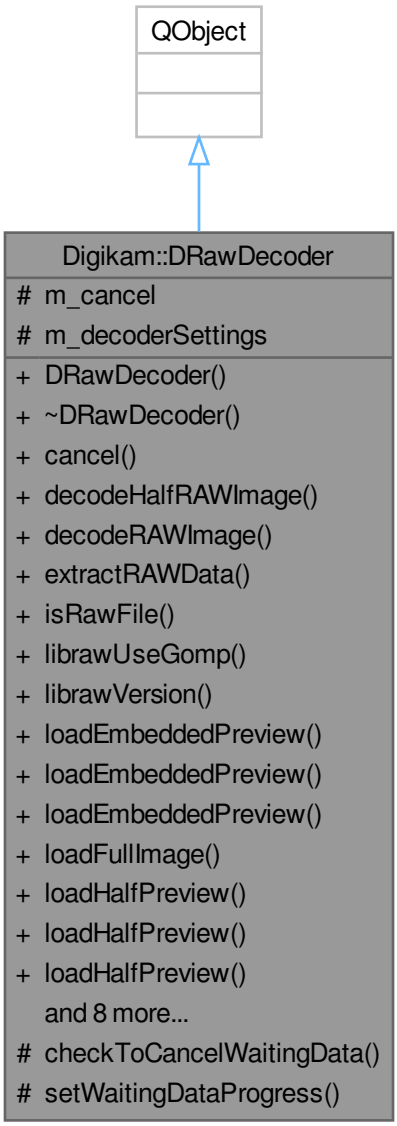
- **DragHandle** (QDockWidget ∗const)
- QSize **minimumSizeHint** () const override
- QSize **sizeHint** () const override

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 dcraw 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 **dcbEnhanceFl** = 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 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 $\sim$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 IMPULSENR : 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.

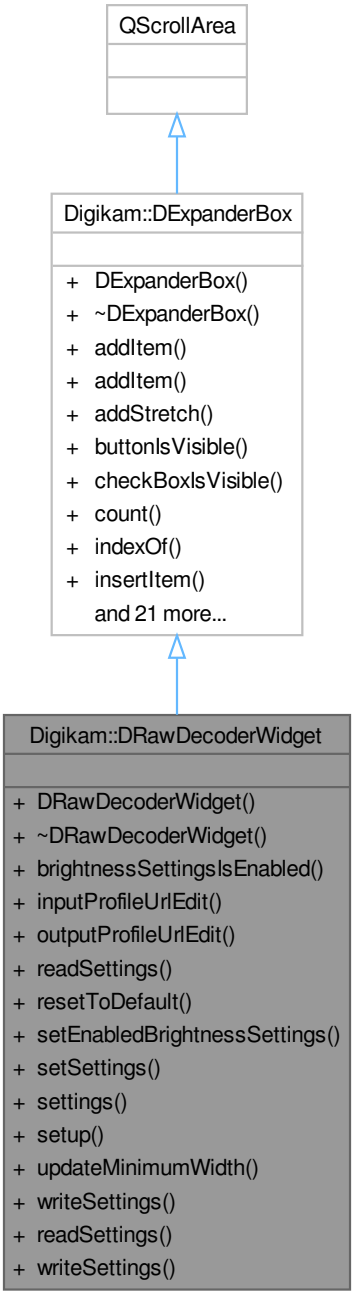
9.448.2.11 whiteBalance

```
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 **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 expand↩ Bydefault)
- 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

parent	the parent widget instance
advSettings	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/s2.
- 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 demosaising 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 dcraw.
- 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

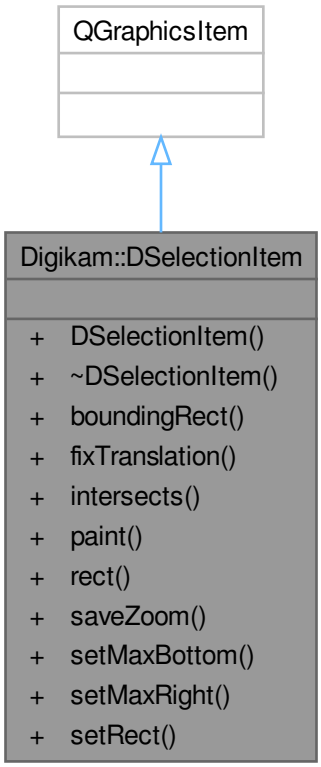
- **DSaveSettingsWidget** (QWidget ∗const parent)
- FileSaveConflictBox::ConflictRule **conflictRule** () const
- QString **extension** () const
- OutputFormat **fileFormat** () const
- void **readSettings** (KConfigGroup &group)
- void **resetToDefault** ()
- void **setConflictRule** (FileSaveConflictBox::ConflictRule 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::DSelectionItem Class Reference

Inheritance diagram for Digikam::DSelectionItem:



Public Types

- enum **Intersects** {
  None , **Top** , **TopRight** , **Right** ,
  BottomRight , **Bottom** , **BottomLeft** , **Left** ,
  TopLeft , **Move** }

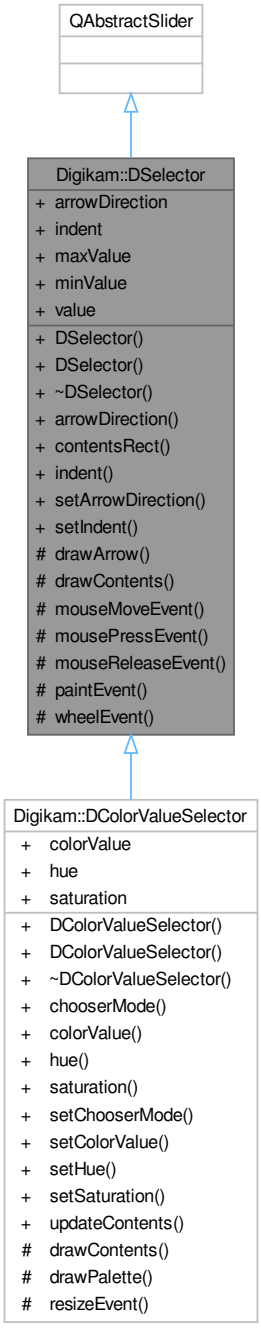
Public Member Functions

- **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 &sinfo)

  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)

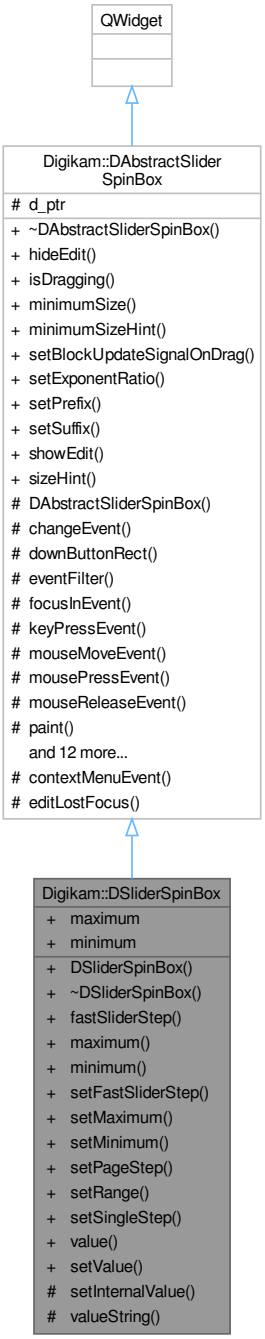
  Linux only: open file urls with the service.
- static bool **runFiles** (const QString &appCmd, const QList< QUrl > &urls, const KService::Ptr &service=KService::Ptr(), const DServiceInfo &serviceInfo=DServiceInfo())

  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.

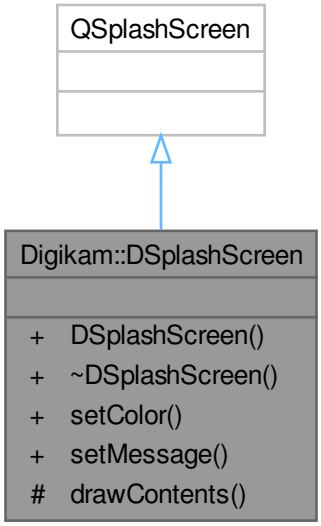
9.457.1.2 valueString()

```
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

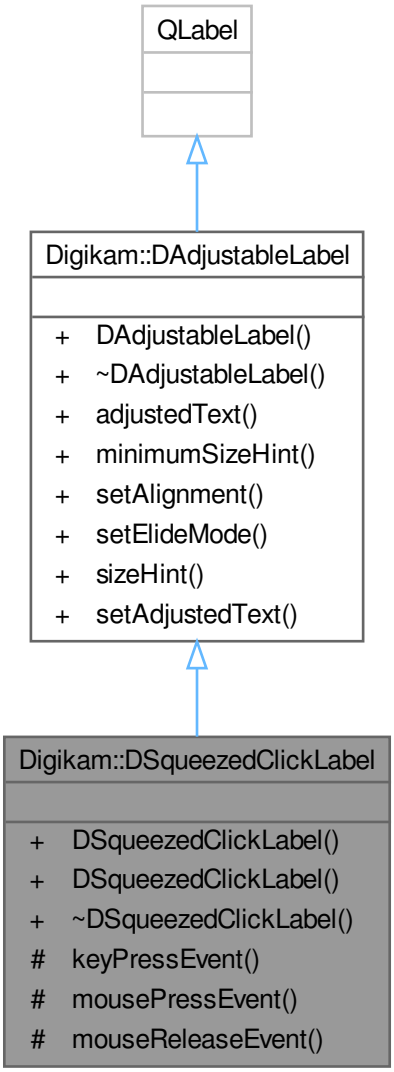
- void **setColor** (const QColor &color)
- void **setMessage** (const QString &message)

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

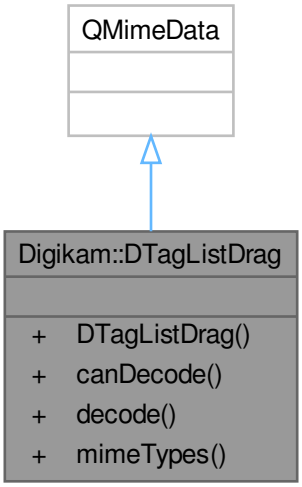
Public Slots inherited from Digikam::DAdjustableLabel

- 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

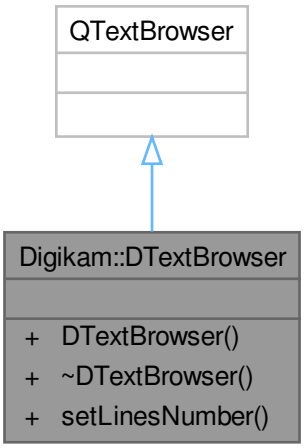
- static bool **canDecode** (const QMimeData *e)
- static bool **decode** (const QMimeData *e, QList< int > &tagIDs)
- static QStringList **mimeTypes** ()

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:



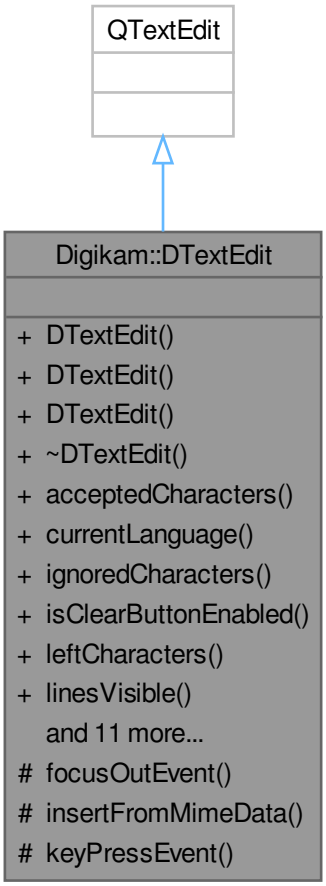
Public Member Functions

- **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 ()

  Emmited 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 QLine↩ Edit. 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 depedencies is not resolved, these method do nothing.

9.462.3.6 setLinesVisible()

```
void Digikam::DTextEdit::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 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.

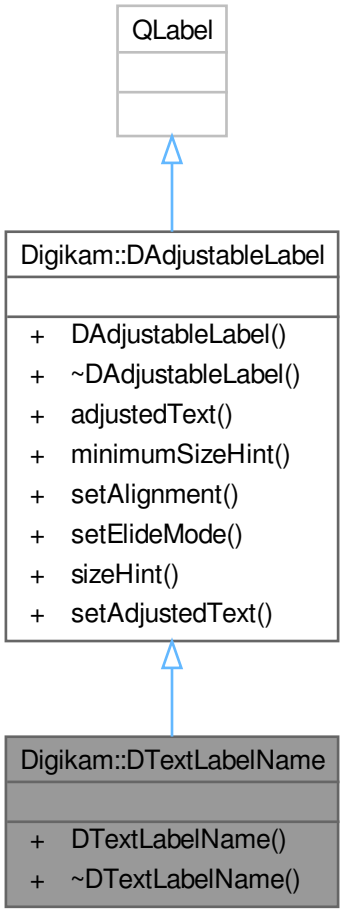
9.462.3.9 text()

```
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

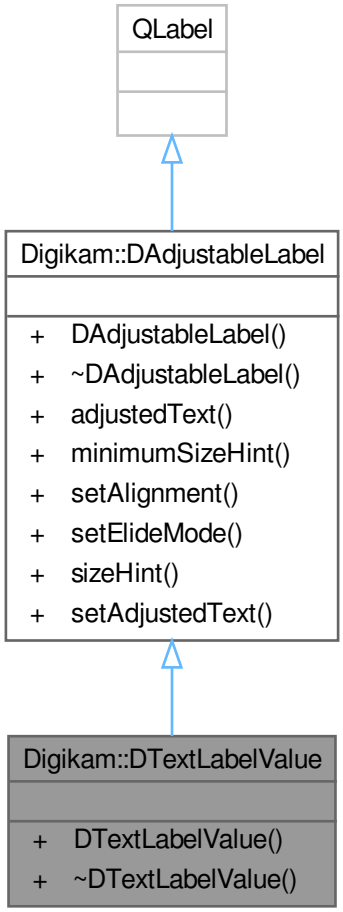
Additional Inherited Members

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

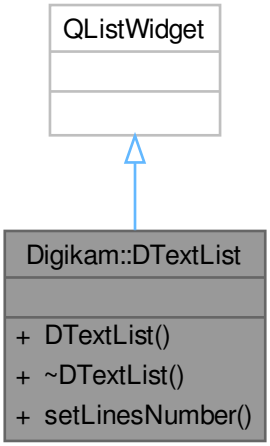
Additional Inherited Members

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 &itemInfo)

  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

dirToDelete	path to folder
deleteTime	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

imagePath	path to image
deleteTime	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

collPath	path to collection
baseName	name of the file inside the trash
itemInfo	item to extract data to it

9.468 Digikam::DTrashItemInfo Class Reference

Public Member Functions

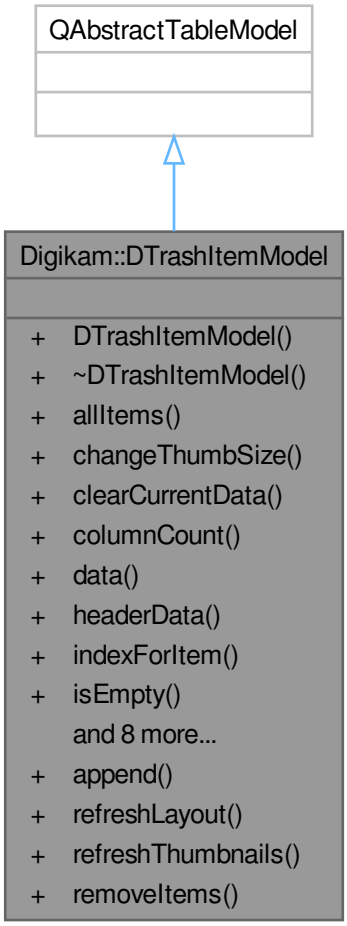
- bool **isNull** () const
- bool **operator==** (const DTrashItemInfo &itemInfo) const

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

itemInfo	item to append

9.469.1.2 changeThumbSize()

```
void Digikam::DTrashItemModel::changeThumbSize (
            int size )
```

Parameters

size	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

colPath	path to collection to load items for

9.469.1.5 pixmapForItem()

```
bool Digikam::DTrashItemModel::pixmapForItem (
            const QString & path,
            QPixmap & pix ) const
```

Parameters

path	path of image in trash
pix	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

desc	loading description from thumbnail load thread
pix	pixmap from thumbnail load thread

9.469.1.7 removeItems

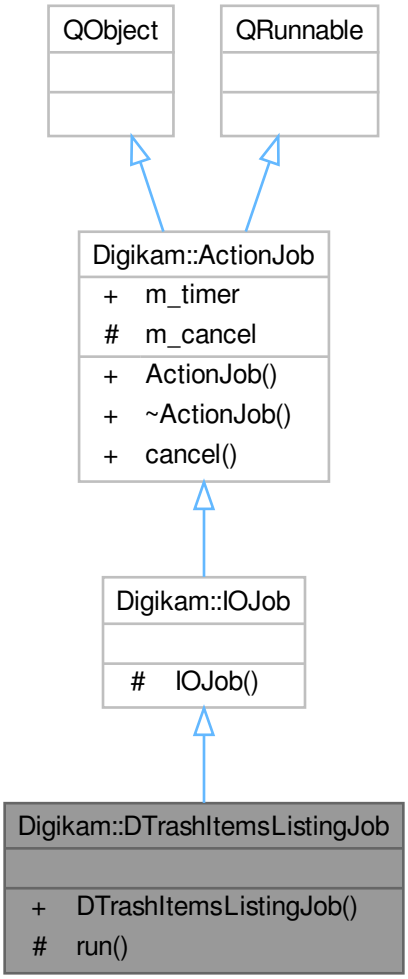
```
void Digikam::DTrashItemModel::removeItems (
            const QModelIndexList & indexes )  [slot]
```

Parameters

indexes	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 **signalOneProccessed** (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.

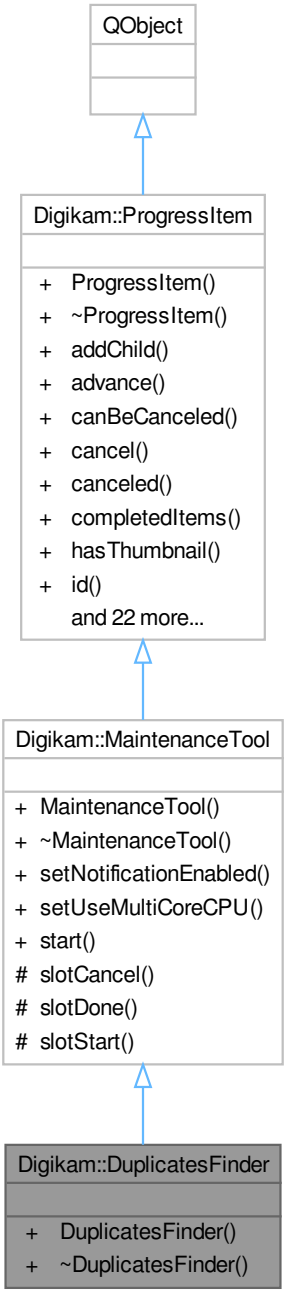
Protected Attributes inherited from Digikam::ActionJob

- 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, HaarIface::RefImageSelMethod method=HaarIface::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 elpased 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

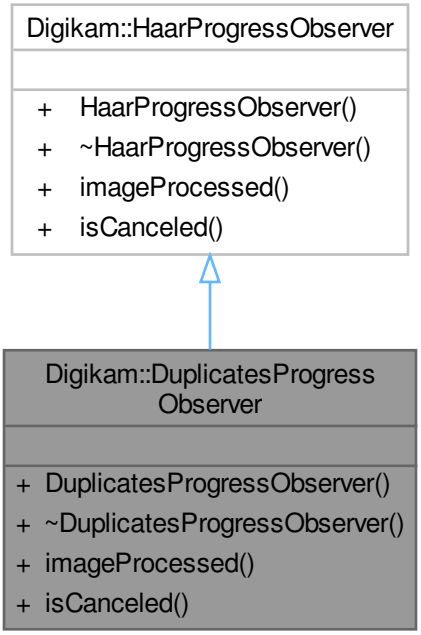
- void **start** ()

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()

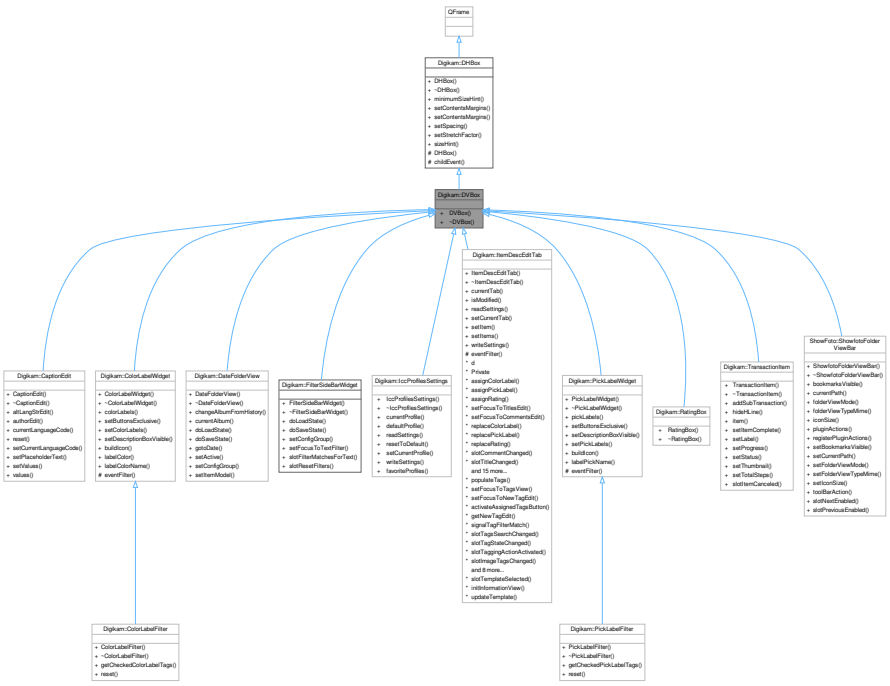
```
bool Digikam::DuplicatesProgressObserver::isCanceled ( )  [override], [virtual]
```

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

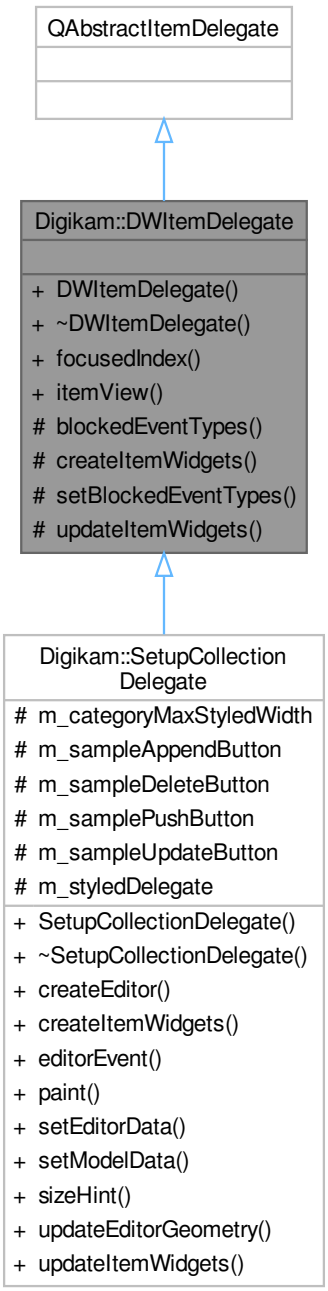
Protected Member Functions inherited from Digikam::DHBox

- **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 QPersistentModelIndex &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

itemView	the item view the new delegate will monitor
parent	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

widget	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

widget	the widget which must block events
types	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

widgets	the widgets to update
option	the current set of style options for the view.
index	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

delegate	the ItemDelegate 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

index	The index to search into.
option	a QStyleOptionViewItem.
updateWidgets	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

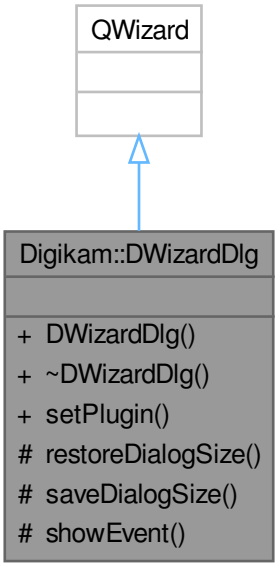
- **DWItemDelegatePoolPrivate** (DWItemDelegate *const dd)

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

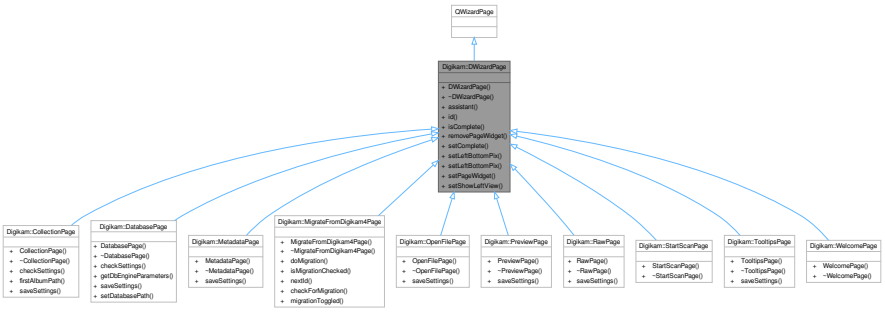
- **DWizardDlg** (QWidget ∗const parent, const QString &objName)
- void **setPlugin** (DPlugin ∗const tool)

Protected Member Functions

- void **restoreDialogSize** ()
- void **saveDialogSize** ()
- void **showEvent** (QShowEvent ∗) override

9.478 Digikam::DWizardPage Class Reference

Inheritance diagram for Digikam::DWizardPage:



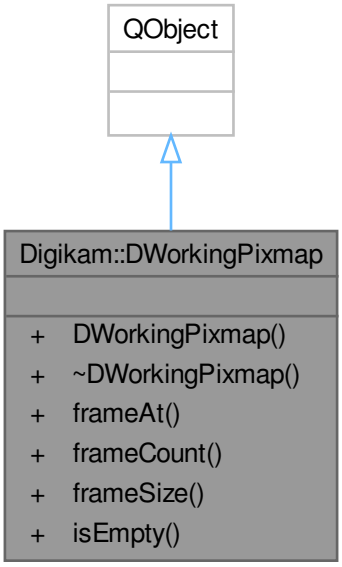
Public Member Functions

- **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:



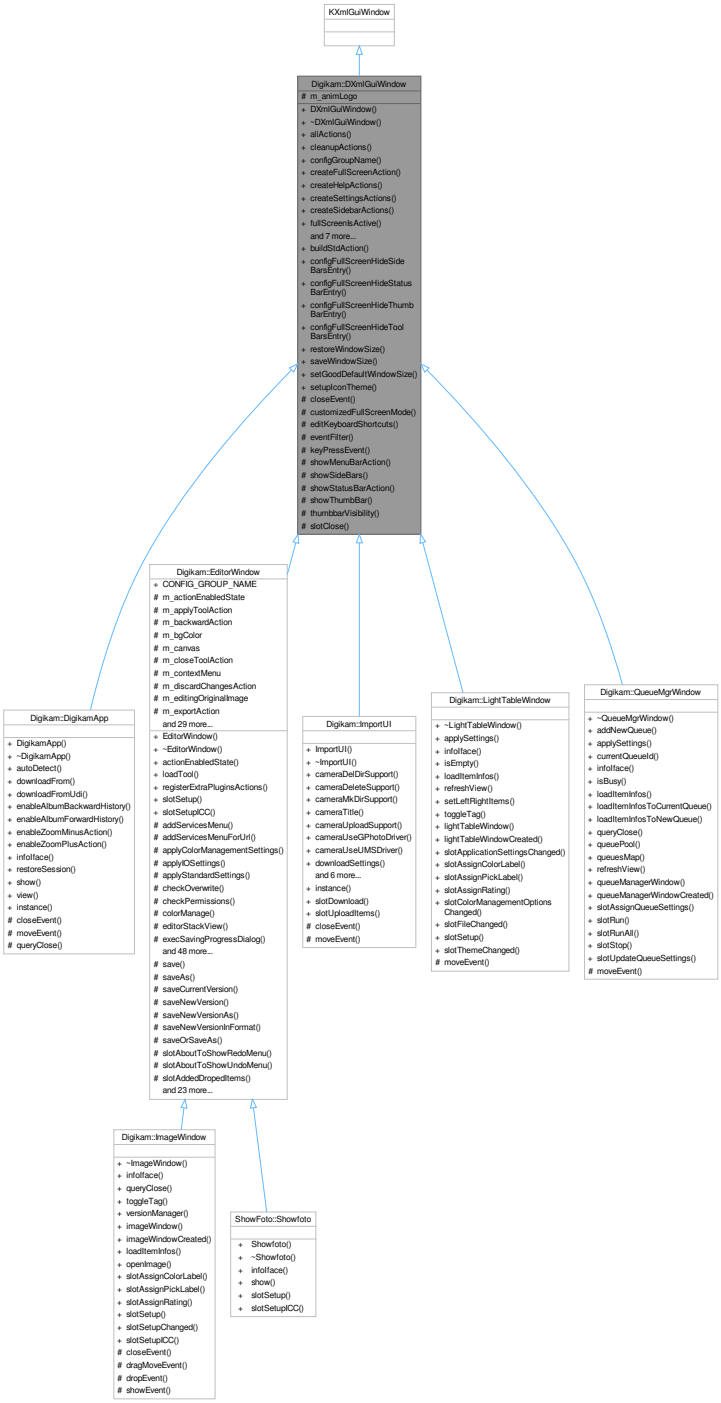
Public Member Functions

- **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 **fullScreenIsActive** () const

  Return true if managed window is currently in Full Screen Mode.
- virtual DInfoInterface * infoIface (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 cutomization.

9.480.1.4 infoIface()

```
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.

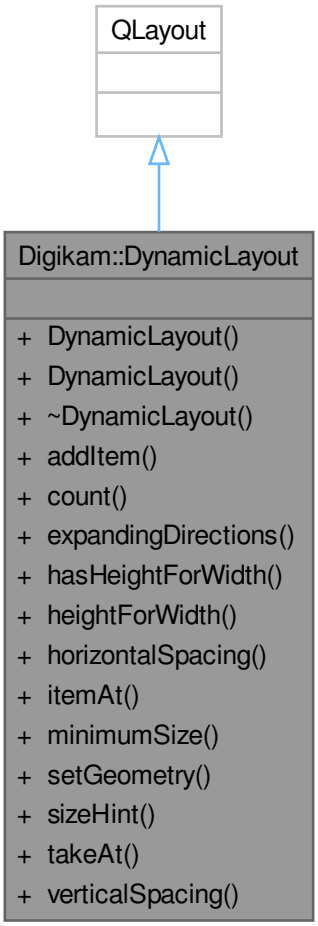
9.480.1.8 thumbbarVisibility()

```
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:

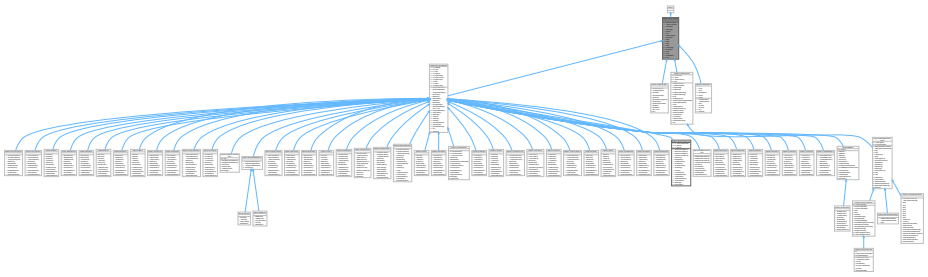


Public Member Functions

- **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 ∗layItem) override
- int **count** () const override
- Qt::Orientations **expandingDirections** () const override
- bool **hasHeightForWidth** () const override
- int **heightForWidth** (int) 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.

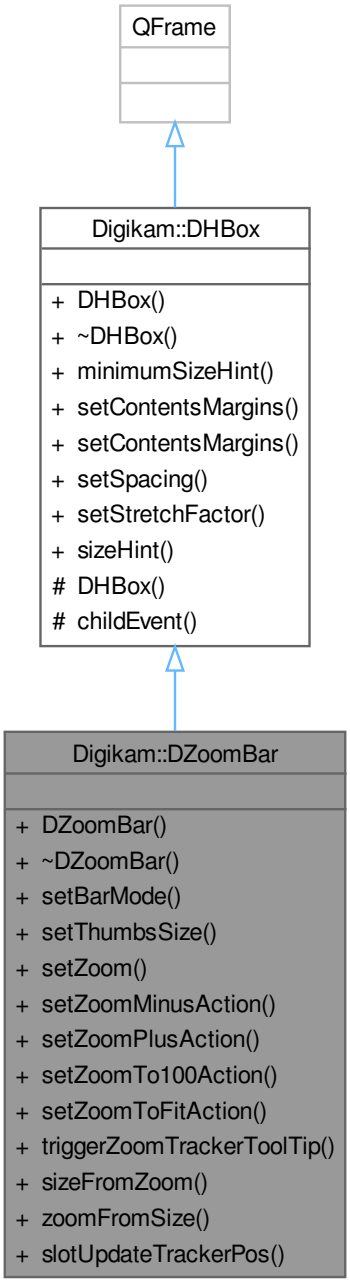
9.482.2.6 wait

```
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
```

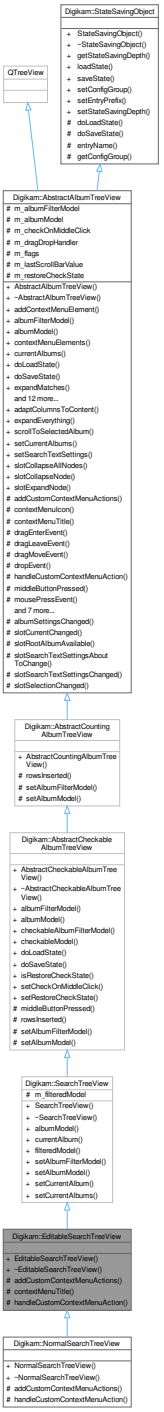
Enumerator

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 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 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 | 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

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

parent	qt parent
searchModel	the model this view should act on
searchModificationHelper	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()

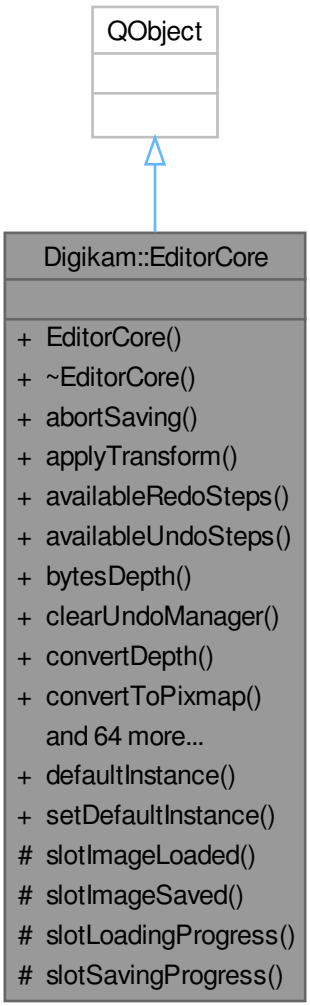
```
void Digikam::EditableSearchTreeView::handleCustomContextMenuAction (
            QAction * action,
            const AlbumPointer< Album > & album )  [override], [protected], [virtual]
```

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

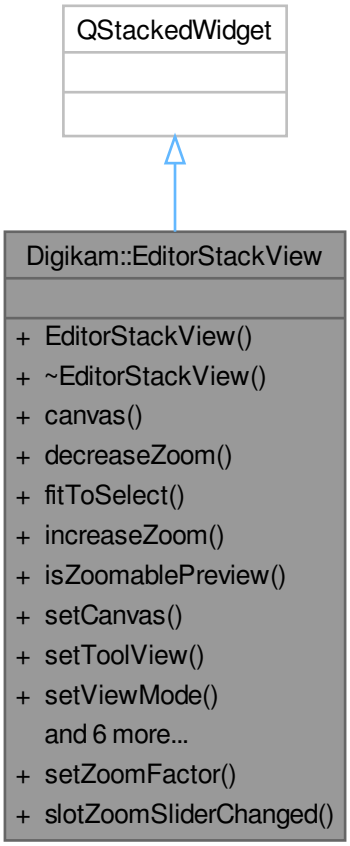
- static EditorCore * **defaultInstance** ()
- static void **setDefaultInstance** (EditorCore *const instance)

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

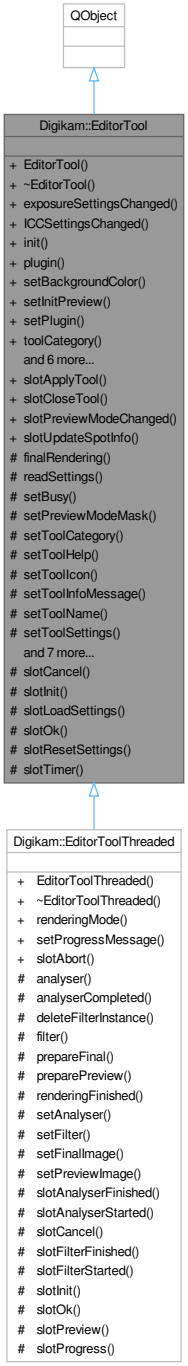
- void **signalToggleOffFitToWindow** ()
- void **signalZoomChanged** (bool isMax, bool isMin, double zoom)

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 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

- 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::EditorToolIface Class Reference

Inheritance diagram for Digikam::EditorToolIface:

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

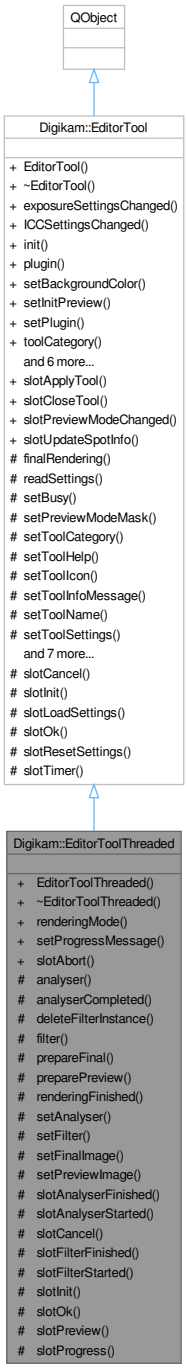
- void **signalCancelClicked** ()
- void **signalChannelChanged** ()
- void **signalColorGuideChanged** ()
- void **signalDefaultClicked** ()
- void **signalLoadClicked** ()
- void **signalOkClicked** ()
- void **signalSaveAsClicked** ()
- void **signalScaleChanged** ()
- void **signalTryClicked** ()

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.

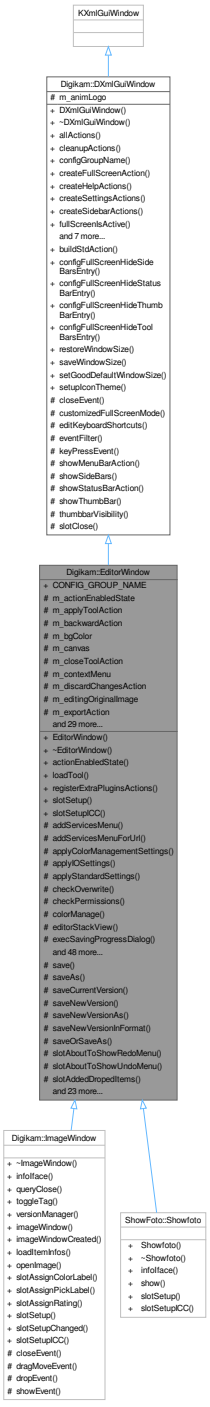
9.490.1.2 setProgressMessage()

```
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 **fullScreenIsActive** () const

  Return true if managed window is currently in Full Screen Mode.
- virtual DInfoInterface * infoIface (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 **slotAddedDropedItems** (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 **saveIsComplete** ()=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 **EditorToolIface**

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*.

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

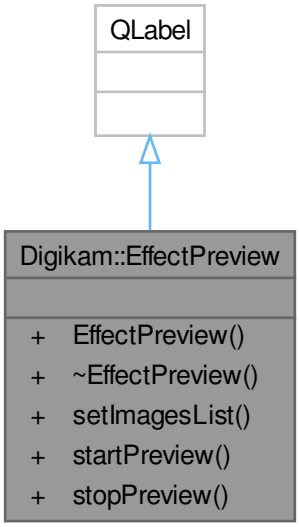
```
enum Digikam::EffectMngr::EffectType
```

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** (EffectMngr::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

name	The ellipsoid name.
semiMajorAxis	The equatorial radius.
semiMinorAxis	The polar radius.
inverseFlattening	The inverse of the flattening value.
ivfDefinitive	`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

name	The ellipsoid name.
semiMajorAxis	The equatorial radius.
semiMinorAxis	The polar radius.

9.494.3.2 createFlattenedSphere()

```
Ellipsoid Digikam::Ellipsoid::createFlattenedSphere (
            const QString & name,
            double semiMajorAxis,
            double inverseFlattening )  [static]
```

Parameters

name	The ellipsoid name.
semiMajorAxis	The equatorial radius.
inverseFlattening	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

x1	Longitude of first point (in decimal degrees).
y1	Latitude of first point (in decimal degrees).
x2	Longitude of second point (in decimal degrees).
y2	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

latitude	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 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** ()

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.

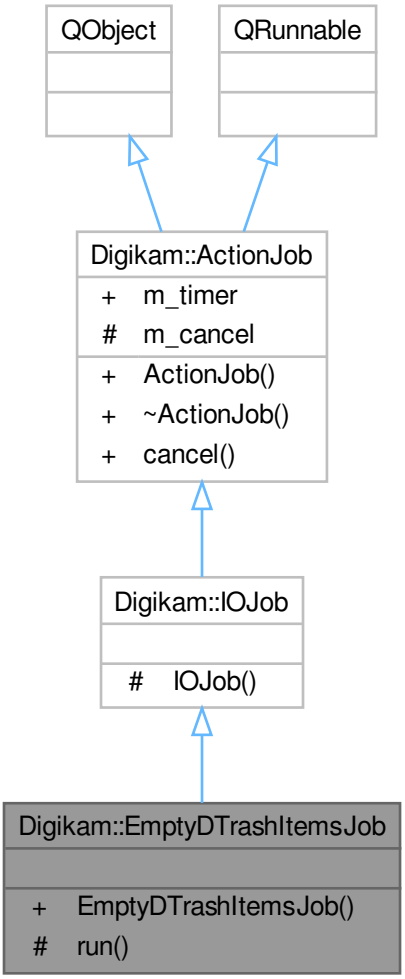
9.495.1.3 readParameters()

```
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 **signalOneProccessed** (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.

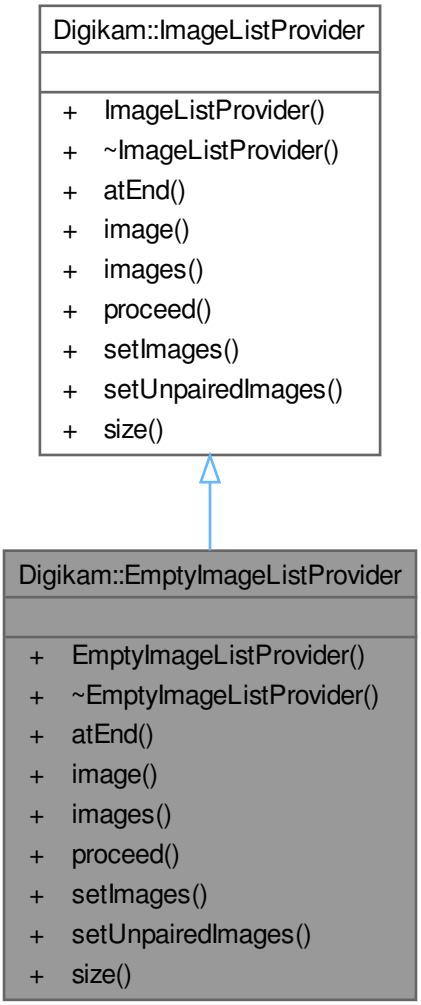
Protected Attributes inherited from Digikam::ActionJob

- 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 refImage, 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 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** ()

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.

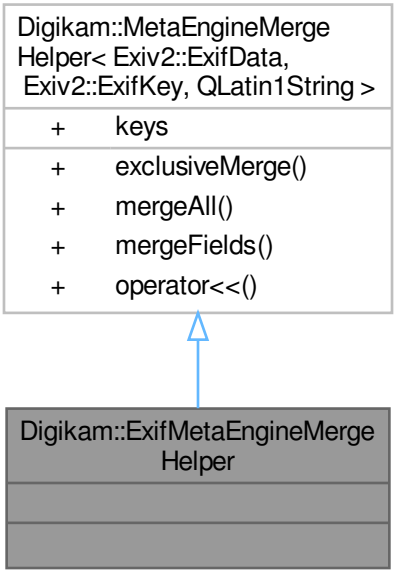
9.498.1.3 readParameters()

```
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 **versionIsRight** () const
- bool **versionIsRight** (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

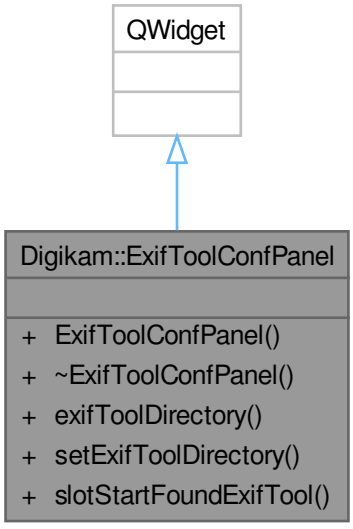
- QString **findHeader** (const QStringList &output, const QString &header) const
- virtual bool **parseHeader** (const QString &output)
- virtual QString **readConfig** ()
- void **setVersion** (QString &version)
- virtual void **writeConfig** ()

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

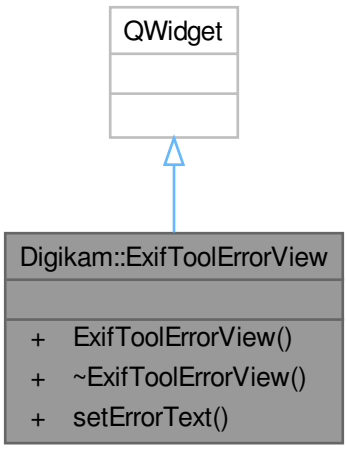
- void **signalExifToolSettingsChanged** (bool available)

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

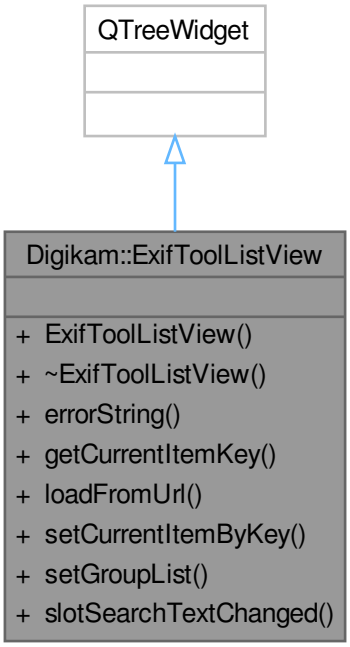
- void **signalSetupExifTool** ()

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()

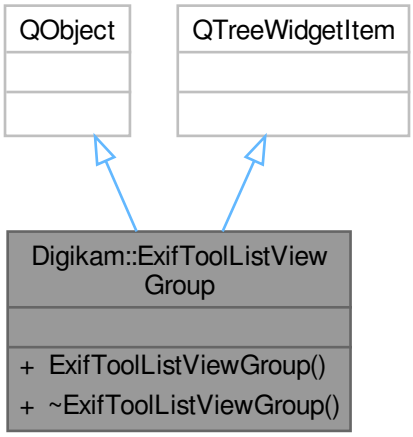
```
void Digikam::ExifToolListView::setGroupList (
            const QStringList & tagsFilter,
            const QStringList & keysFilter = QStringList() )
```

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:

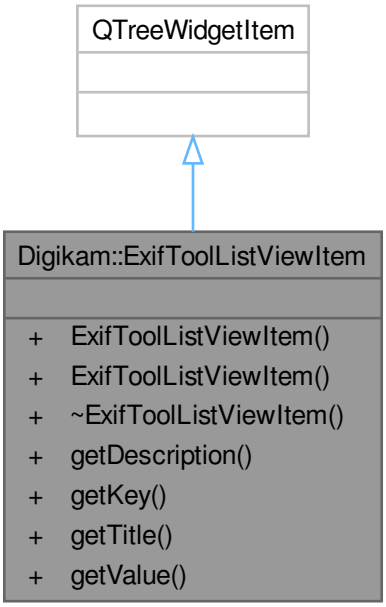


Public Member Functions

- **ExifToolListViewGroup** (QTreeWidget ∗const parent, const QString &group)

9.505 Digikam::ExifToolListViewItem Class Reference

Inheritance diagram for Digikam::ExifToolListViewItem:

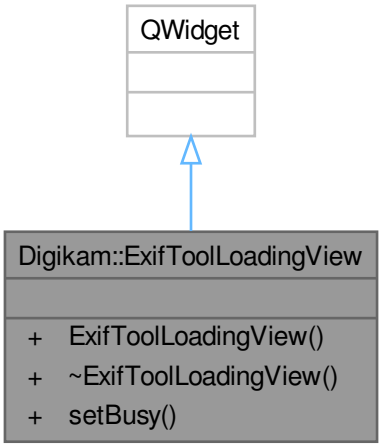


Public Member Functions

- **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:

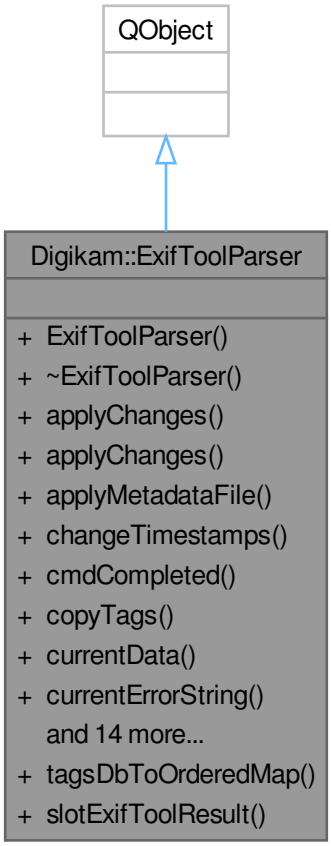


Public Member Functions

- **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 has↩ Xmp=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 write↩ Modes=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 &cmd↩ ErrorChannel)

  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). 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 tupple 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 tupple 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

path	is the target files to change.
newTags	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

path	is the target files to change.
exvTempFile	is the list of changes embedded in EXV container.
hasExif	if the EXV container has Exif metadata restore MarkerNotes.
hasXmp	if the EXV container has Xmp metadata.
hasCSet	if the EXV container has characters set information.

9.507.2.3 applyMetadataFile()

```
bool Digikam::ExifToolParser::applyMetadataFile (
            const QString & path,
            const QString & meta )
```

Parameters

path	is the target file to change.
meta	is the metadata file.

9.507.2.4 changeTimestamps()

```
bool Digikam::ExifToolParser::changeTimestamps (
            const QString & path,
            const QDateTime & dateTime )
```

Parameters

path	is the target file to change.
dateTime	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

src	must be a readable file format supported by ExifTool.

Parameters

dst	must be a writable file format supported by ExifTool.
copyOps	is a OR combination of ExifToolProcess::CopyTagsSource values.
writeModes	is a OR combaniation of ExifToolProcess::WritingTagsMode 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

path	must be a readable file format supported by ExifTool.
transOps	is a OR combination of ExifToolProcess::TranslateTagsOps 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.

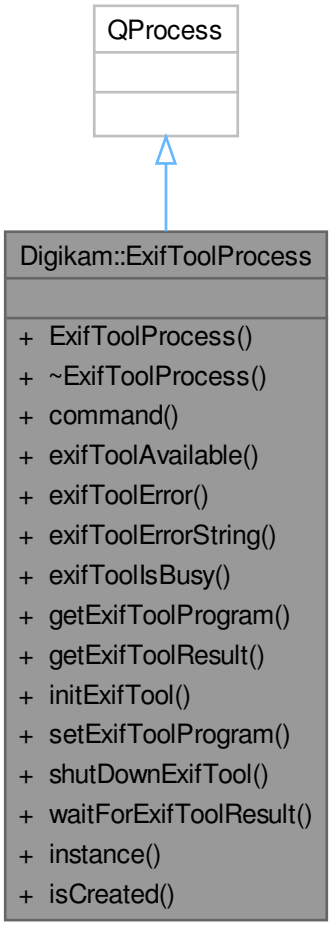
9.507.2.14 writableFormats()

```
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 **exifToolIsBusy** () 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

  WatCondition 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

LOAD_METADATA	Load all metadata from a file with ExifTool.
LOAD_CHUNKS	Load Exif, Iptc, and Xmp chunks from a file as byte-array for MetaEngine.
APPLY_CHANGES	Apply tag changes in a file with ExifTool.
APPLY_CHANGES_EXV	Apply tag changes in a file with ExifTool using an EXV container.
APPLY_METADATA_FILE	Apply a metadata file to a file with ExifTool.
CHANGE_TIMESTAMPS	Change all timestamps in a file with ExifTool.
READ_FORMATS	Return the list of readable ExifTool file formats.
WRITE_FORMATS	Return the list of writable ExifTool file formats.
TRANSLATIONS_LIST	List of ExifTool languages available for translations.
TAGS_DATABASE	List of ExifTool tags from database.
VERSION_STRING	Return the ExifTool version as string.
COPY_TAGS	Copy tags from one file to another one. See CopyTagsSource enum for details.
TRANS_TAGS	Translate tags in file. See TranslateTagsOps enum for details.
NO_ACTION	Last value from this list. Do nothing.

9.508.1.2 CopyTagsSource

enum Digikam::ExifToolProcess::CopyTagsSource

Enumerator

COPY_EXIF	Copy all Exif Tags from source file.
COPY_MAKERNOTES	Copy all Makernotes tags from source file.
RESTORE_PREVIEW	Restore preview image from source file.
COPY_IPTC	Copy all Iptc tags from source file.
COPY_XMP	Copy all Xmp tags from source file.
COPY_ICC	Copy ICC profile from source file.
COPY_ALL	Copy all tags from source file.
COPY_NONE	No copy operation.

9.508.1.3 TranslateTagsOps

enum Digikam::ExifToolProcess::TranslateTagsOps

Enumerator

TRANS_ALL_XMP	Translate all existing Tags from source file to Xmp.
TRANS_ALL_IPTC	Translate all existing Tags from source file to Iptc.
TRANS_ALL_EXIF	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: ExitTool 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.

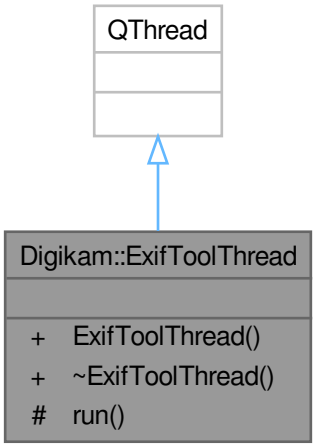
9.509 Digikam::ExifToolProcess::Result Class Reference

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)

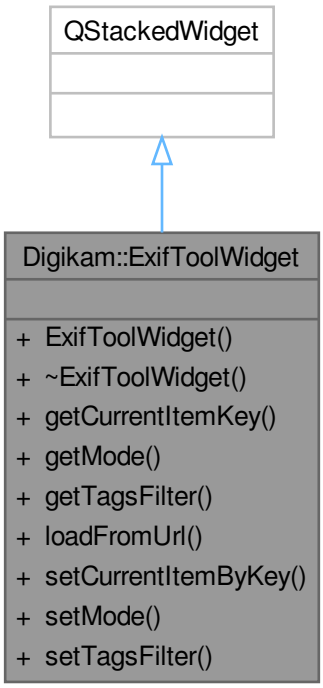
Protected Member Functions

- 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

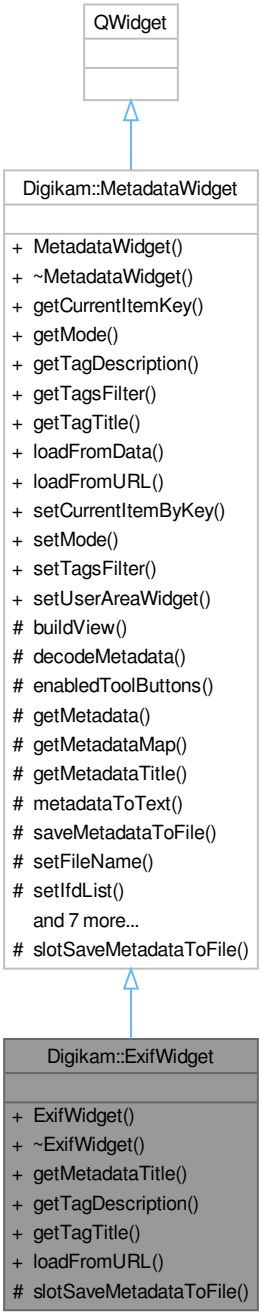
- void **signalSetupExifTool** ()
- void **signalSetupMetadataFilters** ()

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 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

9.512.1 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.

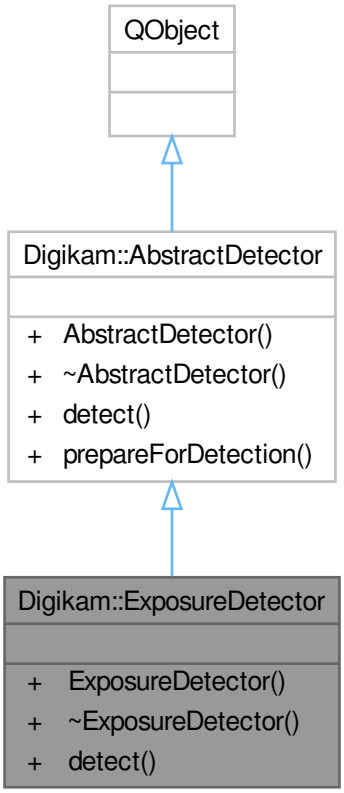
9.512.1.4 loadFromURL()

```
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

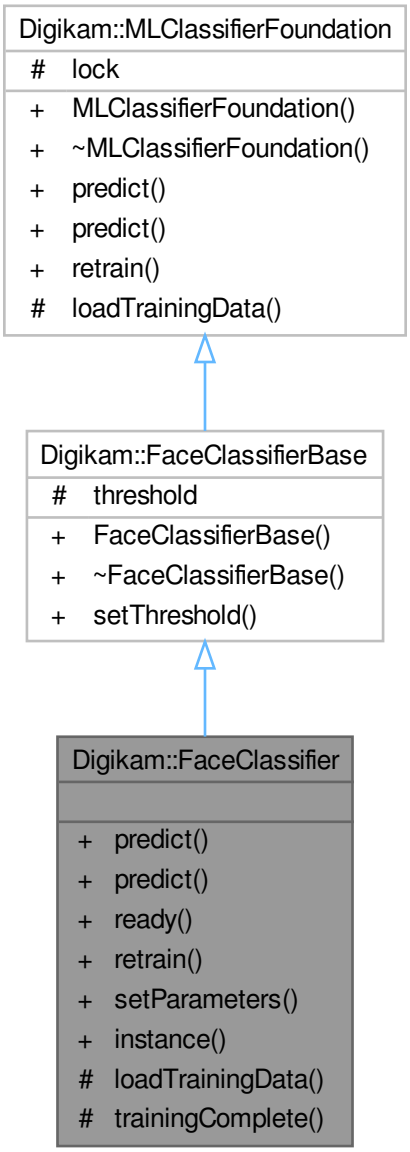
9.514.1.1 exposureIndicatorMode

```
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.

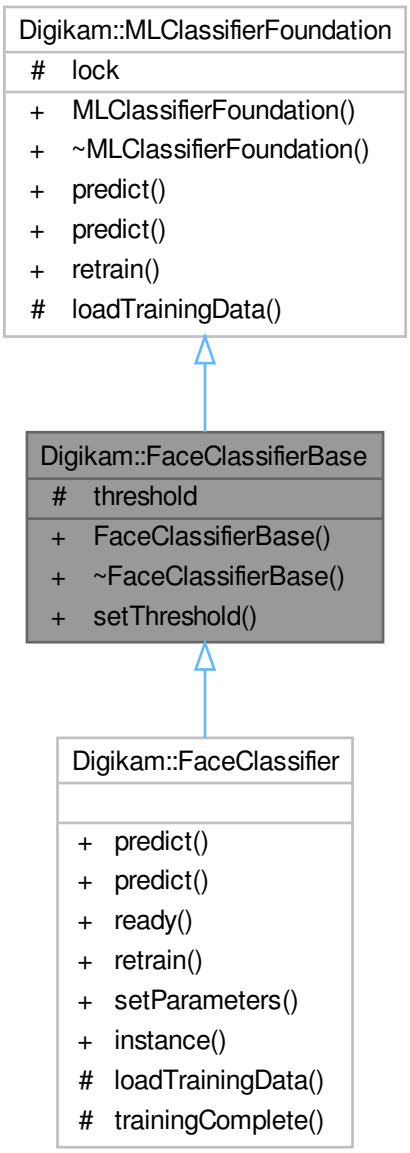
9.515.1.4 retrain()

```
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

identities	in the database.

9.517.1.2 insertFaceVector()

```
int Digikam::FaceDb::insertFaceVector (
            const cv::Mat & faceEmbedding,
            const int label,
            const QString & hash ) const
```

Parameters

faceEmbedding	
label	
hash	

Returns

id of newly inserted entry.

9.517.1.3 removeFaceVector() [1/2]

```
bool Digikam::FaceDb::removeFaceVector (
            const int id ) const
```

Parameters

id	the nodeId (row id) to remove.

Returns

bool

9.517.1.4 removeFaceVector() [2/2]

```
bool Digikam::FaceDb::removeFaceVector (
            const QString & hash ) const
```

Parameters

hash	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

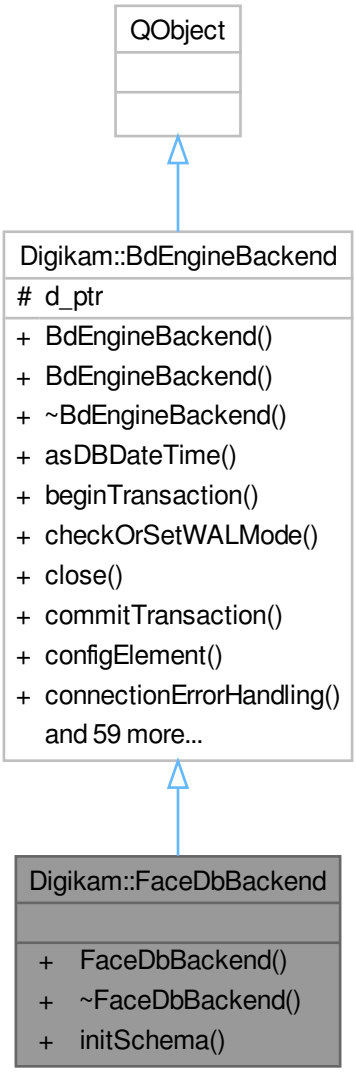
9.519.1.1 FaceDbAccessUnlock()

```
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)
- 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 lastInsert↩Id=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 last↩InsertId=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 &field↩Names, 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 msecs)

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 DImg & 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 )
```

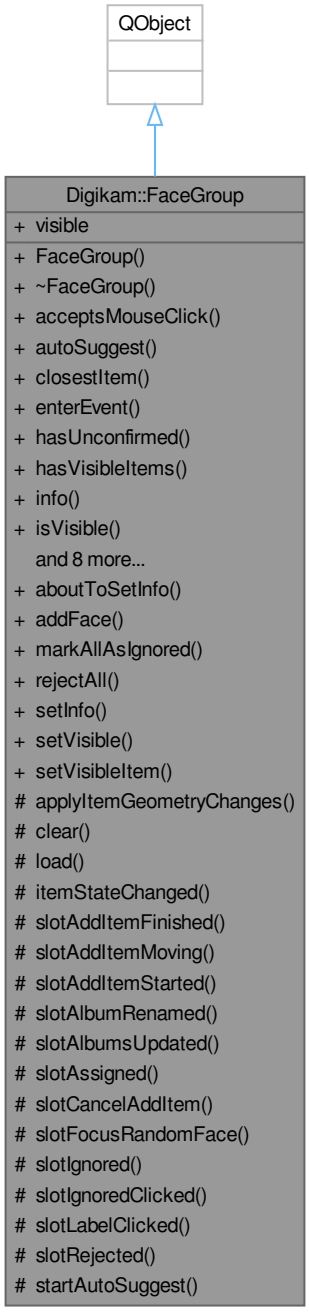
Available parameters:

"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 **acceptsMouseClick** (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::FaceItem Class Reference

Inheritance diagram for Digikam::FaceItem:

Public Member Functions

- **FaceItem** (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::FaceItemRetriever Class Reference

Public Member Functions

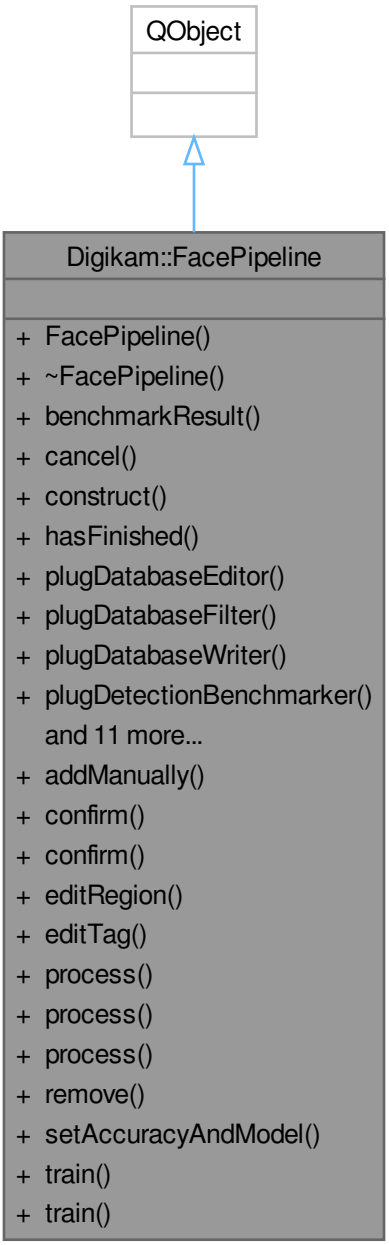
- **FaceItemRetriever** (FacePipeline::Private ∗const d)
- void **cancel** ()
- QList< QImage ∗ > **getDetails** (const DImg &src, const QList< FaceTagsIface > &faces) const
- QList< QImage ∗ > **getDetails** (const DImg &src, const QList< QRectF > &rects) const
- QList< QImage ∗ > **getThumbnails** (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 assigned↩ TagId=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 **plugDetectionBenchmarker** ()
- void **plugFaceDetector** ()
- void **plugFacePreviewLoader** ()
- void **plugFaceRecognizer** ()
- void **plugParallelFaceDetectors** ()
- void **plugRecognitionBenchmarker** ()
- 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

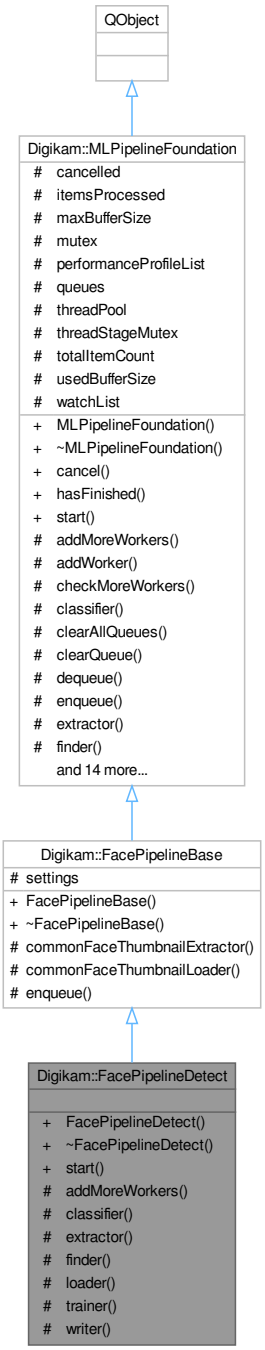
9.528.2.1 enqueue()

```
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.

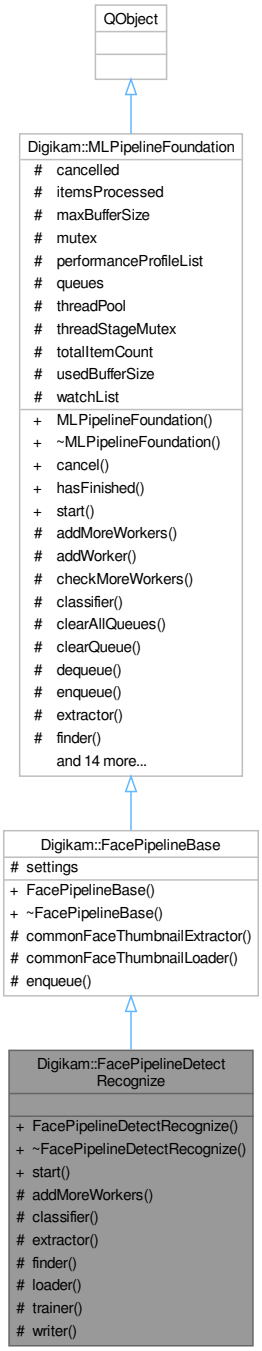
9.529.1.8 writer()

```
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.

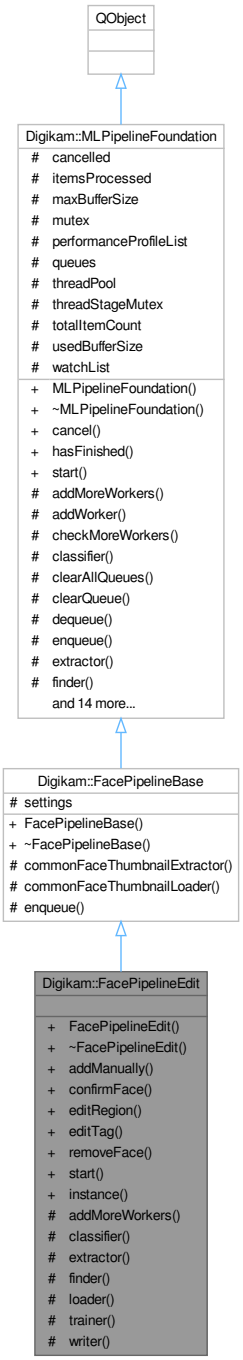
9.530.1.8 writer()

```
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.

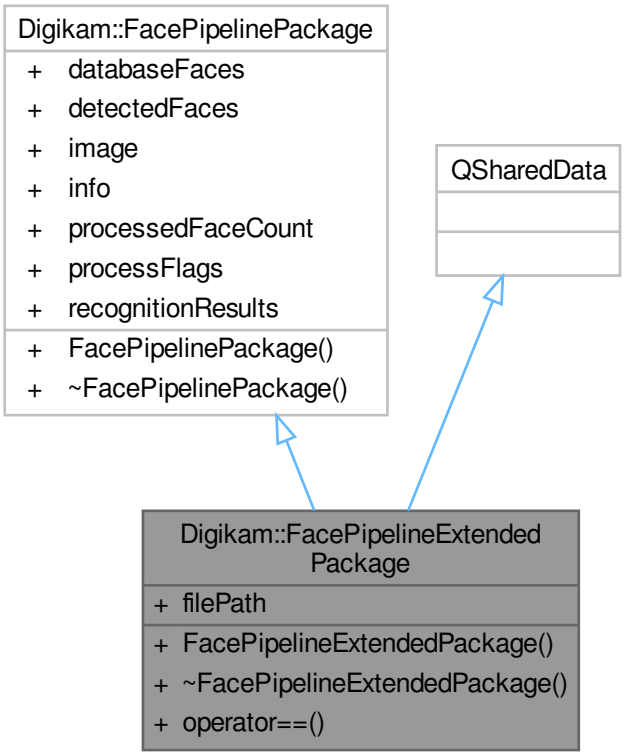
9.531.1.8 writer()

```
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 , **ProcessedBy↩ Recognizer** = 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 imageId, int tagId, const TagRegion &region)
- **FaceTagsIface** (Type type, qlonglong imageId, int tagId, const TagRegion &region)
- QString **getAutodetectedPersonString** () const

  Returns the string tagId + ',' + unconfirmedFace + ',' + regionXml.
- const QString **hash** () const

  Generate a hash based on the imageId, tagId, and rect to uniquely identify this entry in the face training DB.
- qlonglong **imageId** () 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 imageId, 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** (qlonglong 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

- qlonglong **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

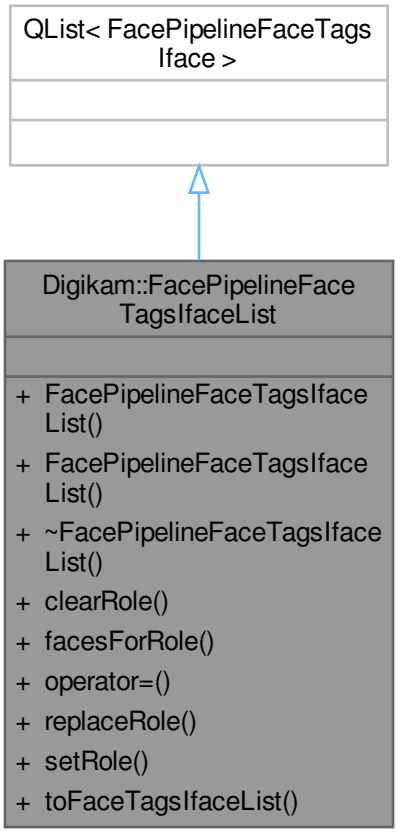
```
enum Digikam::FacePipelineFaceTagsIface::Role
```

Enumerator

GivenAsArgument	Source.
ForRecognition	Task.
Confirmed	Executed action (task is cleared).

9.534 Digikam::FacePipelineFaceTagsIfaceList Class Reference

Inheritance diagram for Digikam::FacePipelineFaceTagsIfaceList:

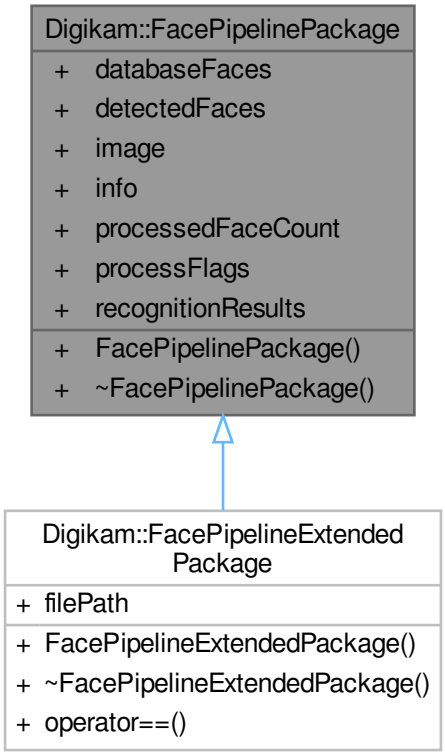


Public Member Functions

- **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

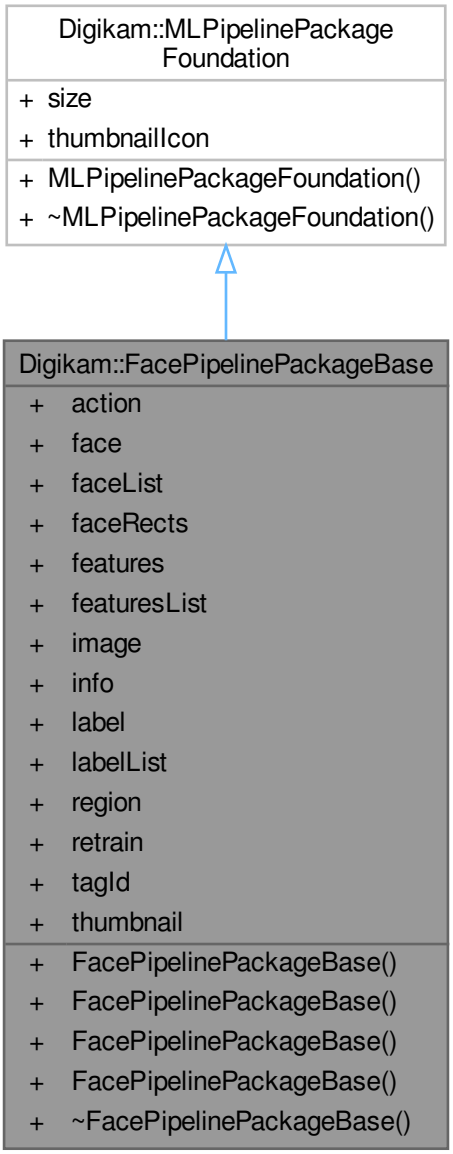
- enum **ProcessFlag** {
  NotProcessed = 0 , **PreviewImageLoaded** = 1 << 0 , **ProcessedByDetector** = 1 << 1 , **ProcessedBy↩ Recognizer** = 1 << 2 ,
  WrittenToDatabase = 1 << 3 , **ProcessedByTrainer** = 1 << 4 }
- typedef QFlags< ProcessFlag > **ProcessFlags**

Public Attributes

- FacePipelineFaceTagsIfaceList **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** (qlonglong _imageId)
- **FacePipelinePackageBase** (qlonglong _imageId, const FaceTagsIface &_face)

Public Attributes

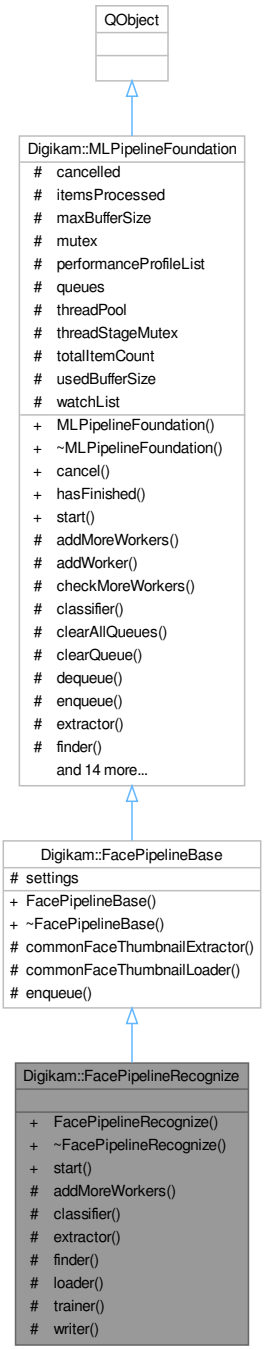
- EditPipelineAction **action** = EditPipelineAction::Confirm
- FaceTagsIface **face**
- QList< FaceTagsIface > **faceList**
- QList< QRectF > **faceRects**
- cv::Mat **features**
- QList< cv::Mat > **featuresList**
- DImg **image**
- ItemInfo **info**
- int **label** = -1
- QList< int > **labelList**
- 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.

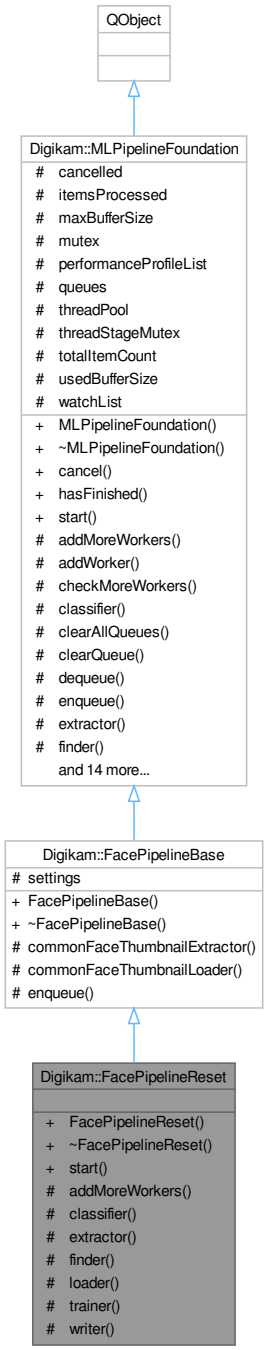
9.537.1.8 writer()

```
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.

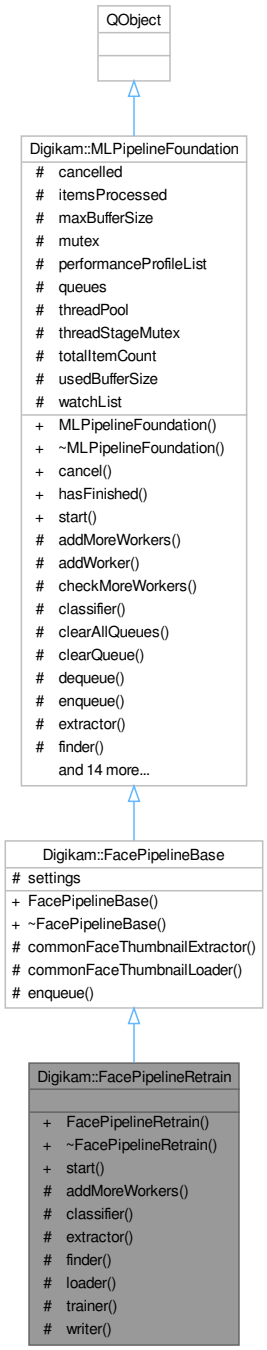
9.538.1.8 writer()

```
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.

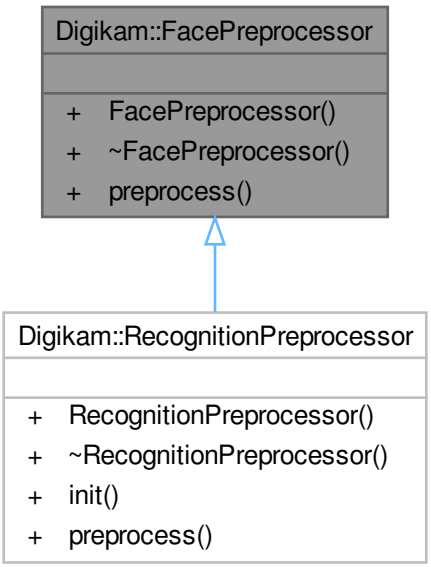
9.539.1.8 writer()

```
bool Digikam::FacePipelineRetrain::writer ( )  [override], [protected], [virtual]
```

Implements Digikam::MLPipelineFoundation.

9.540 Digikam::FacePreprocessor Class Reference

Inheritance diagram for Digikam::FacePreprocessor:

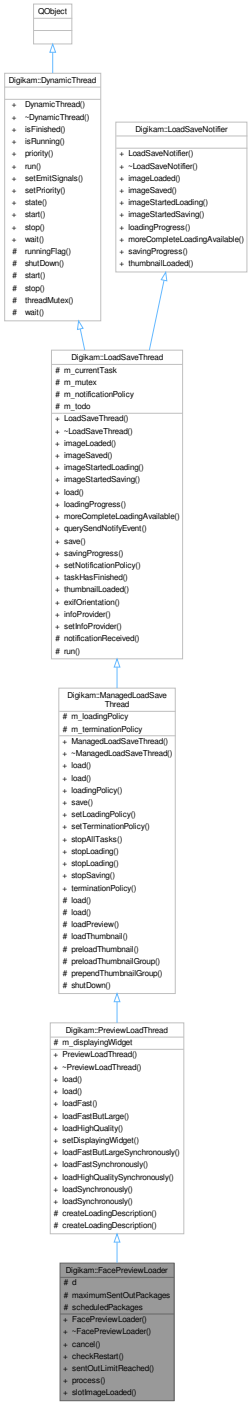


Public Member Functions

- 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 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 inherited from Digikam::PreviewLoadThread

- static LoadingDescription **createLoadingDescription** (const QString &filePath, const PreviewSettings &settings, int size, const IccProfile &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 overlayed 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

- QTimeLine ∗ **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 [https://github.com/arunponnusamy/cvlib].
YOLOv3	YOLO neural network inference [https://github.com/sthanhng/yoloface].
YuNet	YuNet neural network inference [https://github.com/opencv/opencv_zoo/tree/main].

9.544.1.3 FaceRecognitionModel

enum Digikam::FaceScanSettings::FaceRecognitionModel

Enumerator

OpenFace	OpenFace pre-trained neural network model [https://github.com/sahilshah/openface/tree/master].
SFace	SFace pre-trained neural network model [https://github.com/opencv/opencv_zoo/blob/main/models/face_recognition_sface/].

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

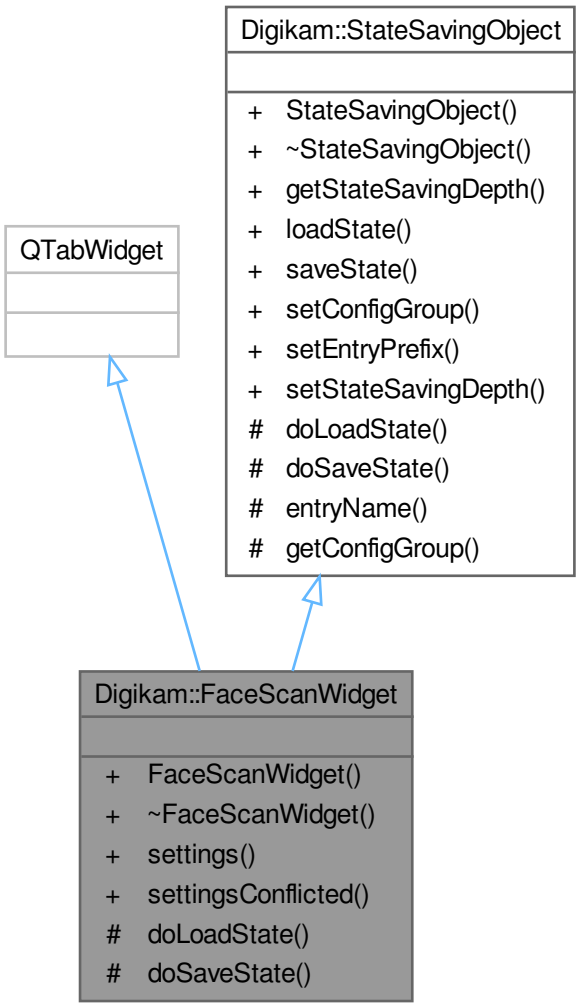
9.544.2.2 recognizeAccuracy

```
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()

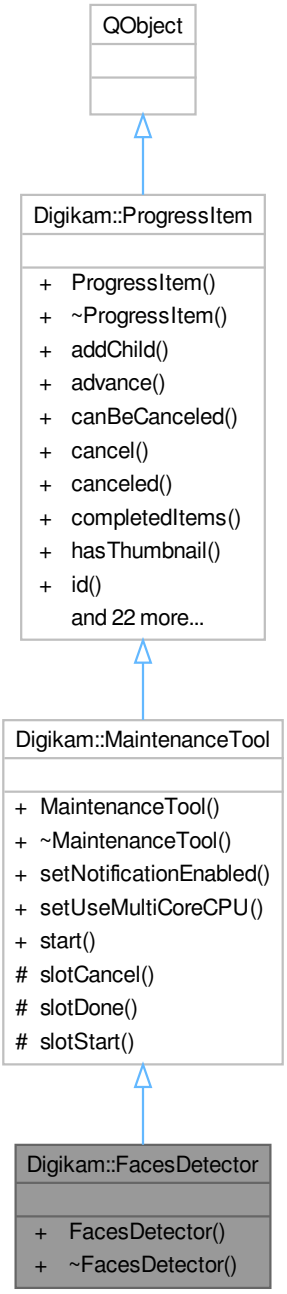
```
void Digikam::FaceScanWidget::doSaveState ( )  [override], [protected], [virtual]
```

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 elpased 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

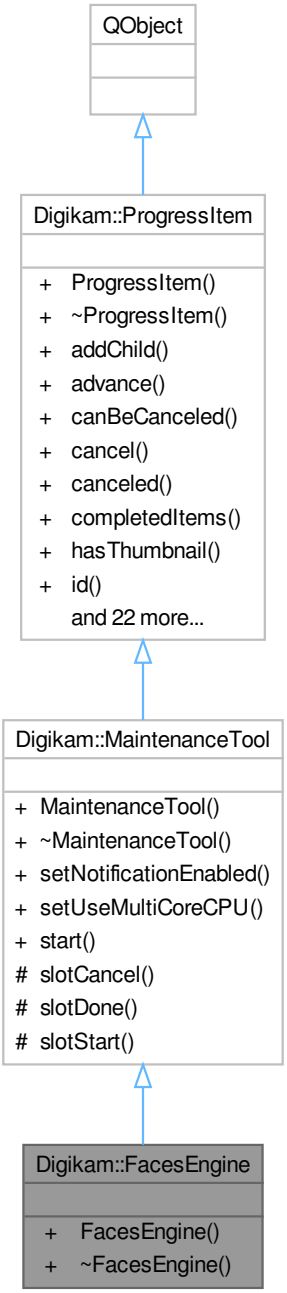
- void **start** ()

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 elpased 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 QMultiMap< 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** (qlonglong imageid, const QRect &faceRect)
- static int **getOrCreateTagForIdentity** (const QMultiMap< 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 QMultiMap< 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 QMultiMap< 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.

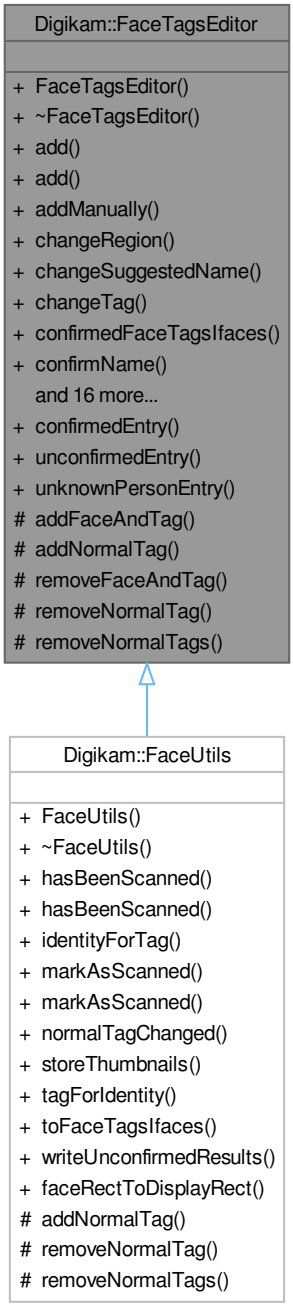
9.548.1.4 tagForPerson()

```
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** (qlonglong 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** (qlonglong imageid) const
- FaceTagsIface confirmName (const FaceTagsIface &face, int tagId=-1, const TagRegion &confirmed↩ Region=TagRegion())

  Assign the name tag for given face entry.
- QList< FaceTagsIface > **databaseFaces** (qlonglong imageid) const

  Reads the FaceTagsIfaces for the given image id from the database.
- QList< FaceTagsIface > **databaseFaces** (qlonglong imageId, FaceTagsIface::TypeFlags flags) const
- QList< FaceTagsIface > **databaseFacesForTraining** (qlonglong imageid) const
- int **faceCountForPersonInImage** (qlonglong imageid, int tagId) const

  Returns the number of faces a particular person has in the specified image.
- QList< ItemTagPair > **faceItemTagPairs** (qlonglong imageid, FaceTagsIface::TypeFlags flags) const
- QMap< QString, QString > getSuggestedNames (qlonglong id) const

  Returns a Map of Tag Regions (in XML format) to Suggested Name (from Face Recognition) for the given image.
- QList< QRect > getTagRects (qlonglong imageid) const

  Returns a list of all tag rectangles for the image.
- QList< FaceTagsIface > **ignoredFaceTagsIfaces** (qlonglong imageid) const
- int **numberOfFaces** (qlonglong imageid) const

  Returns the number of faces present in an image.
- void **removeAllFaces** (qlonglong 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** (qlonglong 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** (qlonglong imageId, const QSize &size, int oldOrientation, int newOrientation)

  Rotate face tags.
- QList< FaceTagsIface > unconfirmedFaceTagsIfaces (qlonglong imageid) const

  Returns list of Unconfirmed and Unknown faces in the Image.
- QList< FaceTagsIface > unconfirmedNameFaceTagsIfaces (qlonglong 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 (qlonglong imageId, int tagId, const TagRegion &region)

  Returns the entry that would be added if the given face is autodetected.
- static FaceTagsIface **unknownPersonEntry** (qlonglong imageId, const TagRegion &region)

Protected Member Functions

- void **addFaceAndTag** (ItemTagPair &pair, const FaceTagsIface &face, const QStringList &properties, bool addTag)
- virtual void addNormalTag (qlonglong imageId, int tagId)
- void **removeFaceAndTag** (ItemTagPair &pair, const FaceTagsIface &face, bool touchTags)
- virtual void removeNormalTag (qlonglong imageId, int tagId)
- virtual void **removeNormalTags** (qlonglong 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 (
            qlonglong 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 , **TypeFirst** = UnknownName , **TypeLast** = FaceForTraining }
- typedef QFlags< Type > **TypeFlags**

Public Member Functions

- **FaceTagsIface** (const FaceTagsIface &other)
- **FaceTagsIface** (const QString &attribute, qlonglong imageId, int tagId, const TagRegion &region)
- **FaceTagsIface** (Type type, qlonglong imageId, int tagId, const TagRegion &region)
- QString **getAutodetectedPersonString** () const

  Returns the string tagId + ',' + unconfirmedFace + ',' + regionXml.
- const QString **hash** () const

  Generate a hash based on the imageId, tagId, and rect to uniquely identify this entry in the face training DB.
- qlonglong **imageId** () 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 imageId, 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

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 FaceTagsIface **fromListing** (qlonglong 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

- qlonglong **m_imageId** = 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==.

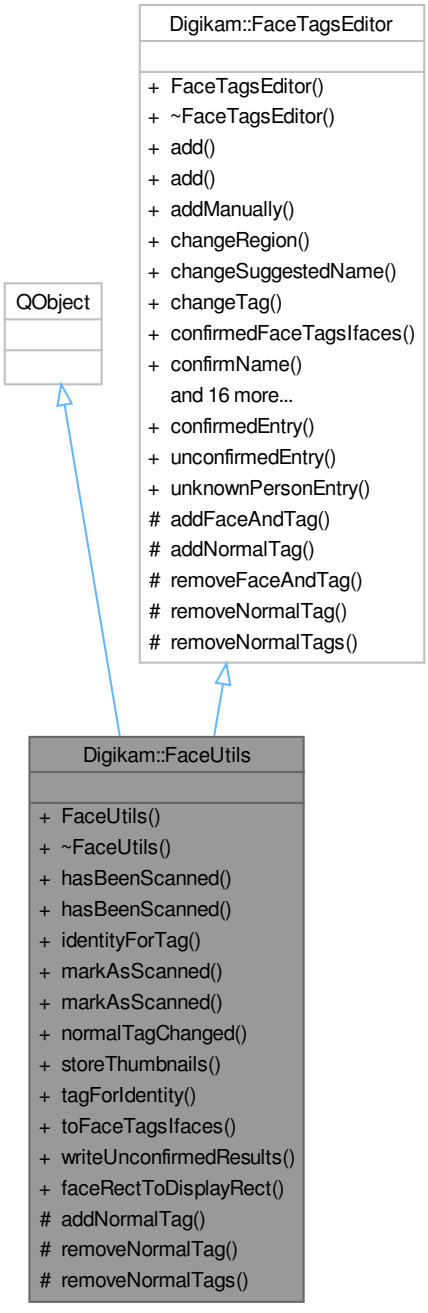
9.550.1.2 typeForAttribute()

```
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** (qlonglong imageid) const
- Identity **identityForTag** (int tagId) const
- void **markAsScanned** (const ItemInfo &info, bool hasBeenScanned=true) const
- void **markAsScanned** (qlonglong 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 (qlonglong imageid, const QList< QRectF > &detectedFaces, const QList< Identity > &recognitionResults, const QSize &fullSize) const

  Conversion.
- QList< FaceTagsIface > writeUnconfirmedResults (qlonglong imageid, const QList< QRectF > &detected↩Faces, 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** (qlonglong 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** (qlonglong imageid) const
- FaceTagsIface confirmName (const FaceTagsIface &face, int tagId=-1, const TagRegion &confirmed↩Region=TagRegion())

  Assign the name tag for given face entry.
- QList< FaceTagsIface > **databaseFaces** (qlonglong imageid) const

  Reads the FaceTagsIfaces for the given image id from the database.
- QList< FaceTagsIface > **databaseFaces** (qlonglong imageId, FaceTagsIface::TypeFlags flags) const
- QList< FaceTagsIface > **databaseFacesForTraining** (qlonglong imageid) const
- int **faceCountForPersonInImage** (qlonglong imageid, int tagId) const

  Returns the number of faces a particular person has in the specified image.
- QList< ItemTagPair > **faceItemTagPairs** (qlonglong imageid, FaceTagsIface::TypeFlags flags) const
- QMap< QString, QString > getSuggestedNames (qlonglong id) const

  Returns a Map of Tag Regions (in XML format) to Suggested Name (from Face Recognition) for the given image.
- QList< QRect > getTagRects (qlonglong imageid) const

  Returns a list of all tag rectangles for the image.
- QList< FaceTagsIface > **ignoredFaceTagsIfaces** (qlonglong imageid) const
- int **numberOfFaces** (qlonglong imageid) const

  Returns the number of faces present in an image.

- void **removeAllFaces** (qlonglong 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** (qlonglong 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** (qlonglong imageId, const QSize &size, int oldOrientation, int newOrientation)

  Rotate face tags.
- QList< FaceTagsIface > unconfirmedFaceTagsIfaces (qlonglong imageid) const

  Returns list of Unconfirmed and Unknown faces in the Image.
- QList< FaceTagsIface > unconfirmedNameFaceTagsIfaces (qlonglong 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 (qlonglong imageId, int tagId, const TagRegion &region)

  Returns the entry that would be added if the given face is autodetected.
- static FaceTagsIface **unknownPersonEntry** (qlonglong imageId, const TagRegion &region)

Protected Member Functions

- void addNormalTag (qlonglong imageId, int tagId) override

  Reimplemented from parent class.
- void removeNormalTag (qlonglong imageId, int tagId) override
- void removeNormalTags (qlonglong 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 (
            qlonglong 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 (
            qlonglong 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 (
            qlonglong 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 (
            qlonglong 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 QMultiMap< 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 QMultiMap< QString, QString > &attributes)

  Adds or sets, resp., the attributes of an identity.
- Identity **addIdentityDebug** (const QMultiMap< 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 QMultiMap< 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 QMultiMap< 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 QMultiMap< 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 QMultiMap< 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 **versionIsRight** () const
- bool **versionIsRight** (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.

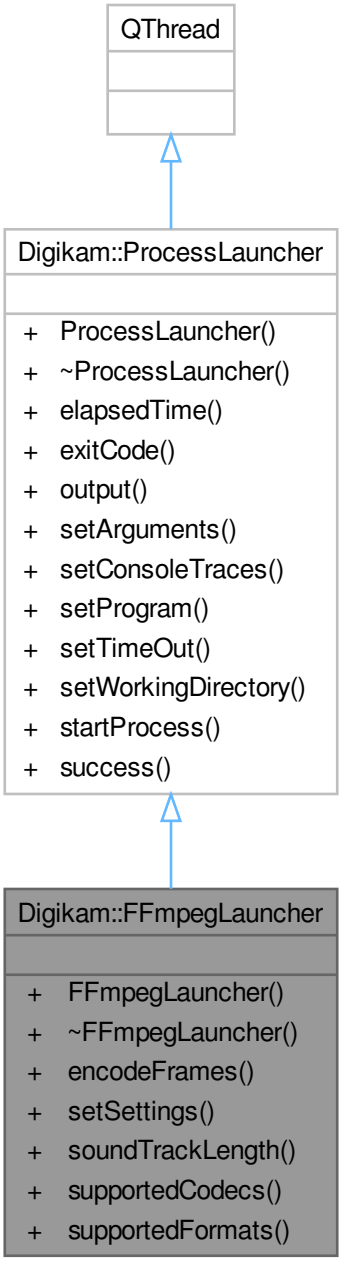
9.554.1.3 getVideoCodecsProperties()

```
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 outout 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 msecs)
- void **setWorkingDirectory** (const QString &dir)
- void **startProcess** ()

  Start the process.

- bool **success** () const

  Return true if the process is startd 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

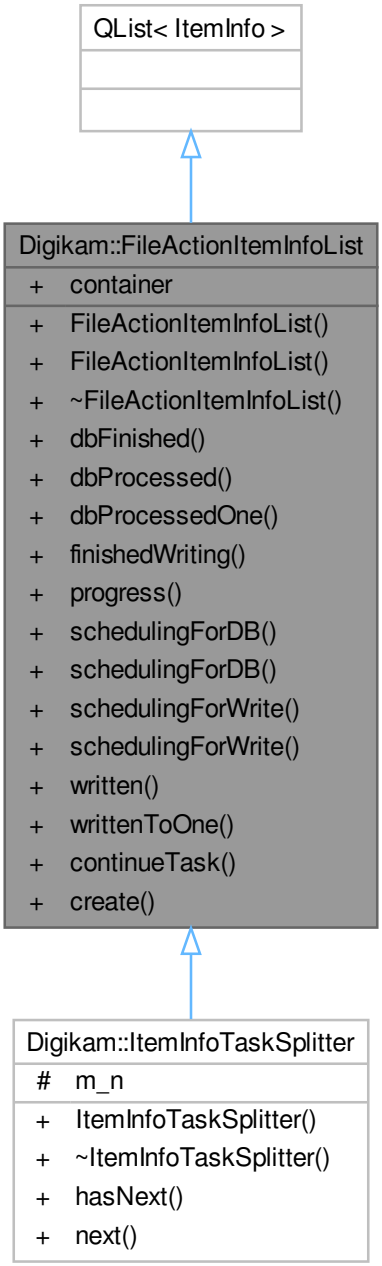
- **FieldQueryBuilder** (QString &sql, SearchXmlCachingReader &reader, QList< QVariant > *boundValues, ItemQueryPostHooks *const hooks, SearchXml::Relation relation)
- void **addChoiceIntField** (const QString &name)
- void **addChoiceStringField** (const QString &name)
- void **addDateField** (const 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

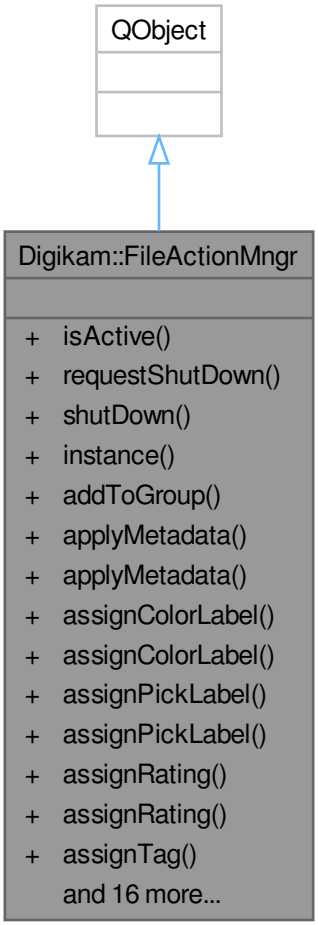
- static FileActionItemInfoList **continueTask** (const QList< ItemInfo > &list, FileActionProgressItemContainer *const container)
- static FileActionItemInfoList **create** (const QList< ItemInfo > &list)

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 FileActionMngr * **instance** ()

Friends

- class **FileActionMngrCreator**

9.558.1 Member Function Documentation

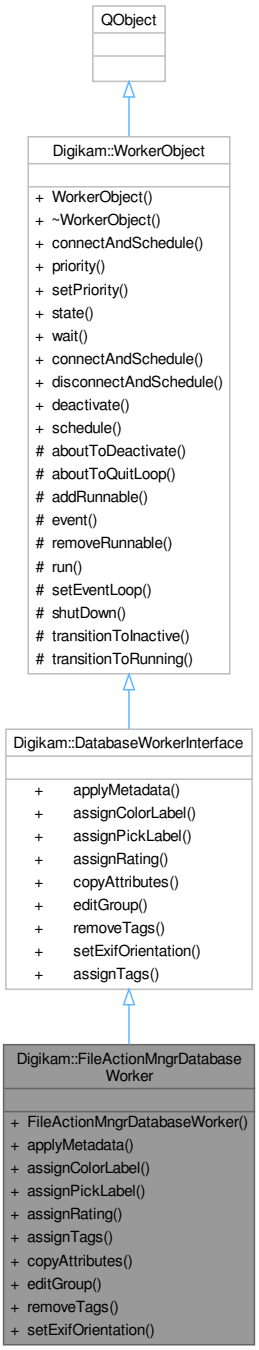
9.558.1.1 transform

```
void Digikam::FileActionMngr::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.

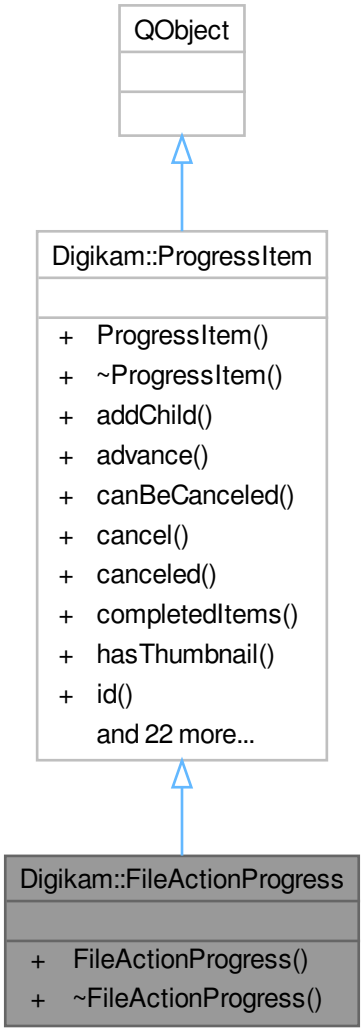
9.560.1.4 writeOrientationToFiles()

```
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)

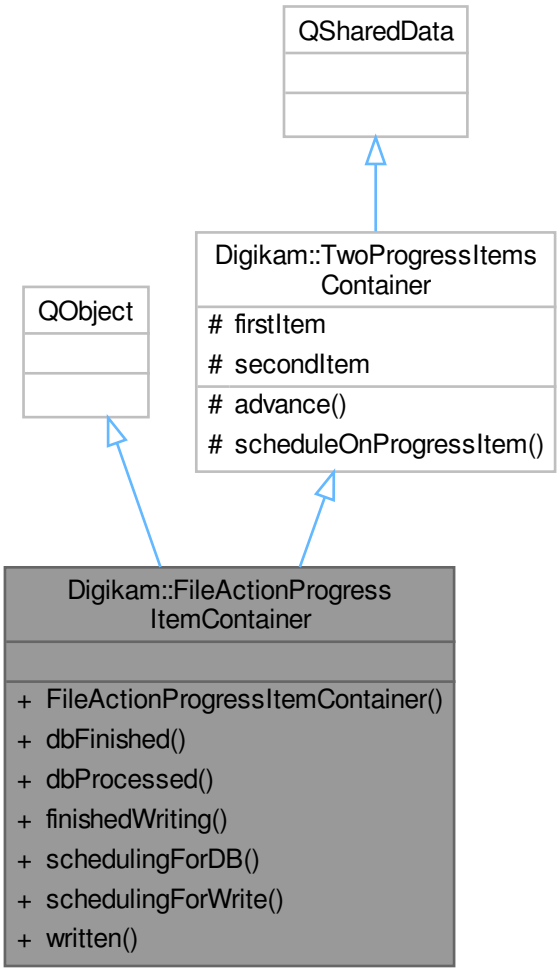
  Sets whether this item has a thumbnail.
- void **setTotalItems** (unsigned int v)
- void setUsesBusyIndicator (bool useBusyIndicator)

  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 **signalWrittingDone** ()

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**

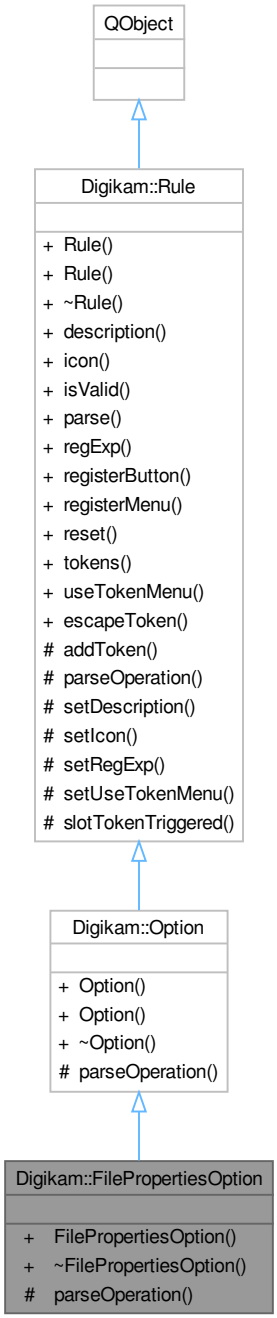
9.563 Digikam::FileActionProgressItemCreator Class Reference

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

settings	contains settings
match	result of the regular expression match done in Option::parse()

Returns

Implements Digikam::Option.

9.565 Digikam::FileReadLocker Class Reference

Public Member Functions

- **FileReadLocker** (const QString &filePath)

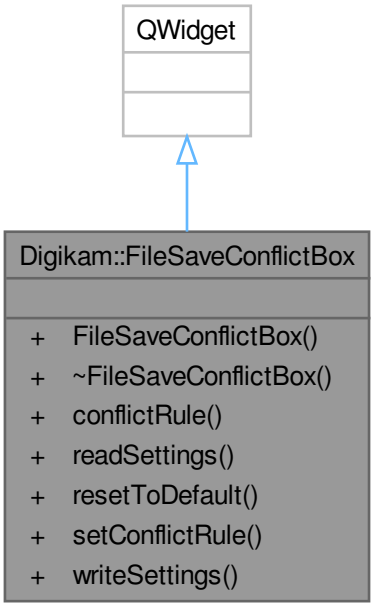
9.566 Digikam::FileReadWriteLockKey Class Reference

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

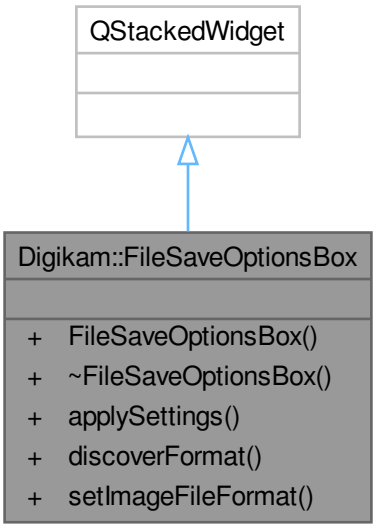
- void **signalConflictButtonChanged** (int)

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	**Warning** Order is important here. See filesaveoptionbox.cpp which use these values to fill a stack of widgets.

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

parent	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

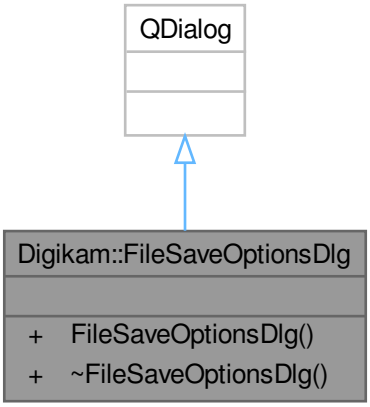
filename	file name to discover the desired format from
fallback	the fallback format to return if no format could be discovered based on the filename

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:

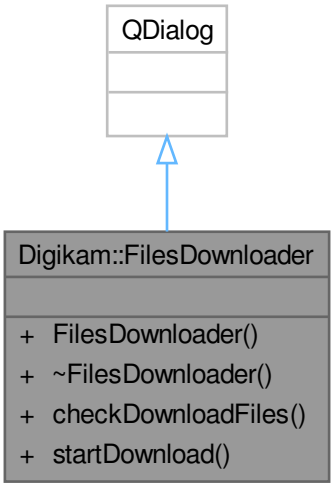


Public Member Functions

- **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

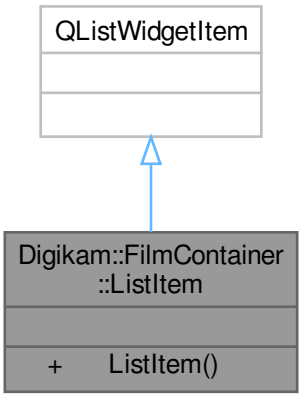
- static QList< ListItem * > **profileItemList** (QListWidget *const view)

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** (DImg ∗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 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** ()

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 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 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.

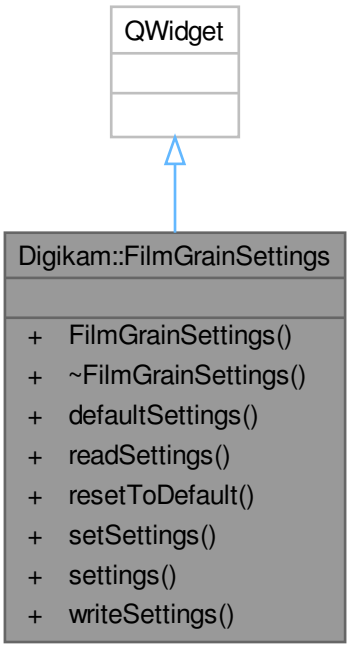
9.577.1.3 readParameters()

```
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

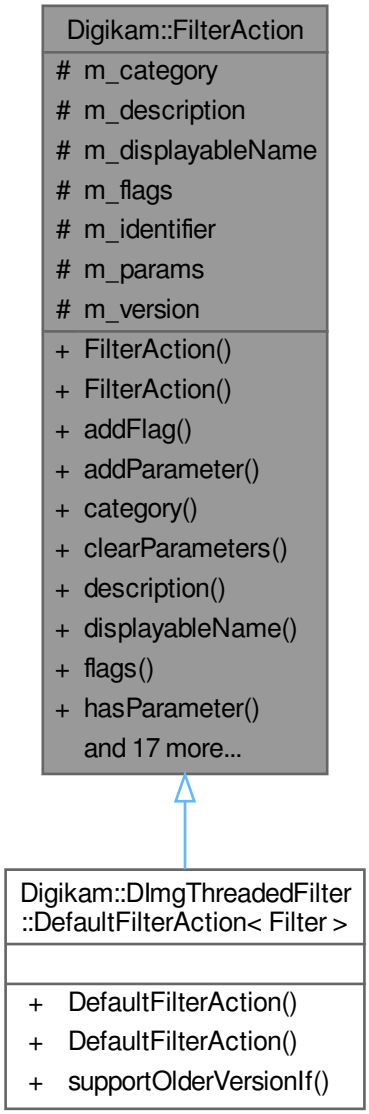
- void **fromString** (const QString &filter)
- bool **match** (const QStringList &wildcards, const QString &name)
- bool **matchesCurrentFilter** (const CamItemInfo &item)
- const QStringList & **mimeWildcards** (const QString &mime)
- const QRegularExpression & **regexp** (const QString &wildcard)
- 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 **displayableName** () 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 **setDisplayableName** (const QString &displayableName)
- 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. **Note** 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

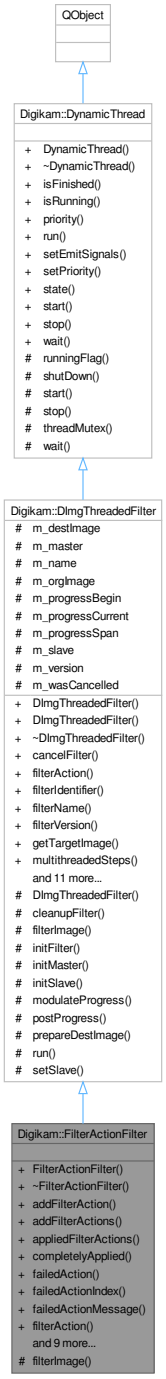
```
Category Digikam::FilterAction::m_category = ReproducibleFilter  [protected]
```

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 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

- 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.

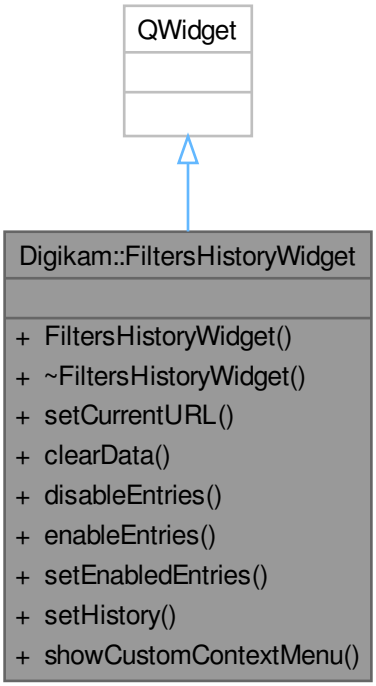
9.581.1.8 setContinueOnError()

```
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)

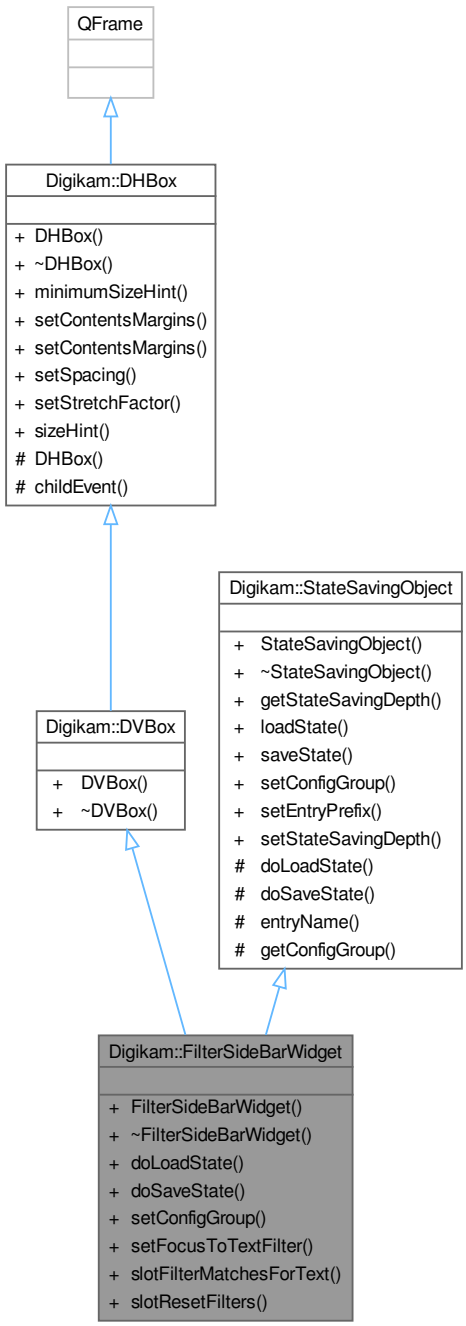
Public Member Functions

- **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

parent	the parent for qt parent child mechanism
tagFilterModel	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

group	config group to use for state saving and restoring

Reimplemented from Digikam::StateSavingObject.

9.583.3.4 signalTagFilterChanged

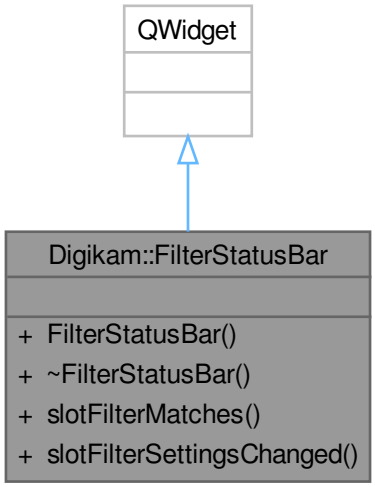
```
void Digikam::FilterSideBarWidget::signalTagFilterChanged (
            const QList< int > & includedTags,
            const QList< int > & excludedTags,
            ItemFilterSettings::MatchingCondition matchingCond,
            bool showUnTagged,
            const QList< int > & clTagIds,
            const QList< int > & plTagIds )  [signal]
```

Parameters

includedTags	a list of included tag ids
excludedTags	a list of excluded tag ids
matchingCond	condition to join the selected tags
showUnTagged	if this is true, only photos without a tag shall be shown
clTagIds	a list of color label tag ids
plTagIds	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** ()

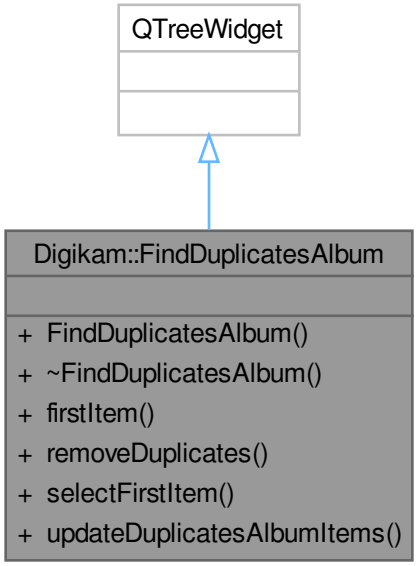
Public Member Functions

- **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:

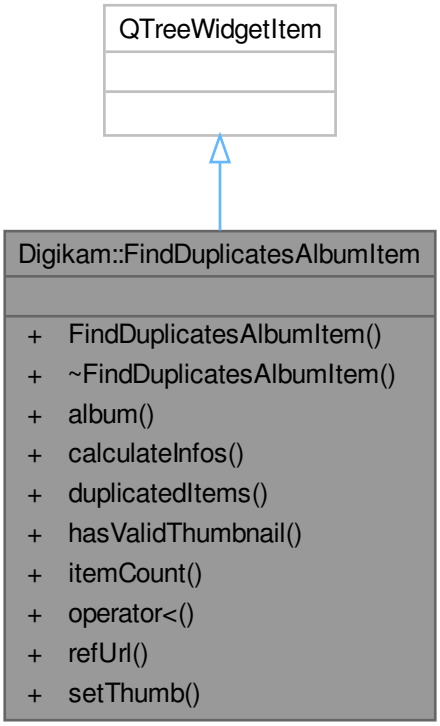


Public Member Functions

- **FindDuplicatesAlbum** (QWidget *const parent=nullptr)
- QTreeWidgetItem * **firstItem** ()
- void **removeDuplicates** ()
- void **selectFirstItem** ()
- void **updateDuplicatesAlbumItems** (const QList< SAlbum * > &sAlbumsToRebuild, const QList< qlonglong > &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

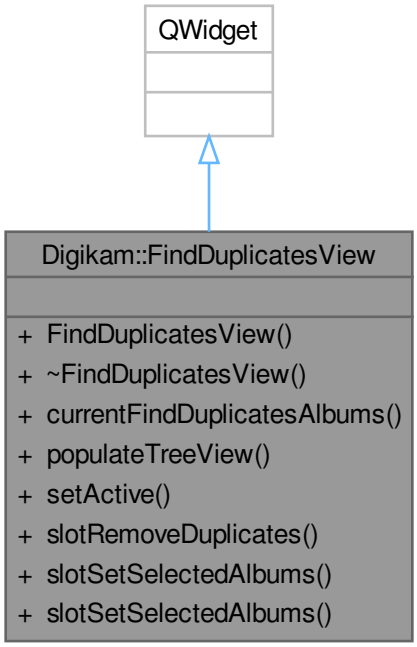
- **FindDuplicatesAlbumItem** (QTreeWidget ∗const parent, SAlbum ∗const album)
- SAlbum ∗ **album** () const
- void **calculateInfos** (const QList< qlonglong > &deletedImages=QList< qlonglong >())

  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

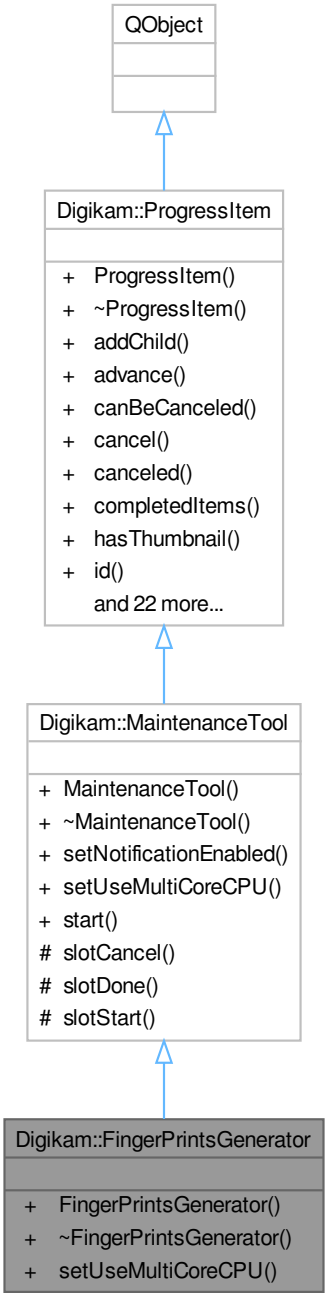
- void **signalScanNotification** (const QString &msg, int type)

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 elpased 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

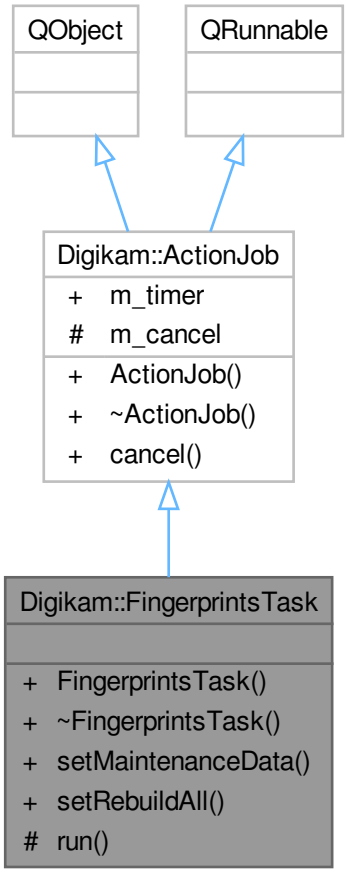
9.588.2.1 setUseMultiCoreCPU()

```
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.

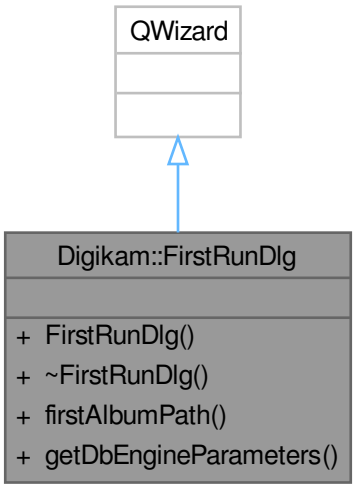
Protected Attributes inherited from Digikam::ActionJob

- 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 contructors.
- 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

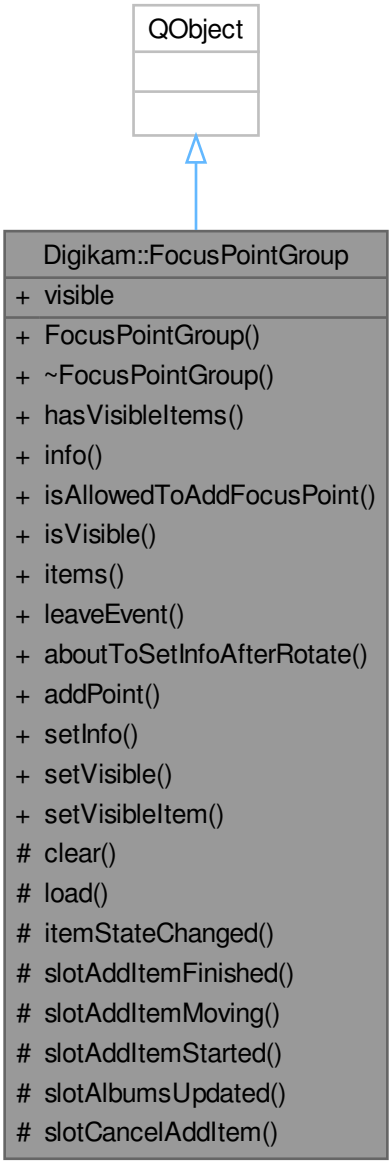
9.591.3.1 setType()

```
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

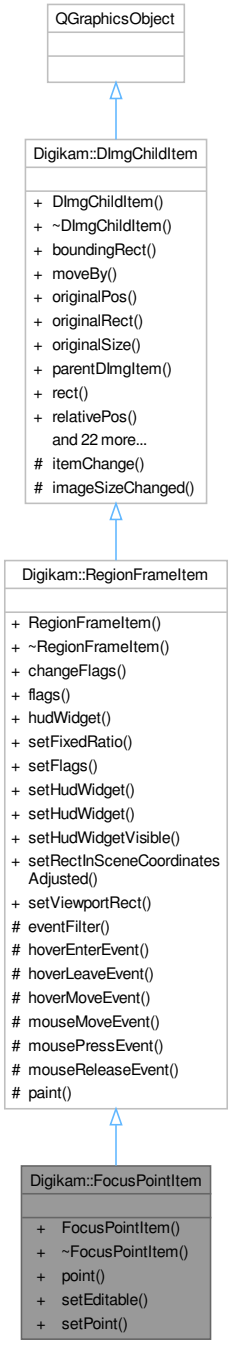
- void **clear** ()
- void **load** ()

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

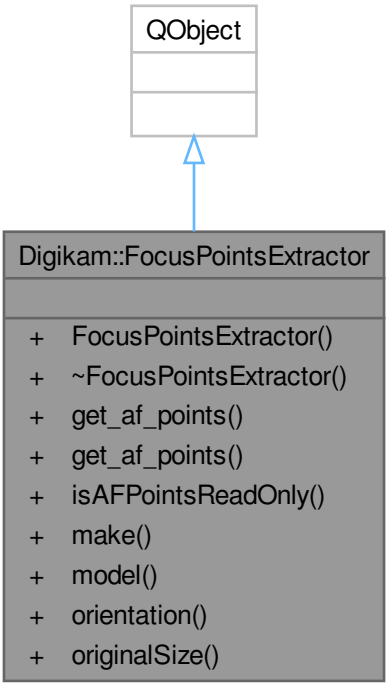
- bool **eventFilter** (QObject *watched, QEvent *event) override
- void **hoverEnterEvent** (QGraphicsSceneHoverEvent *event) override
- void **hoverLeaveEvent** (QGraphicsSceneHoverEvent *event) override
- void **hoverMoveEvent** (QGraphicsSceneHoverEvent *event) override
- void **mouseMoveEvent** (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

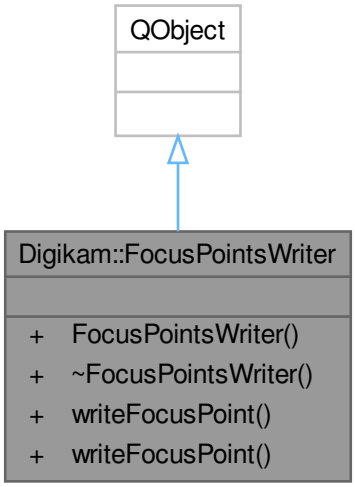
9.594.1.1 ListAFPoints

```
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

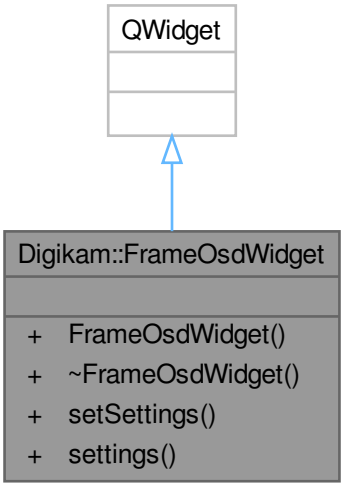
  Print rating while streaming.
- bool **printTags** = false

  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 &timg, 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 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 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.

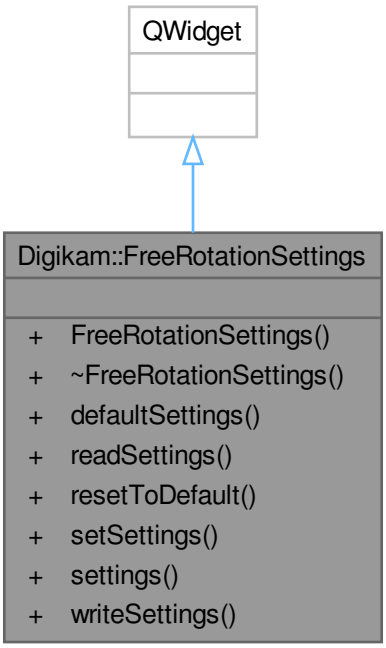
9.601.1.3 readParameters()

```
void Digikam::FreeRotationFilter::readParameters (
            const FilterAction & action )  [override], [virtual]
```

Implements Digikam::DImgThreadedFilter.

9.602 Digikam::FreeRotationSettings Class Reference

Inheritance diagram for Digikam::FreeRotationSettings:



Signals

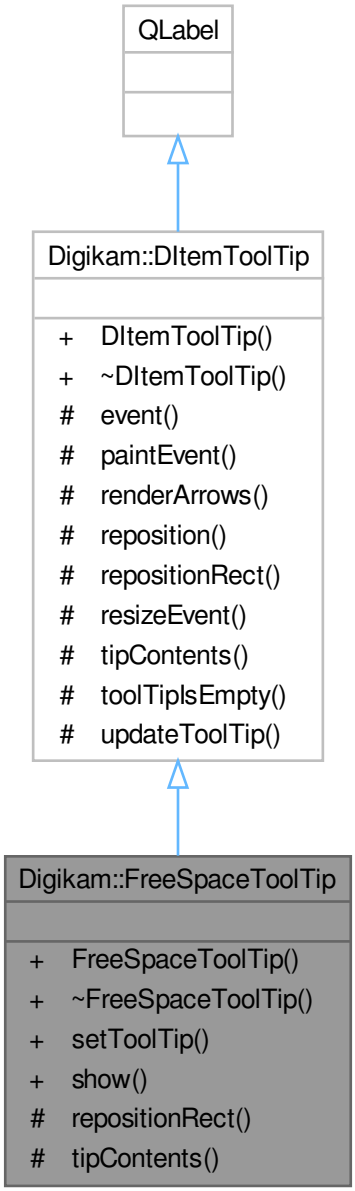
- void **signalSettingsChanged** ()

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 **toolTipIsEmpty** () const
- void **updateToolTip** ()

9.603.1 Member Function Documentation

9.603.1.1 repositionRect()

```
QRect Digikam::FreeSpaceToolTip::repositionRect ( )  [override], [protected], [virtual]
```

Implements Digikam::DItemToolTip.

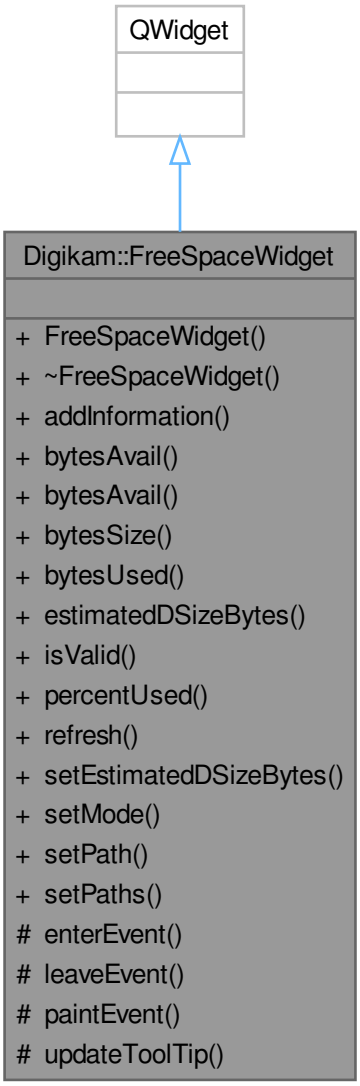
9.603.1.2 tipContents()

```
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** ()

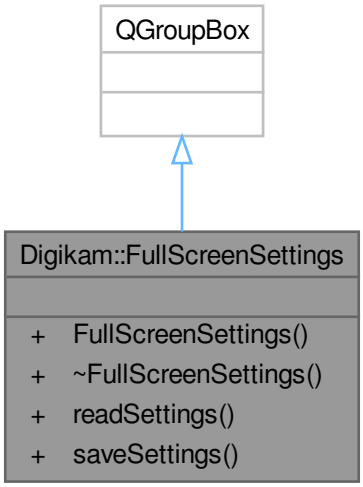
9.605 Digikam::FullObjectDetection Class Reference

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:

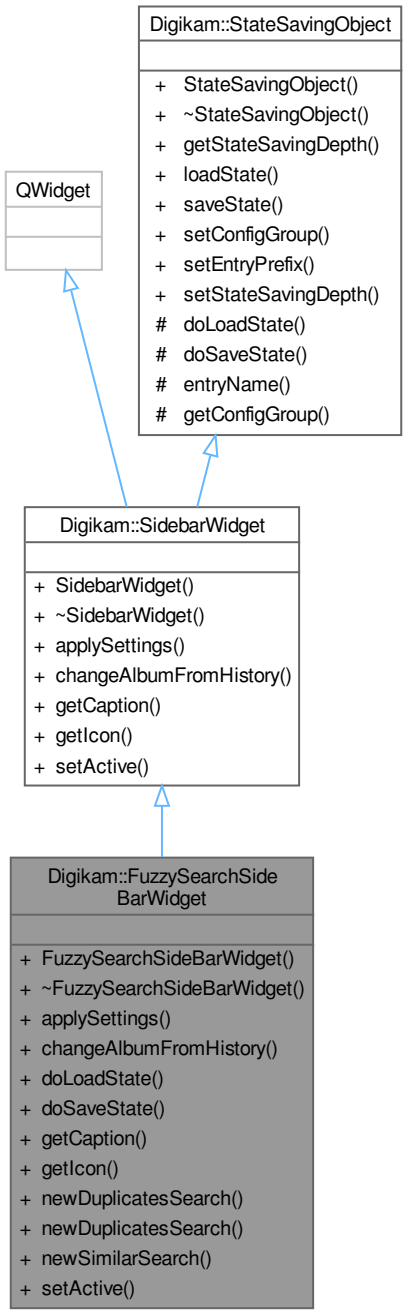


Public Member Functions

- **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< PAlbum ∗ > &albums)
- void **newDuplicatesSearch** (const QList< TAlbum ∗ > &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]
```

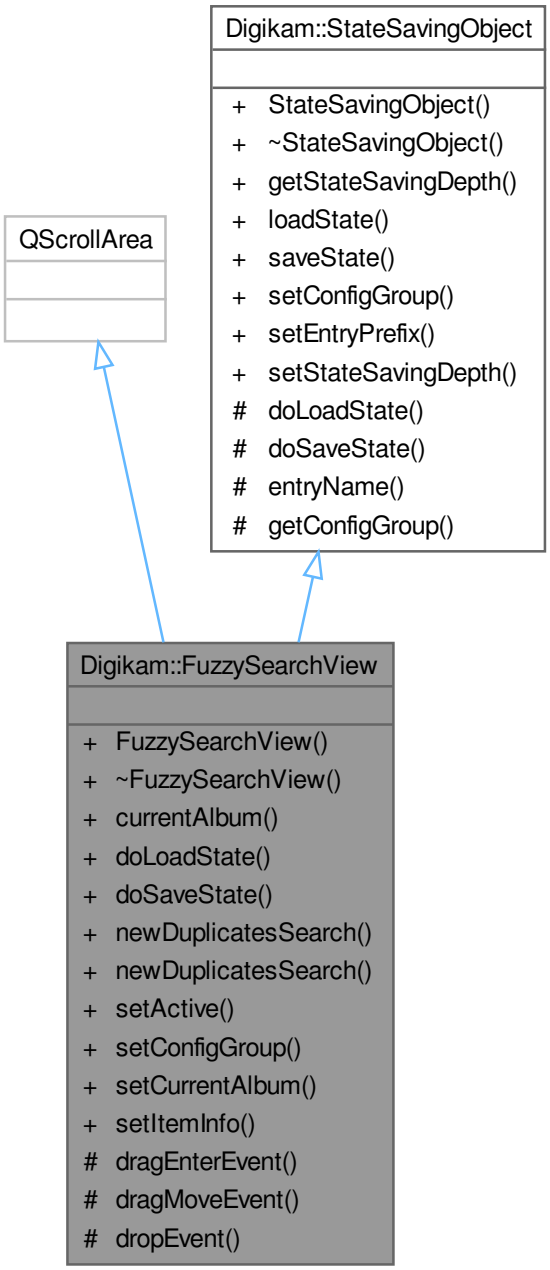
Parameters

active	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 search↩ ModificationHelper, QWidget *const parent=nullptr)
- SAlbum * **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< PAlbum * > &albums)
- void **newDuplicatesSearch** (const QList< TAlbum * > &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** (SAlbum *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

group	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

azimuth	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

latitude	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

longitude	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

distance	The orthodromic distance value.

9.610.2.6 computeDirection()

```
bool Digikam::GeodeticCalculator::computeDirection ( )
```

Computes the azimuth and orthodromic distance from the startingGeographicPoint() and the destination↩ GeographicPoint().

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 set↩ Direction(...) 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

latitude1	The latitude of the first point (in decimal degrees).
latitude2	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

P1	The latitude of the first point (in radians).
P2	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

longitude	The longitude in decimal degrees between -180 and +180°
latitude	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

azimuth	The azimuth in decimal degrees from -180° to 180°.
distance	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

longitude	The longitude in decimal degrees between -180 and +180°
latitude	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.

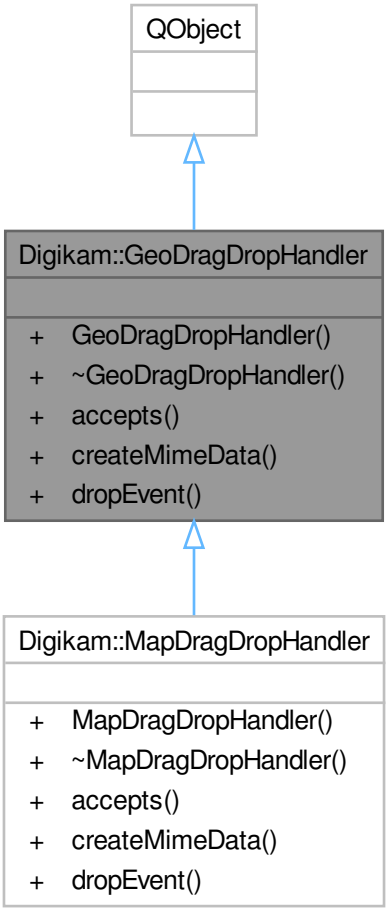
9.610.3.6 m_TOLERANCE_CHECK

```
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::GeoIfaceCluster Class Reference

Public Types

- typedef QList< GeoIfaceCluster > **List**
- enum **PixmapType** { **PixmapMarker** , **PixmapCircle** , **PixmapImage** }

Public Attributes

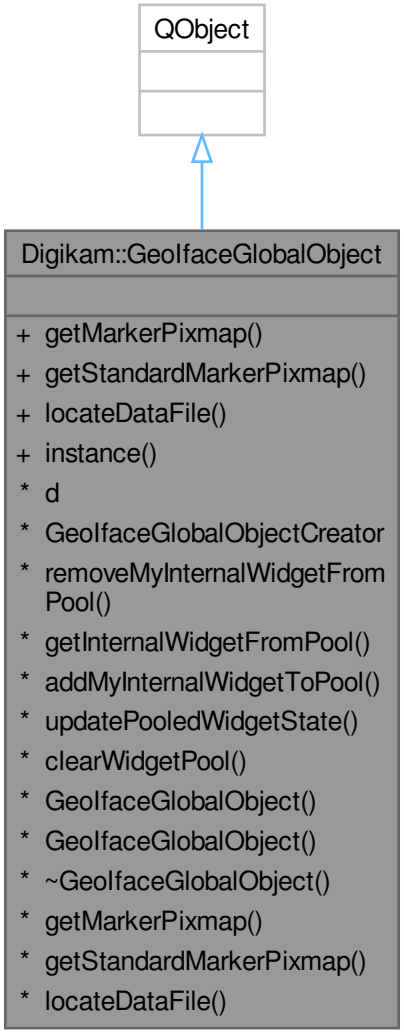
- GeoCoordinates **coordinates**
- GeoGroupState **groupState**
- int **markerCount**
- int **markerSelectedCount**
- QPoint **pixelPos**
- QPoint **pixmapOffset**

  anchor point of the image, measured from bottom-left
- QSize **pixmapSize**
- enum Digikam::GeoIfaceCluster::PixmapType **pixmapType**
- QMap< int, QVariant > **representativeMarkers**
- QList< TileIndex > **tileIndicesList**

9.613 Digikam::GeoIfaceGlobalObject Class Reference

Global object for geolocation interface to hold items common to all geolocation interface Widget instances.

Inheritance diagram for Digikam::GeoIfaceGlobalObject:



Public Member Functions

Shared pixmaps

- QPixmap **getMarkerPixmap** (const QString &pixmapId)
- QPixmap **getStandardMarkerPixmap** ()
- QUrl **locateDataFile** (const QString &filename)

Static Public Member Functions

- static GeoIfaceGlobalObject * **instance** ()

Shared internal map widgets

- class **GeoIfaceGlobalObjectCreator**
- void **removeMyInternalWidgetFromPool** (const MapBackend ∗const mapBackend)
- bool **getInternalWidgetFromPool** (const MapBackend ∗const mapBackend, GeoIfaceInternalWidgetInfo ∗const targetInfo)
- void **addMyInternalWidgetToPool** (const GeoIfaceInternalWidgetInfo &info)
- void **updatePooledWidgetState** (const QWidget ∗const widget, const GeoIfaceInternalWidgetInfo::↩ InternalWidgetState newState)
- void **clearWidgetPool** ()

9.614 Digikam::GeoIfaceInternalWidgetInfo Class Reference

Class to hold information about map widgets stored in the GeoIfaceGlobalObject.

Public Types

- typedef void(∗ **DeleteFunction**) (GeoIfaceInternalWidgetInfo ∗const info)
- enum **InternalWidgetState** { **InternalWidgetReleased** = 1 , **InternalWidgetUndocked** = 2 , **Internal**↩ **WidgetStillDocked** = 4 }
- typedef QFlags< InternalWidgetState > **InternalWidgetStates**

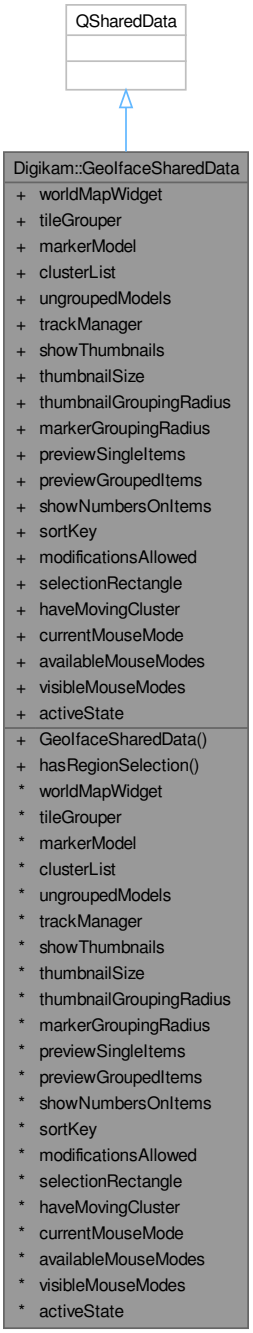
Public Attributes

- QVariant **backendData**
- QString **backendName**
- QPointer< QObject > **currentOwner**
- DeleteFunction **deleteFunction**
- InternalWidgetStates **state**
- QPointer< QWidget > **widget**

9.614.1 Detailed Description

9.615 Digikam::GeoIfaceSharedData Class Reference

Inheritance diagram for Digikam::GeoIfaceSharedData:



Public Member Functions

- bool hasRegionSelection () const

Public Attributes

Objects

- MapWidget * **worldMapWidget**
- TileGrouper * **tileGrouper**
- AbstractMarkerTiler * **markerModel**
- GeoIfaceCluster::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**

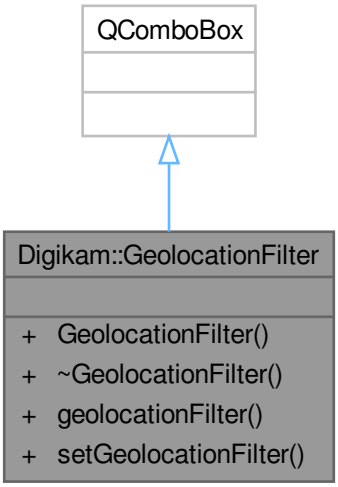
9.615.1 Member Function Documentation

9.615.1.1 hasRegionSelection()

```
bool Digikam::GeoIfaceSharedData::hasRegionSelection ( ) const  [inline]
```

9.616 Digikam::GeolocationFilter Class Reference

Inheritance diagram for Digikam::GeolocationFilter:



Signals

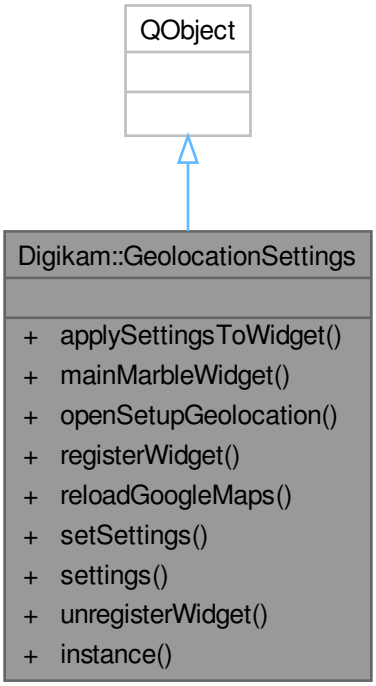
- void **signalFilterChanged** (const ItemFilterSettings::GeolocationCondition &condition)

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

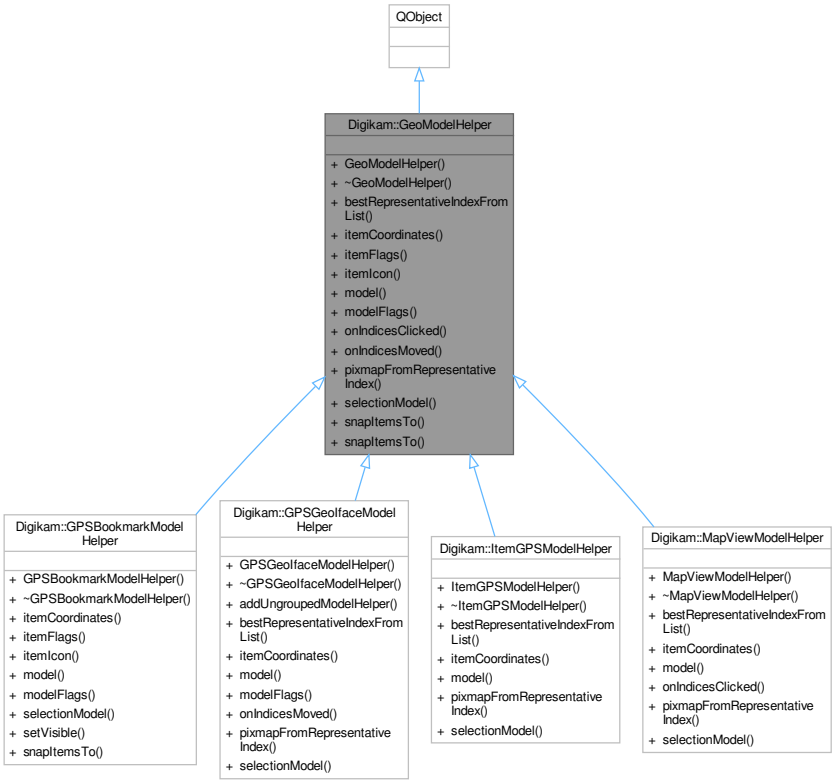
Public Attributes

- 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 > &snapped↩ Indices)

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

index	Modelindex of the marker.
offset	Offset of the zero point in the icon, given from the top-left.
size	the size of the icon, only populated if a URL is returned.
pixmap	Holder for the pixmap of the icon.
url	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::GPSGeoIfaceModelHelper.

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::GPSGeoIfaceModelHelper.

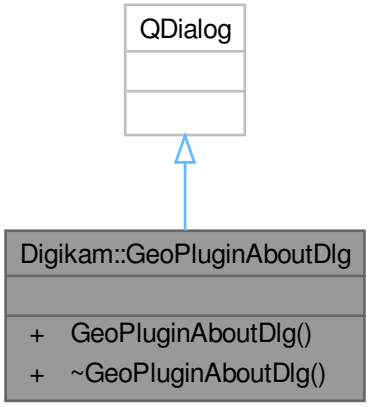
9.619.2.7 selectionModel()

```
virtual QItemSelectionModel * Digikam::GeoModelHelper::selectionModel ( ) const  [pure virtual]
```

Implemented in Digikam::MapViewModelHelper.

9.620 Digikam::GeoPluginAboutDlg Class Reference

Inheritance diagram for Digikam::GeoPluginAboutDlg:



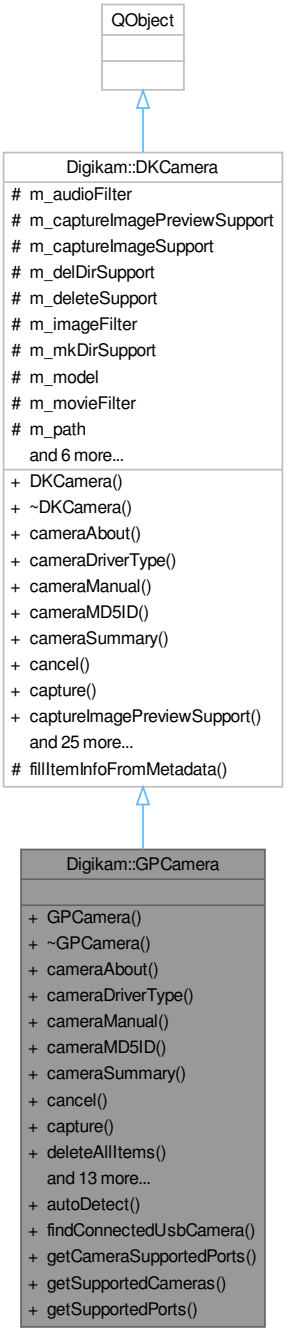
Public Member Functions

- **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 (
            qint64 & bytesSize,
            qint64 & 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.

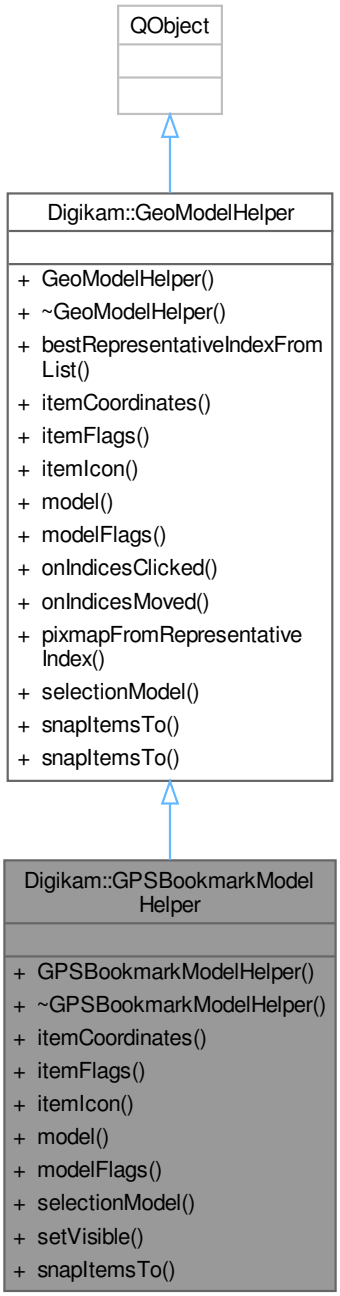
9.621.1.19 uploadItem()

```
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 > &snapped↩ Indices)

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

index	Modelindex of the marker.
offset	Offset of the zero point in the icon, given from the top-left.
size	the size of the icon, only populated if a URL is returned.
pixmap	Holder for the pixmap of the icon.
url	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.

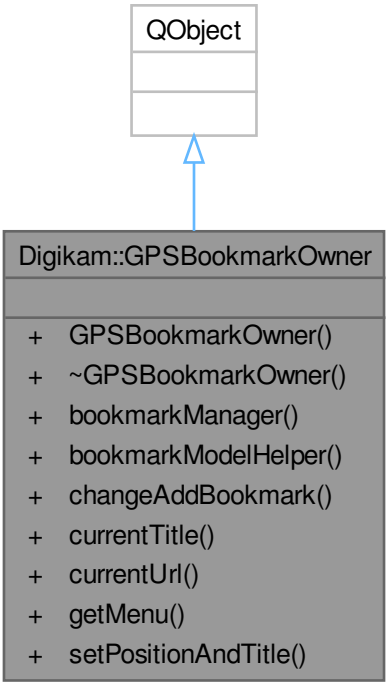
9.622.1.7 snapItemsTo()

```
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

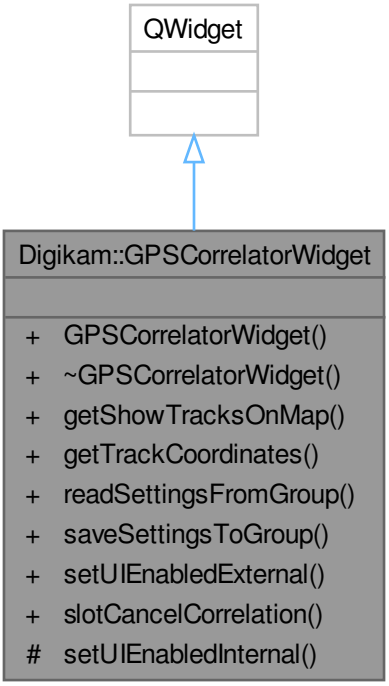
- void **positionSelected** (const GPSDataContainer &position)

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 &cancel↩ Slot)
- 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

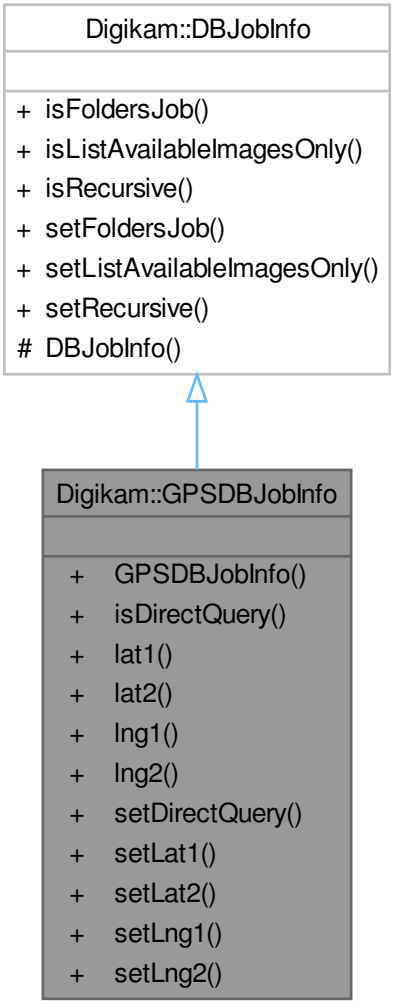
- void **clear** ()
- void **clearAltitude** ()
- void **clearDop** ()
- void **clearFixType** ()
- void **clearNonCoordinates** ()
- void **clearNSatellites** ()
- void **clearSpeed** ()
- HasFlags **flags** () const
- GeoCoordinates **getCoordinates** () const
- qreal **getDop** () const
- qreal **getFixType** () 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

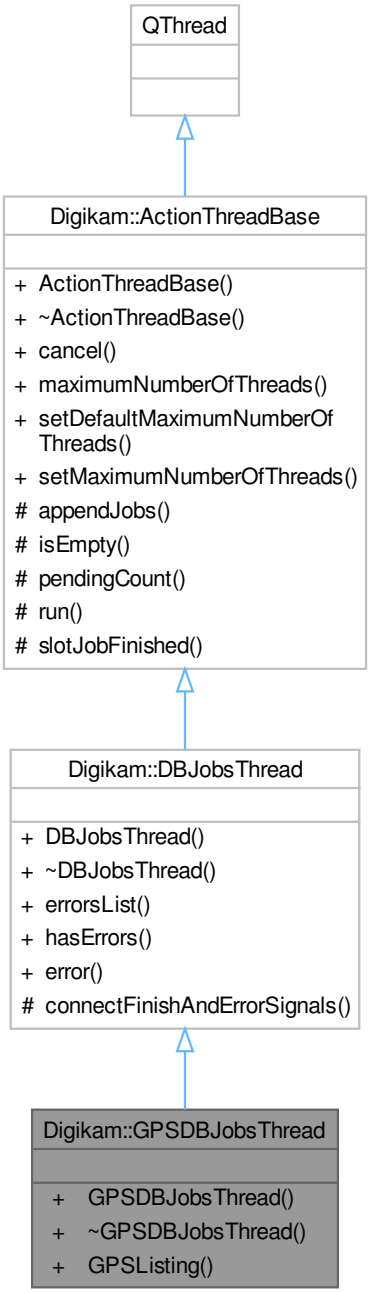
- bool **isDirectQuery** () const
- qreal **lat1** () const
- qreal **lat2** () const
- qreal **lng1** () const
- qreal **lng2** () const
- void **setDirectQuery** ()
- void **setLat1** (qreal lat)
- void **setLat2** (qreal lat)
- void **setLng1** (qreal lng)
- void **setLng2** (qreal lng)

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()

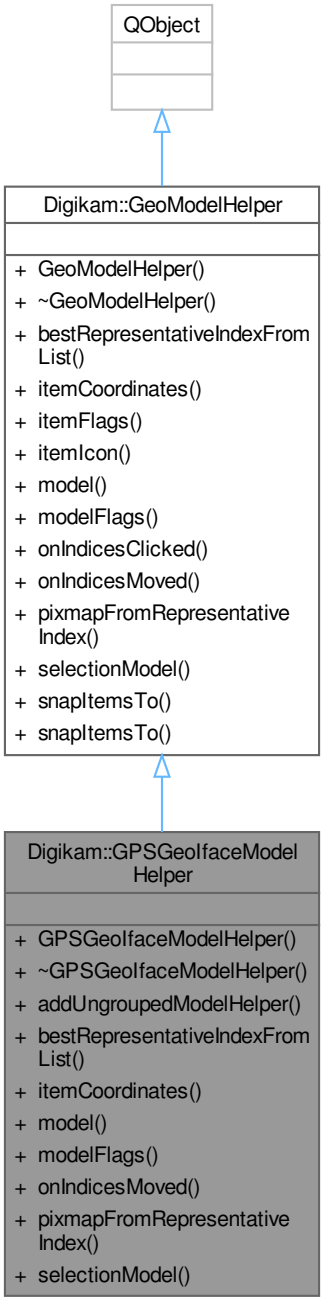
```
void Digikam::GPSDBJobsThread::GPSListing (
            const GPSDBJobInfo & info )
```

Parameters

info	represents the GPS job info

9.628 Digikam::GPSGeoIfaceModelHelper Class Reference

Inheritance diagram for Digikam::GPSGeoIfaceModelHelper:



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

- **GPSGeoIfaceModelHelper** (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

  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.
- 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 )
```

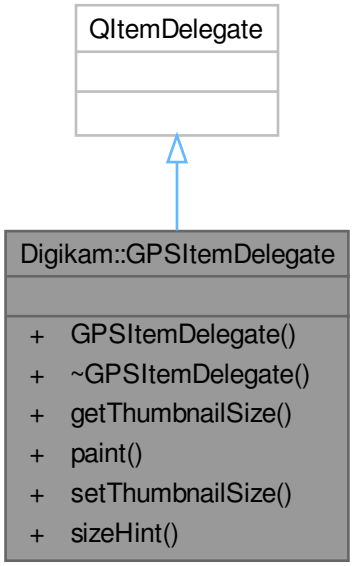
This function adds them.

Parameters

externalTagList	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

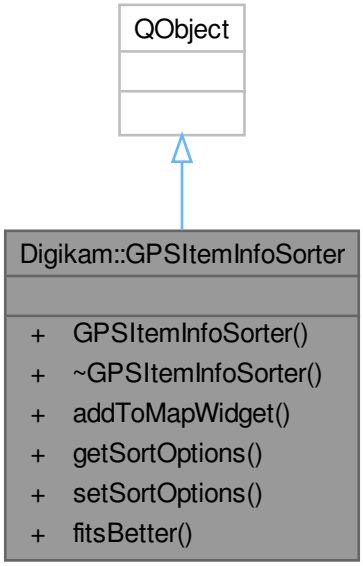
- static GPSItemInfo **fromIdCoordinatesRatingDateTime** (const qlonglong p_id, const GeoCoordinates &p↩ _coordinates, const int p_rating, const QDateTime &p_creationDate)

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

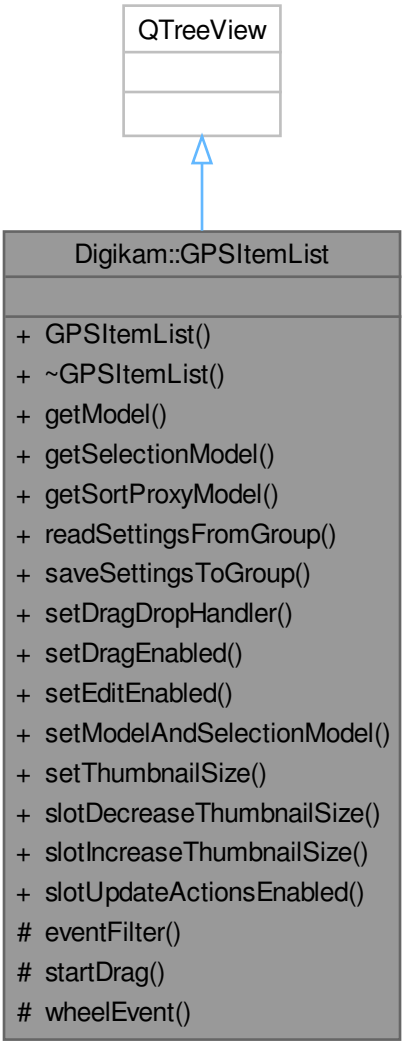
- **GPSItemInfoSorter** (QObject *const parent)
- void **addToMapWidget** (MapWidget *const mapWidget)
- SortOptions **getSortOptions** () const
- void **setSortOptions** (const SortOptions sortOptions)

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

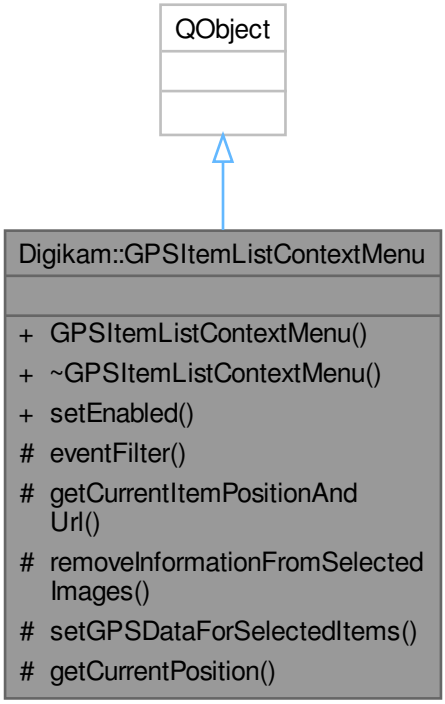
- **GPSItemList** (QWidget ∗const parent=nullptr)
- GPSItemModel ∗ **getModel** () const
- QItemSelectionModel ∗ **getSelectionModel** () const
- GPSItemSortProxyModel ∗ **getSortProxyModel** () const
- void **readSettingsFromGroup** (const KConfigGroup ∗const group)
- void **saveSettingsToGroup** (KConfigGroup ∗const group)
- void **setDragDropHandler** (ItemListDragDropHandler ∗const dragDropHandler)
- void **setDragEnabled** (const bool state)
- void **setEditEnabled** (const bool state)
- void **setModelAndSelectionModel** (GPSItemModel ∗const model, QItemSelectionModel ∗const selection↩ Model)
- 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 &cancel↩ Slot)
- void **signalUndoCommand** (GPSUndoCommand ∗undoCommand)

Public Member Functions

- **GPSItemListContextMenu** (GPSItemList ∗const imagesList, GPSBookmarkOwner ∗const bookmark↩ Owner=nullptr)
- void **setEnabled** (const bool state)

Protected Member Functions

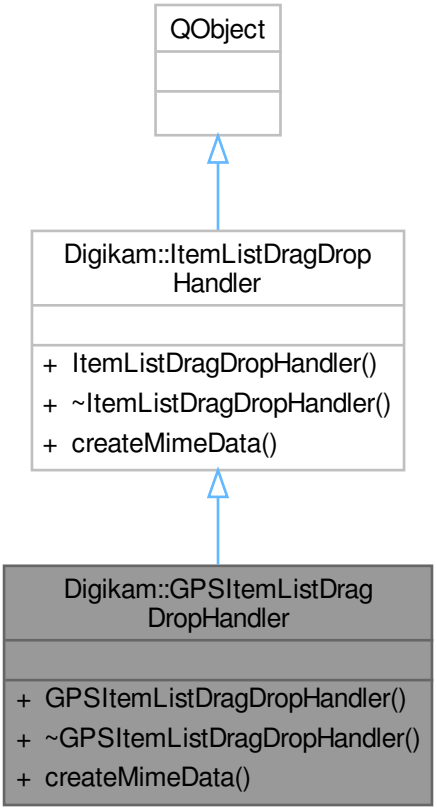
- bool **eventFilter** (QObject ∗watched, QEvent ∗event) override
- bool **getCurrentItemPositionAndUrl** (GPSDataContainer ∗const gpsInfo, QUrl ∗const itemUrl)
- void **removeInformationFromSelectedImages** (const GPSDataContainer::HasFlags flagsToClear, const QString &undoDescription)
- void **setGPSDataForSelectedItems** (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

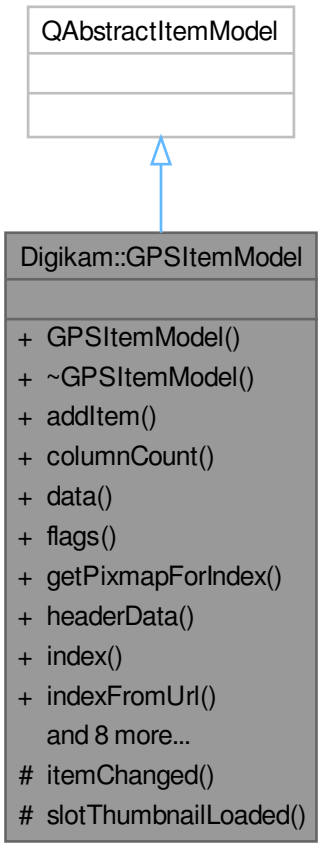
9.635.1.1 createMimeData()

```
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

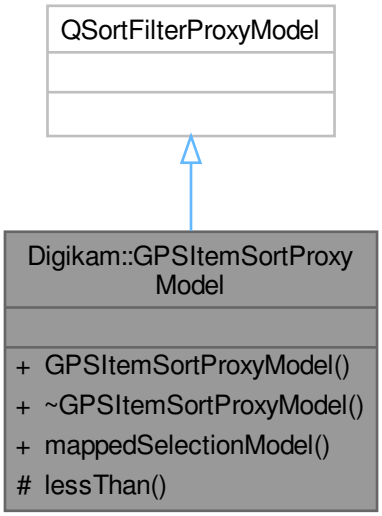
- void **itemChanged** (GPSItemContainer ∗const changedItem)

Friends

- class **GPSItemContainer**

9.637 Digikam::GPSItemSortProxyModel Class Reference

Inheritance diagram for Digikam::GPSItemSortProxyModel:



Public Member Functions

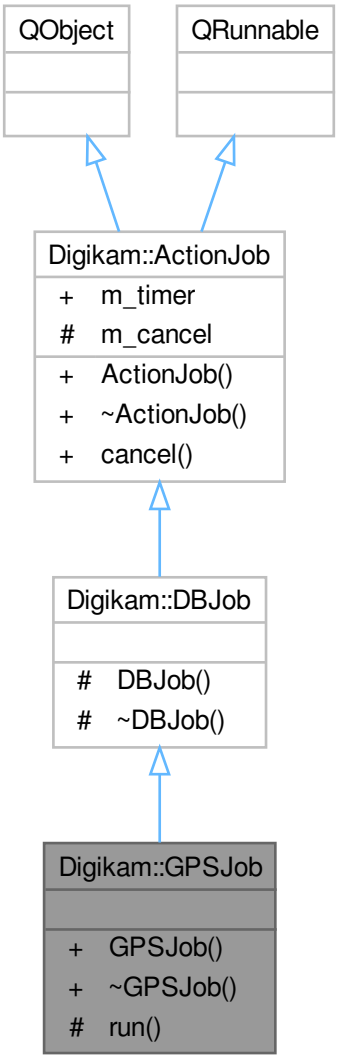
- **GPSItemSortProxyModel** (GPSItemModel *const imageModel, QItemSelectionModel *const source↩ SelectionModel)
- QItemSelectionModel * **mappedSelectionModel** () const

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

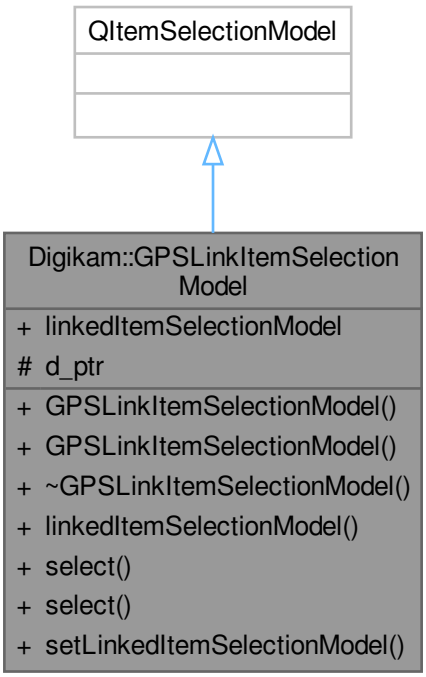
- bool **m_cancel** = false

  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

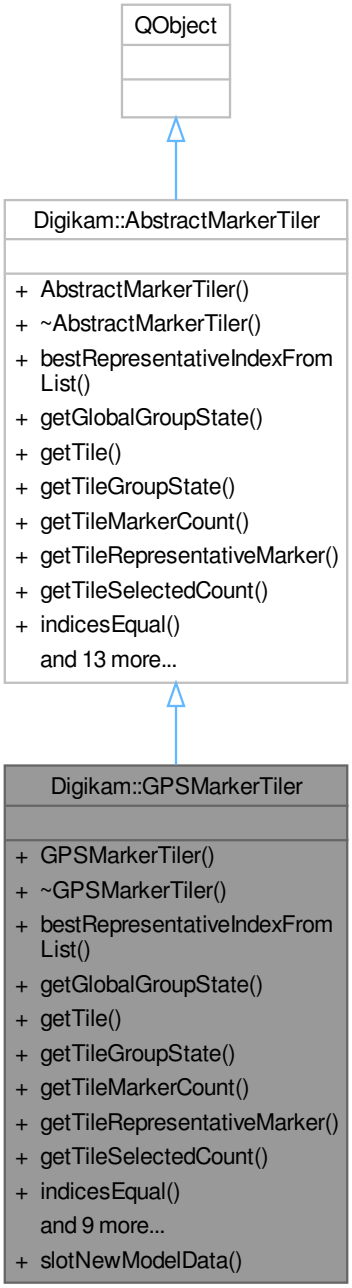
Although Qt documentation, multiple views can share the same QItemSelectionModel, the views then need to have the same source model.

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::GPSMarkerTiler 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 QIntList &a, const QIntList &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 GPSMarkerTiler()

```
Digikam::GPSMarkerTiler::GPSMarkerTiler (
            QObject *const parent,
            ItemFilterModel *const imageFilterModel,
            QItemSelectionModel *const selectionModel )  [explicit]
```

Parameters

parent	The parent object
imageFilterModel	The image filter instance
selectionModel	The selection model instance

9.640.3 Member Function Documentation

9.640.3.1 bestRepresentativeIndexFromList()

```
QVariant Digikam::GPSMarkerTiler::bestRepresentativeIndexFromList (
            const QList< QVariant > & indices,
            const int sortKey )  [override], [virtual]
```

This is needed to display a thumbnail for a marker group.

Parameters

indices	A list containing markers, obtained by getTileRepresentativeMarker.
sortKey	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.

9.640.3.2 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

tileIndex	The index of a tile.
stopIfEmpty	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

tileIndex	Index of the tile from which the best marker should be found.
sortKey	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::GPSMarkerTiler::getTileSelectedCount (
            const TileIndex & tileIndex )  [override], [virtual]
```

Implements Digikam::AbstractMarkerTiler.

9.640.3.8 indicesEqual()

```
bool Digikam::GPSMarkerTiler::indicesEqual (
            const QVariant & a,
            const QVariant & b ) const  [override], [virtual]
```

Implements Digikam::AbstractMarkerTiler.

9.640.3.9 onIndicesClicked()

```
void Digikam::GPSMarkerTiler::onIndicesClicked (
            const ClickInfo & clickInfo )  [override], [virtual]
```

Reimplemented from Digikam::AbstractMarkerTiler.

9.640.3.10 pixmapFromRepresentativeIndex()

```
QPixmap Digikam::GPSMarkerTiler::pixmapFromRepresentativeIndex (
            const QVariant & index,
            const QSize & size )  [override], [virtual]
```

Parameters

index	The marker's index.
size	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::GPSMarkerTiler::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

upperLeft	The North-West point.
lowerRight	The South-East point.
level	The requested tiling level.

Implements Digikam::AbstractMarkerTiler.

9.640.3.12 regenerateTiles()

```
void Digikam::GPSMarkerTiler::regenerateTiles ( )  [override], [virtual]
```

Implements Digikam::AbstractMarkerTiler.

9.640.3.13 setActive()

```
void Digikam::GPSMarkerTiler::setActive (
            const bool state )  [override], [virtual]
```

Parameters

state	New state of the map, true means active.

Implements Digikam::AbstractMarkerTiler.

9.640.3.14 setPositiveFilterIsActive()

```
void Digikam::GPSMarkerTiler::setPositiveFilterIsActive (
            const bool state )
```

9.640.3.15 slotNewModelData

```
void Digikam::GPSMarkerTiler::slotNewModelData (
            const QList< ItemInfo > & infoList )  [slot]
```

9.640.3.16 tileNew()

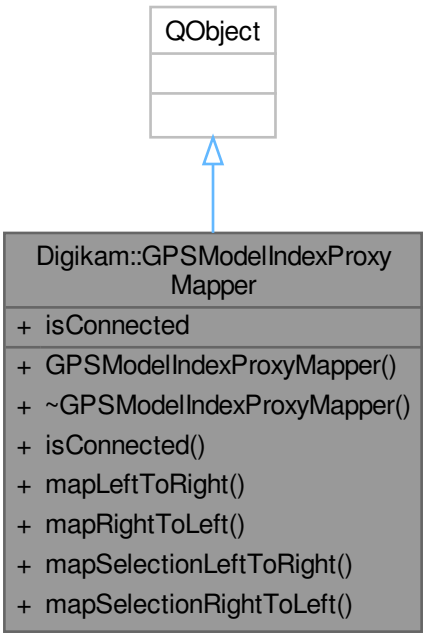
```
AbstractMarkerTiler::Tile * Digikam::GPSMarkerTiler::tileNew ( )  [override], [virtual]
```

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

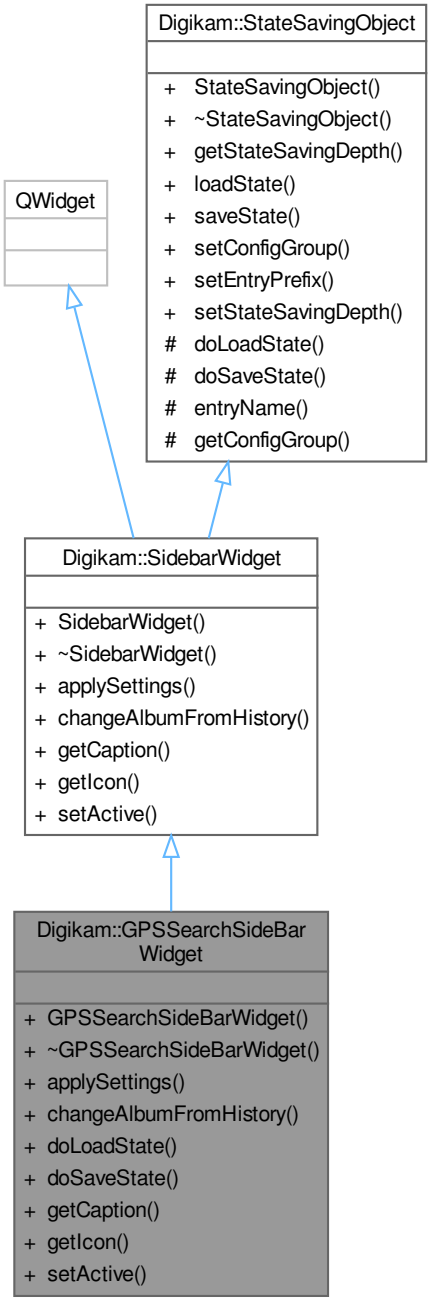
9.641.2.1 isConnected

```
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 **signalMapSoloItems** (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]
```

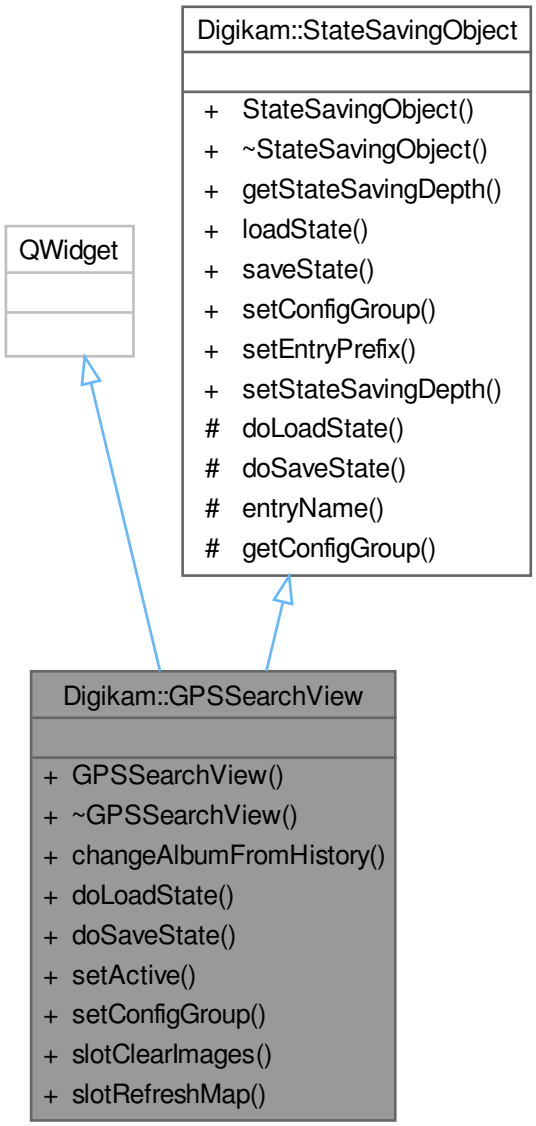
Parameters

active	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 **signalMapSoloItems** (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

parent	The parent object.
searchModel	The model that stores the searches.
searchModificationHelper	The helper instance to perform the searches.
imageFilterModel	The image model used by displaying the selected images on map.
itemSelectionModel	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

state	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.

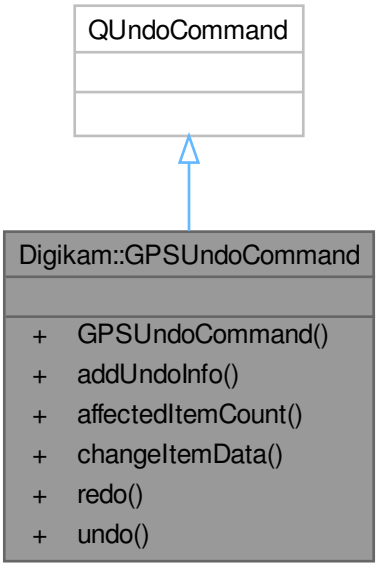
Parameters

group	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)

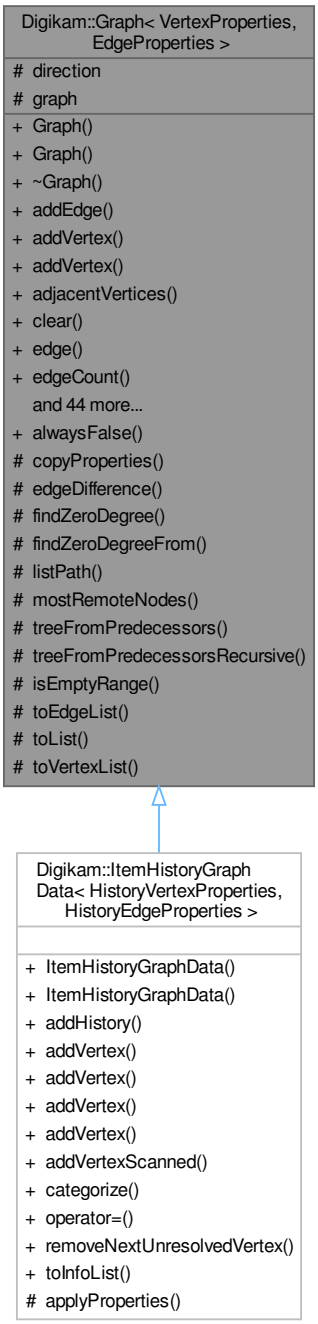
Public Attributes

- 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=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.

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 >::verticesDominatedBy↩
DepthFirstSorted (
            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

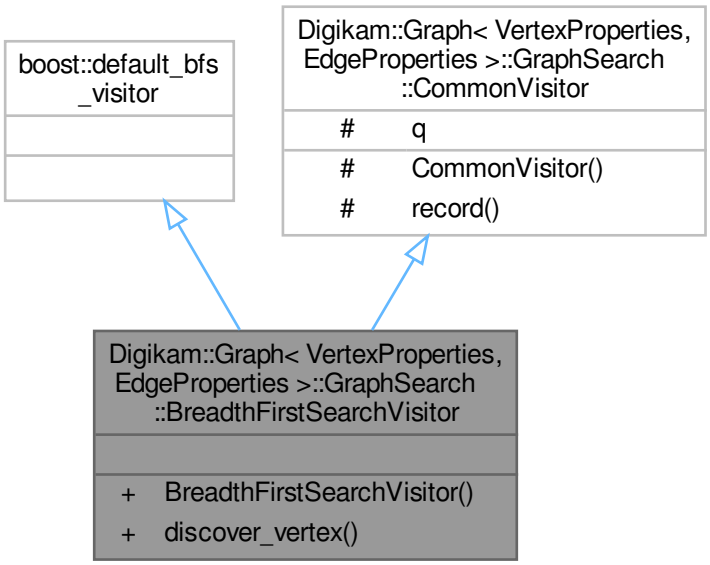
9.649.1.1 depth_first_search_sorted()

```
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::BreadthFirst↩ SearchVisitor:



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

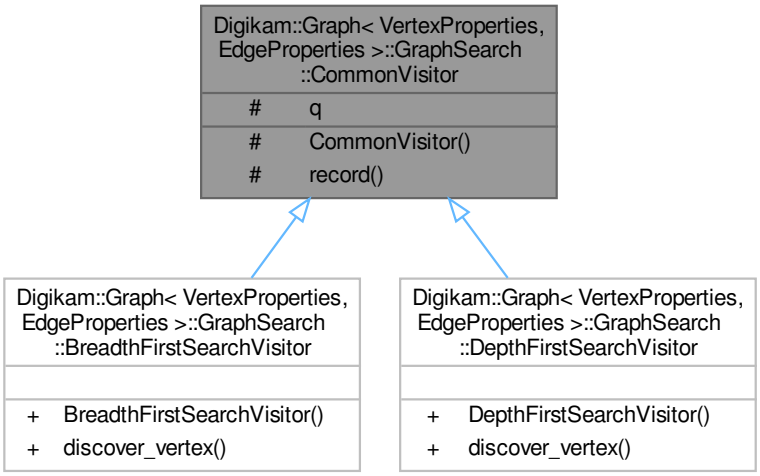
- **CommonVisitor** (GraphSearch ∗const qq)
- void **record** (const Vertex &v) const

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

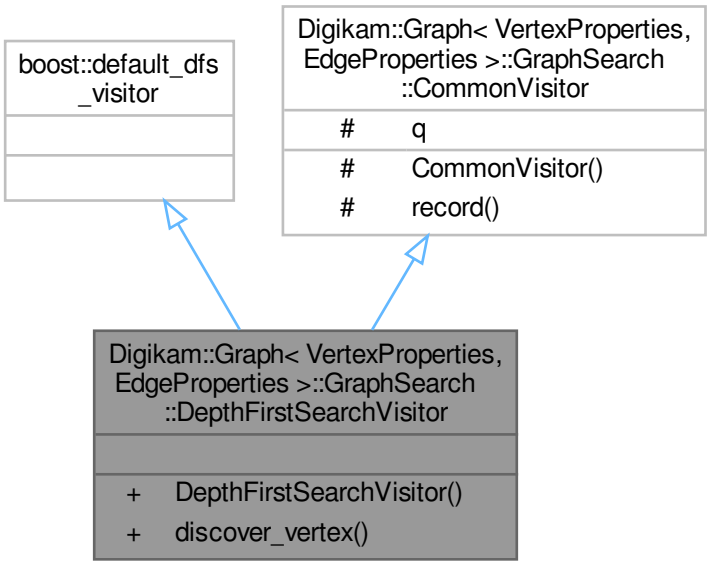
- **CommonVisitor** (GraphSearch ∗const qq)
- void **record** (const Vertex &v) const

Protected Attributes

- GraphSearch ∗const **q** = nullptr

9.652 Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch::DepthFirstSearchVisitor Class Reference

Inheritance diagram for Digikam::Graph< VertexProperties, EdgeProperties >::GraphSearch::DepthFirstSearch↩ Visitor:



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 QMaps, 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 QMaps, 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

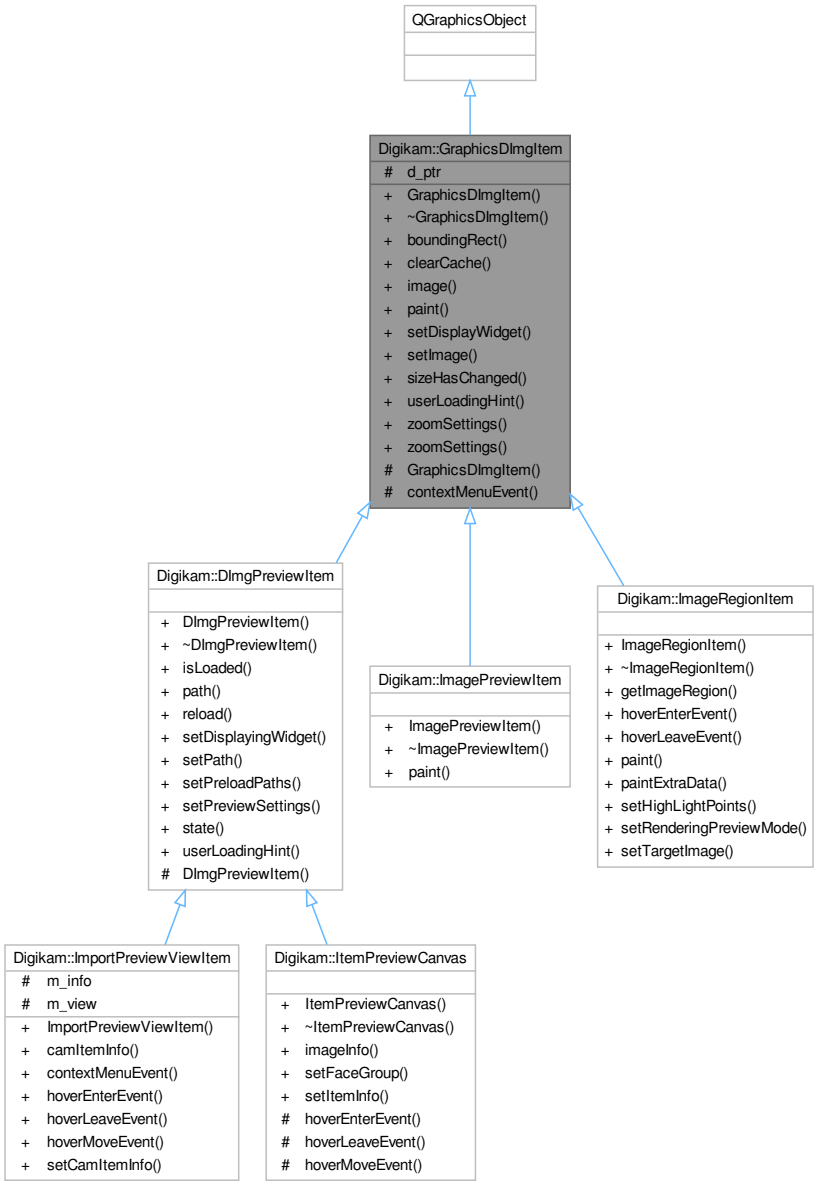
- **Vertex** (const vertex_t &vv)
- bool **isNull** () const
- **operator const vertex_t &** () const
- **operator vertex_t &** ()
- bool **operator!=** (const vertex_t &other) const
- Vertex & **operator=** (const vertex_t &other)
- bool **operator==** (const vertex_t &other) const

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

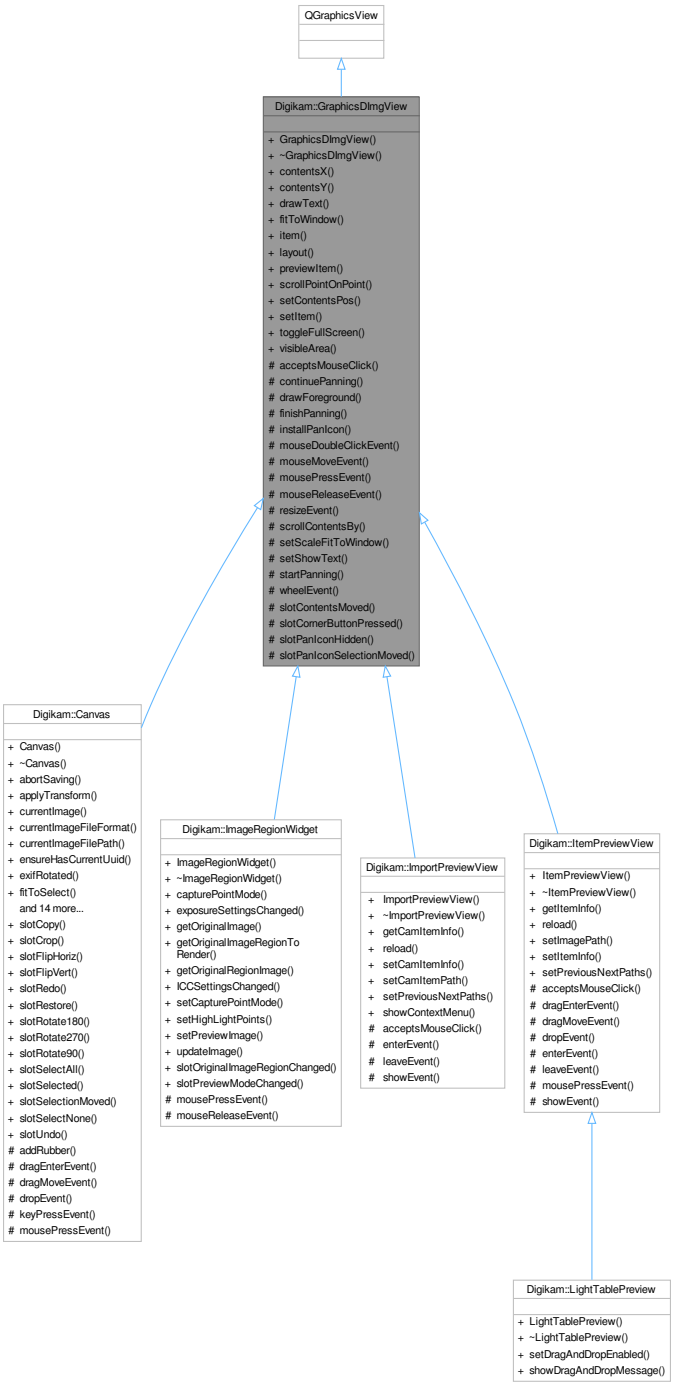
9.656.1.1 setImage()

```
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 **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 Slots

- void **slotContentsMoved** ()
- void **slotCornerButtonPressed** ()
- void **slotPanIconHidden** ()
- virtual void **slotPanIconSelectionMoved** (const QRect &, bool)

Protected Member Functions

- virtual bool **acceptsMouseClick** (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::GraphicsDImgView::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::GraphicsDImgView::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 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 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(), setOriginal↩ Image(), 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.

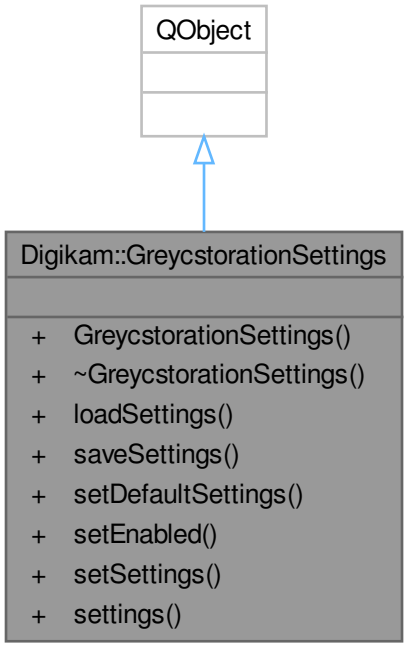
9.659.3.4 readParameters()

```
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

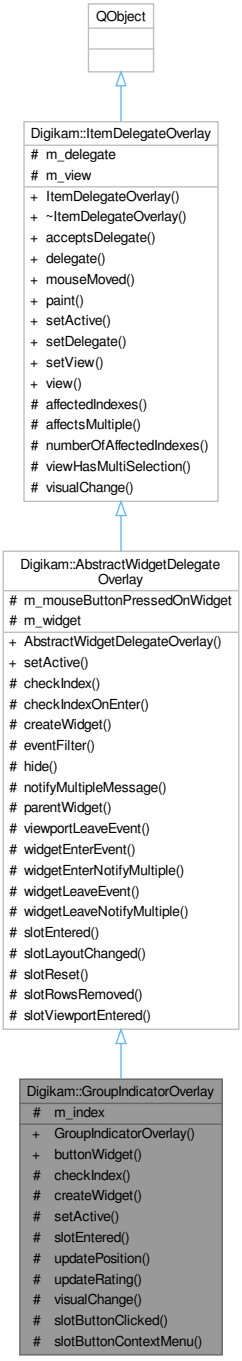
9.661.1.1 GroupedImagesFinder()

```
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 overlayed 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.

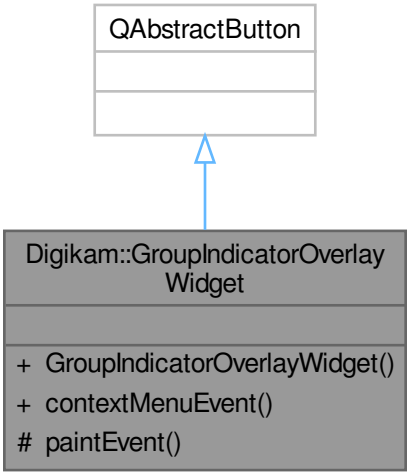
9.662.1.5 visualChange()

```
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

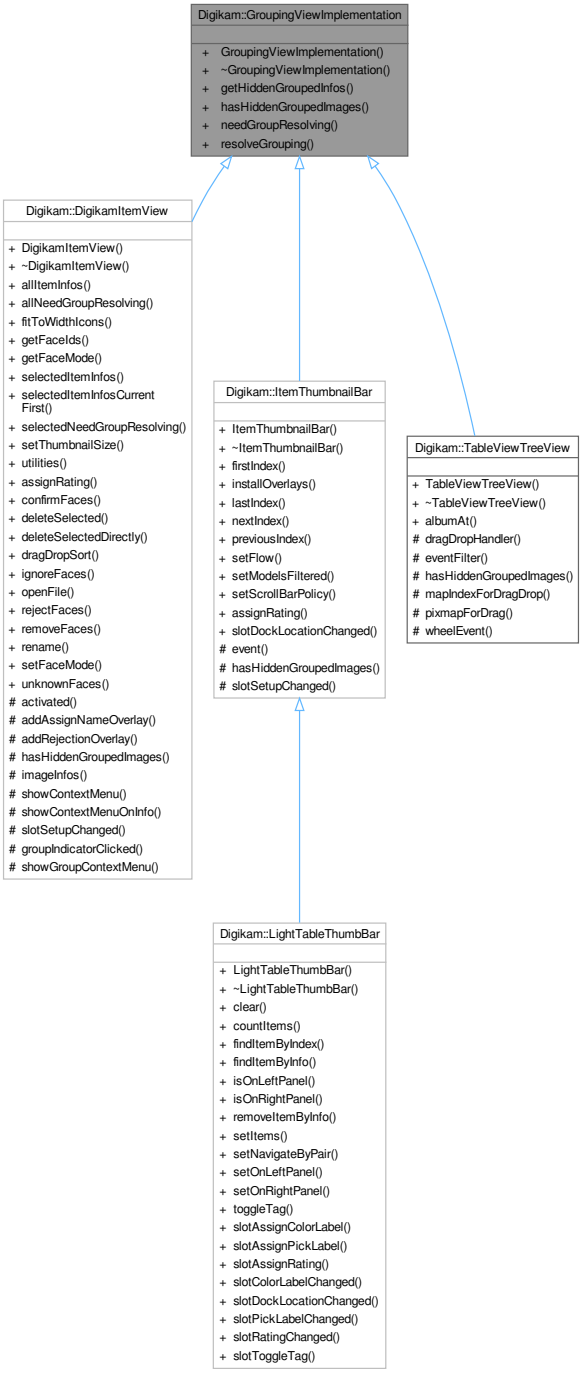
- **GroupIndicatorOverlayWidget** (QWidget ∗const parent=nullptr)
- void **contextMenuEvent** (QContextMenuEvent ∗event) override

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 x NUM_PIXELS^2 x 2log(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 **fillPixelData** (const DImg &image)

  Write pixels of a DImg in three arrays (one per color channel, pixels linearly)
- void **fillPixelData** (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∗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::HaarIface 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

- **HaarIface** (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 refImageSelectionMethod, const QSet< qlonglong > &refs, double requiredPercentage, double maximum↩Percentage, 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 HaarIface::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

filename	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

results	Map of duplicates images found over a list of images.
isAlbumUpdate	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.

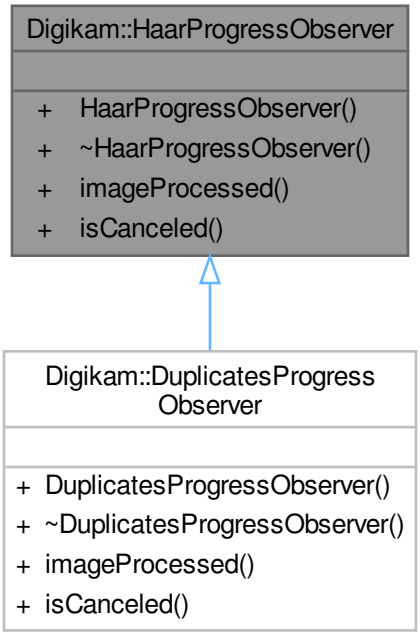
9.673.2.8 signatureAsText()

```
QString Digikam::HaarIface::signatureAsText (
            const QImage & image )
```

Can be used for bestMatchesForSignature.

9.674 Digikam::HaarProgressObserver Class Reference

Inheritance diagram for Digikam::HaarProgressObserver:

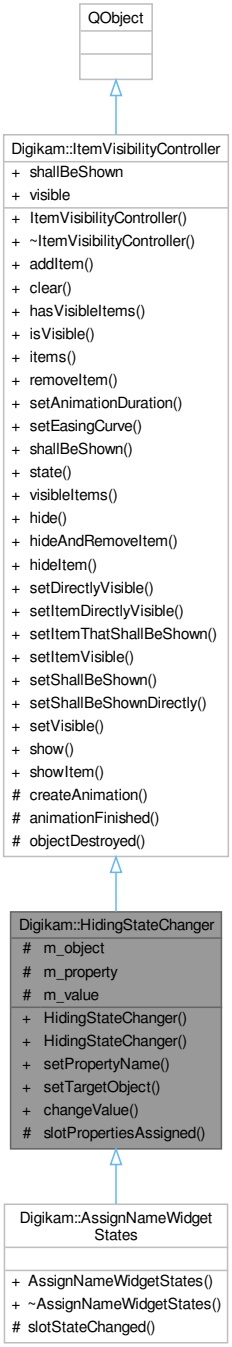


Public Member Functions

- 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 msecs)
- 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

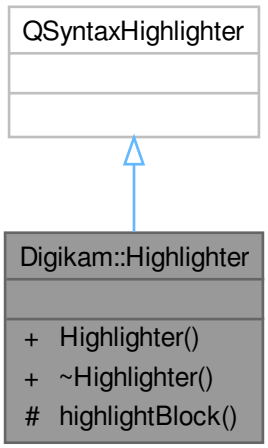
9.675.1.1 HidingStateChanger()

```
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

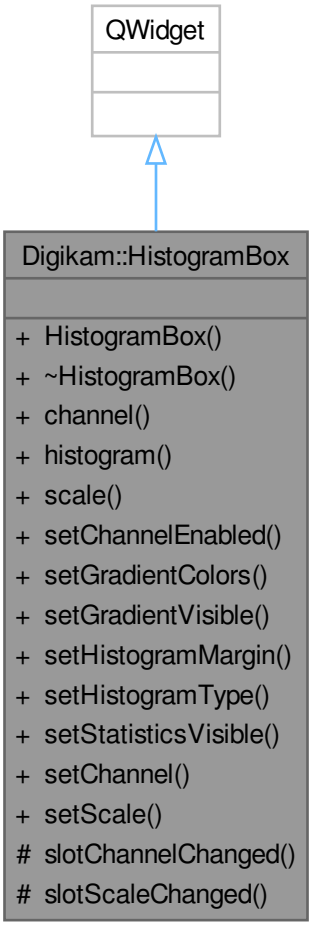
- **Highlighter** (QTextDocument *const document, Parser *const _parser)

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 selectMode=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)

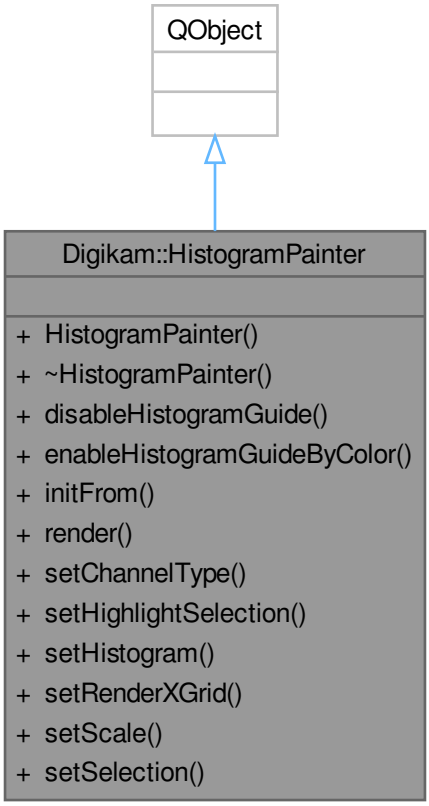
Protected Slots

- 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

parent	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

color	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

widget	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

paintDevice	pixmap to paint the histogram on

9.678.3.4 setChannelType()

```
void Digikam::HistogramPainter::setChannelType (
            ChannelType channelType )
```

Parameters

channelType	channel type to render

9.678.3.5 setHighlightSelection()

```
void Digikam::HistogramPainter::setHighlightSelection (
            bool highlightSelection )
```

The selection must be defined with setHighlightSelection.

Parameters

highlightSelection	if true, a selection will be highlighted

9.678.3.6 setHistogram()

```
void Digikam::HistogramPainter::setHistogram (
            ImageHistogram *const histogram )
```

Parameters

histogram	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

renderXGrid	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

scale	scal to paint histogram with

9.678.3.9 setSelection()

```
void Digikam::HistogramPainter::setSelection (
            double selectionMin,
            double selectionMax )
```

Parameters

selectionMin	0 <= value <= 1, percent of the histogram width to start highlighting as percent. Ensure that this value is smaller then selectionMax.
selectionMax	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 v2: v1 -> actions[0] -> ... -> actions[n] = v2.

9.681 Digikam::HistoryImageId 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==** (qlonglong 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** (DImg *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 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** ()

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

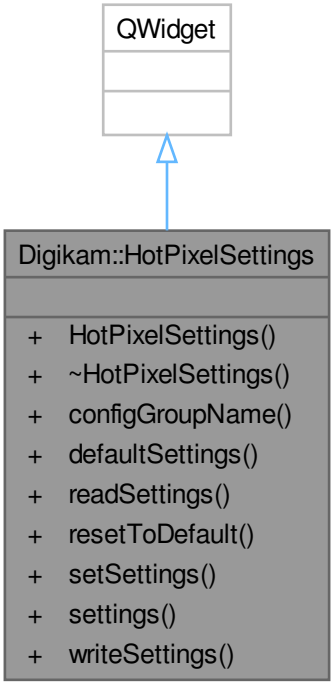
9.685.1.1 operator==()

```
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

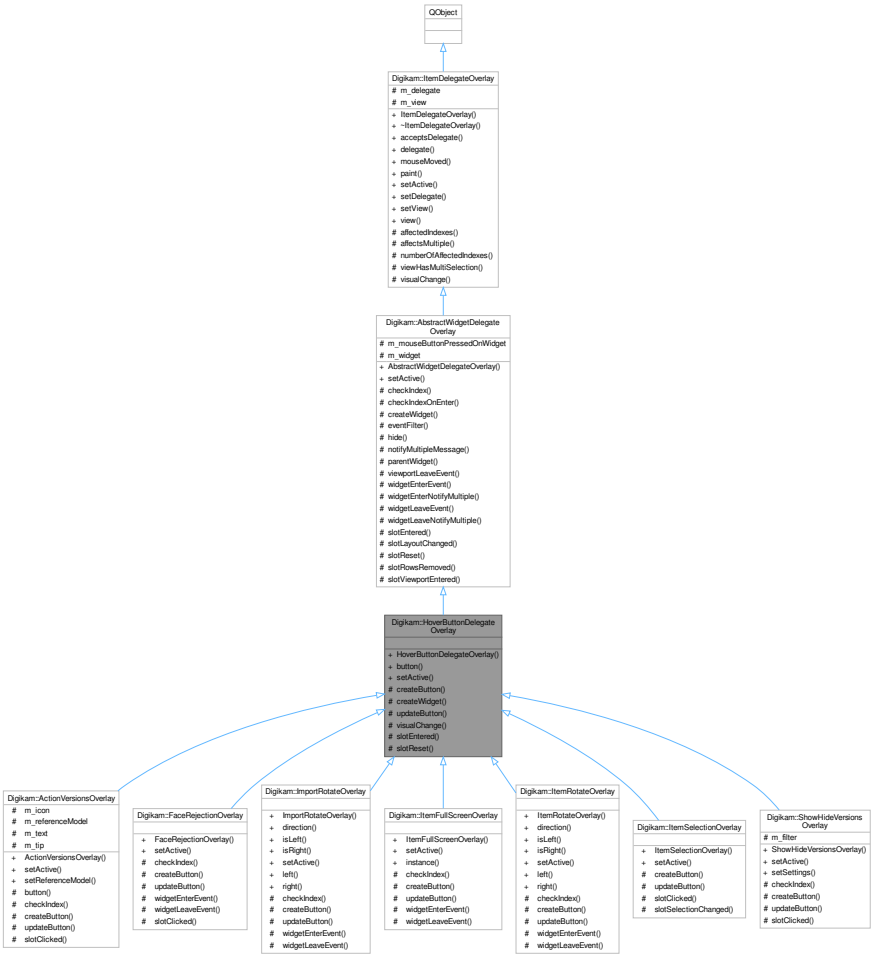
- **HotPixelsWeights** (const HotPixelsWeights &w)
- void **calculateHotPixelsWeights** ()
- unsigned int **height** () const
- HotPixelsWeights & **operator=** (const HotPixelsWeights &w)
- bool **operator==** (const HotPixelsWeights &ws) const
- double ∗∗ **operator[ ]** (int n) const
- unsigned int **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 overlayed 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 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** ()

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.

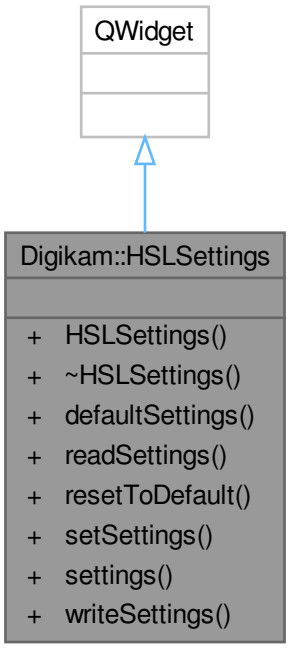
9.690.1.3 readParameters()

```
void Digikam::HSLFilter::readParameters (
            const FilterAction & action )  [override], [virtual]
```

Implements Digikam::DImgThreadedFilter.

9.691 Digikam::HSLSettings Class Reference

Inheritance diagram for Digikam::HSLSettings:



Signals

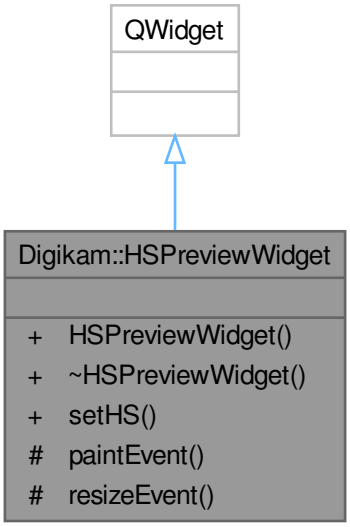
- void **signalSettingsChanged** ()

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

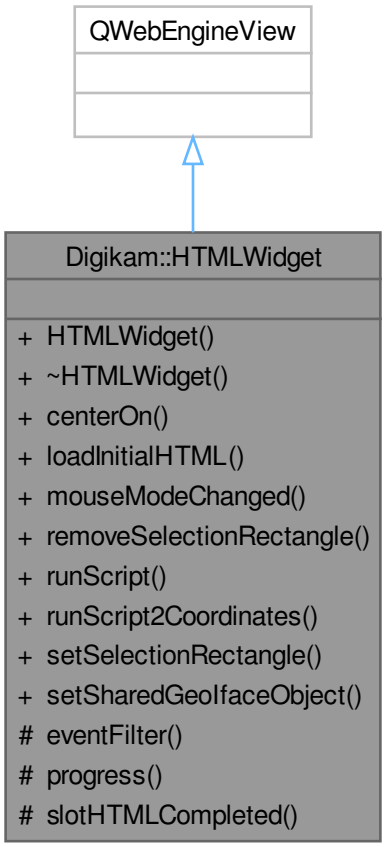
- **HSPreviewWidget** (QWidget ∗const parent=nullptr)
- void **setHS** (double hue, double sat)

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 **setSharedGeoIfaceObject** (GeoIfaceSharedData ∗const sharedData)

Protected Slots

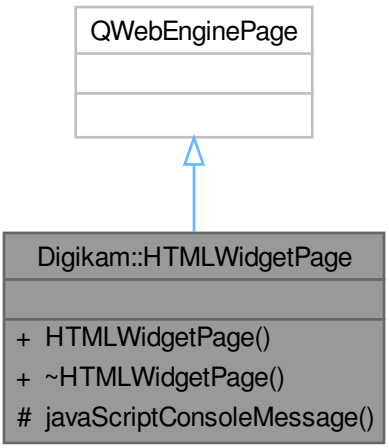
- void **progress** (int progress)
- void **slotHTMLCompleted** (bool ok)

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

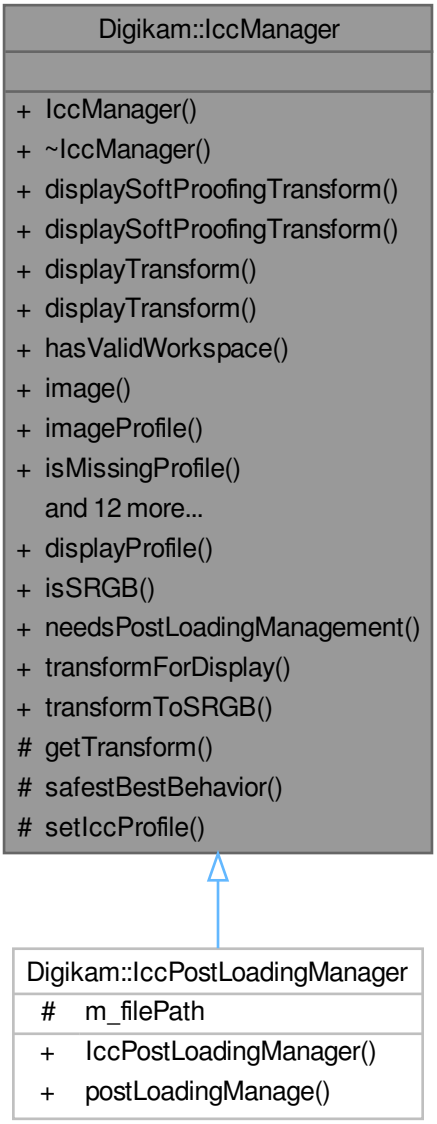
- **HTMLWidgetPage** (HTMLWidget ∗const parent=nullptr)

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 &display↩ Profile)
- IccTransform **displaySoftProofingTransform** (const IccProfile &deviceProfile, QWidget ∗const displaying↩ 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.

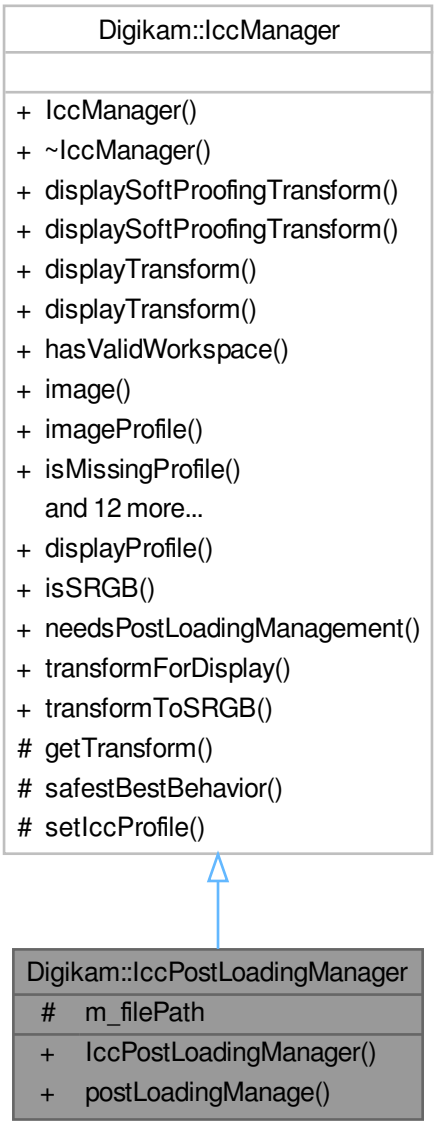
9.695.2.3 transformForDisplay()

```
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::IccManager

- IccManager (const DImg &image, const ICCSettingsContainer &settings=IccSettings::instance() ->settings())

  Constructs an IccManager object.
- IccTransform **displaySoftProofingTransform** (const IccProfile &deviceProfile, const IccProfile &display↩ Profile)
- IccTransform **displaySoftProofingTransform** (const IccProfile &deviceProfile, QWidget ∗const displaying↩ 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.

Protected Attributes

- QString **m_filePath**

Additional Inherited Members

Static Public Member Functions inherited from Digikam::IccManager

- 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 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

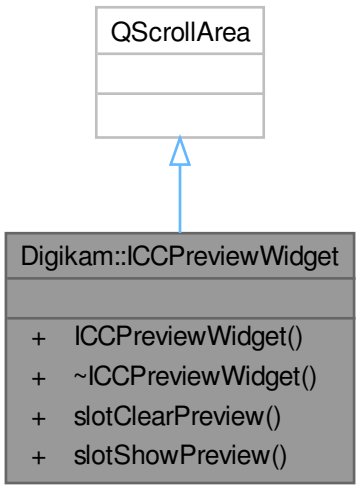
9.696.2.1 postLoadingManage()

```
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.

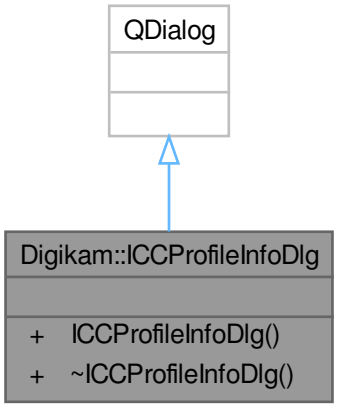
9.698.2.9 type()

```
IccProfile::ProfileType Digikam::IccProfile::type ( )
```

< 'nkbf', proprietary in Nikon profiles

9.699 Digikam::ICCProfileInfoDlg Class Reference

Inheritance diagram for Digikam::ICCProfileInfoDlg:

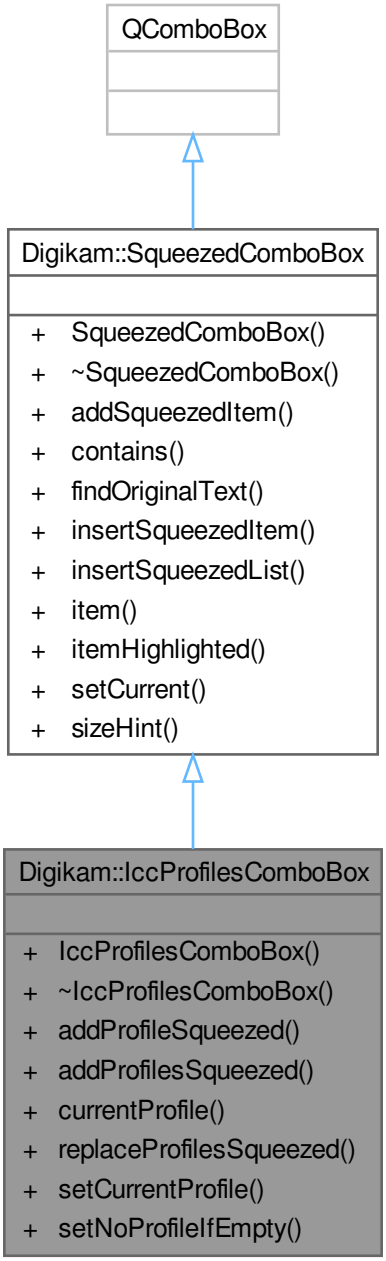


Public Member Functions

- **ICCProfileInfoDlg** (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.

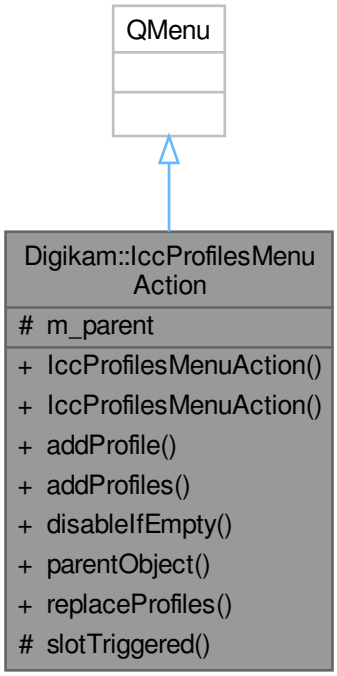
9.700.2.2 setCurrentProfile()

```
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

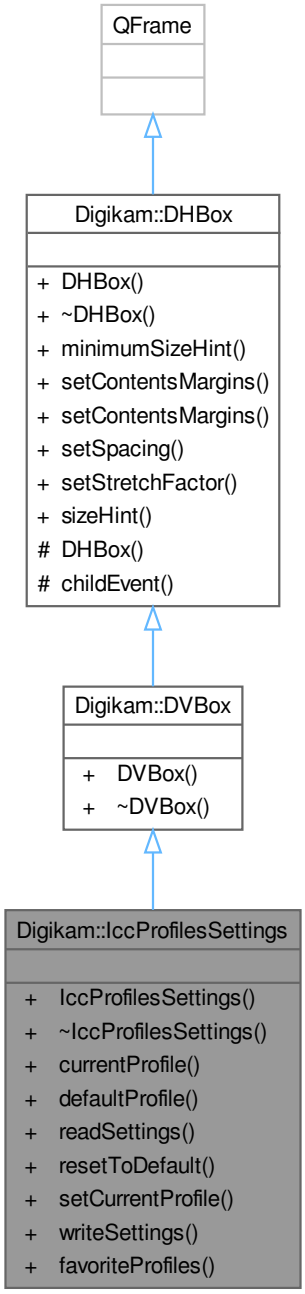
9.701.1.1 addProfile()

```
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::ICCProfileWidget Class Reference

Inheritance diagram for Digikam::ICCProfileWidget:

Public Member Functions

- **ICCProfileWidget** (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::ICCProfileWidget::getMetadataTitle ( ) const  [override], [virtual]
```

Implements Digikam::MetadataWidget.

9.703.1.2 getTagDescription()

```
QString Digikam::ICCProfileWidget::getTagDescription (
            const QString & key )  [override], [virtual]
```

Reimplemented from Digikam::MetadataWidget.

9.703.1.3 getTagTitle()

```
QString Digikam::ICCProfileWidget::getTagTitle (
            const QString & key )  [override], [virtual]
```

Reimplemented from Digikam::MetadataWidget.

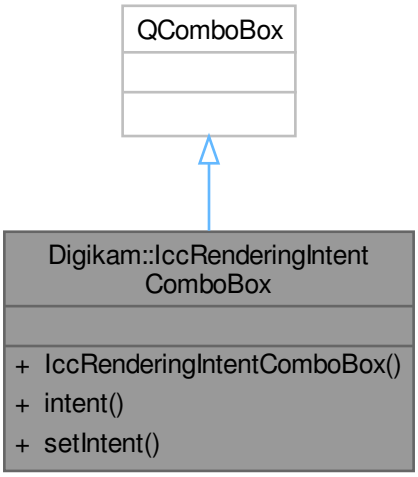
9.703.1.4 loadFromURL()

```
bool Digikam::ICCProfileWidget::loadFromURL (
            const QUrl & url )  [override], [virtual]
```

Implements Digikam::MetadataWidget.

9.704 Digikam::IccRenderingIntentComboBox Class Reference

Inheritance diagram for Digikam::IccRenderingIntentComboBox:

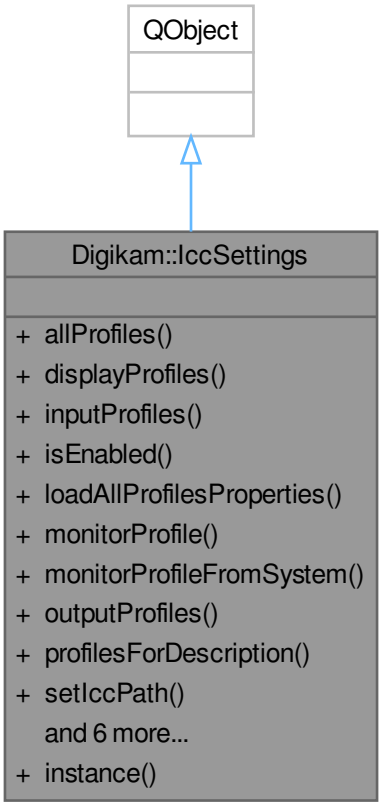


Public Member Functions

- **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 , **Do↩ NotInterpret** = 1 << 6 ,
  KeepProfile = 1 << 10 , **ConvertToWorkspace** = 1 << 11 , LeaveFileUntagged = 1 << 18 , **AskUser** = 1 << 20 ,
  SafestBestAction = 1 << 21 , PreserveEmbeddedProfile = UseEmbeddedProfile | KeepProfile , **EmbeddedToWorkspace** = UseEmbeddedProfile | ConvertToWorkspace , **SRGBToWorkspace** = Use↩ SRGB | ConvertToWorkspace ,
  AutoToWorkspace = AutomaticColors | ConvertToWorkspace , **InputToWorkspace** = UseDefaultInput↩ Profile | 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

InvalidBehavior	Note: Values are stored in config - keep them constant.
UseEmbeddedProfile	Interpretation of the image data.
KeepProfile	Transformation / target profile.
LeaveFileUntagged	Special flags and values.
PreserveEmbeddedProfile	ready combinations for convenience

9.707 Digikam::IccTransform Class Reference

Public Types

- enum **RenderingIntent** { **Perceptual** = 0 , **RelativeColorimetric** = 1 , **Saturation** = 2 , **Absolute↩ Colorimetric** = 3 }

Public Member Functions

- **IccTransform** (const IccTransform &other)
- bool apply (DImg &image, DImgLoaderObserver *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.
- IccProfile **effectiveInputProfile** () const

  Returns the embedded profile; if none is set, the input profile; if none is set, sRGB.
- IccProfile **embeddedProfile** () const

  Returns the contained profiles.
- IccProfile **inputProfile** () const
- RenderingIntent **intent** () const
- bool **isCheckingGamut** () const
- bool **isUsingBlackPointCompensation** () const
- IccTransform & **operator=** (const IccTransform &other)
- IccProfile **outputProfile** () const
- RenderingIntent **proofIntent** () const
- IccProfile **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 DImg &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.

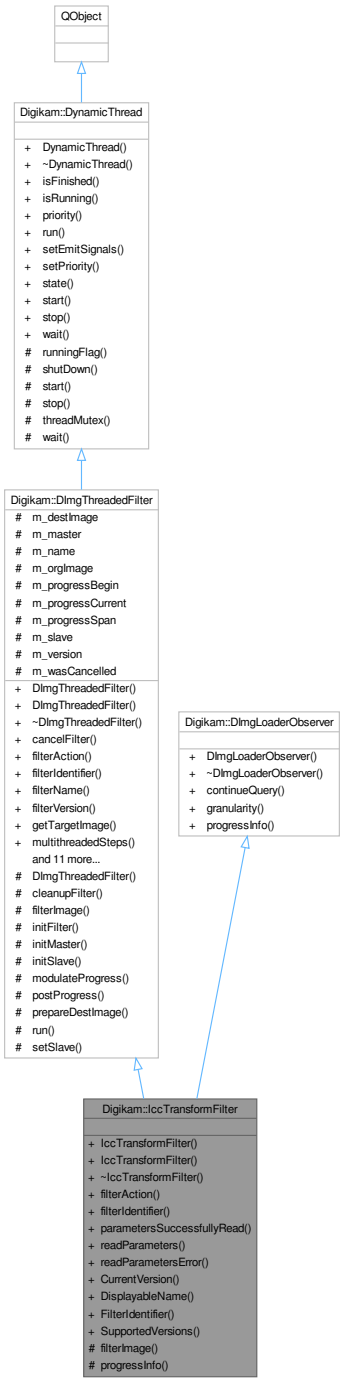
9.707.1.6 willHaveEffect()

```
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 **DisplayableName** ()
- 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::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

9.708.1 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.
- QMultiMap< 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 QMultiMap< 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 QMultiMap< QString, QString > &attributes)

  Adds a new identity with the specified attributes.
- Identity **addIdentityDebug** (const QMultiMap< 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 identites 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 QMultiMap< 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 QMultiMap< QString, QString > & attributes )
```

Please note that a UUID is automatically generated.

9.710.1.2 findIdentity() [1/2]

```
Identity Digikam::IdentityProvider::findIdentity (
            const QMultiMap< 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

otherCurves	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()

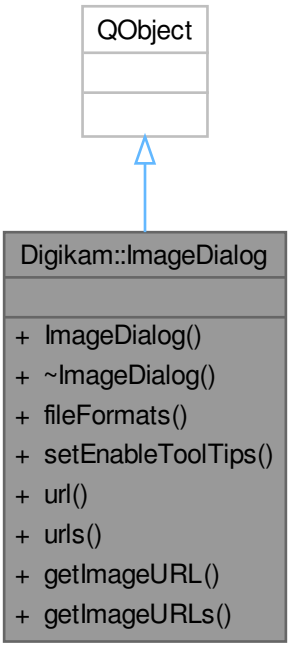
```
void Digikam::ImageCurves::setContainer (
            const CurvesContainer & container )
```

Note

bits depth must match

9.714 Digikam::ImageDialog Class Reference

Inheritance diagram for Digikam::ImageDialog:



Public Member Functions

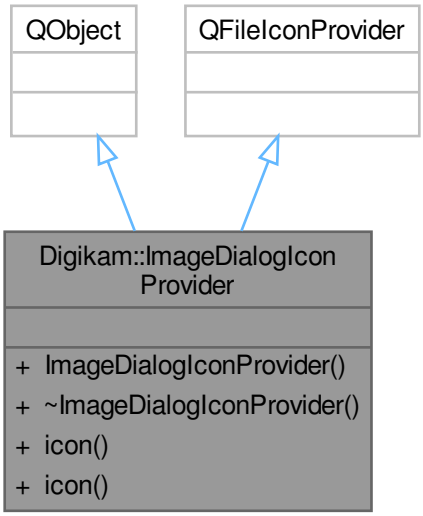
- **ImageDialog** (QWidget *const parent, const QUrl &url, bool singleSelect=false, const QString &caption=QString())
- QStringList **fileFormats** () const
- void **setEnableToolTips** (bool val)
- QUrl **url** () const
- QList< QUrl > **urls** () const

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

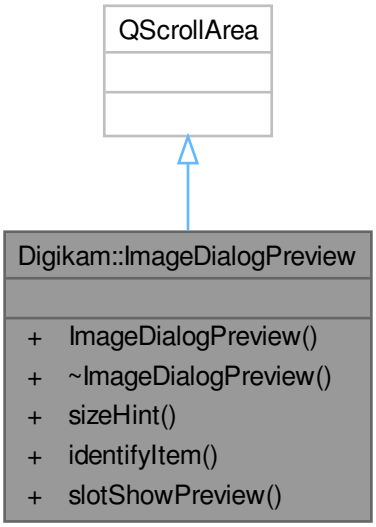
- void **signalThumbnailRefresh** ()

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 **toolTipIsEmpty** () 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, ImageIface::PreviewType type=ImageIface::FullImage)
- void **exposureSettingsChanged** ()
- QImage **getMask** () const
- DColor **getSpotColor** (int getColorFrom) const
- QPoint **getSpotPosition** () const
- void **ICCSettingsChanged** ()
- ImageIface * **imageIface** () 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::ImageIface Class Reference

Public Types

- enum PreviewType { FullImage , ImageSelection }

Public Member Functions

- ImageIface (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)

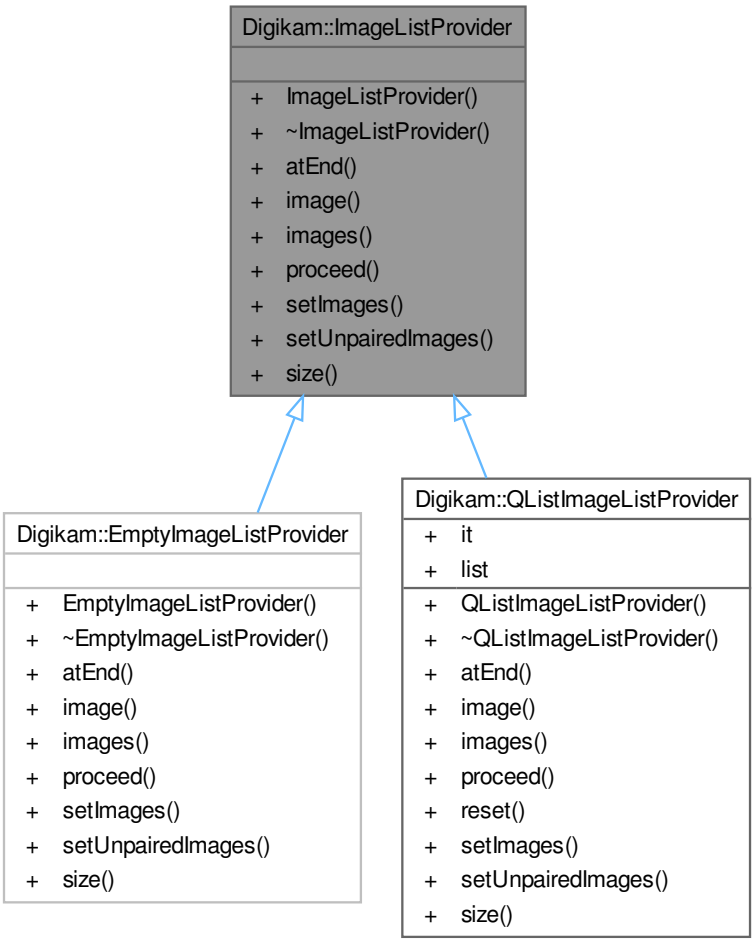
  Methods to save/load the levels values to/from a Gimp levels text file.
- void **setLevelGammaValue** (int channel, double val)

  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

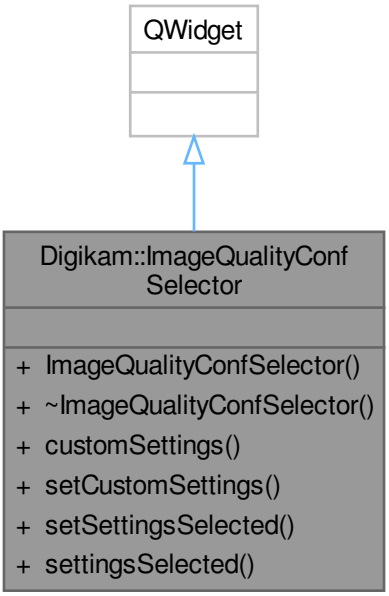
9.727 Digikam::ImageQualityCalculator::ResultDetection Struct Reference

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 avaialble in setup dialog.
CustomSettings	Sttings cusomized 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**

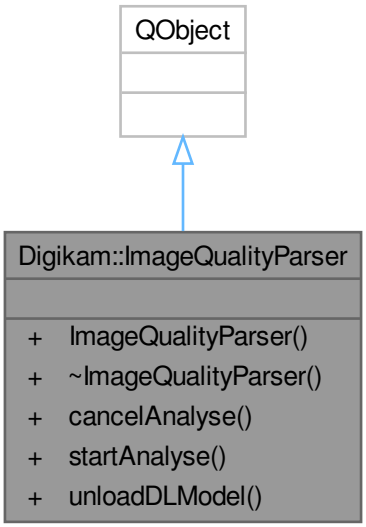
  Item noise level.
- int **pendingThreshold**

  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** ()

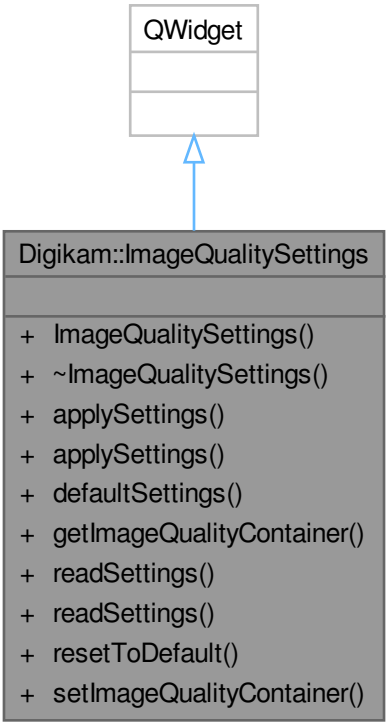
  Perform quality estimation and fill Pick Label value accordingly.

Static Public Member Functions

- static void **unloadDLModel** ()

9.731 Digikam::ImageQualitySettings Class Reference

Inheritance diagram for Digikam::ImageQualitySettings:



Signals

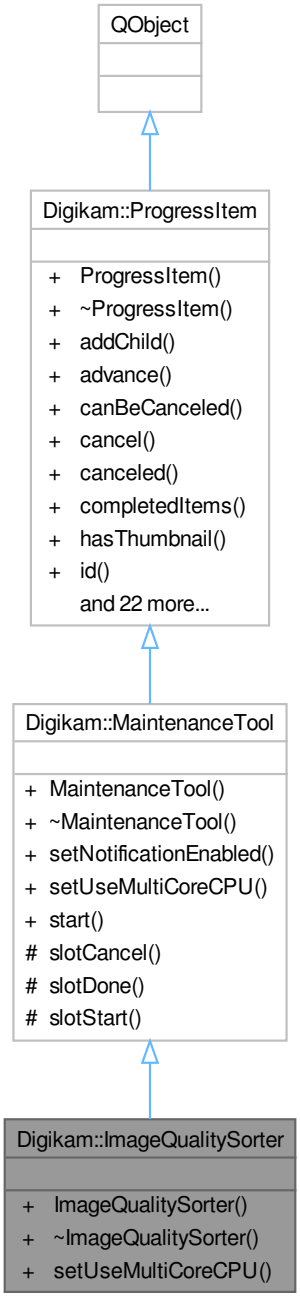
- void **signalSettingsChanged** ()

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 elpased 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

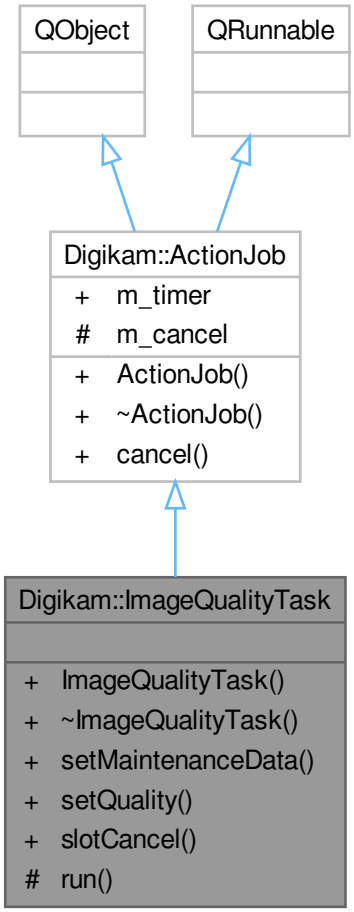
9.732.3.1 setUseMultiCoreCPU()

```
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.

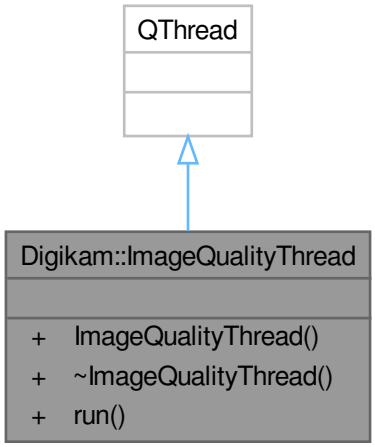
Protected Attributes inherited from Digikam::ActionJob

- 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:

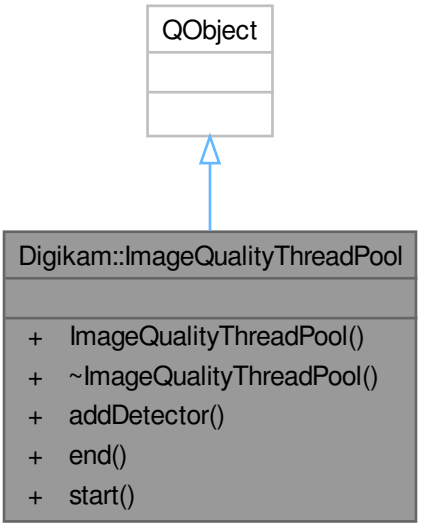


Public Member Functions

- **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:

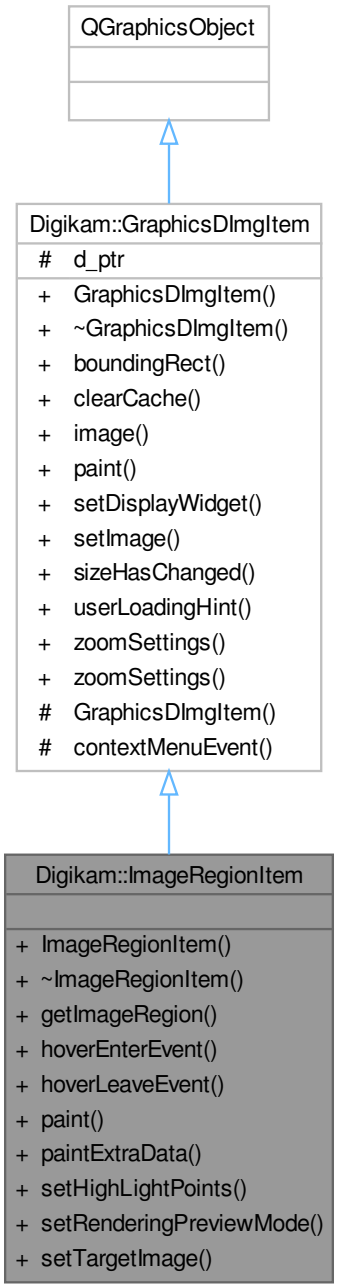


Public Member Functions

- **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

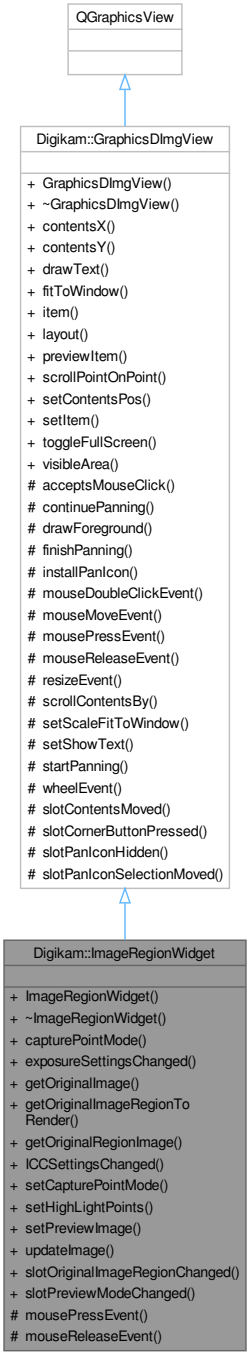
- **GraphicsDImgItem** (GraphicsDImgItemPrivate &dd, QGraphicsItem ∗const parent)
- void **contextMenuEvent** (QGraphicsSceneContextMenuEvent ∗e) override

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 **acceptsMouseClick** (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.

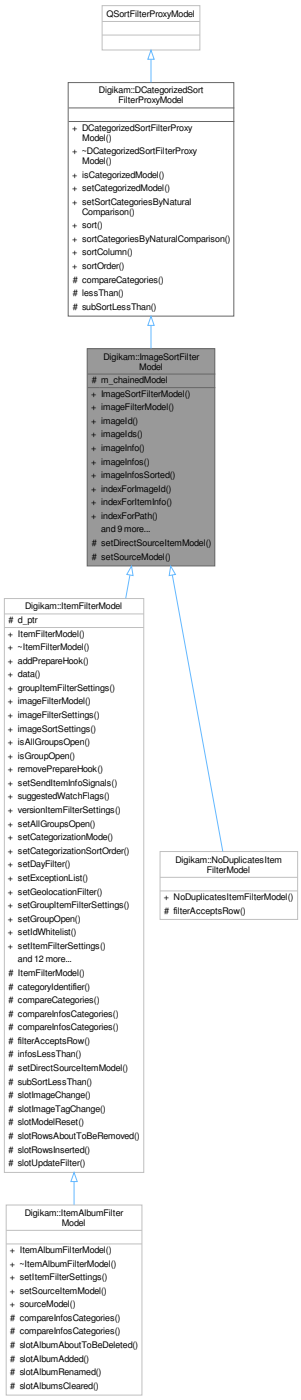
9.738 Digikam::ImageRelation Class Reference

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)

  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 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.

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 ∗ infoIface (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 **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.
- 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** = Overwrite↩
  WithoutAsking ,
  AlwaysNewVersion = AlwaysSaveAs }

Protected Slots inherited from Digikam::EditorWindow

- virtual bool **saveOrSaveAs** ()
- void **slotAboutToShowRedoMenu** ()
- void **slotAboutToShowUndoMenu** ()
- virtual void **slotAddedDropedItems** (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.

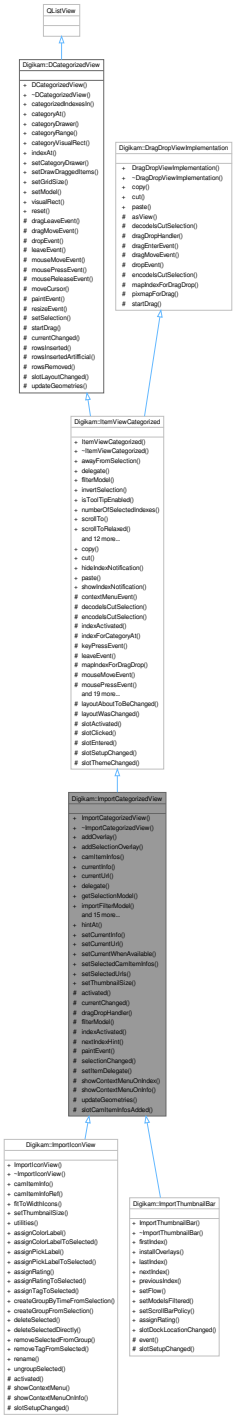
9.744.1.6 zoomedSize()

```
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** (qlonglong 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 **rowsInsertedArtifficial** (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 **decodeIsCutSelection** (const QMimeData ∗mimeData)
- void **encodeIsCutSelection** (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 **decodeIsCutSelection** (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 **encodeIsCutSelection** (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.

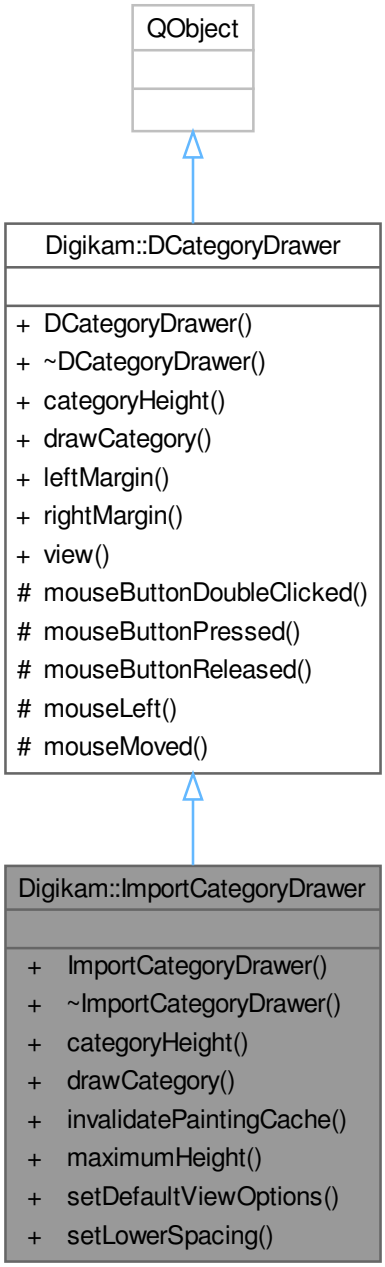
9.745.1.12 showContextMenuOnIndex()

```
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

index	with the given
sortRole	sorting role
option	painter style options
painter	painter instance

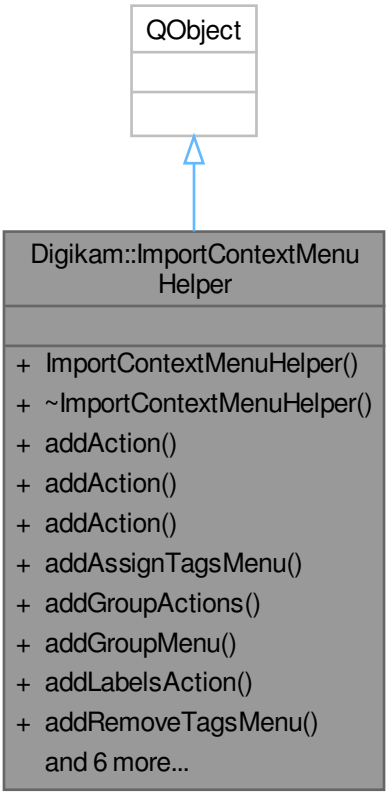
Note

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,
            KActionCollection *const actionCollection = nullptr )  [explicit]
```

Parameters

parent	the menu the helper class is linked to
actionCollection	the actionCollection that should be used. If not set, the standard action from DigikamApp 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

name	the name of the action in the actionCollection
addDisabled	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

action	the action to add
addDisabled	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

action	the action to add
recv	the receiver of the triggered action
slot	the slot to connect the triggered action to
addDisabled	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

ids	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

ids	the selected items

See also

exec()

signalRemoveTag()

9.747.2.8 addRotateMenu()

```
void Digikam::ImportContextMenuHelper::addRotateMenu (
            itemIds & ids )
```

Parameters

ids	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

selectedItems	the list of selected items

9.747.2.10 addSubMenu()

```
void Digikam::ImportContextMenuHelper::addSubMenu (
            QMenu * subMenu )
```

Parameters

subMenu	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

pos	position of the triggered action in the registered menu
at	the action that should be at the position pos

Returns

the triggered action

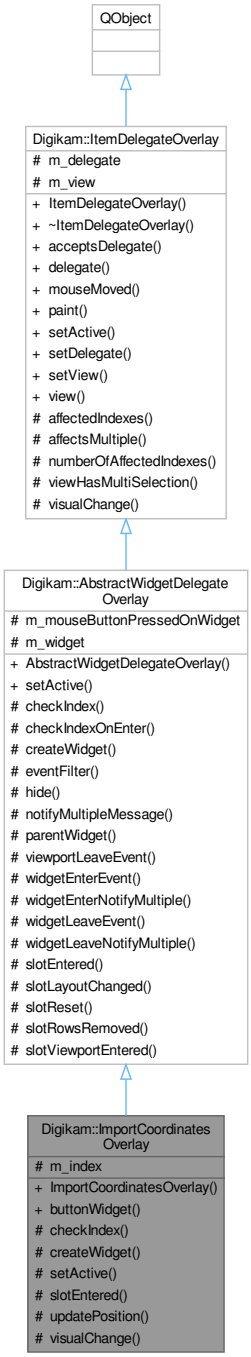
9.747.2.12 setImportFilterModel()

```
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 overlayed 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.

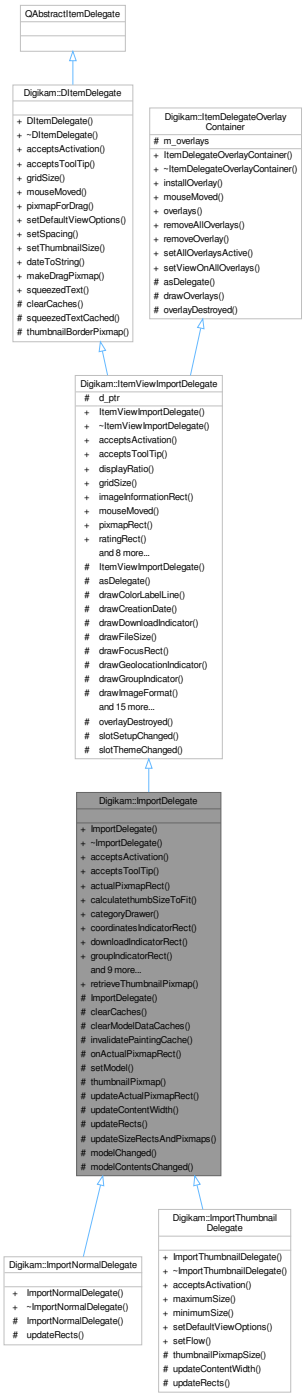
9.748.1.5 visualChange()

```
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 &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 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 is↩ Selected) 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.

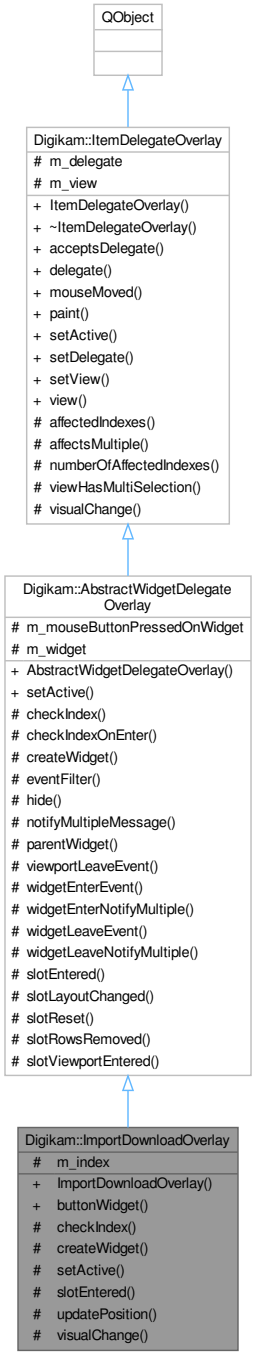
9.749.1.12 updateSizeRectsAndPixmaps()

```
void Digikam::ImportDelegate::updateSizeRectsAndPixmaps ( )  [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 overlayed 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.

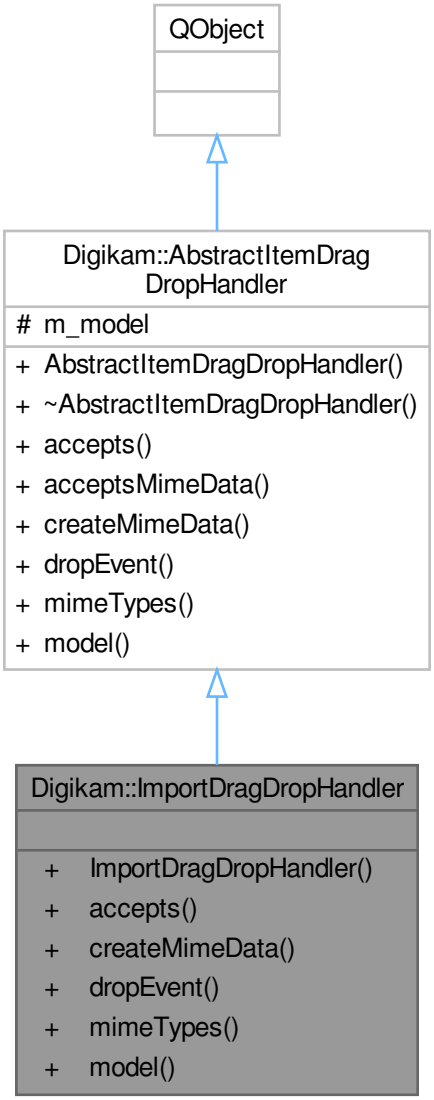
9.750.1.5 visualChange()

```
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 mimeTypes()

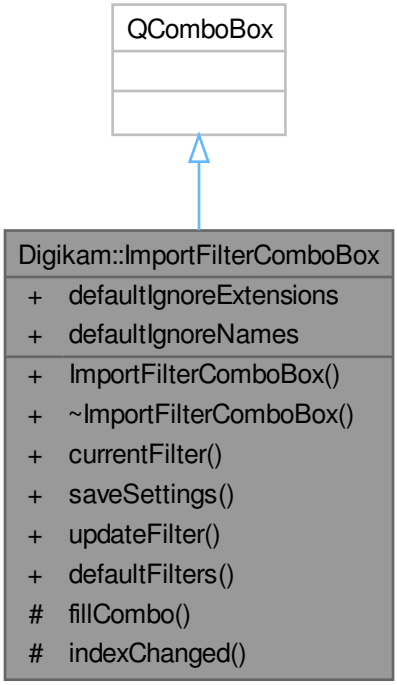
```
QStringList Digikam::ImportDragDropHandler::mimeTypes ( ) const  [override], [virtual]
```

Called by the default implementation of model's mimeTypes().

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

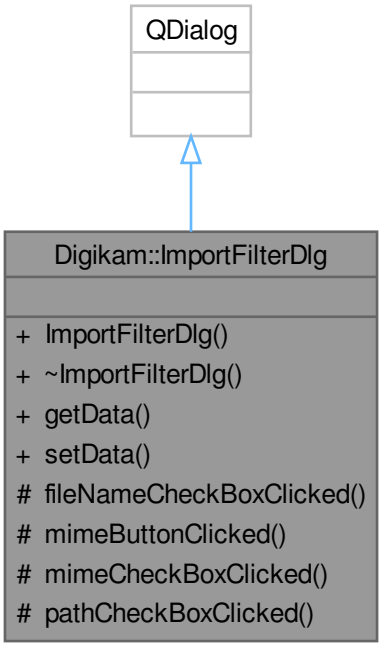
- void **indexChanged** (int index)

Protected Member Functions

- void **fillCombo** ()

9.753 Digikam::ImportFilterDlg Class Reference

Inheritance diagram for Digikam::ImportFilterDlg:



Public Member Functions

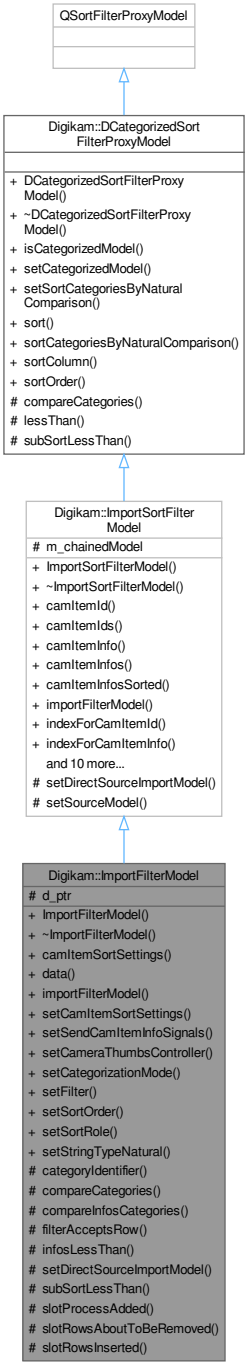
- **ImportFilterDlg** (QWidget ∗const parent=nullptr)
- void **getData** (Filter ∗const filter)
- void **setData** (const Filter &filter)

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)
- 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.
- 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 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

CategorizationModeRole	Returns the current categorization mode.
SortOrderRole	Returns the current sort order.
CategoryFormatRole	Returns the format of the index which is used for category.
CategoryDateRole	Returns the date of the index which is used for category.
ImportFilterModelPointerRole	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 setSend↩ ItemInfoSignals().

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 Category↩ SortRole. In order to correctly sort categories, the data() method of the model should return a qlonglong (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.

9.755 Digikam::ImportIconView Class Reference

Inheritance diagram for Digikam::ImportIconView:

Public Slots

- void **assignColorLabel** (const QModelIndex &index, int colorId)
- void **assignColorLabelToSelected** (int colorId)

- 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** (qlonglong 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 **decodeIsCutSelection** (const QMimeData ∗mimeData)
- void **encodeIsCutSelection** (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 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 **decodeIsCutSelection** (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 **encodeIsCutSelection** (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.

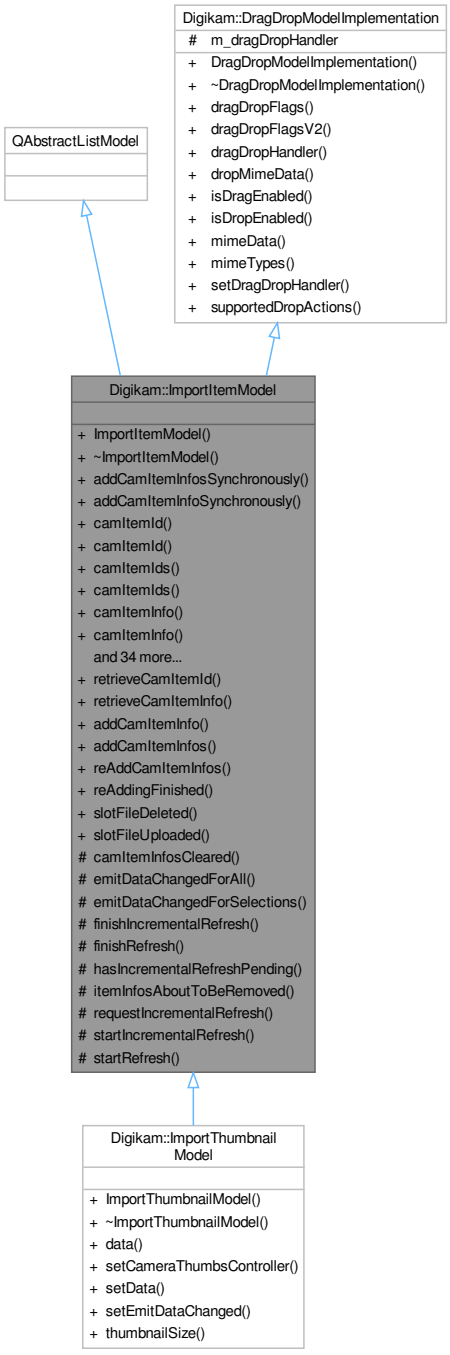
9.755.1.5 slotSetupChanged()

```
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 prepocessor 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 QModel↩ Index, 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** (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 > **indexesForCamItemId** (qlonglong id) const
- QList< QModelIndex > **indexesForCamItemInfo** (const CamItemInfo &info) const
- QList< QModelIndex > **indexesForUrl** (const QUrl &fileUrl) const
- QModelIndex **indexForCamItemId** (qlonglong 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** (qlonglong 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 **removeIndexs** (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 QSortFilter↩ ProxyModel 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 finish↩ Refresh() 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

ImportItemModelPointerRole	An ImportItemModel* 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.

9.756.2 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 camItemInfoId 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().

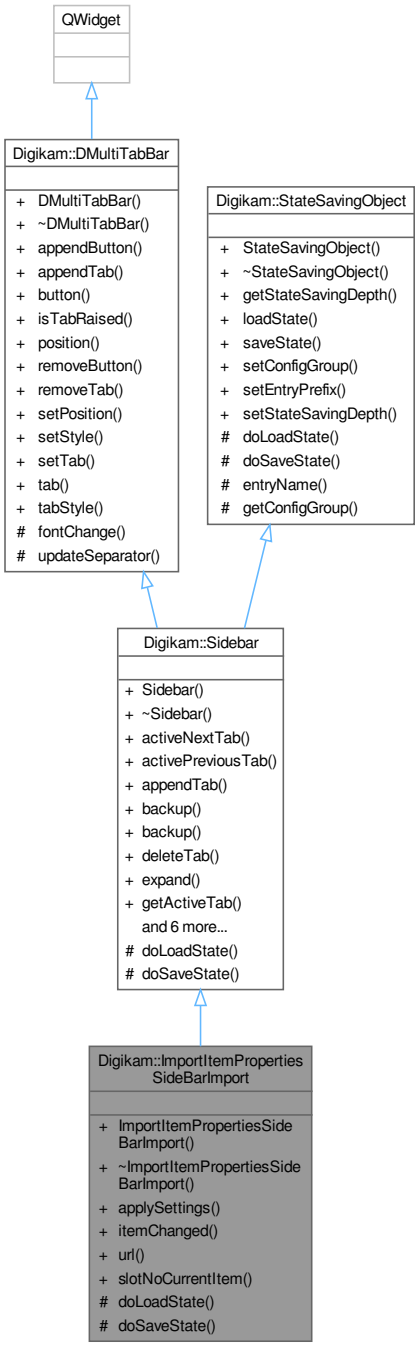
9.756.2.17 startRefresh()

```
void Digikam::ImportItemModel::startRefresh ( )  [protected]
```

No incremental refreshes will be started while listing. A clearCamItemInfos() always stops listing, calling finish↩ Refresh() 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.

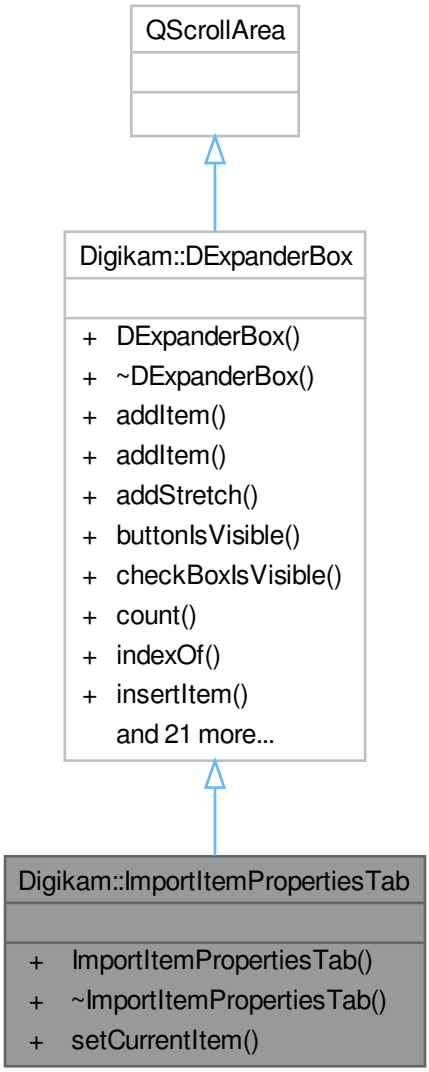
9.757.1.3 doSaveState()

```
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 expand↩ Bydefault)
- 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)

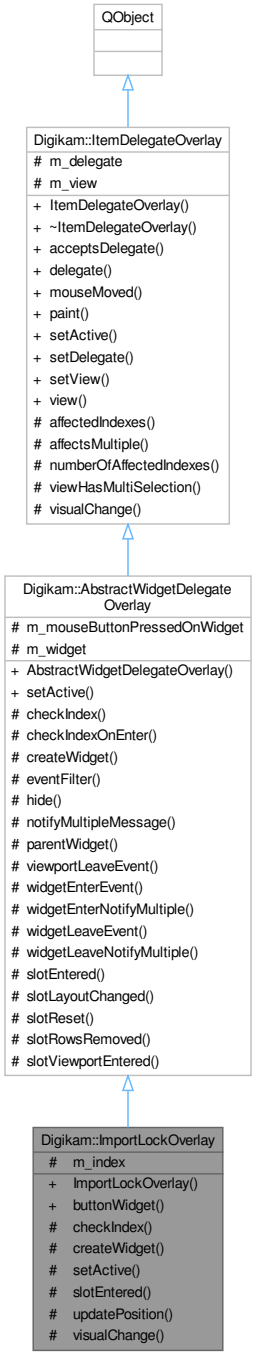
Additional Inherited Members

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 overlayed 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 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 is↩ Selected) 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** ()

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()

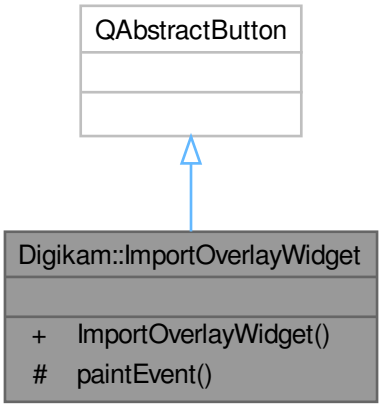
```
void Digikam::ImportNormalDelegate::updateRects ( )  [override], [protected], [virtual]
```

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 **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

- bool acceptsMouseClick (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

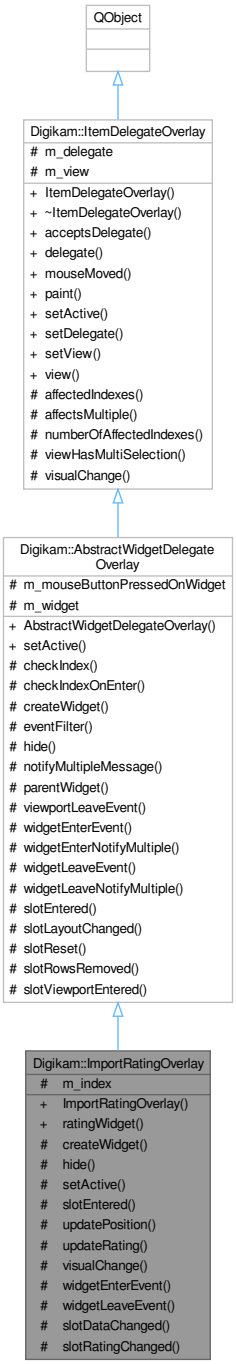
9.762.1.1 acceptsMouseClick()

```
bool Digikam::ImportPreviewView::acceptsMouseClick (
            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 overlayed 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.

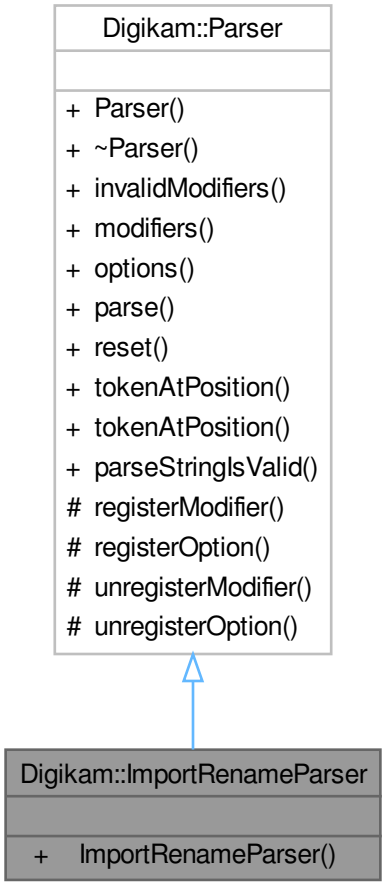
9.763.1.7 widgetLeaveEvent()

```
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 parseStringIsValid (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

- QTimeLine ∗ **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()

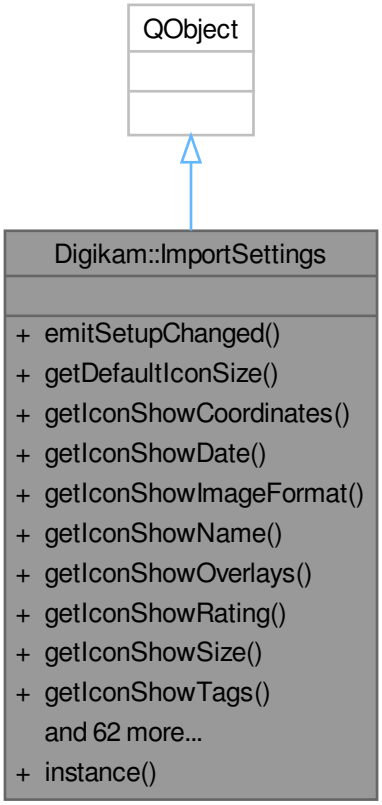
```
void Digikam::ImportRotateOverlayButton::updateToolTip ( )  [override], [protected], [virtual]
```

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

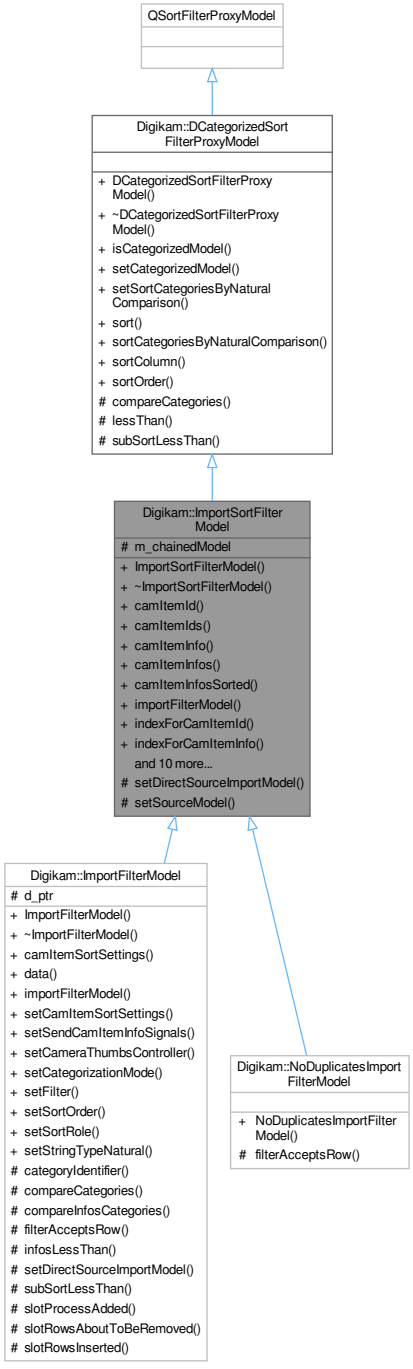
- static ImportSettings * **instance** ()

Friends

- class **ImportSettingsCreator**

9.768 Digikam::ImportSortFilterModel Class Reference

Inheritance diagram for Digikam::ImportSortFilterModel:



Public Member Functions

- **ImportSortFilterModel** (QObject ∗const parent=nullptr)
- 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

- 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.

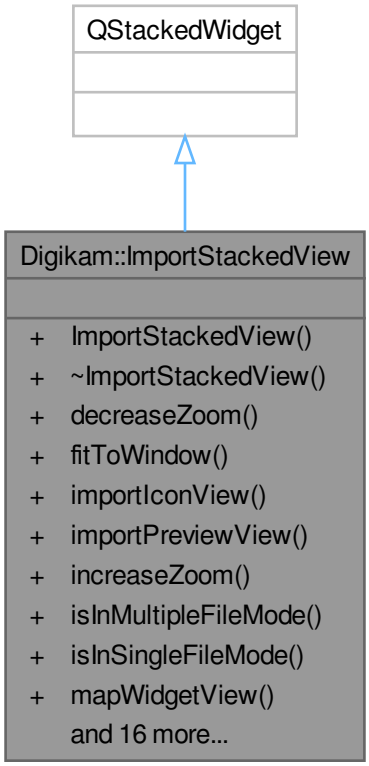
9.768.1.4 setDirectSourceImportModel()

```
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** , **Media↩ PlayerMode** }

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** (qlonglong 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 **decodeIsCutSelection** (const QMimeData *mimeData)
- void **encodeIsCutSelection** (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 **decodeIsCutSelection** (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 **encodeIsCutSelection** (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().

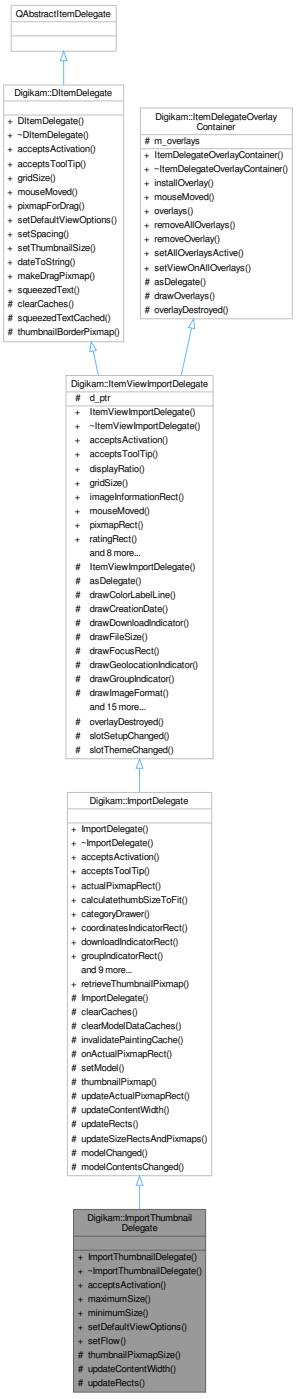
9.770.1.2 slotSetupChanged()

```
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 is↩ Selected) 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** ()

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()

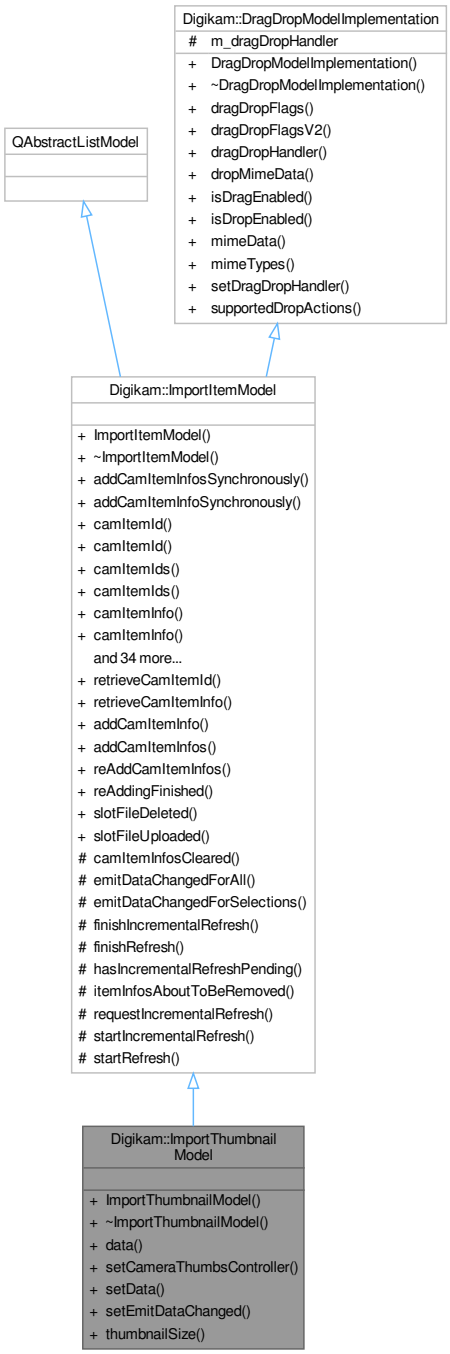
```
void Digikam::ImportThumbnailDelegate::updateRects ( )  [override], [protected], [virtual]
```

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 prepocessor 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 QModel↩ Index, 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** (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 > **indexesForCamItemId** (qlonglong id) const
- QList< QModelIndex > **indexesForCamItemInfo** (const CamItemInfo &info) const
- QList< QModelIndex > **indexesForUrl** (const QUrl &fileUrl) const
- QModelIndex **indexForCamItemId** (qlonglong 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** (qlonglong 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 **removeIndexs** (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

- 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

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 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 QSortFilter↩ ProxyModel 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 * infoIface (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 **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 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

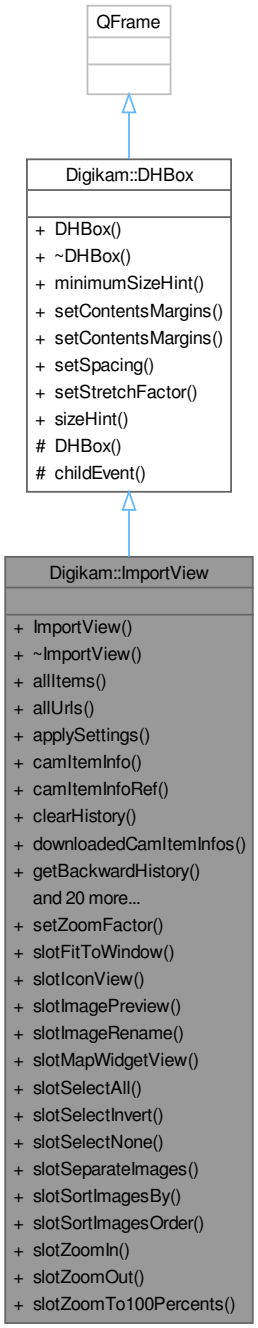
9.773.1.1 infoIface()

```
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 **slotMapWidgetView** ()
- 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

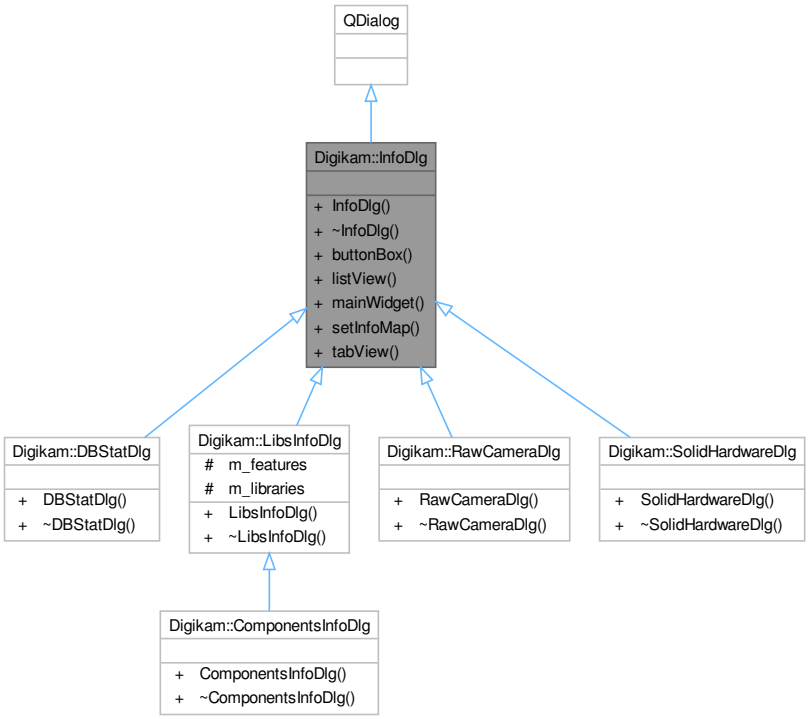
Additional Inherited Members

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** (DImg *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 **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

9.777.1 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.

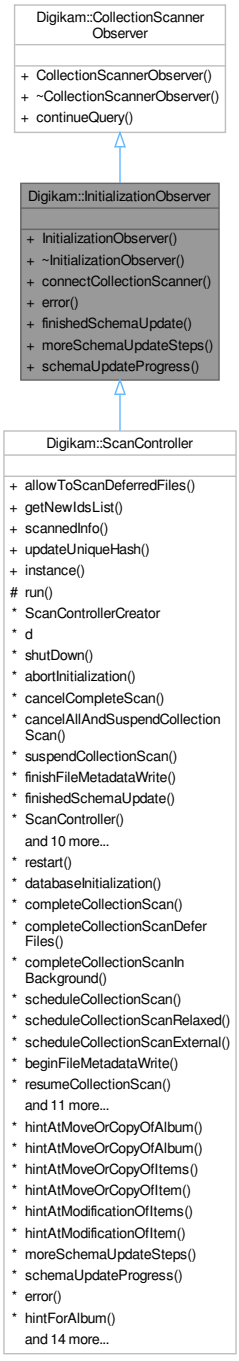
9.777.1.3 readParameters()

```
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

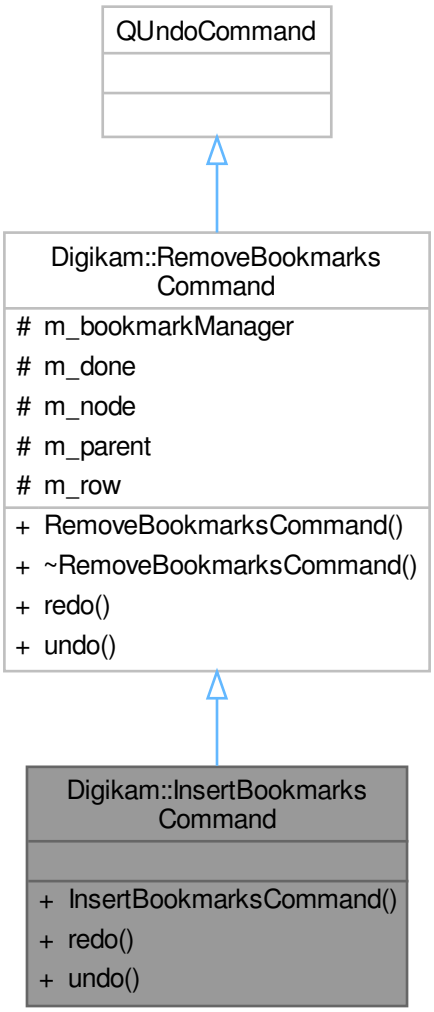
- virtual void **connectCollectionScanner** (CollectionScanner ∗const scanner)=0
- virtual void **error** (const QString &errorMessage)=0
- virtual void **finishedSchemaUpdate** (UpdateResult result)=0
- virtual void **moreSchemaUpdateSteps** (int numberOfSteps)=0
- virtual void **schemaUpdateProgress** (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 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** ()

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 **rawImportToolIid** = 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

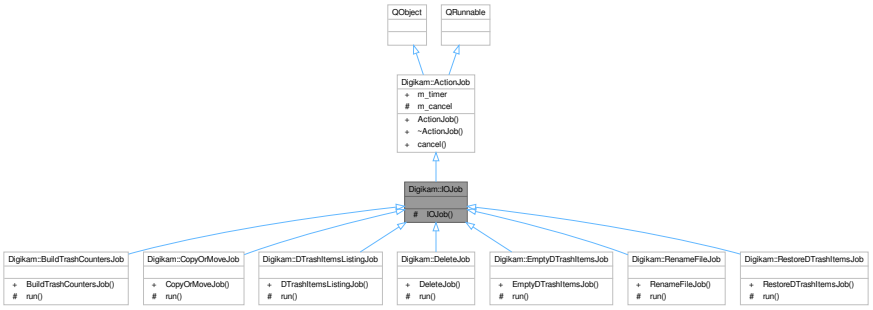
9.782.1.1 JPEGSubSampling

```
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 **signalOneProccessed** (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

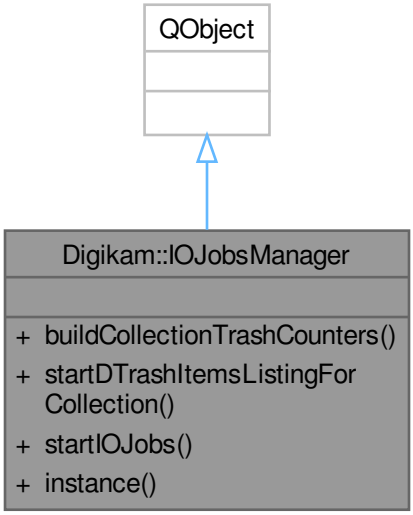
- enum **FileConflict** { **Continue** = 0 , **AutoRename** , **Overwrite** }
- enum **Operation** {
  Unknown = 0 , **CopyAlbum** , **CopyImage** , **CopyFiles** ,
  CopyToExt , **MoveAlbum** , **MoveImage** , **MoveFiles** ,
  Restore , **Rename** , **Delete** , **Trash** ,
  Empty }

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

collectionPath	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

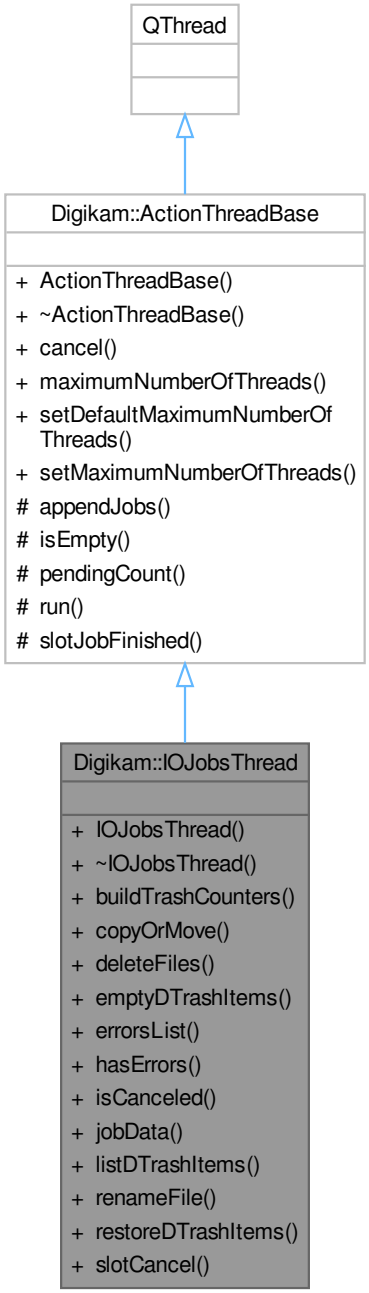
data	IOJobData container with source and destination url

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 **signalOneProccessed** (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

data	IOJobsData container

9.786.1.2 deleteFiles()

```
void Digikam::IOJobsThread::deleteFiles (
            IOJobData *const data )
```

Parameters

data	IOJobsData container

9.786.1.3 emptyDTrashItems()

```
void Digikam::IOJobsThread::emptyDTrashItems (
            IOJobData *const data )
```

Parameters

data	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

collectionPath	

9.786.1.9 renameFile()

```
void Digikam::IOJobsThread::renameFile (
            IOJobData *const data )
```

Parameters

data	IOJobsData container

9.786.1.10 restoreDTrashItems()

```
void Digikam::IOJobsThread::restoreDTrashItems (
            IOJobData *const data )
```

Parameters

data	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

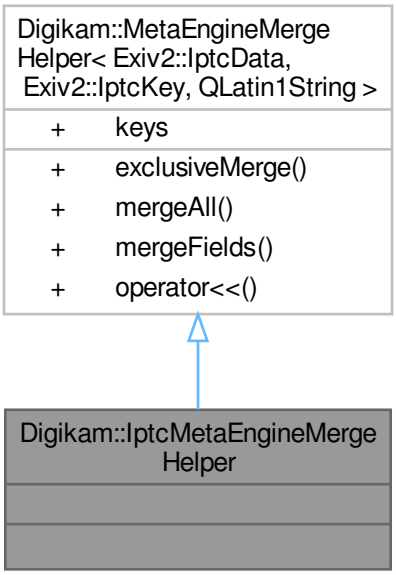
- bool **isEmpty** () const
- bool **isNull** () const
- void **merge** (const IptcCoreLocationInfo &t)
- bool **operator==** (const IptcCoreLocationInfo &t) const

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)

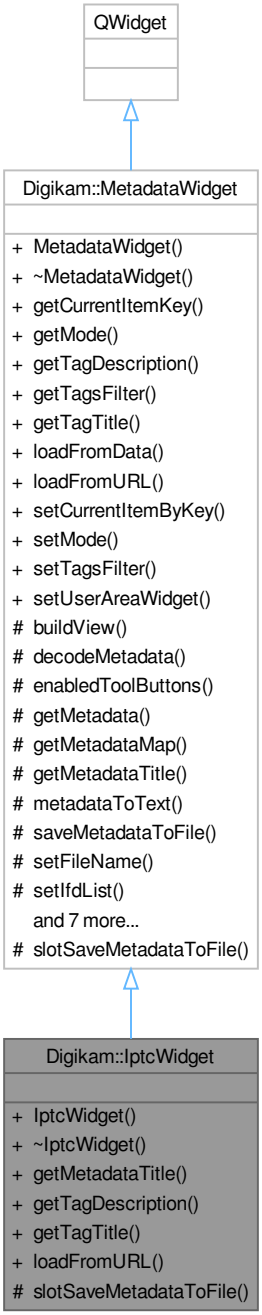
  Merge two (Exif,IPTC,Xmp) Data packages, the result is stored in dest.
- MetaEngineMergeHelper & **operator**<< (const QLatin1String &key)

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::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

9.790.1 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** (qlonglong 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)
- 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)

  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 , GroupIsOpenRole = 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 **setUrlWhitelist** (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.

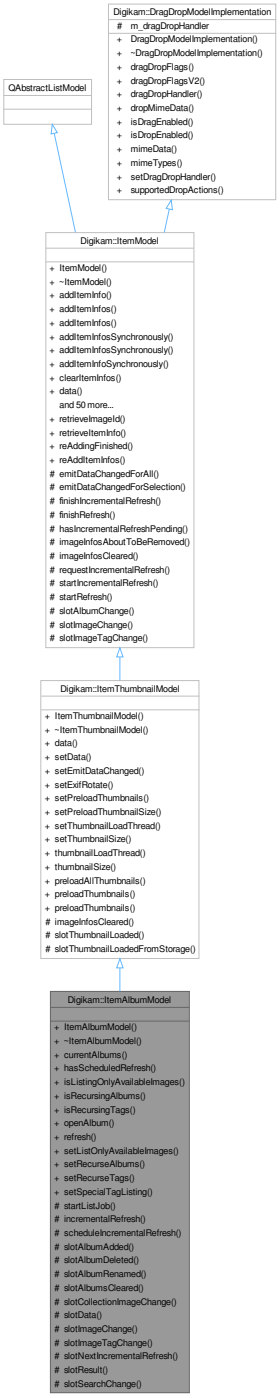
9.791.1.3 setItemFilterSettings()

```
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 **isRecursingAlbums** () const
- bool **isRecursingTags** () 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 prepocessor 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** (qlonglong id) const
- bool **hasImage** (qlonglong id, const QVariant &extraValue) const
- QVariant **headerData** (int section, Qt::Orientation orientation, int role=Qt::DisplayRole) const override
- qlonglong **imageId** (const QModelIndex &index) const
- qlonglong **imageId** (int row) const
- QList< qlonglong > **imageIds** () const
- QList< qlonglong > **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** (qlonglong id) const
- QList< QModelIndex > **indexesForItemInfo** (const ItemInfo &info) const
- QList< QModelIndex > **indexesForPath** (const QString &filePath) const
- QModelIndex **indexForImageId** (qlonglong id) const
- QModelIndex **indexForImageId** (qlonglong 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** (qlonglong 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

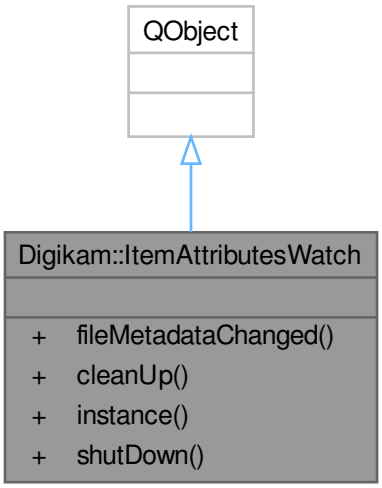
9.792.1.2 slotImageChange

```
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 if the given file has been changed (a write operation on the file on disk).
- void **signalImageCaptionChanged** (qlonglong imageId)
- void **signalImageDateChanged** (qlonglong imageId)
- void signalImageRatingChanged (qlonglong imageId)

  These signals indicated 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 (qlonglong 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** (qlonglong 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 **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 **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 **rowsInsertedArtifficial** (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 **decodeIsCutSelection** (const QMimeData ∗mimeData)
- void **encodeIsCutSelection** (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 **decodeIsCutSelection** (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 **encodeIsCutSelection** (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.

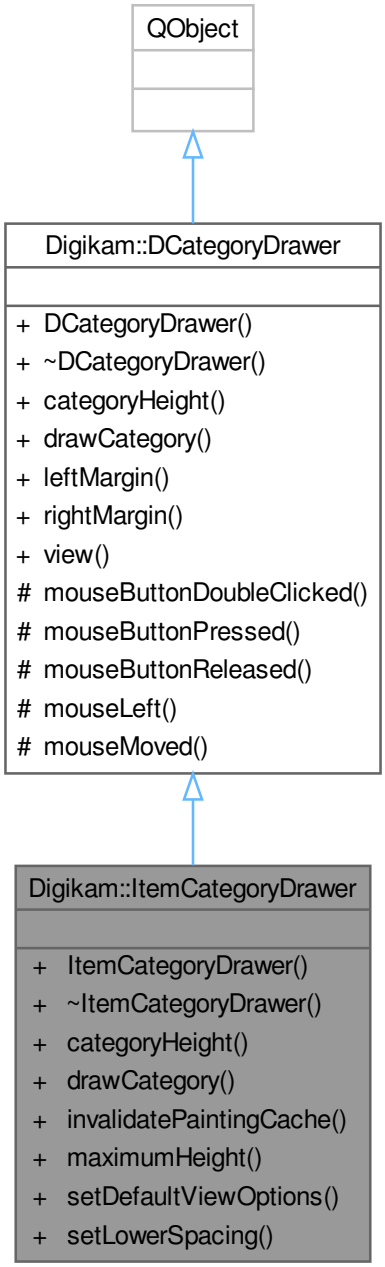
9.794.1.8 showContextMenuOnIndex()

```
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

index	with the given
sortRole	sorting role
option	painter style options
painter	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, qlonglong imageid)

  Create a ItemComments object for the image with the specified id.
- **ItemComments** (const ItemComments &other)
- **ItemComments** (qlonglong 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=Database↩ Comment::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

- QSharedDataPointer< 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 & Destructor Documentation

9.797.2.1 ItemComments()

```
Digikam::ItemComments::ItemComments (
            const CoreDbAccess & access,
            qlonglong 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,

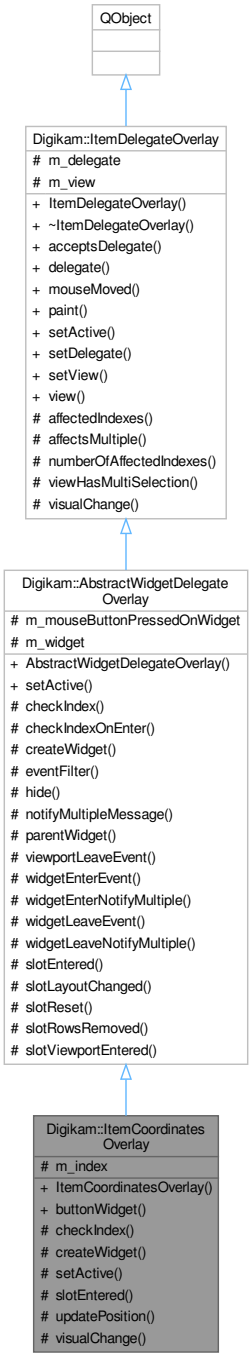
9.797.3.10 type()

```
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 overlayed 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 intents 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.

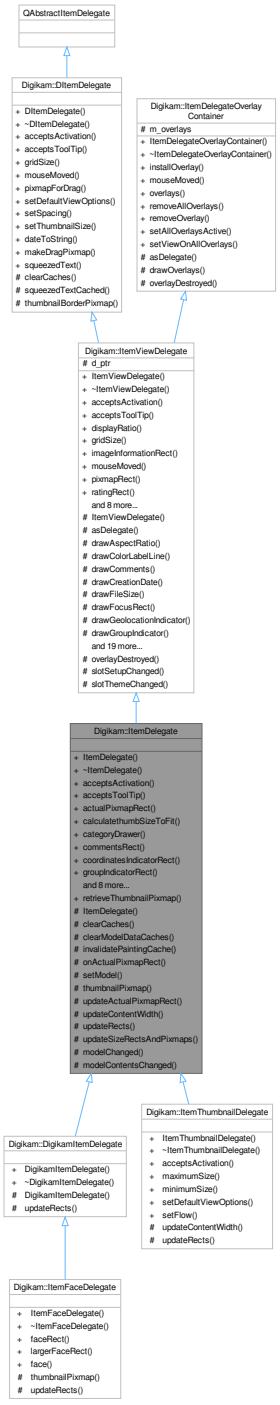
9.800.2.12 source()

```
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 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 **drawPanelSideIcon** (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 is↩ Selected) 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 **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::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.

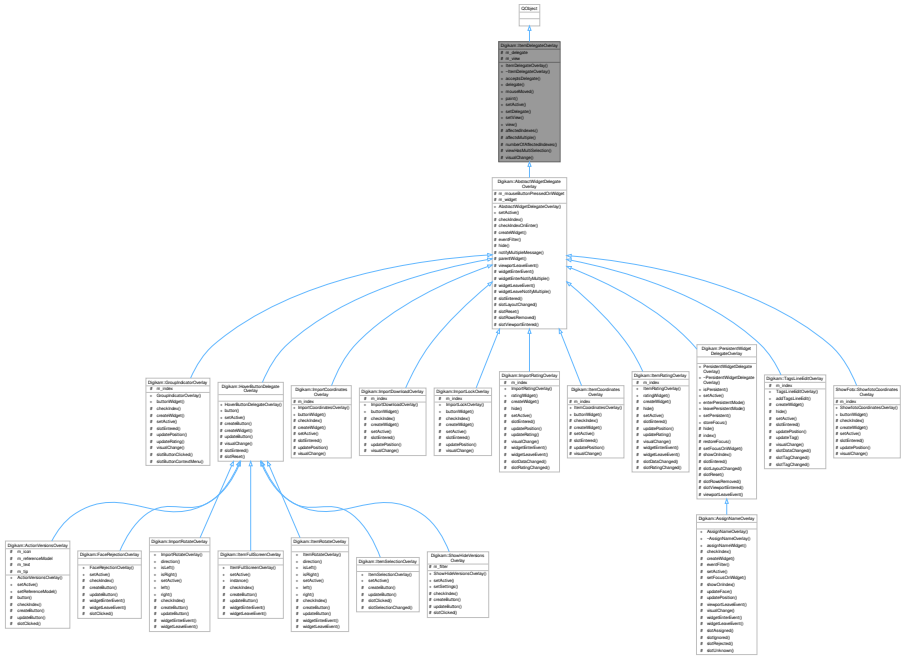
9.801.1.12 updateSizeRectsAndPixmaps()

```
void Digikam::ItemDelegate::updateSizeRectsAndPixmaps ( )  [override], [protected], [virtual]
```

Implements Digikam::ItemViewDelegate.

9.802 Digikam::ItemDelegateOverlay Class Reference

Inheritance diagram for Digikam::ItemDelegateOverlay:



Signals

- void **hideNotification** ()
- void **requestNotification** (const QModelIndex &index, const QString &message)
- void **update** (const QModelIndex &index)

Public Member Functions

- **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)
- virtual void setActive (bool active)

  Called when the overlay was installed and shall begin working, and before it is removed and shall stop.
- void **setDelegate** (QAbstractItemDelegate ∗delegate)
- void **setView** (QAbstractItemView ∗view)
- QAbstractItemView ∗ **view** () const

Protected Slots

- 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

- 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.

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::ShowfotoCoordinates 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

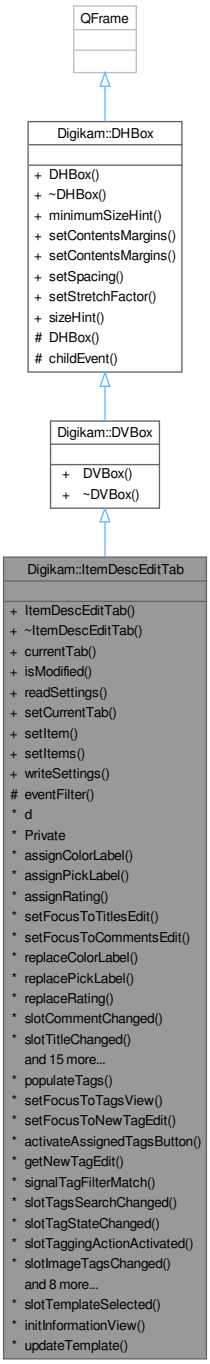
9.803.2.1 asDelegate()

```
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)

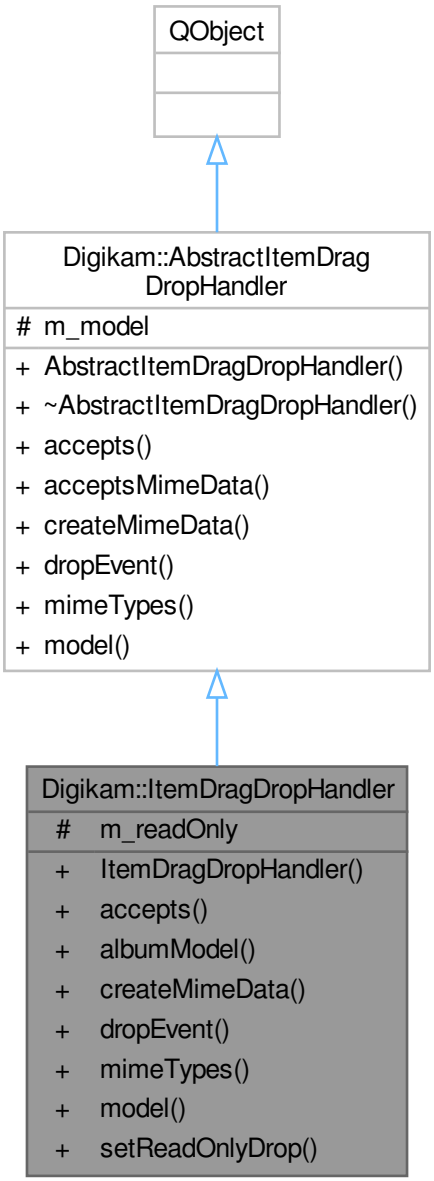
  Description view methods (itemdescedittab_descview.cpp)
- void **assignPickLabel** (int pickId)
- void **assignRating** (int rating)
- void **setFocusToTitlesEdit** ()
- void **setFocusToCommentsEdit** ()
- void **replaceColorLabel** (int colorId)
- void **replacePickLabel** (int pickId)
- void **replaceRating** (int rating)

- void **populateTags** ()

  Tags view methods (itemdescedittab_tagsview.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 mimeTypes()

```
QStringList Digikam::ItemDragDropHandler::mimeTypes ( ) const  [override], [virtual]
```

Called by the default implementation of model's mimeTypes().

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** (qlonglong 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 qlonglong 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 qlonglong imageId, const double value)
- void **setSubjectCode** (const QStringList &subjectCode)
- double similarityTo (const qlonglong 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."

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 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 **drawPanelSideIcon** (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 is↩ Selected) 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 **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::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()

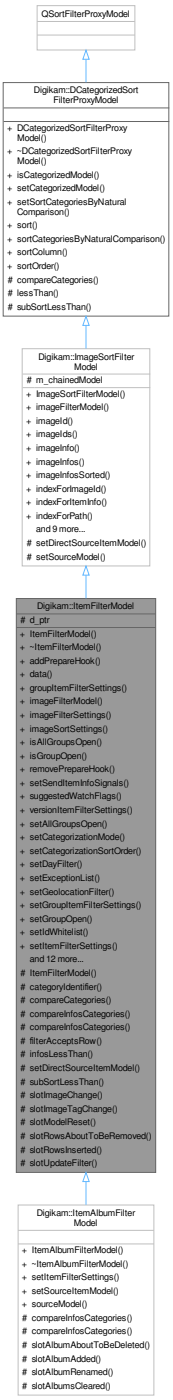
```
void Digikam::ItemFaceDelegate::updateRects ( )  [override], [protected], [virtual]
```

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::Filter↩
ModelRoles + 4 ,
CategoryDateRole = ItemModel::FilterModelRoles + 5 , CategoryFaceRole = ItemModel::FilterModelRoles
+ 6 , GroupIsOpenRole = 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 **setUrlWhitelist** (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** (qlonglong 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)
- 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)

  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 PAlbum 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.
GroupIsOpenRole	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 Category↩ SortRole. In order to correctly sort categories, the data() method of the model should return a qlonglong (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

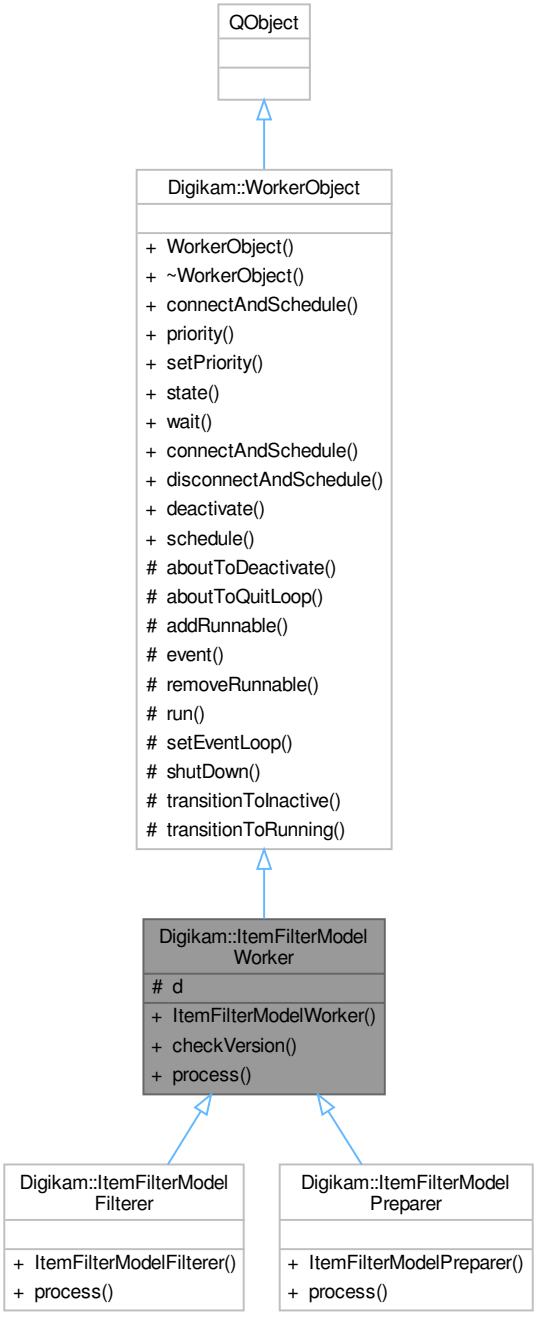
9.811.1.1 process()

```
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.

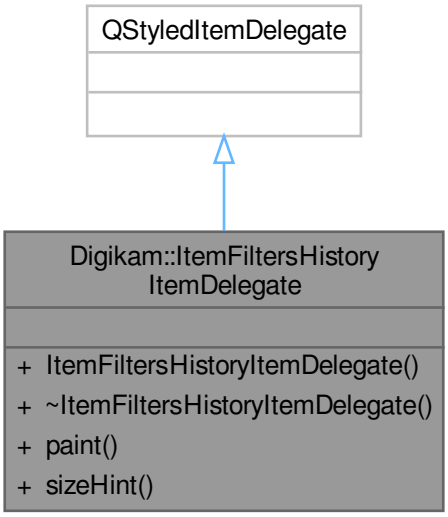
9.813.1.2 watchFlags()

```
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:

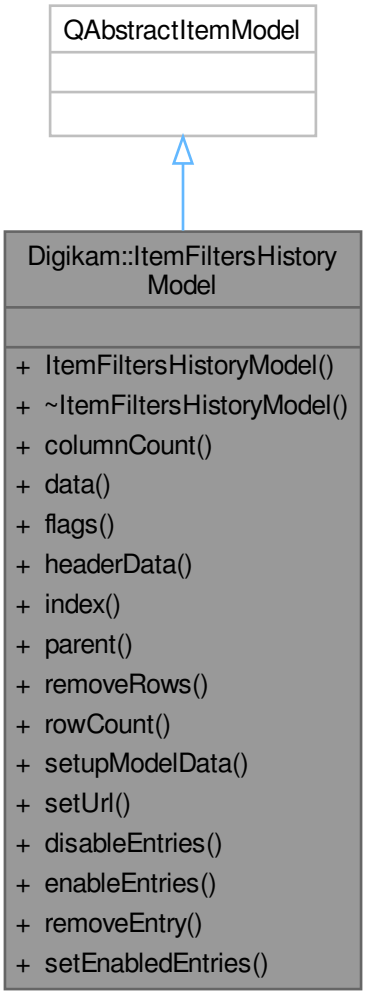


Public Member Functions

- **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 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

9.817.1 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

- QTimeLine ∗ **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()

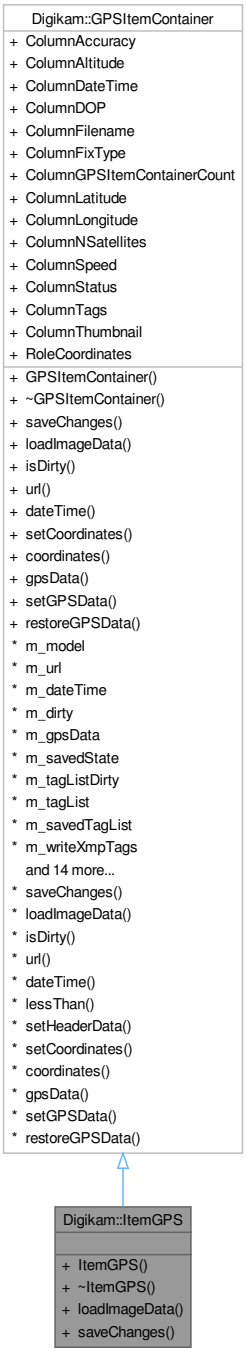
```
void Digikam::ItemFullScreenOverlayButton::updateToolTip ( )  [override], [protected], [virtual]
```

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.

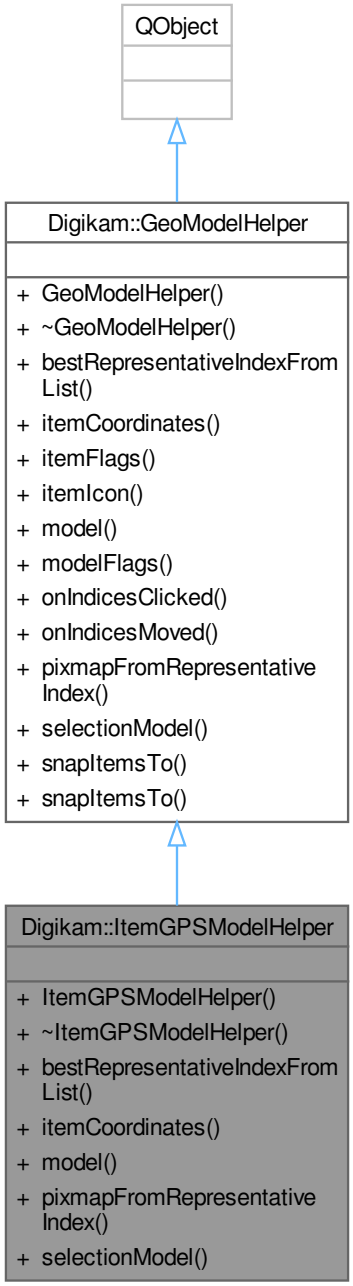
9.819.1.2 saveChanges()

```
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< qlonglong, qlonglong > > &pairs)

  Add images and their relations from the given pairs.
- void addScannedHistory (const DImageHistory &history, qlonglong historySubjectId)

  This is very similar to addHistory.
- QList< qlonglong > **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< qlonglong, qlonglong > > relationCloud () const

  Returns all possible relations between images in this graph, the edges of the transitive closure.
- QPair< QList< qlonglong >, QList< qlonglong > > **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 DImageHistory of the given subject.
LoadLeavesHistory	Will load the DImageHistory 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).

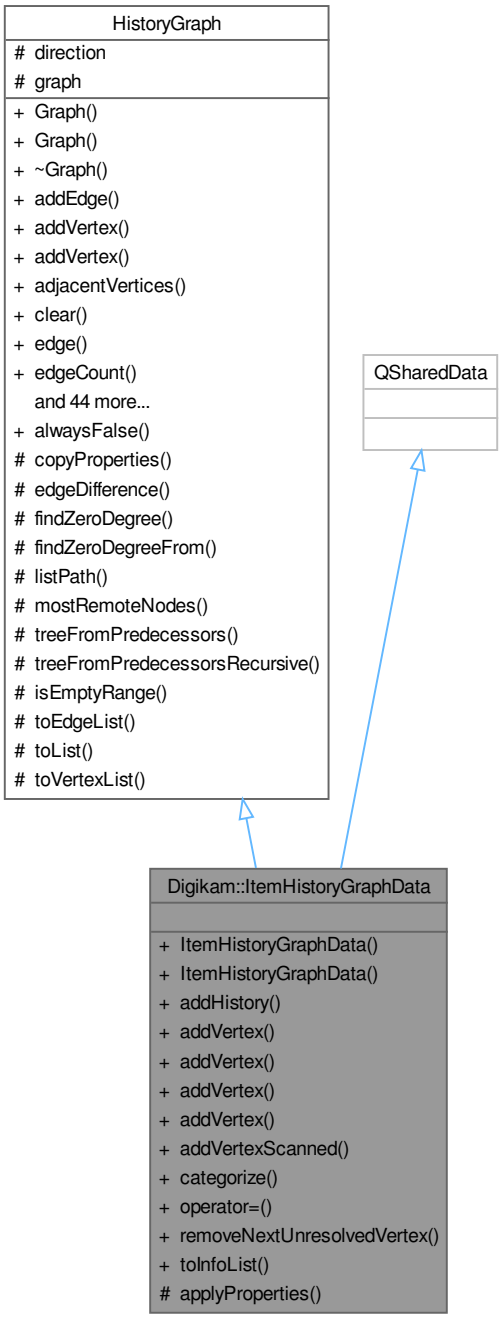
9.821.2.10 rootImages()

```
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 HistoryImageId &id)
- Vertex **addVertex** (const ItemInfo &info)
- Vertex **addVertex** (const QList< HistoryImageId > &imageIds)
- Vertex **addVertex** (qlonglong id)
- Vertex **addVertexScanned** (qlonglong id)
- QHash< Vertex, HistoryImageId::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 **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=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< HistoryImageId > &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 , **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**

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 , **IsHeaderItem↩ Role** = Qt::UserRole + 1002 , **IsCategoryItemRole** = Qt::UserRole + 1003 ,

IsSeparatorItemRole = Qt::UserRole + 1004 , **IsSubjectImageRole** = Qt::UserRole + 1010 , **FilterAction↩ Role** = 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.

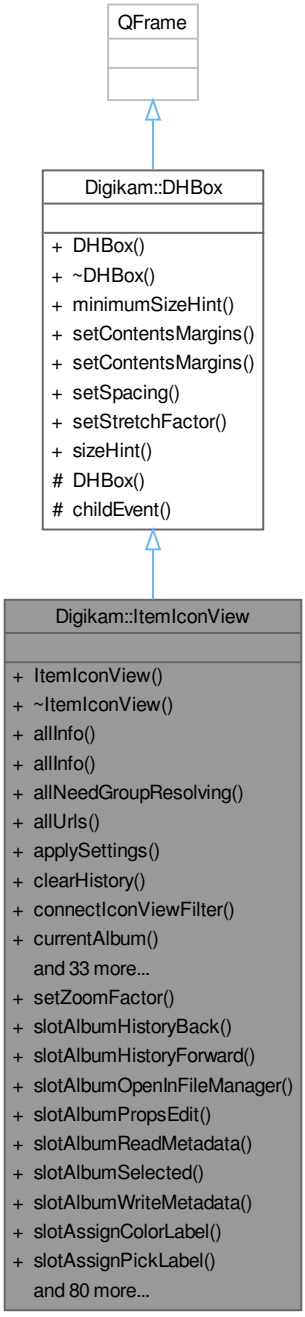
9.823.1.4 setHistory()

```
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 **slotMapWidgetView** ()
- 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 (qlonglong 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
- qlonglong 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.
- qlonglong id () const
- int albumId () const
- int **albumRootId** () const

  The album root id.
- double aspectRatio () const
- qlonglong **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** (qlonglong 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** (qlonglong 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 (
            qlonglong ID )  [explicit]
```

Creates an ItemInfo object without any cached data initially.

Parameters

ID	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

dstAlbumID	destination album id
dstFileName	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()

```
qlonglong 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

tagID	the ID of the tag to remove

9.825.3.23 setDateTime()

```
void Digikam::ItemInfo::setDateTime (
            const QDateTime & dateTime )
```

Parameters

dateTime	the new date and time.

9.825.3.24 setMetadataTemplate()

```
void Digikam::ItemInfo::setMetadataTemplate (
            const Template & t )
```

Parameters

t	the new template data.

9.825.3.25 setModDateTime()

```
void Digikam::ItemInfo::setModDateTime (
            const QDateTime & dateTime )
```

Parameters

dateTime	the new modification date and time.

9.825.3.26 setName()

```
void Digikam::ItemInfo::setName (
            const QString & newName )
```

Parameters

newName	the new name.

9.825.3.27 setTag()

```
void Digikam::ItemInfo::setTag (
            int tagID )
```

Parameters

tagID	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()

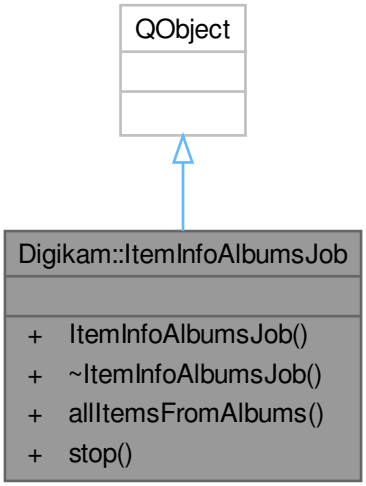
```
QString Digikam::ItemInfo::uniqueHash ( ) const
```

Returns

the unique hash signature as string of the image.

9.826 Digikam::ItemInfoAlbumsJob Class Reference

Inheritance diagram for Digikam::ItemInfoAlbumsJob:



Signals

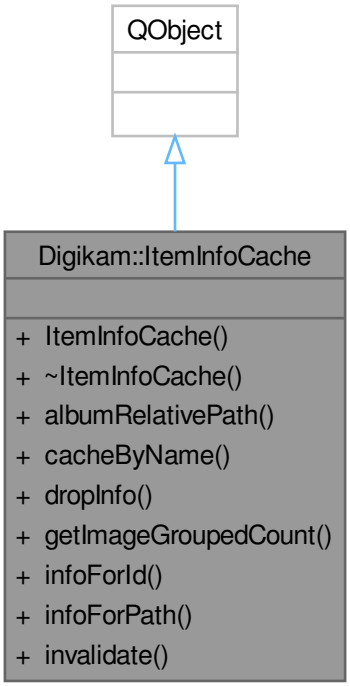
- void **signalCompleted** (const ItemInfoList &items)

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.

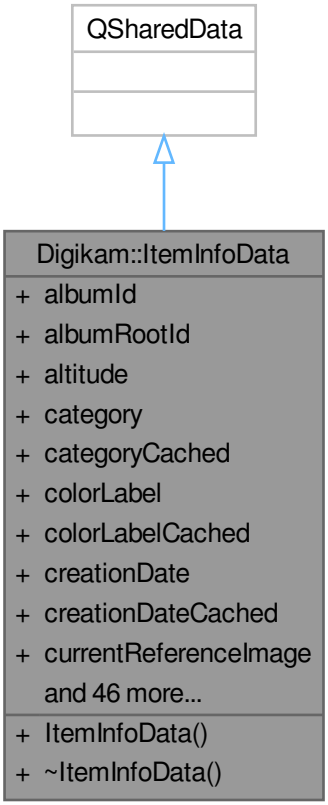
9.827.1.3 infoForPath()

```
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

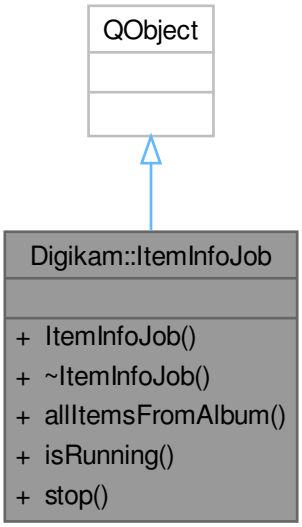
- int **albumId** = -1
- int **albumRootId** = -1
- double **altitude** = 0
- DatabaseItem::Category **category** = DatabaseItem::UndefinedCategory
- bool **categoryCached** = false
- quint8 **colorLabel** = NoColorLabel
- bool **colorLabelCached** = false
- QDateTime **creationDate**
- bool **creationDateCached** = 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
- qint8 **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

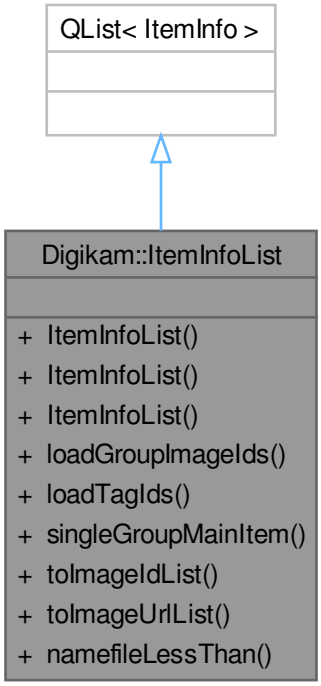
- void **signalCompleted** ()
- void **signalItemsInfo** (const ItemInfoList &items)

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 **loadGroupImageIds** () const
- void **loadTagIds** () const
- ItemInfo singleGroupMainItem () const

  singleGroupMainItem
- QList< qlonglong > **toImageIdList** () 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()

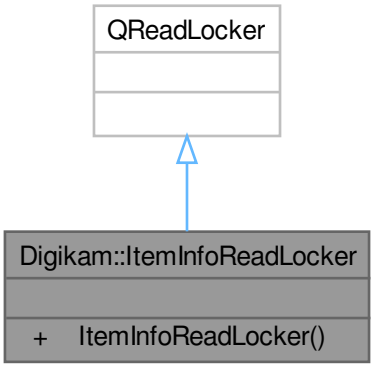
```
ItemInfo Digikam::ItemInfoList::singleGroupMainItem ( ) const
```

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

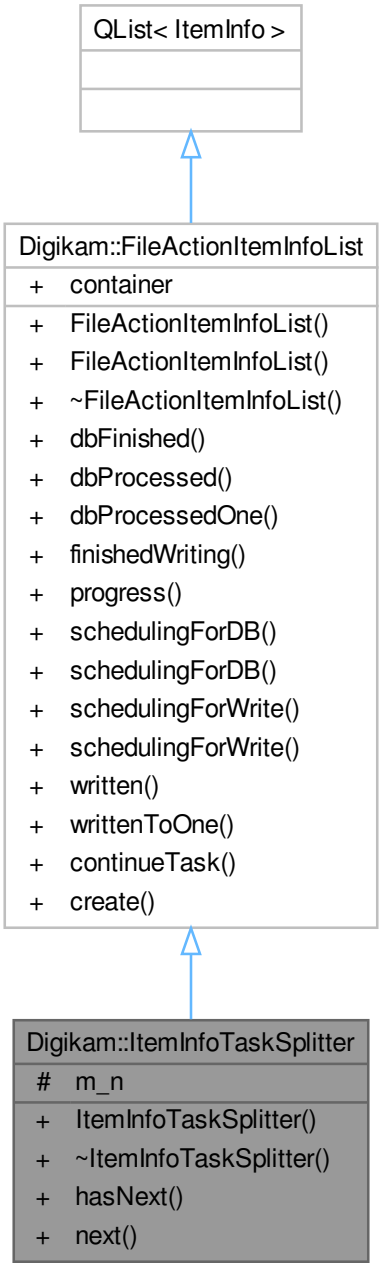
- ItemInfoCache **m_cache**
- QReadWriteLock **m_lock**

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

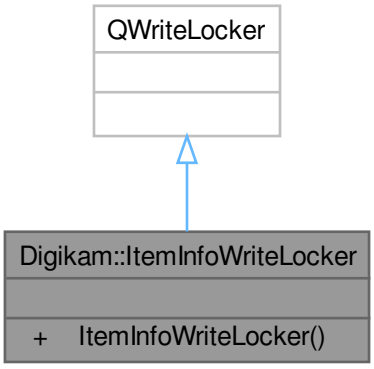
- static FileActionItemInfoList **continueTask** (const QList< ItemInfo > &list, FileActionProgressItemContainer *const container)
- static FileActionItemInfoList **create** (const QList< ItemInfo > &list)

Public Attributes inherited from Digikam::FileActionItemInfoList

- QExplicitlySharedDataPointer< FileActionProgressItemContainer > **container**

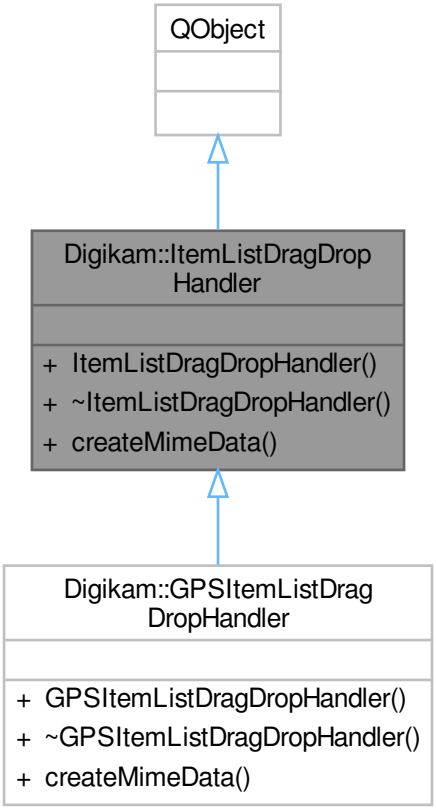
9.835 Digikam::ItemInfoWriteLocker Class Reference

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 referenceImageId=-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

receiver	the receiver for the searches
xml	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

receiver	the receiver for the searches
xml	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

receiver	the receiver for the searches
xml	SearchXml describing the query
limit	the limit the count of the result set. If limit = 0, then no limit is set.
referenceImageId	the id of a reference image in the search query.

9.837.1.5 setListOnlyAvailable()

```
void Digikam::ItemLister::setListOnlyAvailable (
            bool listOnlyAvailable )
```

Default: true.

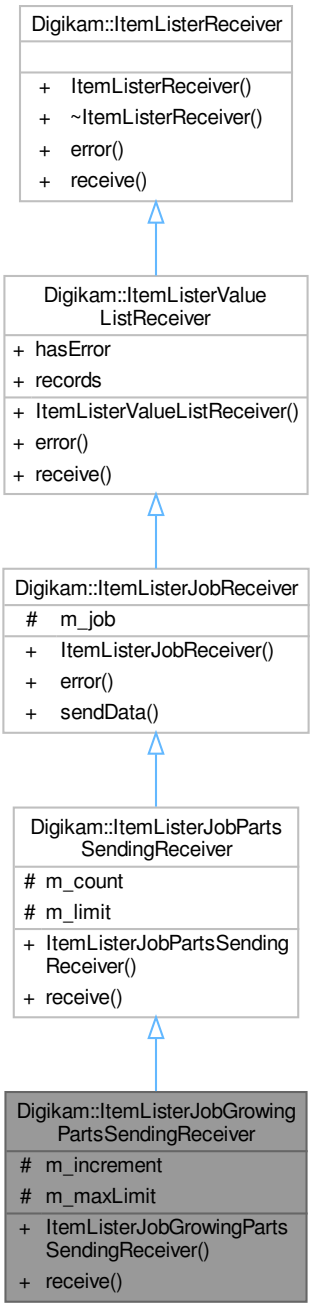
9.837.1.6 setRecursive()

```
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

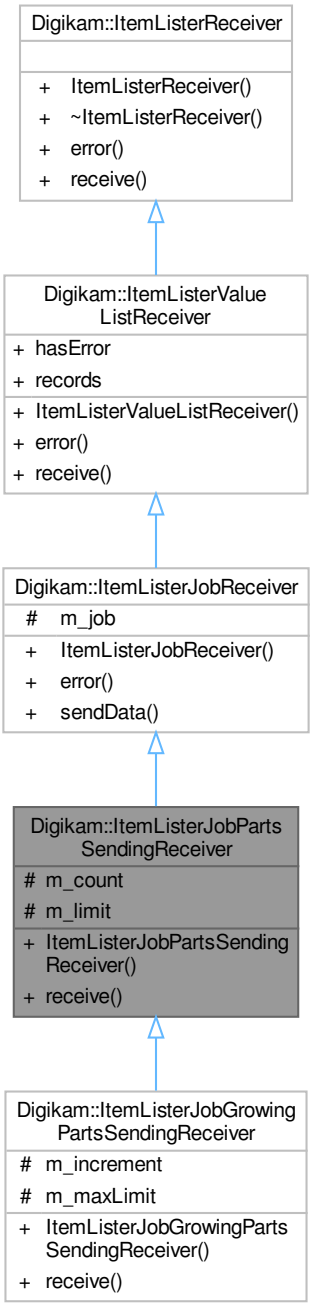
9.838.1.1 receive()

```
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

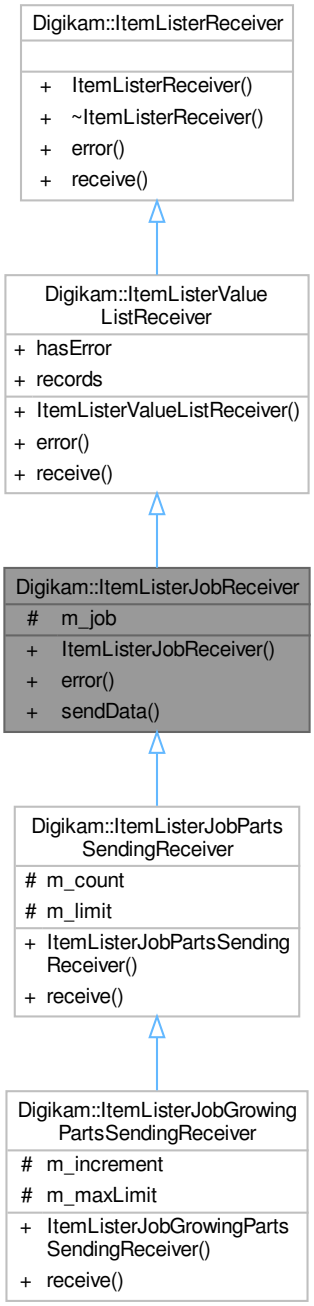
9.839.1.1 receive()

```
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

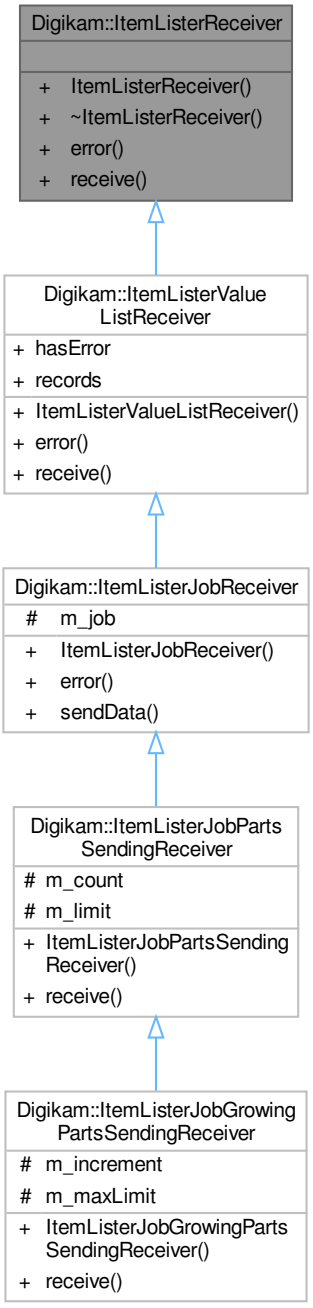
9.840.1.1 error()

```
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

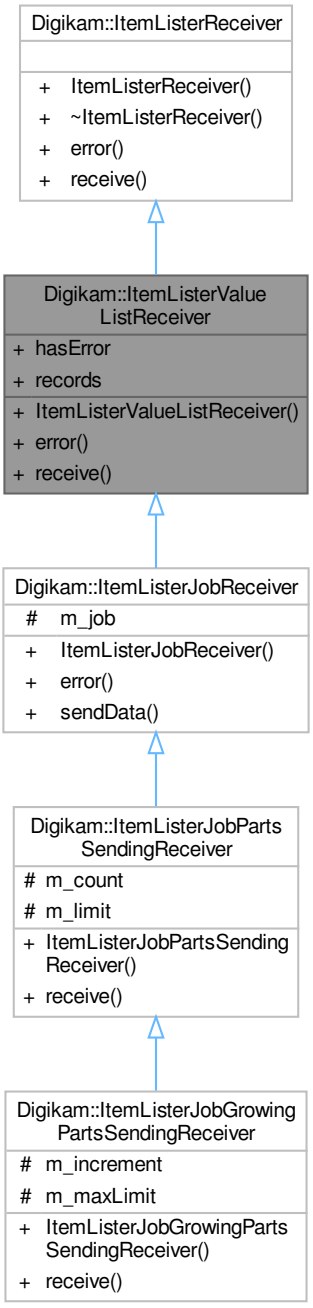
- bool **operator==** (const ItemListerRecord &record) const

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 prepocessor 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** (qlonglong id) const
- bool **hasImage** (qlonglong id, const QVariant &extraValue) const
- QVariant **headerData** (int section, Qt::Orientation orientation, int role=Qt::DisplayRole) const override
- qlonglong **imageId** (const QModelIndex &index) const
- qlonglong **imageId** (int row) const
- QList< qlonglong > **imageIds** () const
- QList< qlonglong > **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** (qlonglong id) const
- QList< QModelIndex > **indexesForItemInfo** (const ItemInfo &info) const
- QList< QModelIndex > **indexesForPath** (const QString &filePath) const
- QModelIndex **indexForImageId** (qlonglong id) const
- QModelIndex **indexForImageId** (qlonglong 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** (qlonglong 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 **retrieveImageId** (const QModelIndex &index)
- static ItemInfo retrieveItemInfo (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 ()

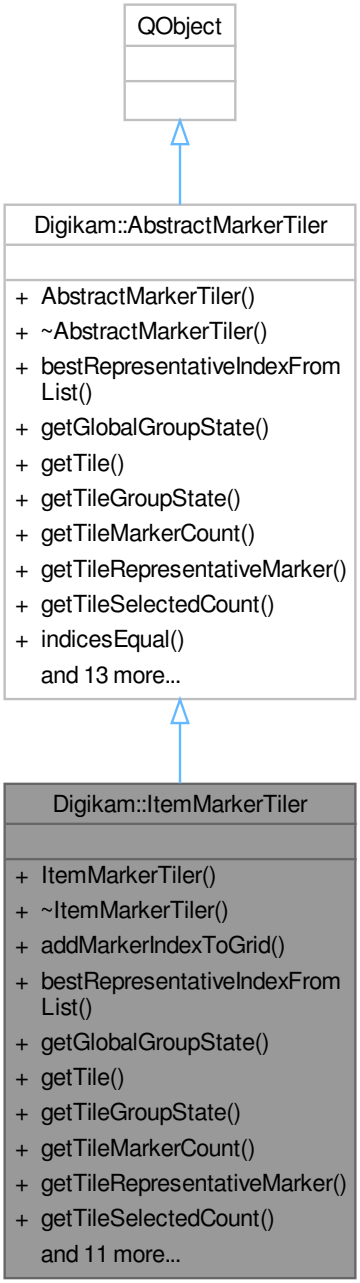
  Subclasses that add ItemInfos in batches shall call startRefresh() when they start sending batches and finishRefresh() when they have finished.

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 QIntList &a, const QIntList &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

markerIndex	The marker index to remove
ignoreSelection	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** (qlonglong id, AdjustmentStatus status, const QDateTime &modification↩ DateOnDisk, qlonglong fileSize)
- AdjustmentStatus **adjustmentStatus** () const
- qlonglong **fileSize** () const
- qlonglong **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

- qlonglong **m_fileSize** = 0
- qlonglong **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
```

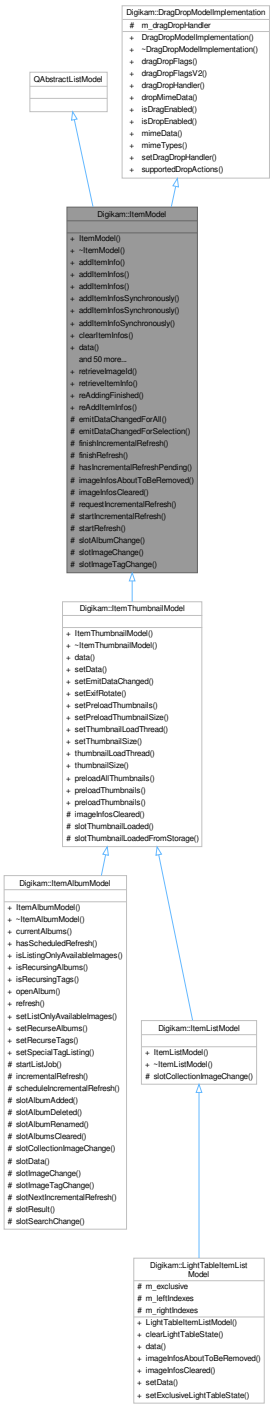
There is no need for a full rescan, optimizations are possible.

Enumerator

AboutToEditMetadata	The file is about to be edited. Suspends scanning. The Finished hint must follow.
MetadataEditingFinished	The file's metadata has been edited as described above.
MetadataEditingAborted	The file's metadata has not been edited, despite sending AboutToEditMedata.

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 , LTLeftPanelRole = Qt::UserRole + 50 , LTRightPanelRole = 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 prepocessor 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** (qlonglong id) const
- bool **hasImage** (qlonglong id, const QVariant &extraValue) const
- QVariant **headerData** (int section, Qt::Orientation orientation, int role=Qt::DisplayRole) const override
- qlonglong **imageId** (const QModelIndex &index) const
- qlonglong **imageId** (int row) const
- QList< qlonglong > **imageIds** () const
- QList< qlonglong > **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** (qlonglong id) const
- QList< QModelIndex > **indexesForItemInfo** (const ItemInfo &info) const
- QList< QModelIndex > **indexesForPath** (const QString &filePath) const
- QModelIndex **indexForImageId** (qlonglong id) const
- QModelIndex **indexForImageId** (qlonglong 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** (qlonglong 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.

Static Public Member Functions

- 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 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.
LTLeftPanelRole	Roles which are defined here but not implemented by ItemModel Returns position of item in Left Light Table preview.
LTRightPanelRole	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 (
            qlonglong 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.

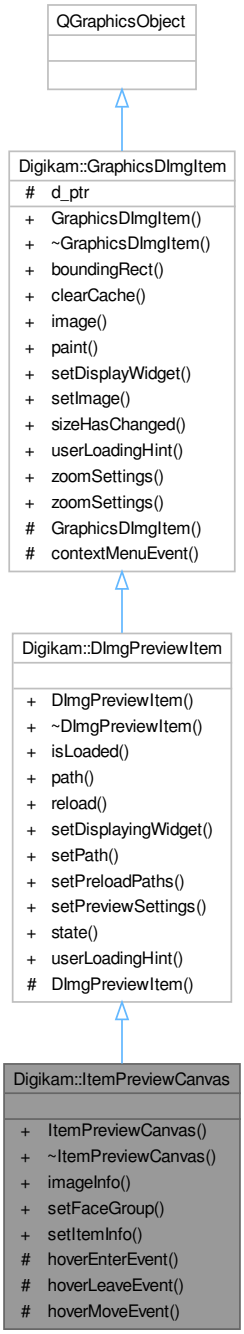
9.848.2.8 setLatitude() [2/2]

```
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 **isLoaded** () 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

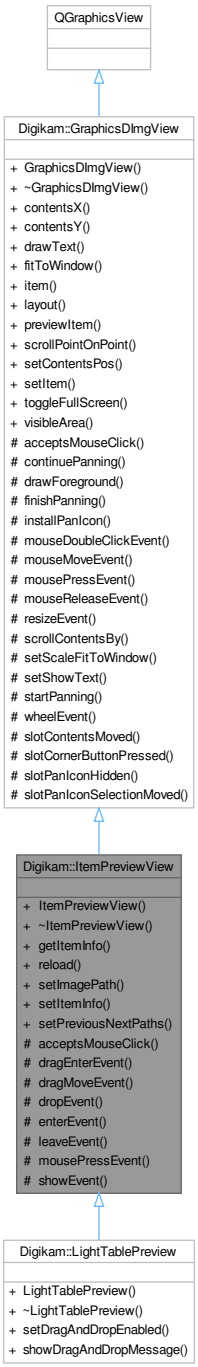
- void **imageChanged** ()
- void **imageSizeChanged** (const QSizeF &size)
- void **showContextMenu** (QGraphicsSceneContextMenuEvent ∗e)

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 **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

- bool acceptsMouseClick (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

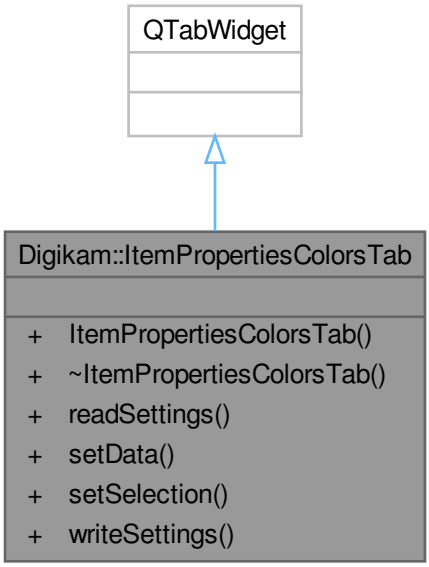
9.850.1.1 acceptsMouseClick()

```
bool Digikam::ItemPreviewView::acceptsMouseClick (
            QMouseEvent * e )  [override], [protected], [virtual]
```

Reimplemented from Digikam::GraphicsDImgView.

9.851 Digikam::ItemPropertiesColorsTab Class Reference

Inheritance diagram for Digikam::ItemPropertiesColorsTab:

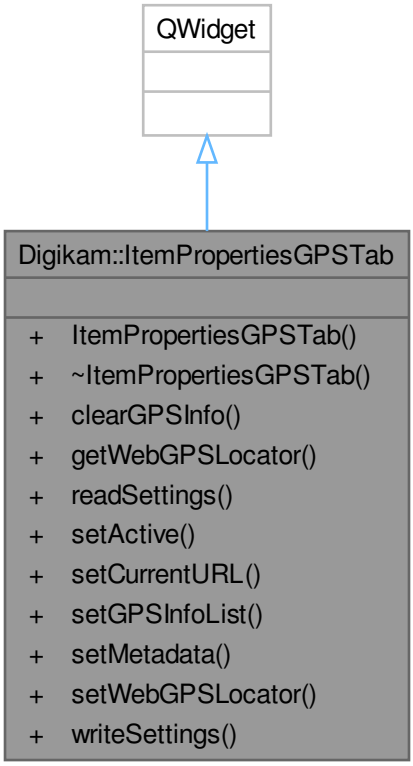


Public Member Functions

- **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

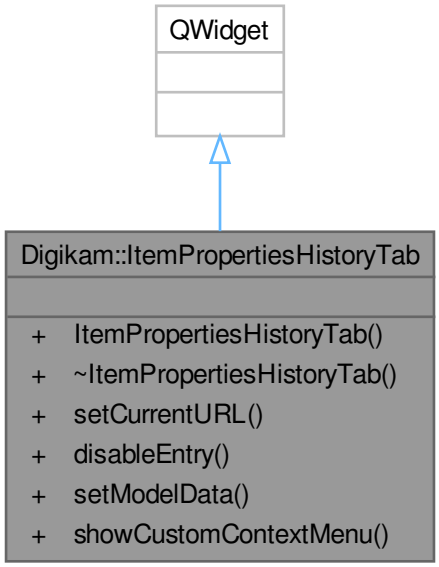
- enum **WebGPSLocator** {
  MapQuest = 0 , **GoogleMaps** , **BingMaps** , **OpenStreetMap** ,
  LocAlizeMaps }

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

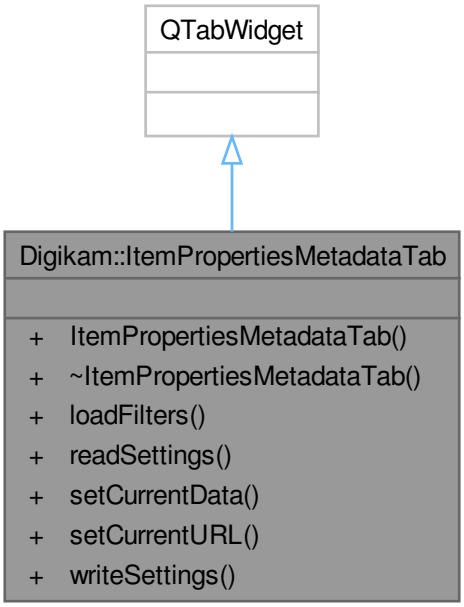
- void **disableEntry** (bool disable)
- void **setModelData** (const QList< DImageHistory::Entry > &entries)
- void **showCustomContextMenu** (const QPoint &position)

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

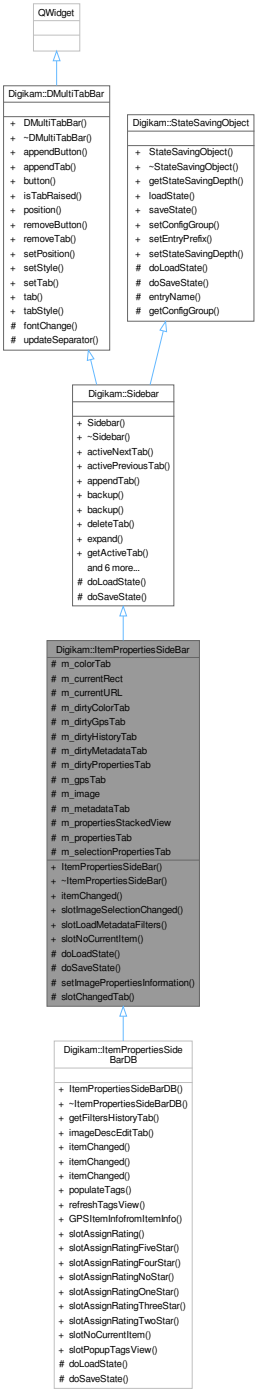
- void **signalSetupExifTool** ()
- void **signalSetupMetadataFilters** (int)

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 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 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()

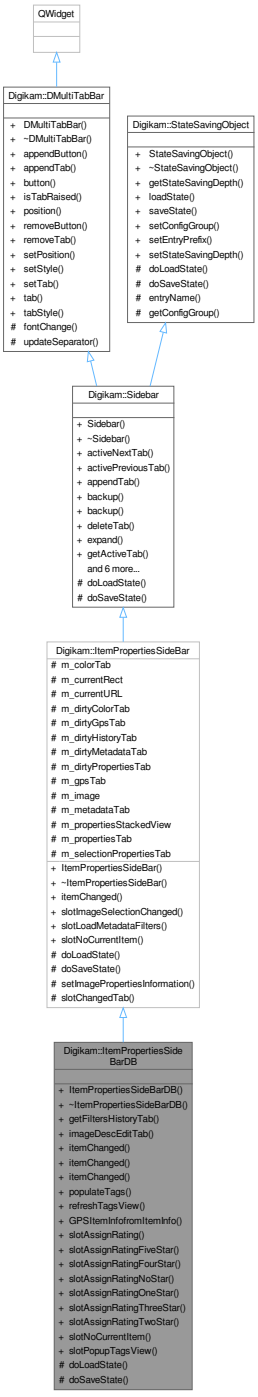
```
void Digikam::ItemPropertiesSideBar::doSaveState ( )  [override], [protected], [virtual]
```

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.

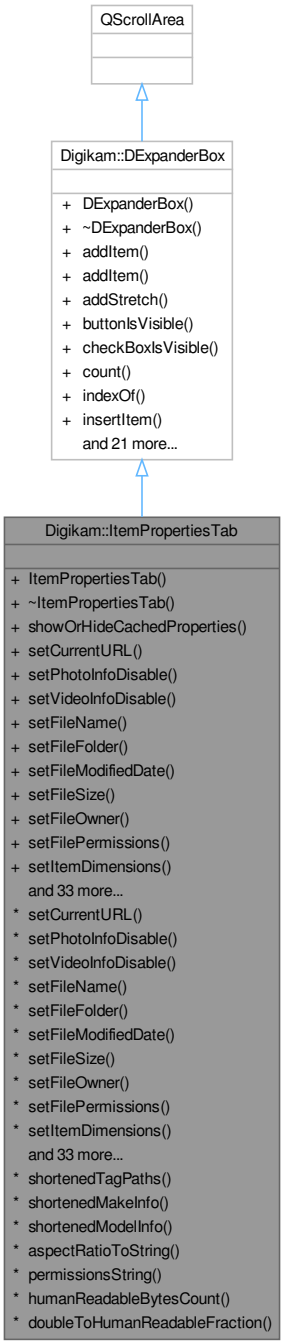
9.856.1.3 itemChanged()

```
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 (itempropertiestab_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 expand↩ Bydefault)
- 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 (itempropertiestab_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 QFileInfo &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.

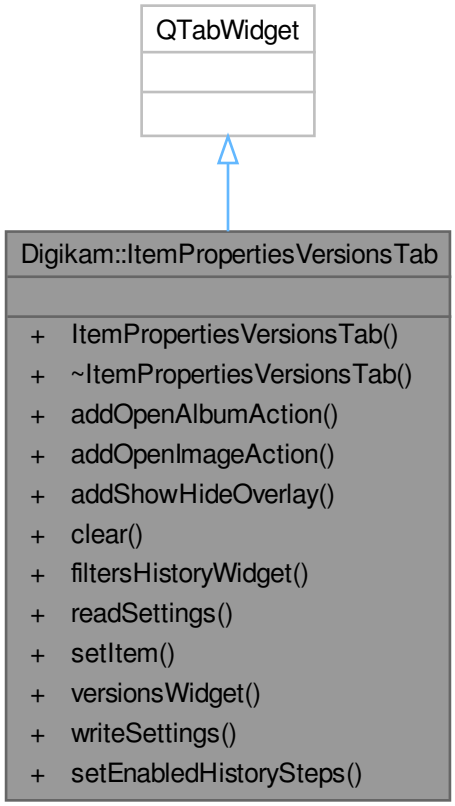
9.857.1.2 shortenedTagPaths()

```
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.

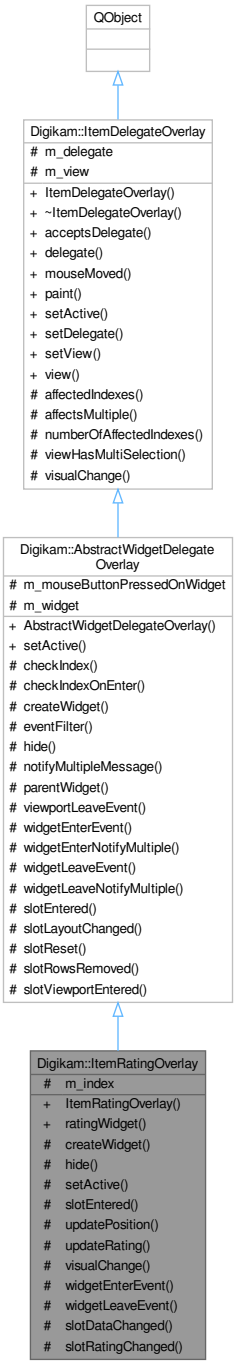
9.861.1.2 checkPosition()

```
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 overlayed 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 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

9.863.1 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

- QTimeLine ∗ **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
- DatabaseItem::Category **category** = DatabaseItem::UndefinedCategory
- qlonglong **fileSize** = 0
- qlonglong **id** = 0
- QString **itemName**
- QDateTime **modificationDate**
- DatabaseItem::Status **status** = DatabaseItem::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 (qlonglong 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 (qlonglong id, QList< qlonglong > *needTaggingIds=nullptr)

  Resolves the image history of the image id by filling the ImageRelations table for all contained referred images.
- static bool **resolveImageHistory** (qlonglong imageId, const QString &historyXml, QList< qlonglong > *needTaggingIds=nullptr)
- static void **tagItemHistoryGraph** (qlonglong 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< qlonglong > **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 (qlonglong 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 (
            qlonglong 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 must↩ BeAvailable 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.

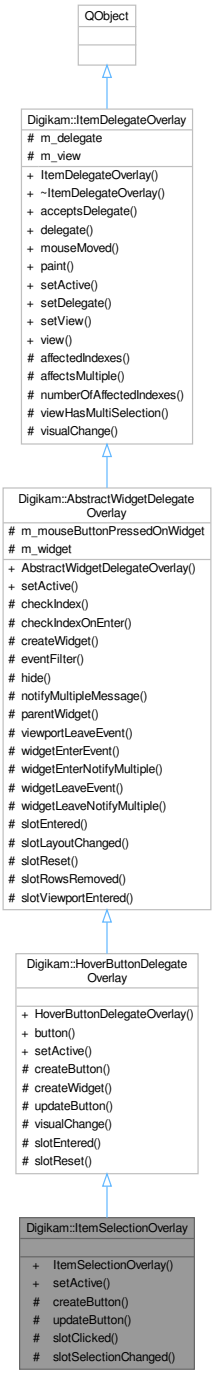
9.866.2.15 sortByProximity()

```
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 QItemSelection &, const QItemSelection &)

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 overlayed 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

- QTimeLine ∗ **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()

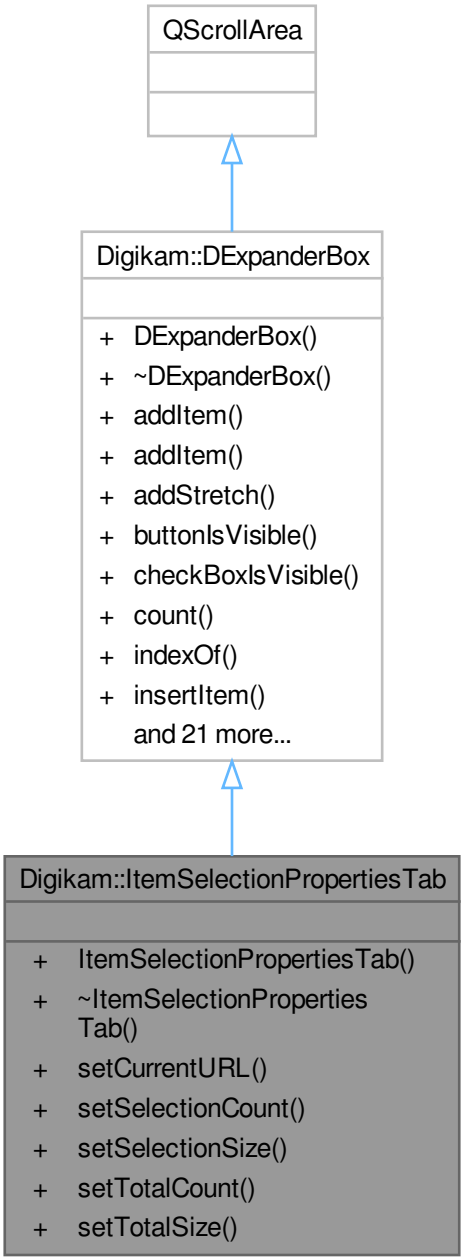
```
void Digikam::ItemSelectionOverlayButton::updateToolTip ( )  [override], [protected], [virtual]
```

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 expand↩ Bydefault)
- 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

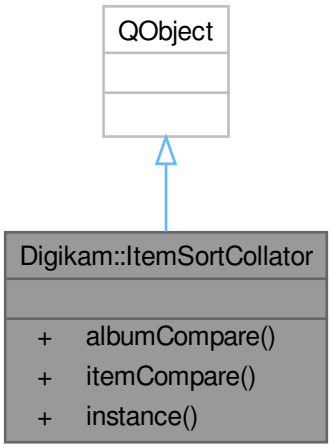
- bool **isNull** () const

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** (qlonglong 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.
- qlonglong **imageId** () const

- bool **isAssigned** () const

  Returns if the tag is assigned to the image.
- bool **isNull** () const
- ItemTagPair & **operator=** (const ItemTagPair &other)
- QMultiMap< 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 (qlonglong 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)

9.873.2.2 availablePairs()

```
QList< ItemTagPair > Digikam::ItemTagPair::availablePairs (
            qlonglong imageId ) [static]
```

This list of tags may not be identical to the tags assigned to the image.

9.874 Digikam::ItemThumbnailBar Class Reference

Inheritance diagram for Digikam::ItemThumbnailBar:

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** (qlonglong 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 **decodeIsCutSelection** (const QMimeData ∗mimeData)
- void **encodeIsCutSelection** (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 **decodeIsCutSelection** (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 **encodeIsCutSelection** (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 **rowsInsertedArtifficial** (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().

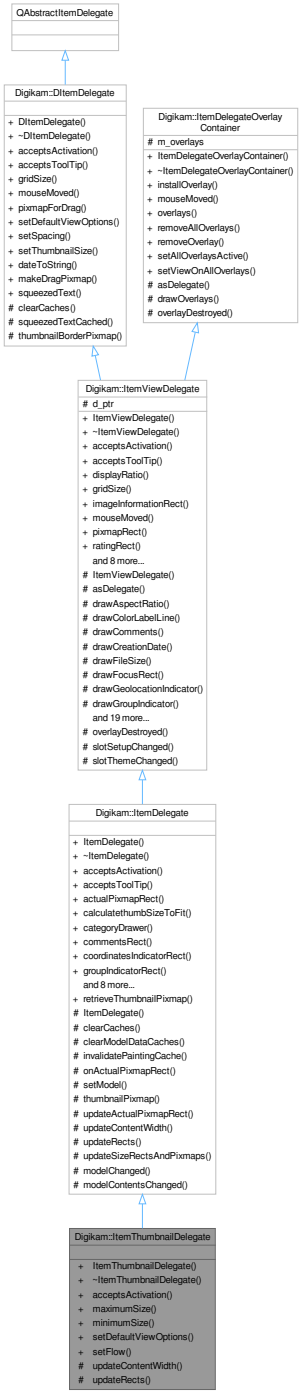
9.874.1.3 slotSetupChanged()

```
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 **drawPanelSideIcon** (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 is↩ Selected) 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 **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::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()

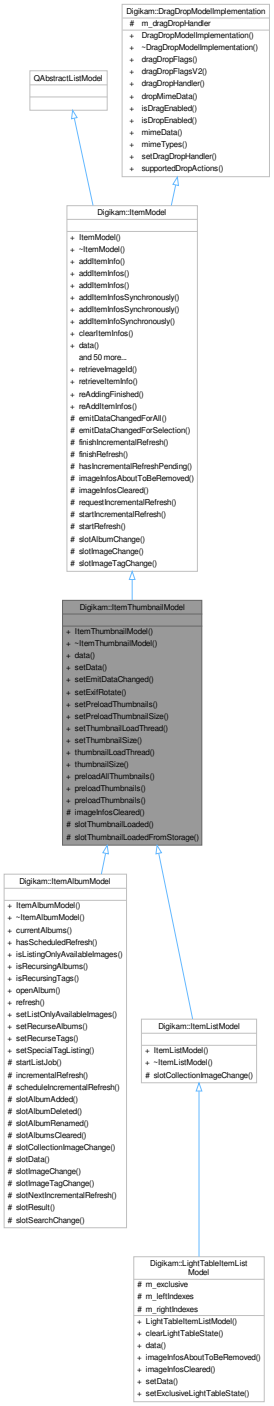
```
void Digikam::ItemThumbnailDelegate::updateRects ( )  [override], [protected], [virtual]
```

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 prepocessor 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** (qlonglong id) const
- bool **hasImage** (qlonglong id, const QVariant &extraValue) const
- QVariant **headerData** (int section, Qt::Orientation orientation, int role=Qt::DisplayRole) const override
- qlonglong **imageId** (const QModelIndex &index) const
- qlonglong **imageId** (int row) const
- QList< qlonglong > **imageIds** () const
- QList< qlonglong > **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** (qlonglong id) const
- QList< QModelIndex > **indexesForItemInfo** (const ItemInfo &info) const
- QList< QModelIndex > **indexesForPath** (const QString &filePath) const
- QModelIndex **indexForImageId** (qlonglong id) const
- QModelIndex **indexForImageId** (qlonglong 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** (qlonglong 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 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

9.876.1 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.

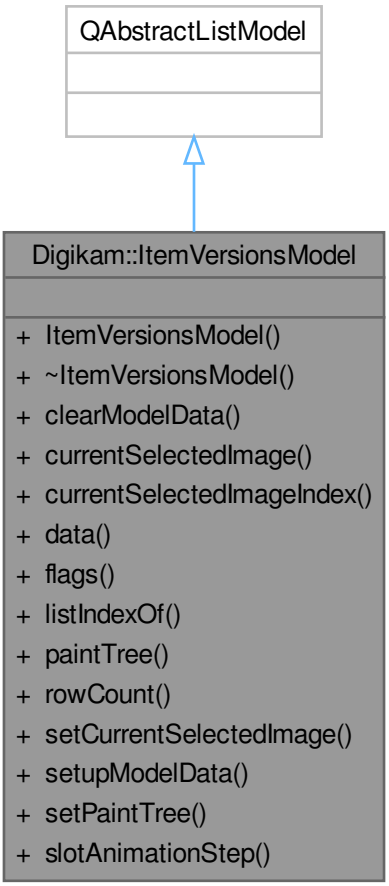
9.876.2.7 setThumbnailLoadThread()

```
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

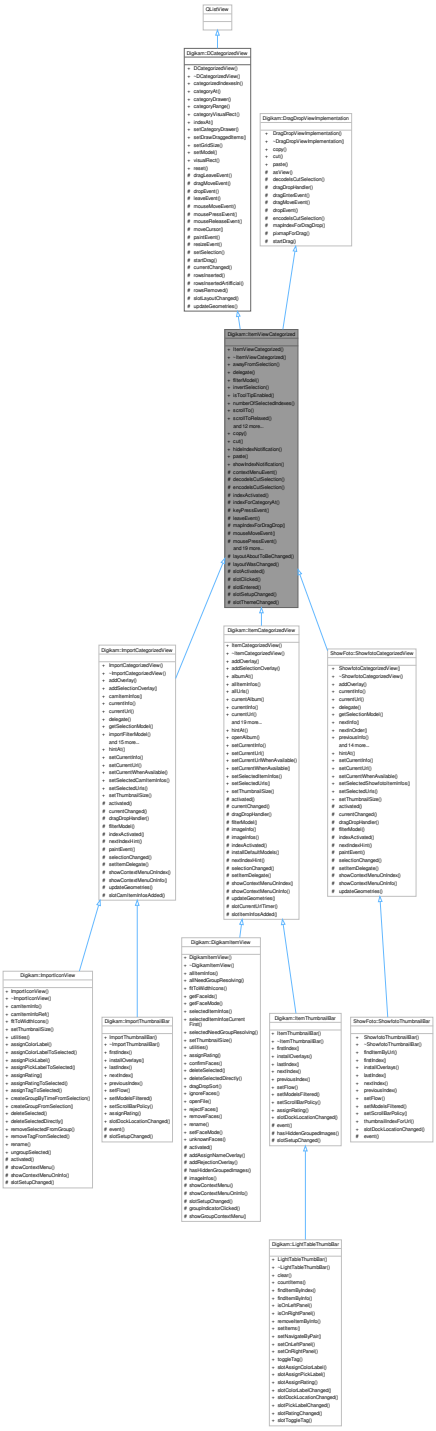
- void **setPaintTree** (bool paint)
- void **slotAnimationStep** ()

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)

9.878 Digikam::ItemViewCategorized Class Reference

Inheritance diagram for Digikam::ItemViewCategorized:



Public Slots

- 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 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 **decodeIsCutSelection** (const QMimeData ∗mimeData)
- void **encodeIsCutSelection** (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 **decodeIsCutSelection** (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 **encodeIsCutSelection** (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.

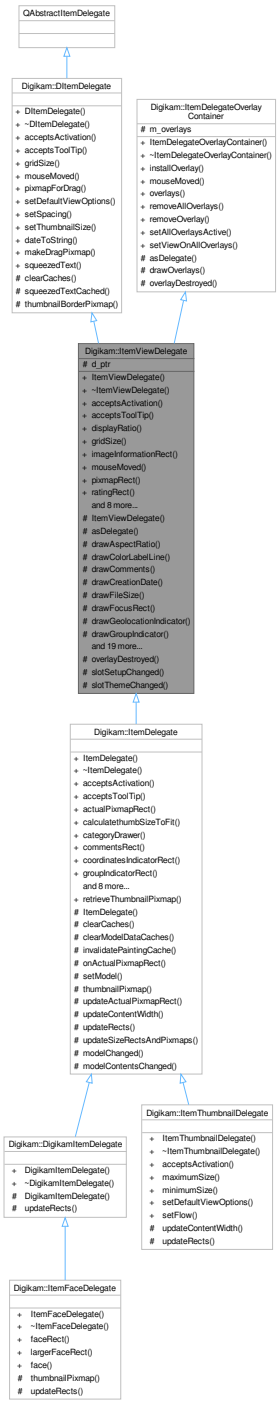
9.878.1.12 showToolTip()

```
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 **drawPanelSideIcon** (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 is↩ Selected) 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 **prepareRatingPixmaps** (bool composeOverBackground=true)
- QPixmap **ratingPixmap** (int rating, bool selected) const

  Returns the relevant pixmap from the cached rating pixmaps.
- 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

- 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.

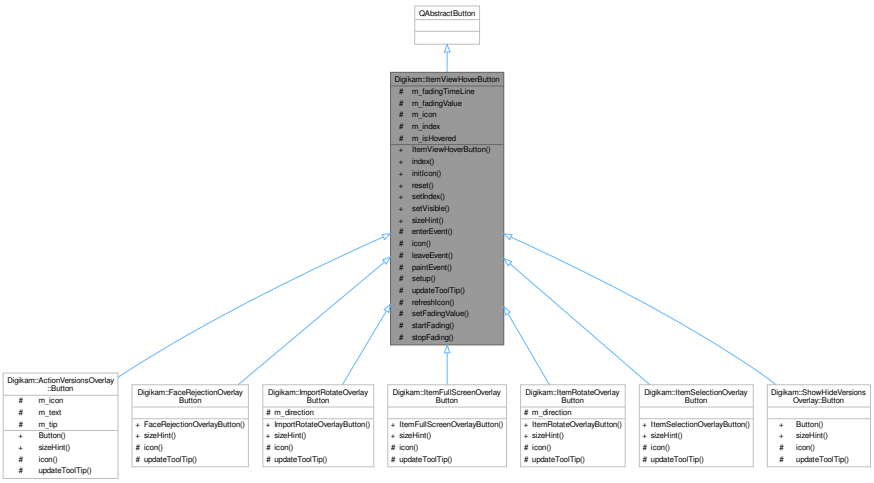
9.879.1.11 setThumbnailSize()

```
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

- QTimeLine ∗ **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()

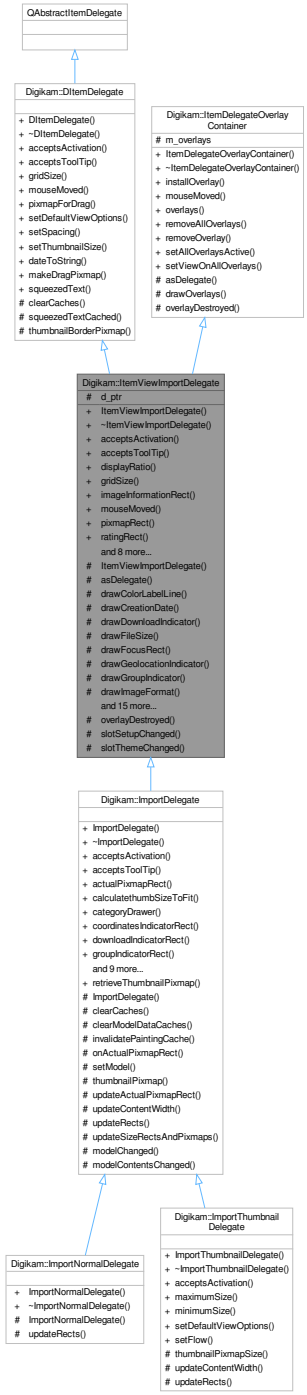
```
void Digikam::ItemViewHoverButton::updateToolTip ( )  [protected], [virtual]
```

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 is↩ Selected) 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 prepareRatingPixmaps (bool composeOverBackground=true)
- QPixmap **ratingPixmap** (int rating, bool selected) const

  Returns the relevant pixmap from the cached rating pixmaps.
- 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

- 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.

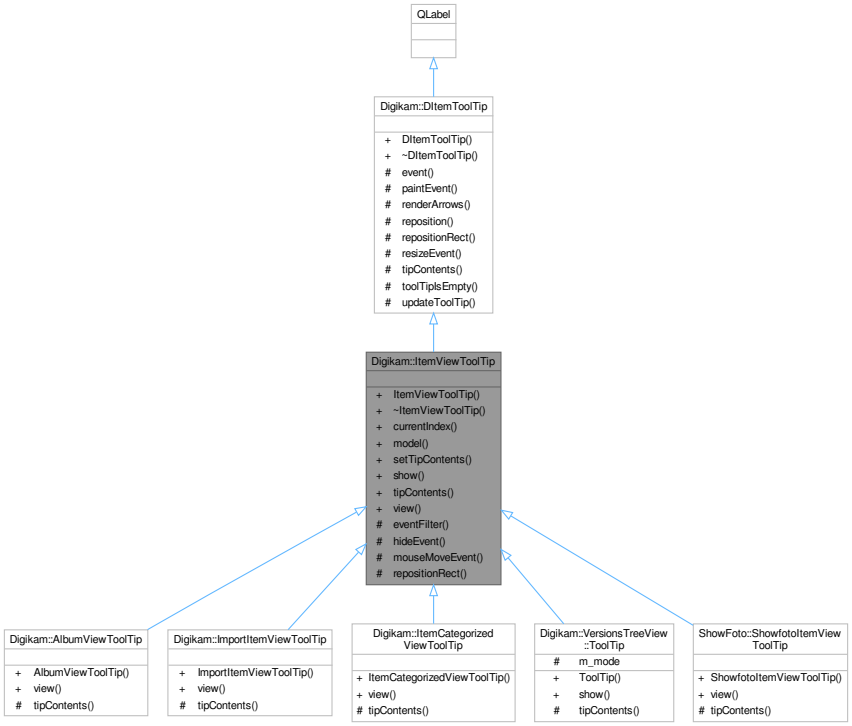
9.881.1.13 setThumbnailSize()

```
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 **toolTipIsEmpty** () 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()

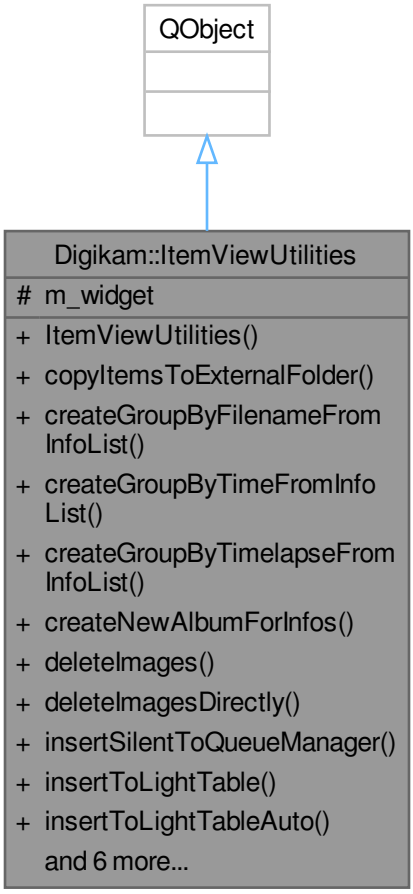
```
QString Digikam::ItemViewToolTip::tipContents ( )  [override], [virtual]
```

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

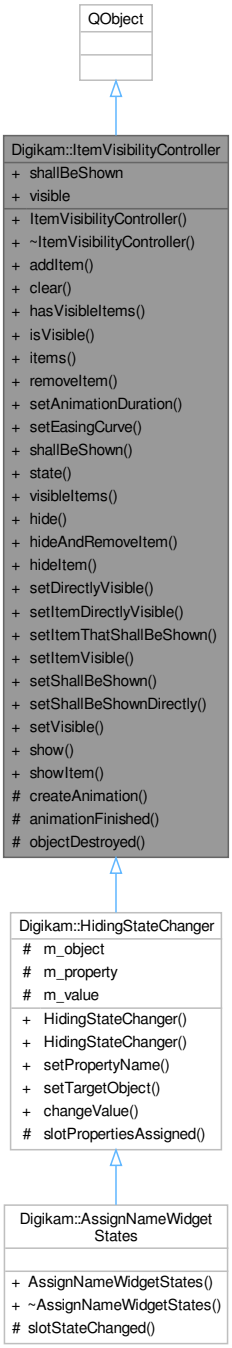
- **ItemViewUtilities** (QWidget *const parentWidget)

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 msecs)
- 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 and animation for "opacity" from 0.0 to 1.0, using default easing curve and duration, which can and will be changed by setEasing↩ Curve 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.

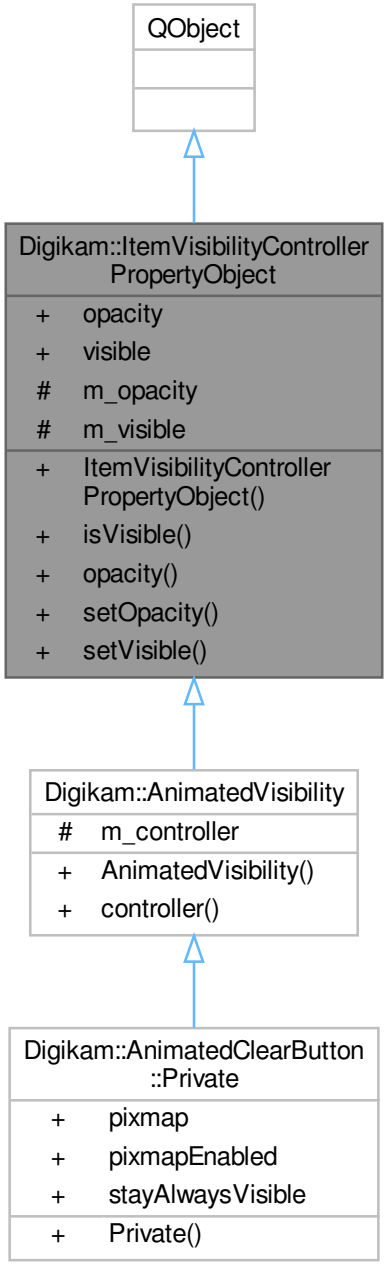
9.884.2.7 showItem

```
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.

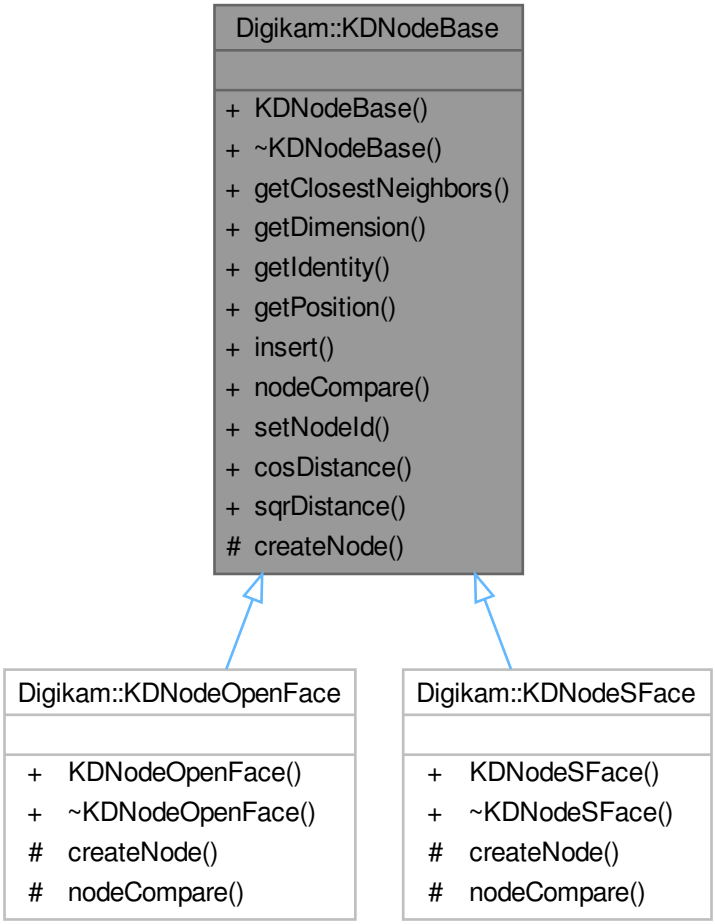
9.887.2.6 setDocumentName()

```
void Digikam::JPEGUtils::JpegRotator::setDocumentName (
            const QString & documentName )
```

Default value is the source's file name

9.888 Digikam::KDNodeBase Class Reference

Inheritance diagram for Digikam::KDNodeBase:



Classes

- struct NodeCompareResult

Public Member Functions

- **KDNodeBase** (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.

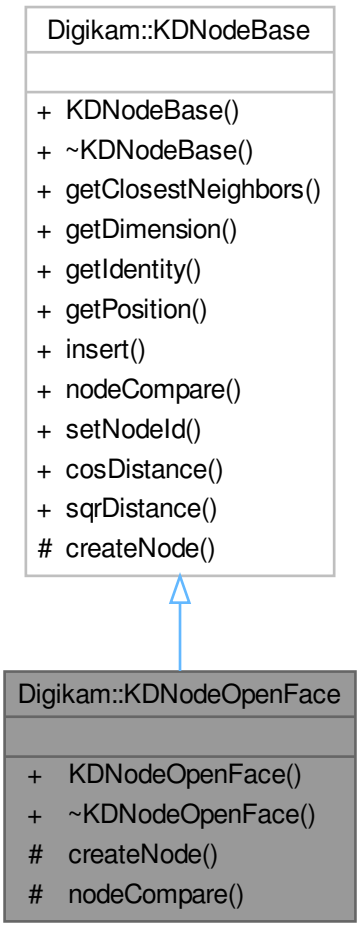
9.889 Digikam::KDNodeBase::NodeCompareResult Struct Reference

Public Attributes

- double **distance1**
- double **distance2**
- bool **result**

9.890 Digikam::KDNodeOpenFace Class Reference

Inheritance diagram for Digikam::KDNodeOpenFace:



Public Member Functions

- **KDNodeOpenFace** (const cv::Mat &nodePos, const int identity, int splitAxis, int dimension)

Public Member Functions inherited from Digikam::KDNodeBase

- **KDNodeBase** (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 &current↩ Position, 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.

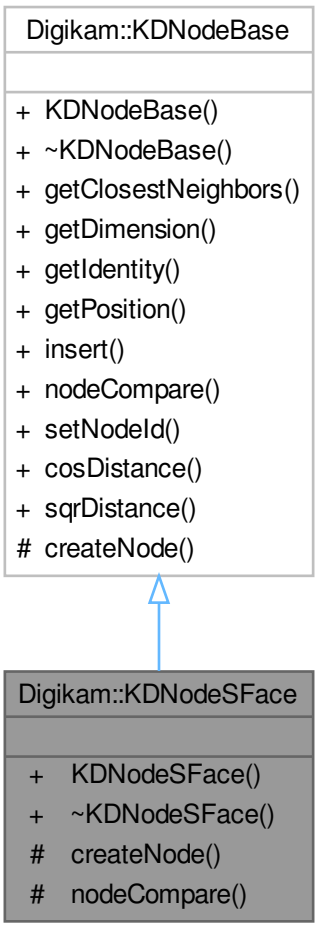
9.890.1.2 nodeCompare()

```
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::KDNodeSFace Class Reference

Inheritance diagram for Digikam::KDNodeSFace:



Public Member Functions

- **KDNodeSFace** (const cv::Mat &nodePos, const int identity, int splitAxis, int dimension)

Public Member Functions inherited from Digikam::KDNodeBase

- **KDNodeBase** (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 &current↩ Position, 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)

9.891.1 Member Function Documentation

9.891.1.1 createNode()

```
KDNodeBase * Digikam::KDNodeSFace::createNode (
            const cv::Mat & nodePos,
            const int identity,
            int splitAxis,
            int dimension )  [override], [protected], [virtual]
```

Implements Digikam::KDNodeBase.

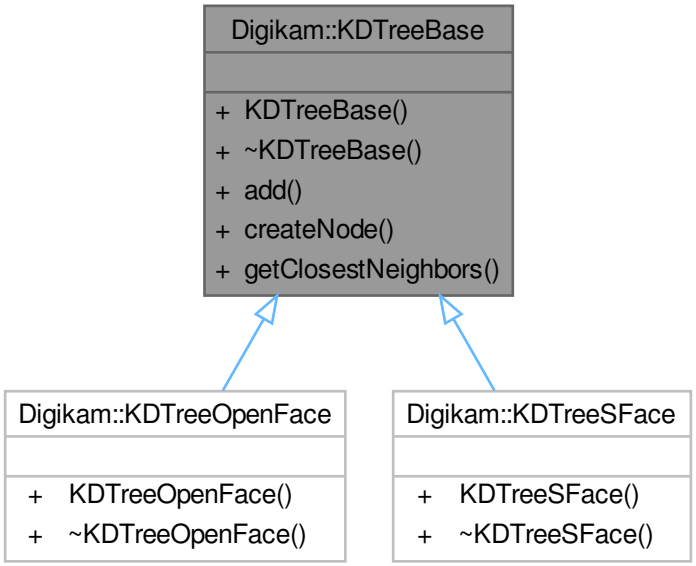
9.891.1.2 nodeCompare()

```
KDNodeBase::NodeCompareResult Digikam::KDNodeSFace::nodeCompare (
            const cv::Mat & queryPosition,
            const cv::Mat & currentPosition,
            float sqRange,
            float cosThreshold,
            int nbDimension ) const  [override], [protected], [virtual]
```

Implements Digikam::KDNodeBase.

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 an ew 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

dim	The dimmension of the tree.
kdtreeThreshold	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 suitabe 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

position	K-dimension vector
identity	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

nodePos	extracted face vectors
identity	identity of this face vector
splitAxis	current axis/dimension of the vector
dimension	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,
```

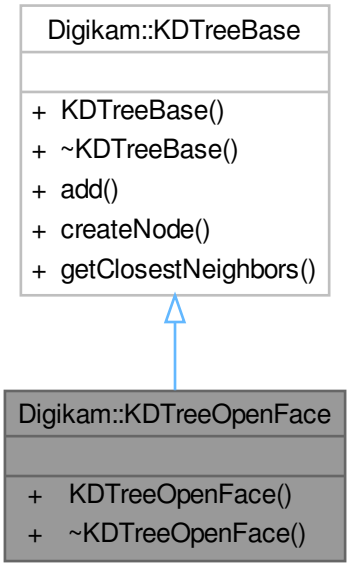
```
float sqRange,
int maxNbNeighbors ) const  [virtual]
```

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

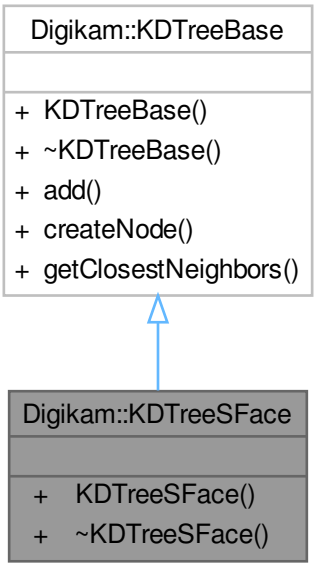
- KDTreeBase (int dim, int kdtreeThreshold=KDTREE_MAP_THRESHOLD)

  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

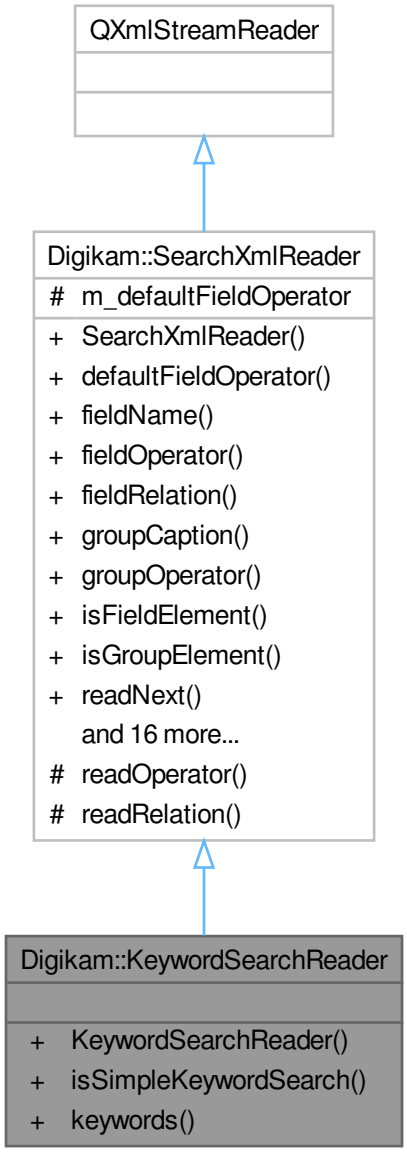
- KDTreeBase (int dim, int kdtreeThreshold=KDTREE_MAP_THRESHOLD)

  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

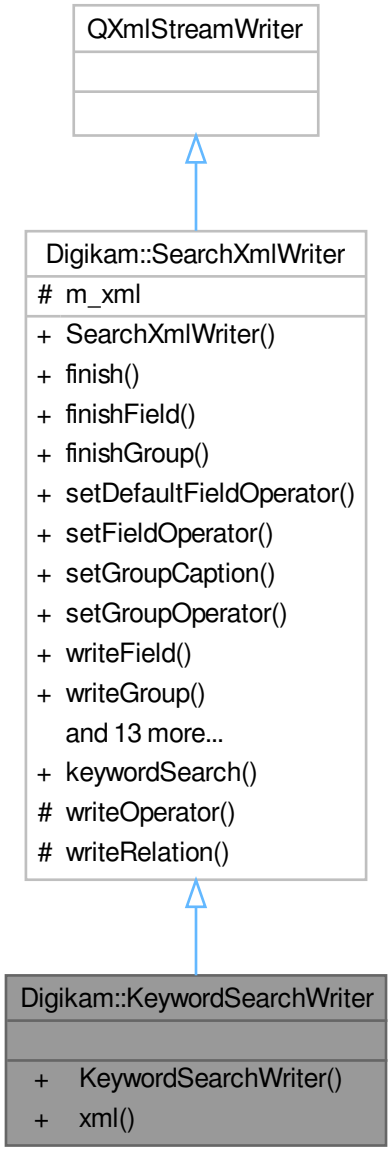
- SearchXml::Operator **readOperator** (const QString &, SearchXml::Operator) const
- SearchXml::Relation **readRelation** (const QString &, SearchXml::Relation) const

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

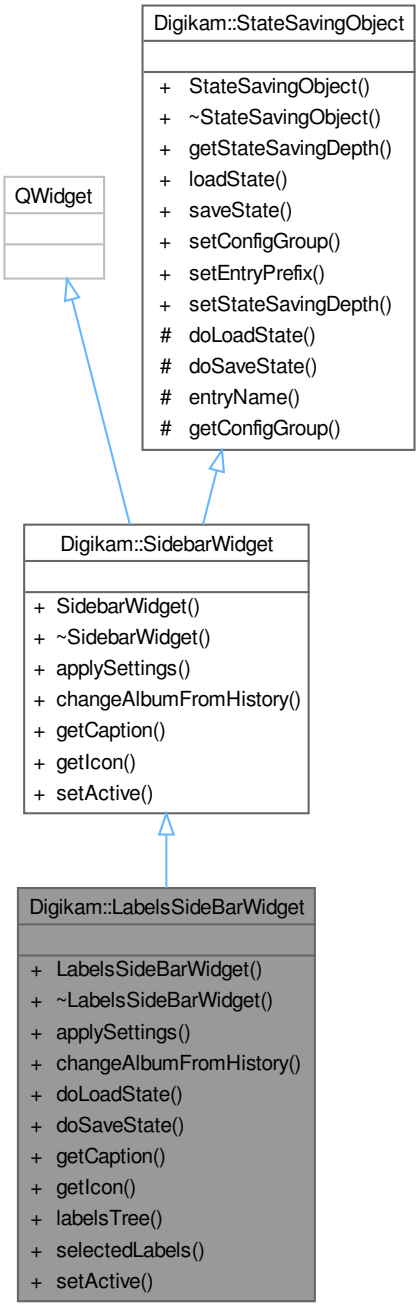
- void **writeOperator** (const QString &, SearchXml::Operator)
- void **writeRelation** (const QString &, SearchXml::Relation)

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]
```

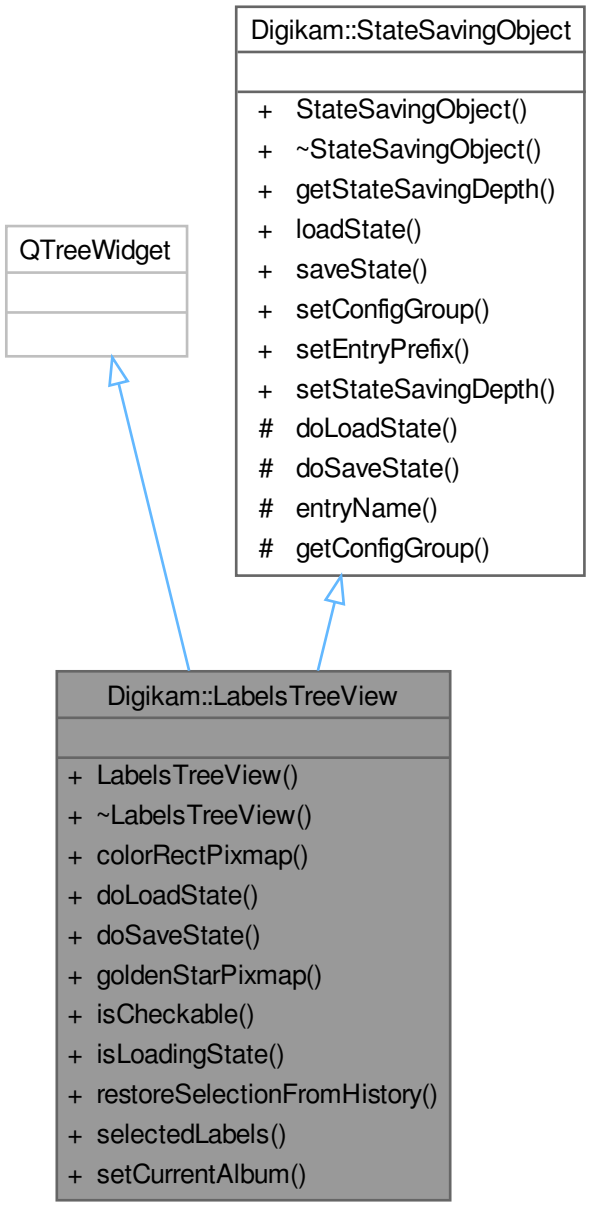
Parameters

active	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

color	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 Pixmap 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

neededLabels	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()

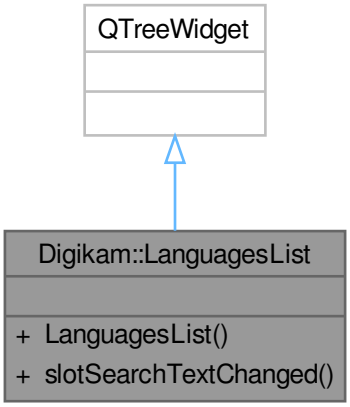
```
QHash< LabelsTreeView::Labels, QList< int > > Digikam::LabelsTreeView::selectedLabels ( )
```

Returns

a QHash 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 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** ()

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

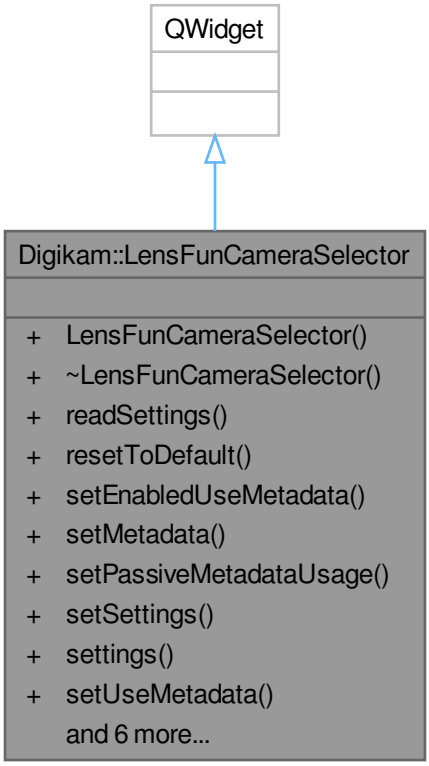
- void **cubicInterpolate** (uchar ∗src, int rowStride, uchar ∗dst, bool sixteenBit, double dx, double dy, double brighten)
- uchar ∗ **pixelAccessAddress** (int i, int j)
- void **pixelAccessDoEdge** (int i, int j)
- void **pixelAccessReposition** (int xInt, int yInt)
- void **pixelAccessSelectRegion** (int n)

9.902.1 Detailed Description

Assuming that accesses are at least slightly coherent, LensDistortionPixelAccess keeps LensDistortionPixel↩ AccessRegions 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

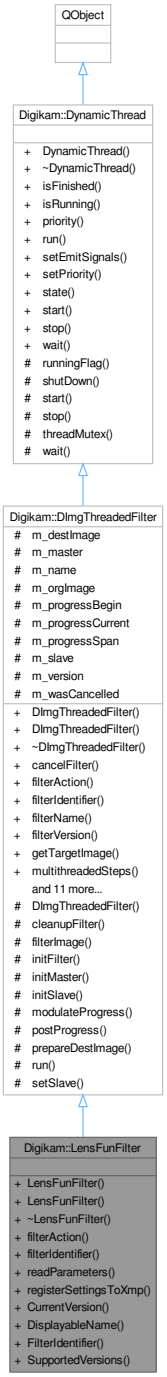
  Distortion Corrections.
- bool **filterGEO** = true

  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 **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

9.905.1 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::LensFunIface Class Reference

Public Types

- typedef const lfCamera ∗ **DevicePtr**
- typedef QList< LensPtr > **LensList**
- typedef const lfLens ∗ **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 lfCamera ∗const ∗ **lensFunCameras** () const
- lfDatabase ∗ **lensFunDataBase** () const
- QString **makeDescription** () const

  Return Camera maker string description found in metadata.
- QString **modelDescription** () const

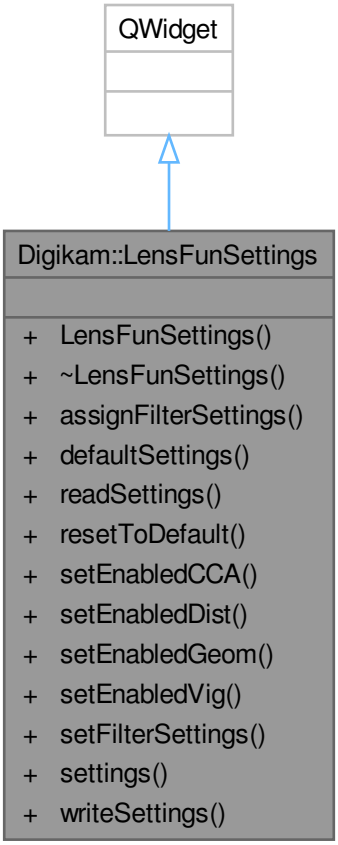
  Return Camera model string description found in metadata.
- void **setFilterSettings** (const LensFunContainer &other)
- void **setSettings** (const LensFunContainer &other)
- LensFunContainer **settings** () const
- void **setUsedCamera** (DevicePtr cam)
- void **setUsedLens** (LensPtr lens)
- bool **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)

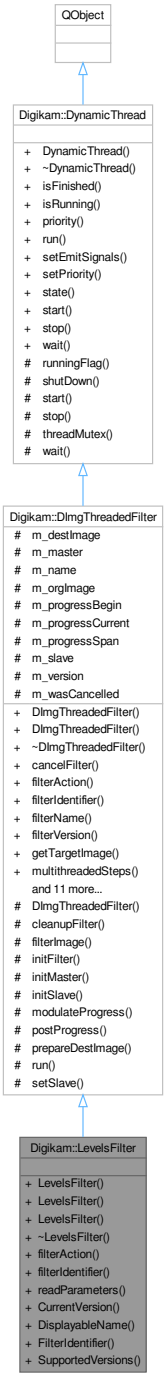
9.908 Digikam::LevelsContainer Class Reference

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 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** ()

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.

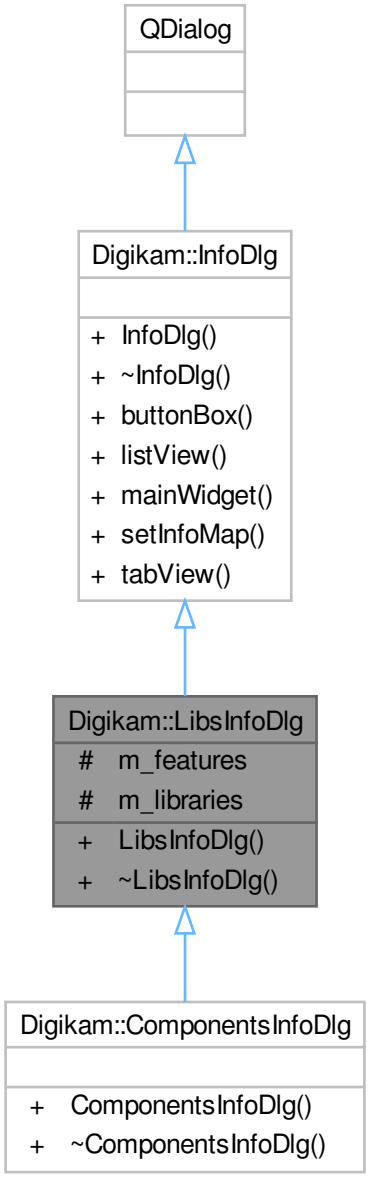
9.909.1.3 readParameters()

```
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

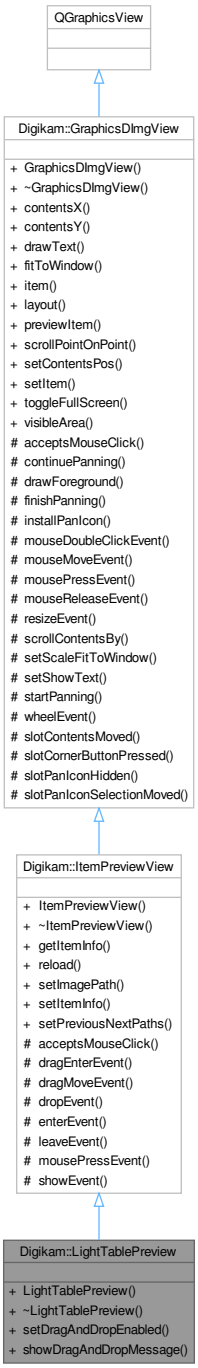
9.910.1.1 LibsInfoDlg()

```
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 acceptsMouseClick (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** (qlonglong 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 **rowsInsertedArtifficial** (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 **decodeIsCutSelection** (const QMimeData *mimeData)
- void **encodeIsCutSelection** (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 **decodeIsCutSelection** (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 **encodeIsCutSelection** (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

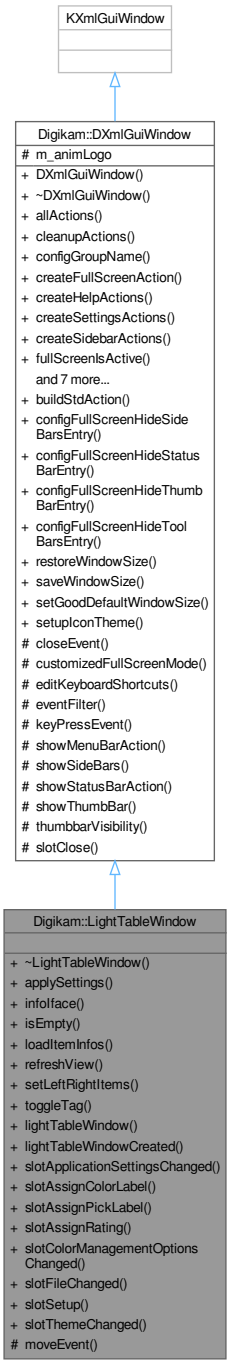
- void **signalDeleteItem** (const ItemInfo &)
- void **signalEditItem** (const ItemInfo &)
- void **signalLeftDroppedItems** (const ItemInfoList &)
- void **signalLeftPanelLeftButtonClicked** ()
- void **signalLeftPopupTagsView** ()
- void **signalLeftPreviewLoaded** (bool)
- void **signalLeftPreviewSelected** (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 * infoIface (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 **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 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 )
```

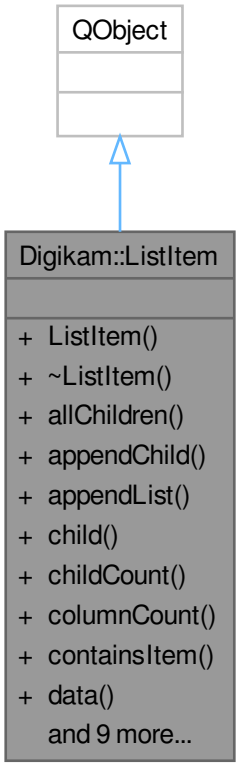
- via CTRL+L (from the albumview) a) digikamapp.cpp: CTRL+key_L leads to slotImageLightTable()) b) digikamview.cpp: void ItemIconView::slotImageLightTable() calls d->iconView->insertToLightTable(list, info); c) albumiconview.cpp: AlbumIconView::insertToLightTable calls ltview->loadItemInfos(list, current);
- 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

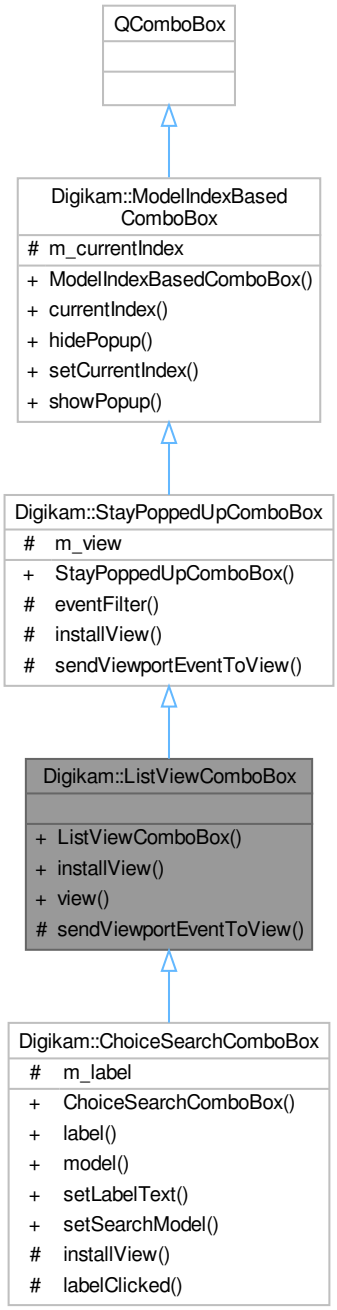
item	- ListItem pointer for which we should search if there is a similar item

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.

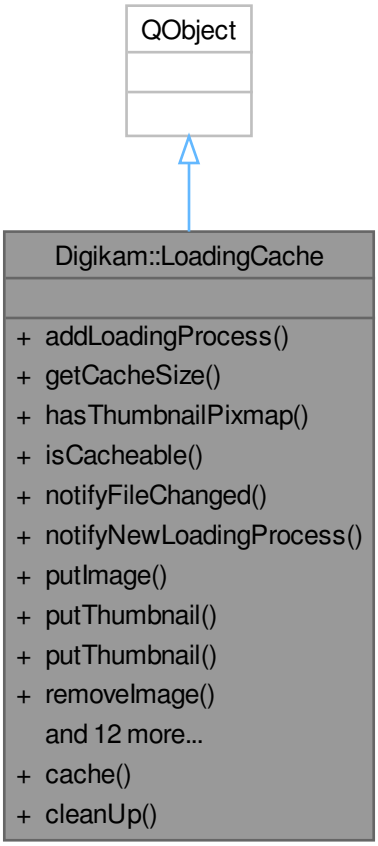
9.916.2.3 view()

```
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
```

QPixmaps 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

numberOfQImages	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.
numberOfQPixmaps	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** ()

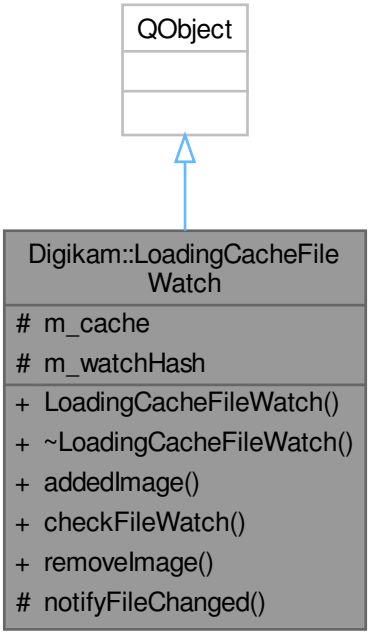
9.918.1 Detailed Description

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 raw↩DecodingHint=RawDecodingCustomSettings, ColorManagementSettings=NoColorConversion)

  Use this for full loading of raw files.
- LoadingDescription (const QString &filePath, const PreviewSettings &settings, int size, ColorManagementSettings=No↩ColorConversion, 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 IccTransform.
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

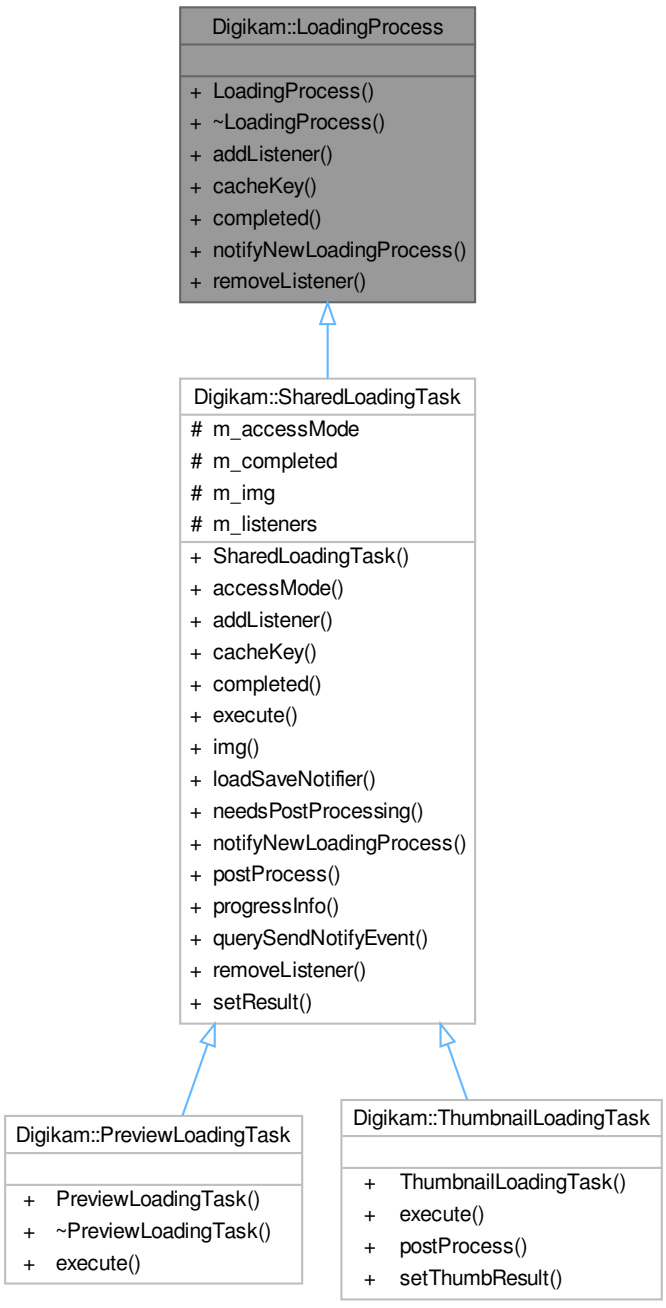
- bool **onlyFromStorage** () const
- bool **onlyPregenerate** () const
- bool **operator==** (const PreviewParameters &other) const

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

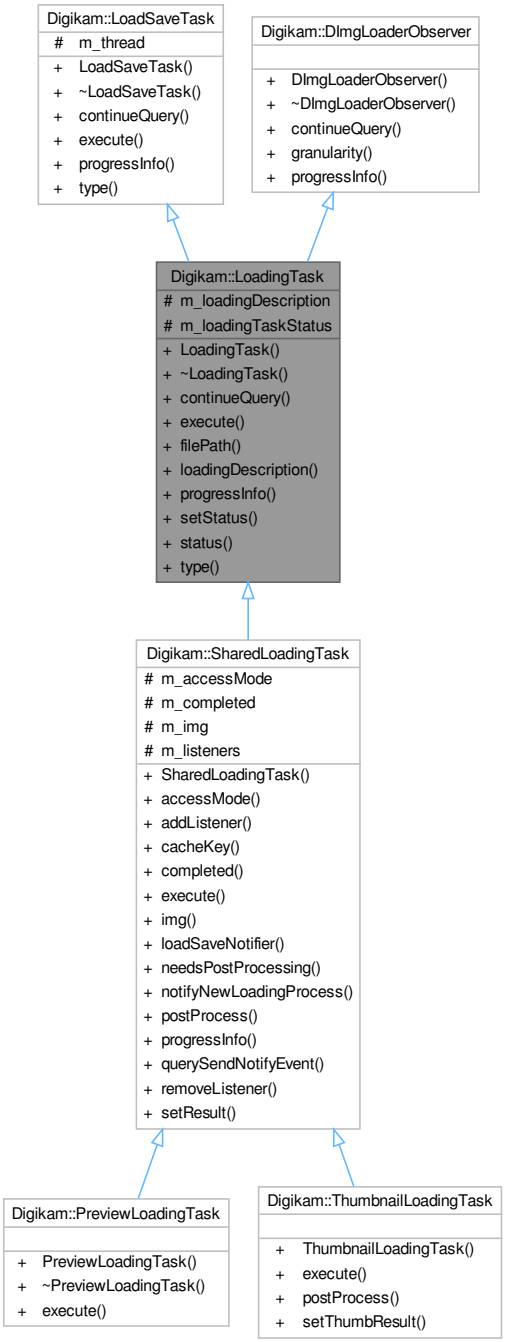
Inheritance diagram for Digikam::LoadingProcessListener:

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.

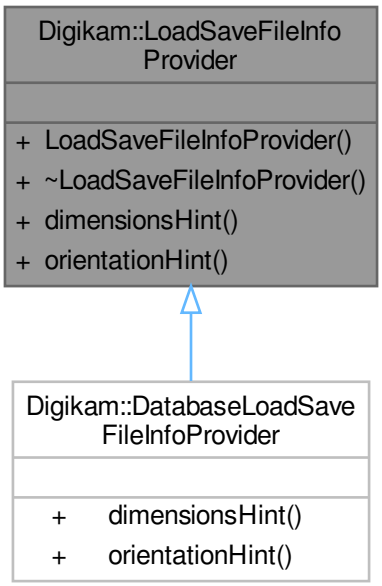
9.926.1.4 type()

```
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()

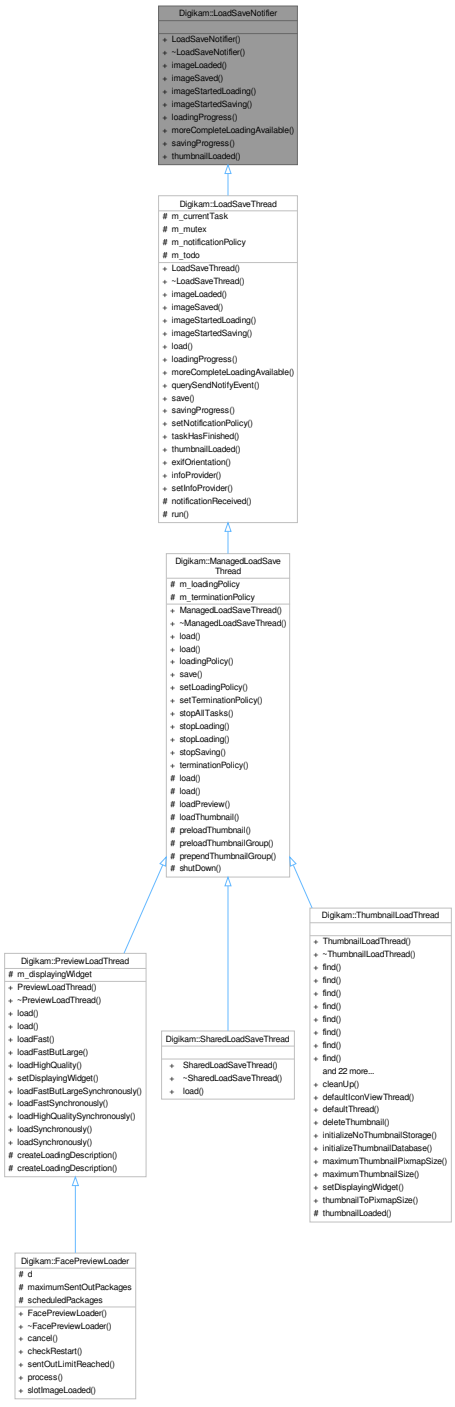
```
virtual int Digikam::LoadSaveFileInfoProvider::orientationHint (
            const QString & path )  [pure virtual]
```

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

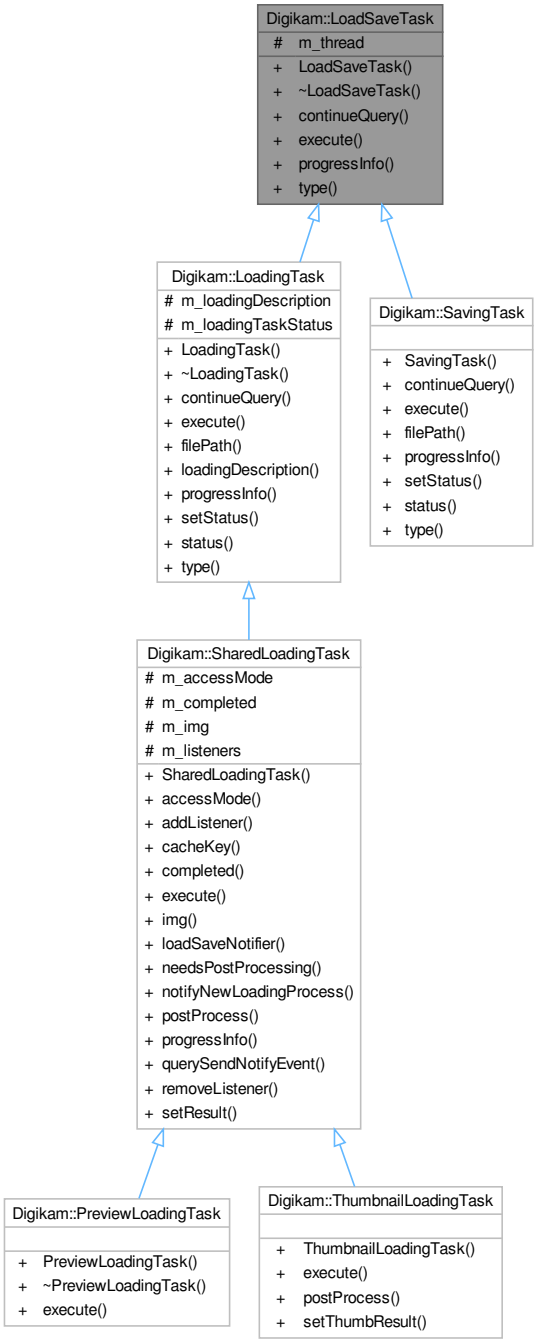
9.928.1.1 thumbnailLoaded()

```
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 DImg & 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 **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

9.932.1 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.

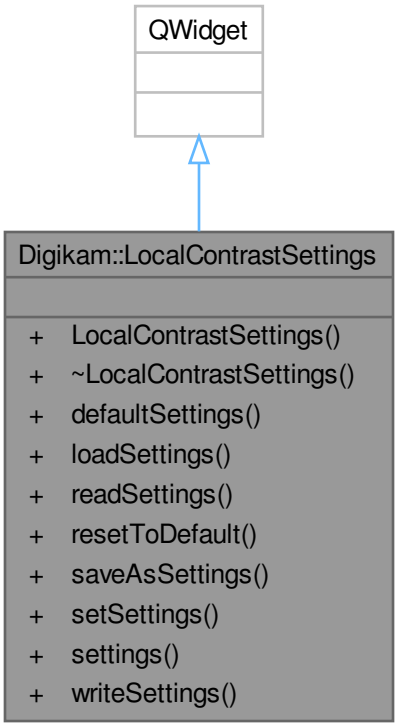
9.932.1.3 readParameters()

```
void Digikam::LocalContrastFilter::readParameters (
            const FilterAction & action )  [override], [virtual]
```

Implements Digikam::DImgThreadedFilter.

9.933 Digikam::LocalContrastSettings Class Reference

Inheritance diagram for Digikam::LocalContrastSettings:



Signals

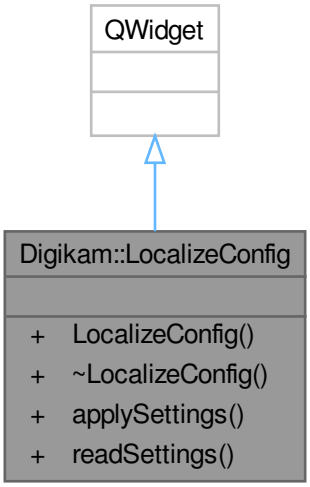
- void **signalSettingsChanged** ()

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

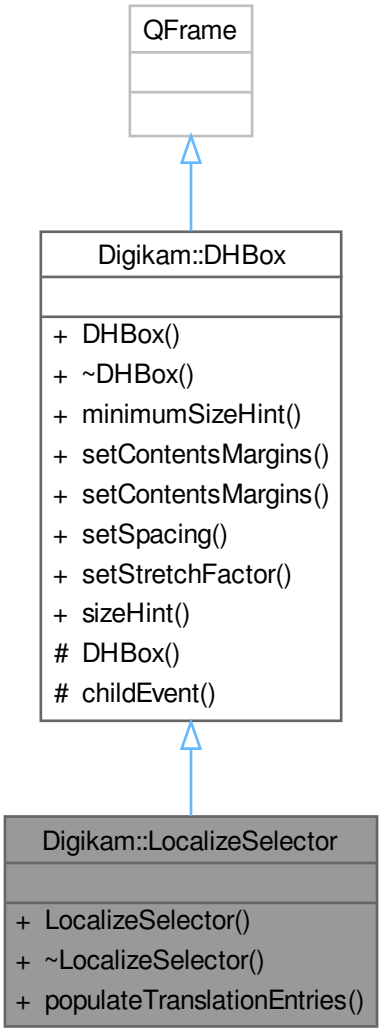
9.935.1.1 ignoredWords

```
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

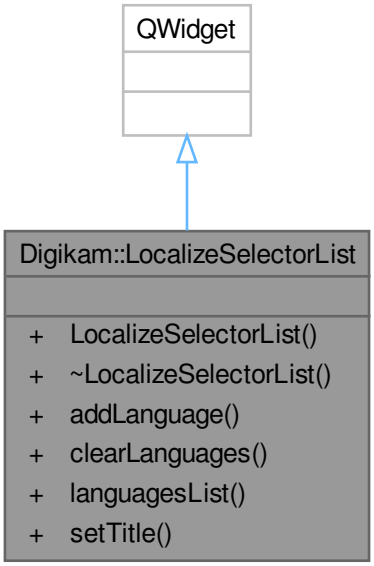
Additional Inherited Members

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

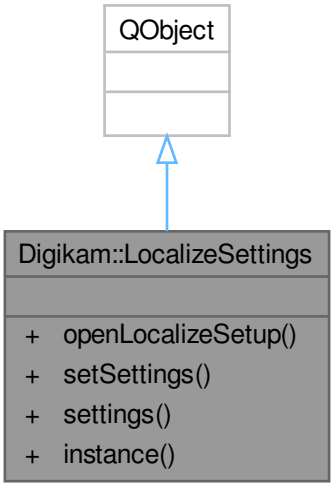
- void **signalSettingsChanged** ()

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

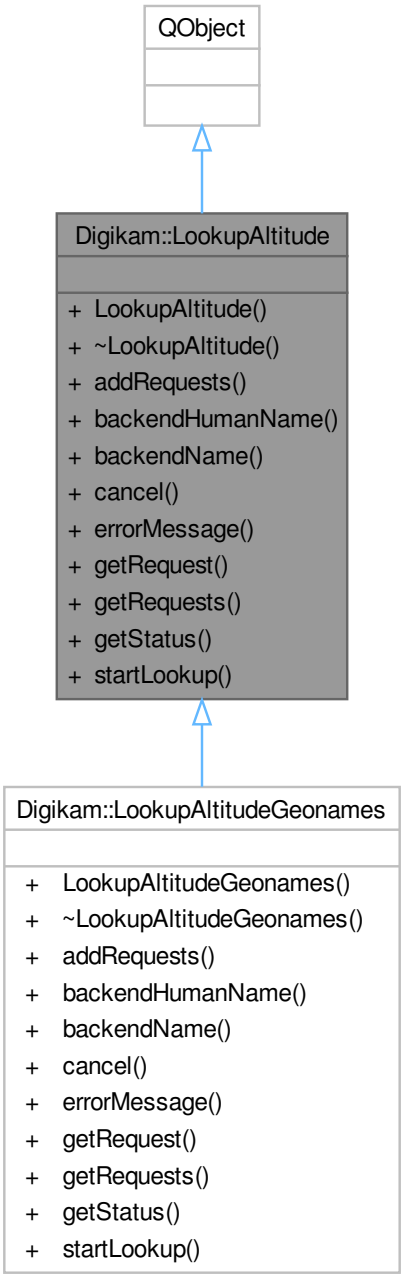
9.938.1.1 instance()

```
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

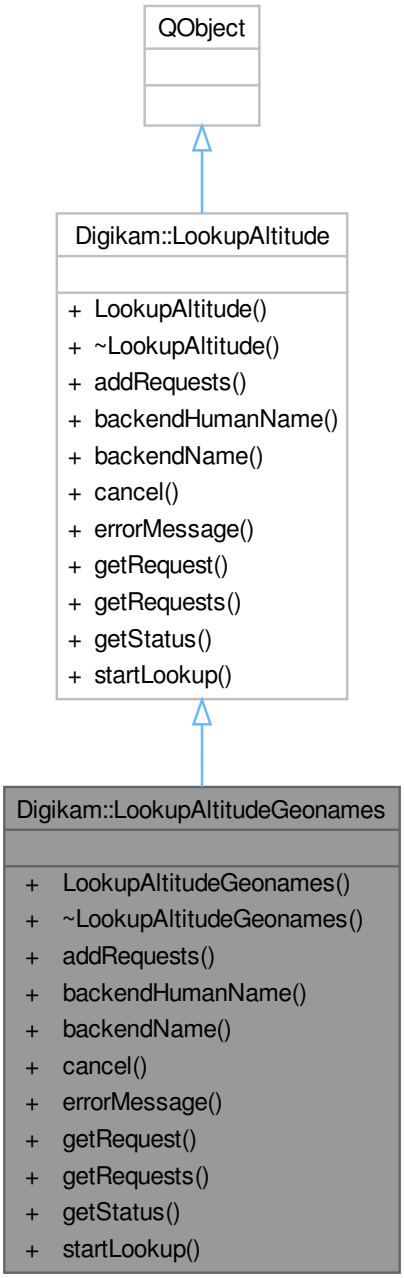
- typedef QList< Request > **List**

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)

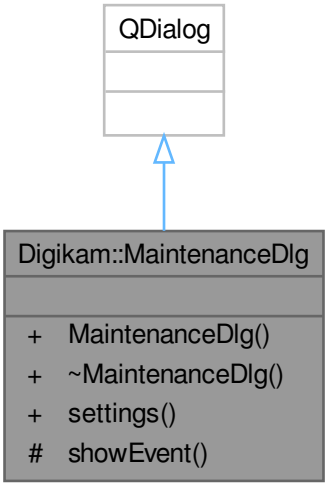
9.943 Digikam::MaintenanceData Class Reference

Public Member Functions

- Identity **getIdentity** () const
- qlonglong **getImageId** () const
- QString **getImagePath** () const
- ItemInfo **getItemInfo** () const
- qlonglong **getSimilarityImageId** () const
- int **getThumbnailId** () const
- void **setIdentities** (const QList< Identity > &identities)
- void **setImageIds** (const QList< qlonglong > &ids)
- void **setImagePaths** (const QList< QString > &paths)
- void **setItemInfos** (const QList< ItemInfo > &infos)
- void **setSimilarityImageIds** (const QList< qlonglong > &ids)
- void **setThumbnailIds** (const QList< int > &ids)

9.944 Digikam::MaintenanceDlg Class Reference

Inheritance diagram for Digikam::MaintenanceDlg:



Public Member Functions

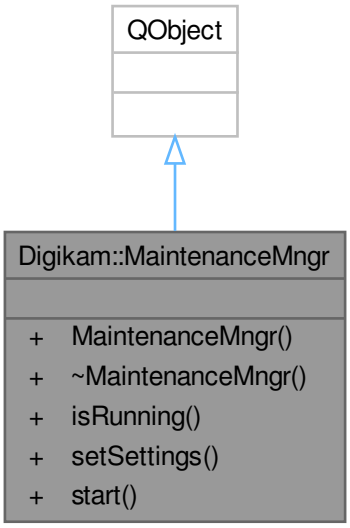
- **MaintenanceDlg** (QWidget *const parent=nullptr)
- MaintenanceSettings **settings** () const

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.
- HaarIface::DuplicatesSearchRestrictions **duplicatesRestriction** = HaarIface::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

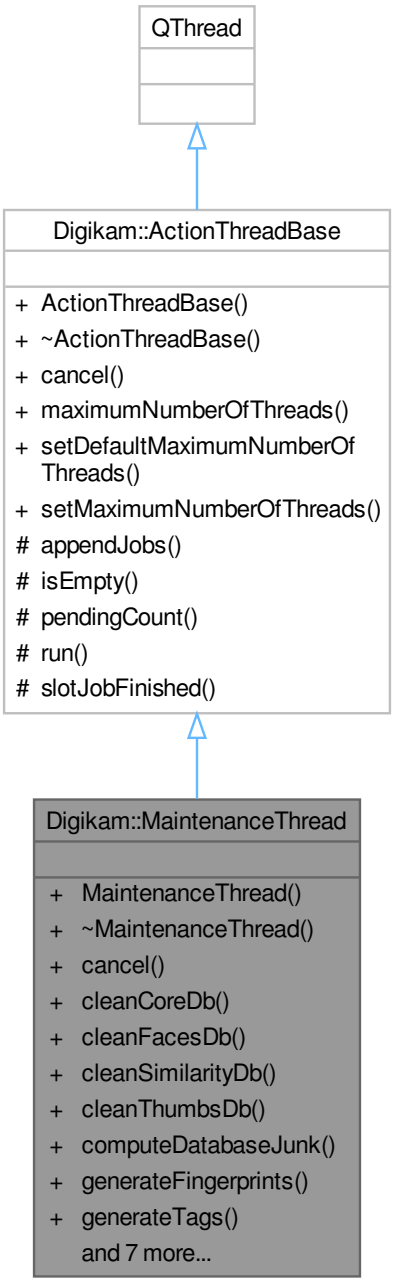
```
int Digikam::MaintenanceSettings::qualityScanMode = ImageQualitySorter::AllItems
```

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 > &thumbnailIds)
- 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

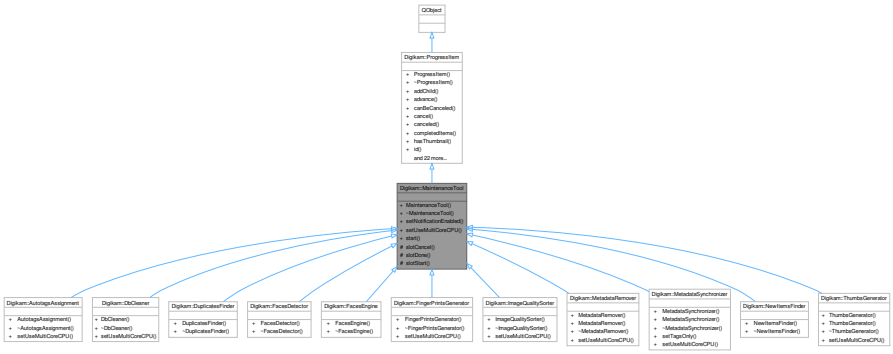
9.947.1.1 signalAdvance

```
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 elpased 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

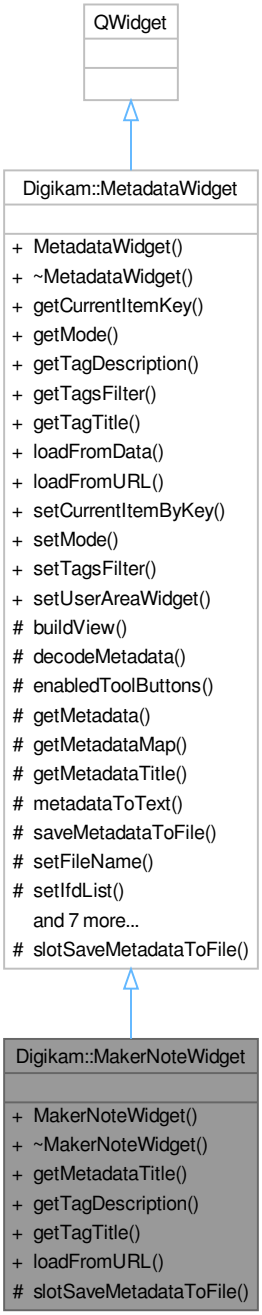
9.948.1.1 setUseMultiCoreCPU()

```
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::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

9.949.1 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.

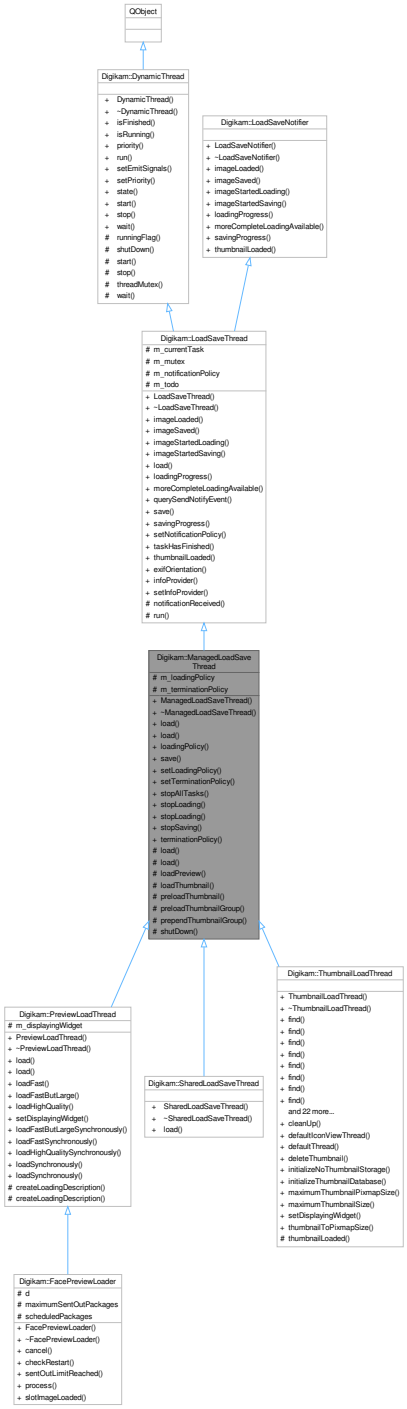
9.949.1.4 loadFromURL()

```
bool Digikam::MakerNoteWidget::loadFromURL (
            const QUrl & url )  [override], [virtual]
```

Implements Digikam::MetadataWidget.

9.950 Digikam::ManagedLoadSaveThread Class Reference

Inheritance diagram for Digikam::ManagedLoadSaveThread:



Public Types

- 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 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 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

- 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

LoadingModeNormal	no sharing of loading process, no caching of image
LoadingModeShared	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 load() operation from LoadSaveThread, 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.

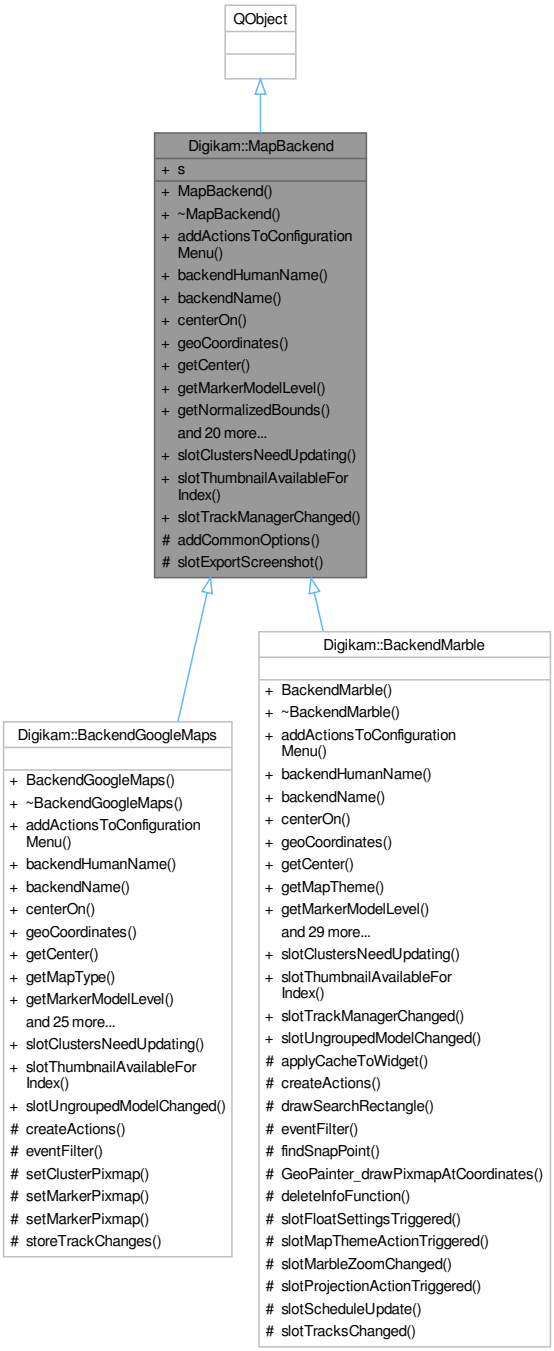
9.950.2.4 stopSaving()

```
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 QIntList &clusterIndices)
- void **signalClustersMoved** (const QIntList &clusterIndices, const QPair< int, QModelIndex > &snapTarget)
- void **signalMarkersMoved** (const QIntList &markerIndices)
- void **signalSelectionHasBeenMade** (const Digikam::GeoCoordinates::Pair &coordinates)
- void **signalZoomChanged** (const QString &newZoom)

Public Member Functions

- **MapBackend** (const QExplicitlySharedDataPointer< GeoIfaceSharedData > &sharedData, QObject *const parent)
- virtual void **addActionsToConfigurationMenu** (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** (GeoIfaceInternalWidgetInfo *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< GeoIfaceSharedData > **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.

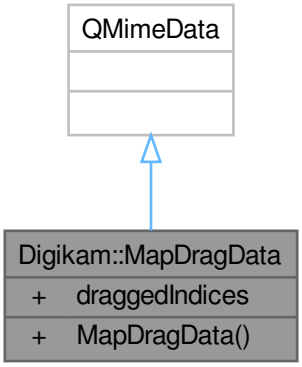
9.951.1.4 setActive()

```
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:

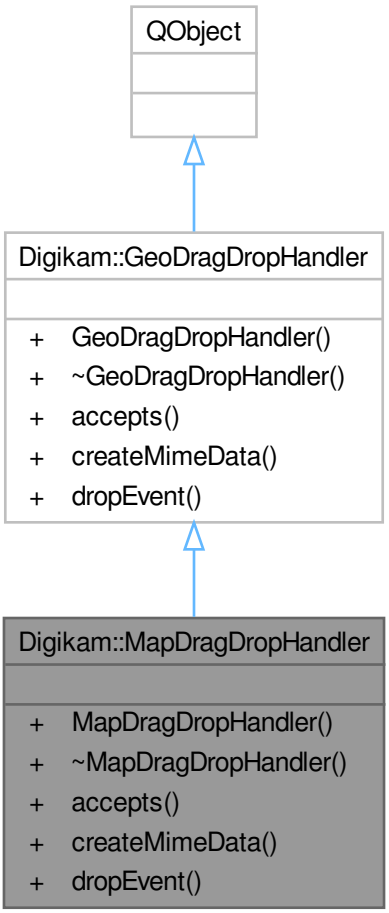


Public Attributes

- QList< QPersistentModelIndex > **draggedIndices**

9.953 Digikam::MapDragDropHandler Class Reference

Inheritance diagram for Digikam::MapDragDropHandler:



Public Member Functions

- **MapDragDropHandler** (QAbstractItemModel *const, GPSGeoIfaceModelHelper *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.

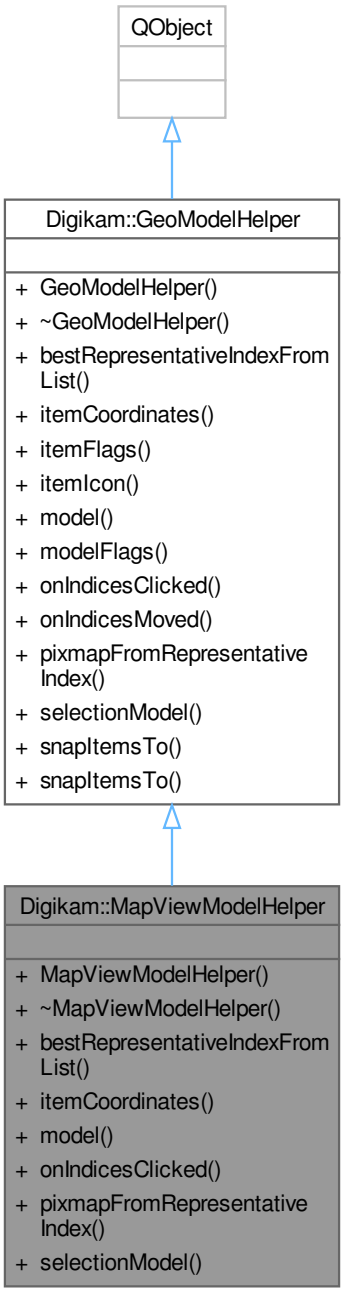
9.953.1.3 dropEvent()

```
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

list	A list containing markers.
sortKey	Determines the sorting options and is actually of type GPSItemInfoSorter::SortOptions

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

index	Current model index.
coordinates	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

clickedIndices	A list containing the marker indices belonging 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

index	The marker's index.
size	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

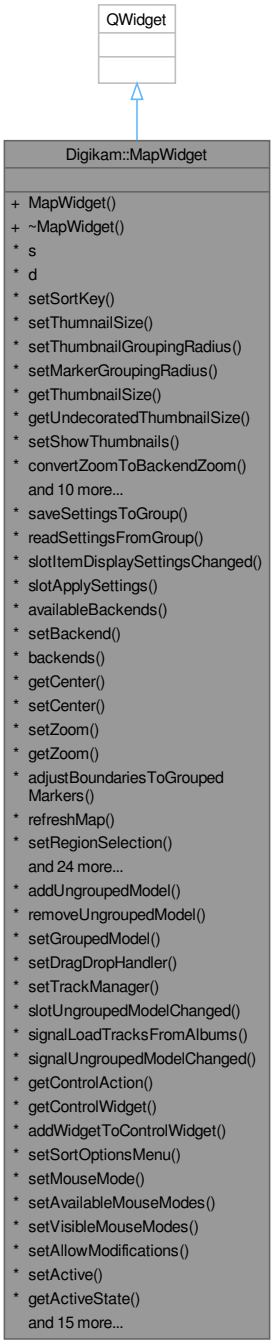
Returns digiKam's selection model.

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 setThumnailSize (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 ∗stroke↩ Style, 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 ∗stroke↩ Color, 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↩ Override, 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 QIntList &clusterIndices, const QPair< int, QModelIndex > &snapTarget)

- void slotClustersClicked (const QIntList &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

useSaneZoomLevel	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

clusterIndex	Index of the cluster
fillColor	Color used to fill the circle
strokeColor	Color used for the stroke around the circle
strokeStyle	Style used to draw the stroke around the circle
labelText	Text for the label
labelColor	Color for the label text
overrideSelection	Get the colors for a different selection state
overrideCount	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 setThumnailSize()

```
void Digikam::MapWidget::setThumnailSize (
            const int newThumbnailSize )
```

9.955.3.14 slotClustersClicked

```
void Digikam::MapWidget::slotClustersClicked (
            const QIntList & clusterIndices )  [protected], [slot]
```

9.955.3.15 slotClustersMoved

```
void Digikam::MapWidget::slotClustersMoved (
            const QIntList & 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]
```

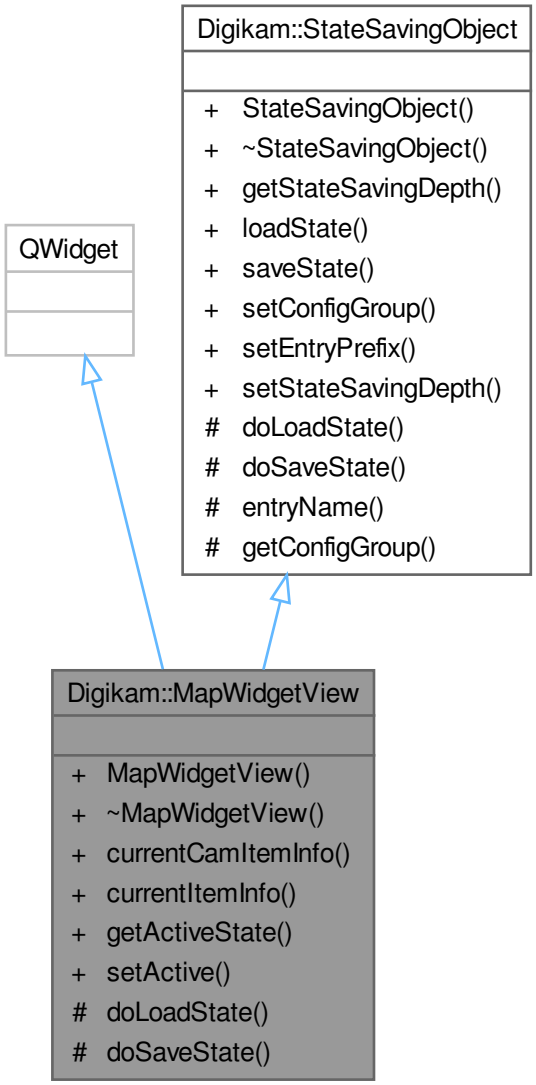
9.955.3.20 updateClusters()

```
void Digikam::MapWidget::updateClusters ( )
```

9.956 Digikam::MapWidgetView Class Reference

Class containing digiKam's central map view.

Inheritance diagram for Digikam::MapWidgetView:



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,
            DCategorizedSortFilterProxyModel *const imageFilterModel,
            QWidget *const parent,
            const Application application )  [explicit]
```

Parameters

selectionModel	digiKam's selection model
imageFilterModel	digikam's filter model
parent	the parent object
application	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

state	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**

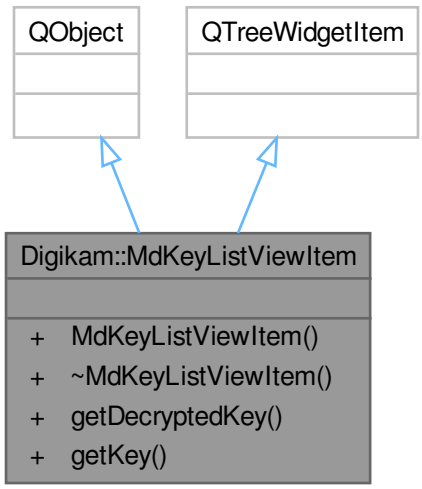
  Number of rows in the matrix.

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:

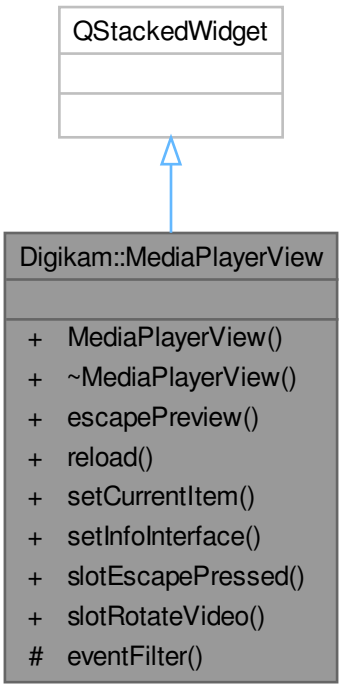


Public Member Functions

- **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 MetaEngineSettingsContaine
  &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 ignoreLazy↩ Sync=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. • metadata fields changed • 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.
PartialWrite	Write only the changed parts. Metadata fields which cannot be changed from MetadataHub (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

toClean	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

writeMode	The mode to write metadata
settings	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

image	The DImg container to retrieve current tags
writeMode	The mode to write metadata
ignoreLazySync	The flag to ignore the lazy sync metadata stage
settings	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

filePath	The file path to retrieve current tags
writeMode	The mode to write metadata
ignoreLazySync	The flag to ignore the lazy sync metadata stage
settings	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

metadata	The metadata backend instance.
writeMode	The mode to write metadata.

Parameters

settings	The MetaEngineSettingsContainer determine whether data is actually set or not. The following metadata fields may be set (depending on settings): • Comment • Date • Rating • Tags • Photographer ID (data from settings) • Credits (data from settings)

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

metadata	meta engine object that apply changes
saveTags	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

filePath	The file path to update current tags
writeMode	The mode to write metadata
settings	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

filePath	path to file to add comments, tags and rating
settings	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

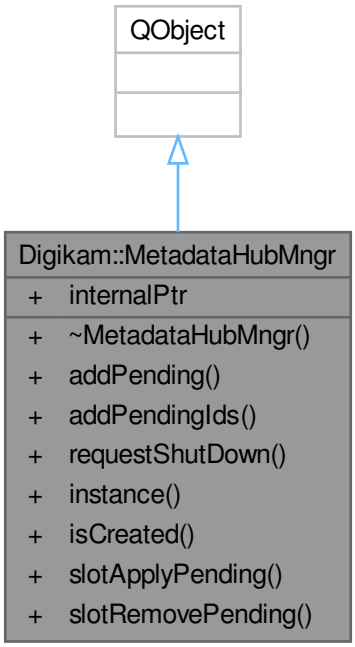
info	The image info to retrieve current tags
writeMode	The mode to write metadata
ignoreLazySync	The flag to ignore the lazy sync metadata stage
settings	The metadata settings to be set

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

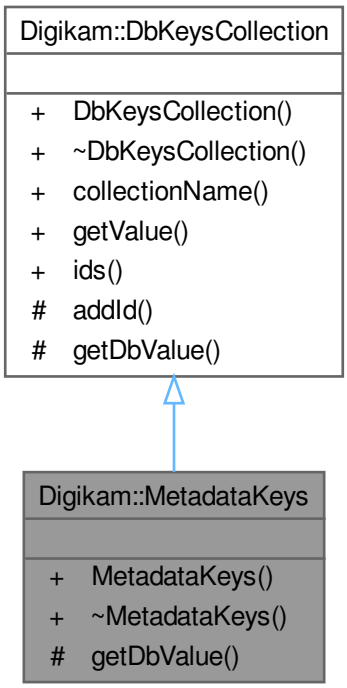
- static MetadataHubMngr * **instance** ()
- static bool **isCreated** ()

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

key	the key representing the value in the database
settings	the ParseSettings object holding all relevant information about the image.

Returns

the value of the given database key

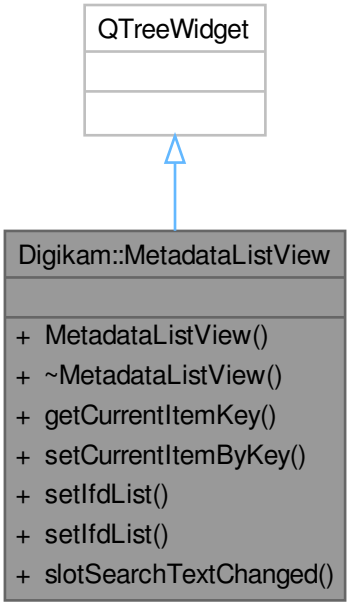
See also

DbKeysCollection::getValue()

Implements Digikam::DbKeysCollection.

9.963 Digikam::MetadataListView Class Reference

Inheritance diagram for Digikam::MetadataListView:



Public Slots

- void **slotSearchTextChanged** (const SearchTextSettings &)

Signals

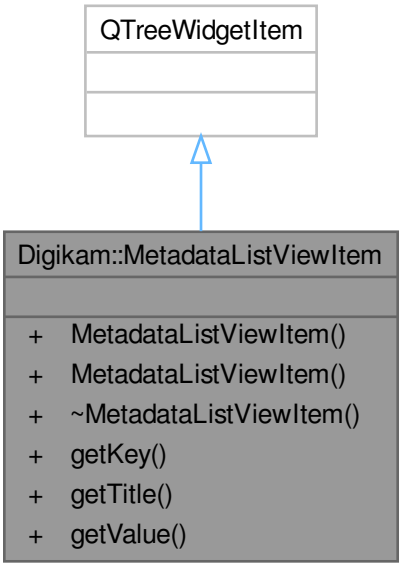
- void **signalTextFilterMatch** (bool)

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

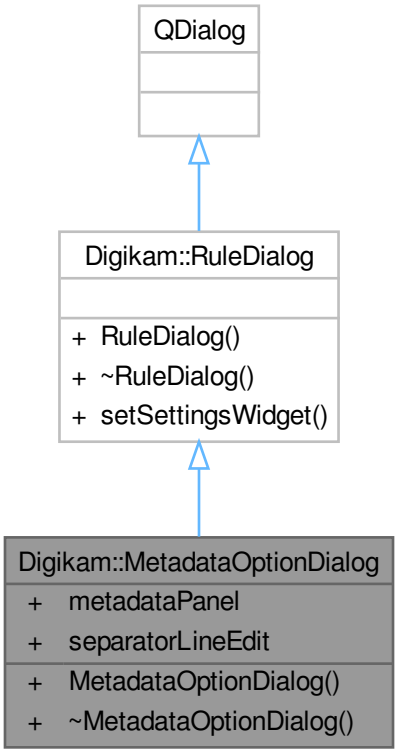
settings	contains settings
match	result of the regular expression match done in Option::parse()

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

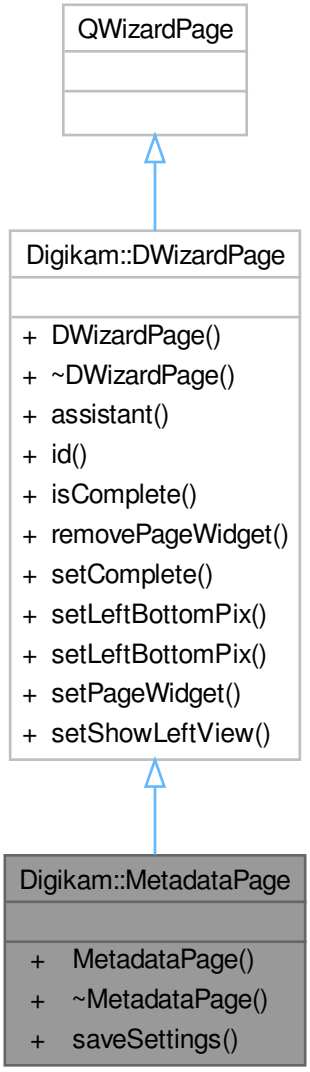
- **RuleDialog** (Rule ∗const parent)
- void **setSettingsWidget** (QWidget ∗const settingsWidget)

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** ()

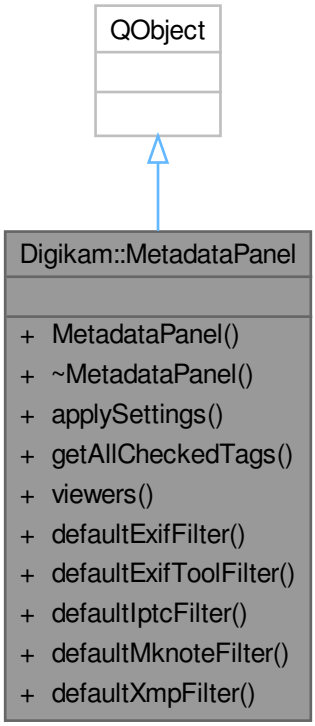
Public Member Functions inherited from Digikam::DWizardPage

- **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

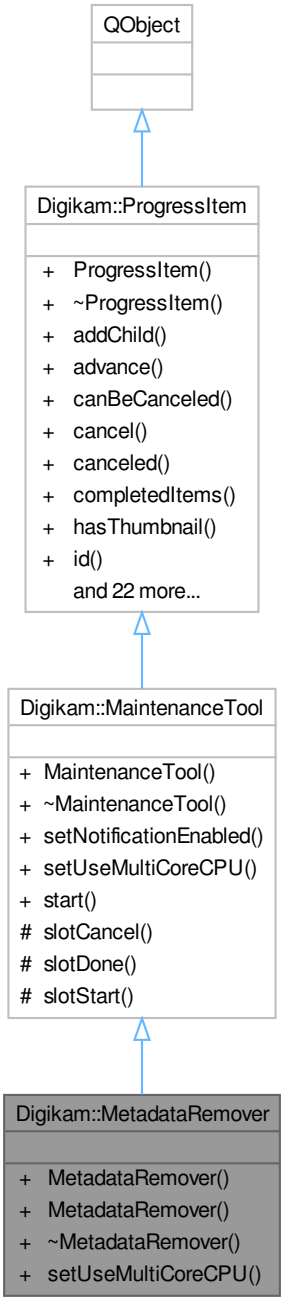
- **MetadataPanel** (QTabWidget ∗const tab)
- void **applySettings** ()
- QStringList **getAllCheckedTags** () const
- QList< MetadataSelectorView ∗ > **viewers** () const

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 elpased 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

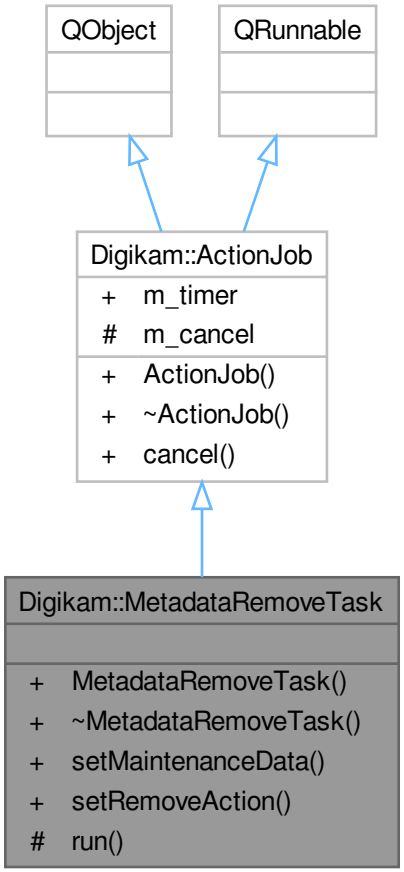
9.969.2.1 setUseMultiCoreCPU()

```
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.

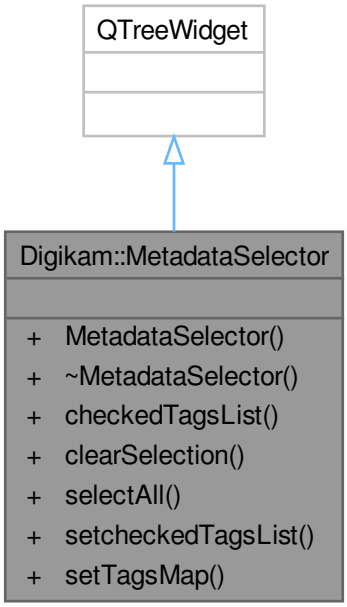
Protected Attributes inherited from Digikam::ActionJob

- 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:

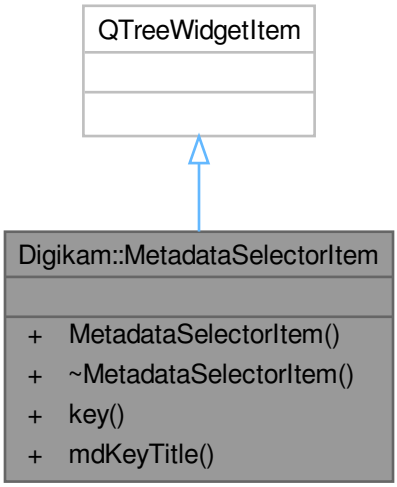


Public Member Functions

- **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:

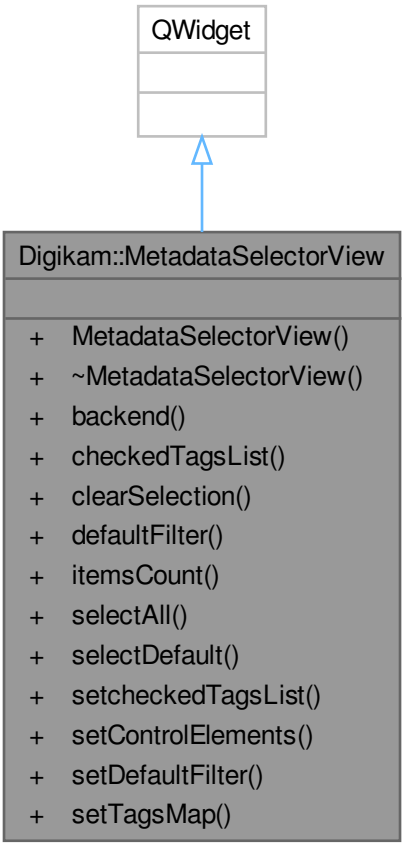


Public Member Functions

- **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**

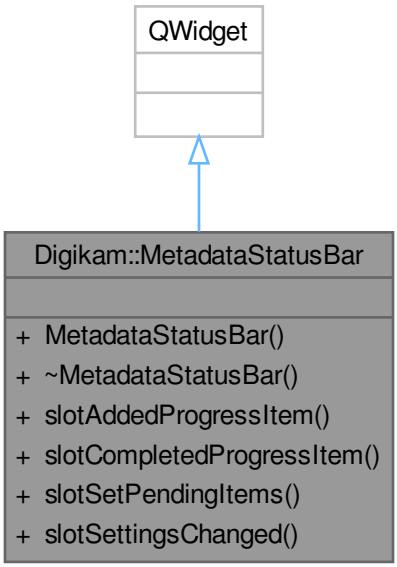
Public Member Functions

- **MetadataSelectorView** (QWidget *const parent, Backend be)
- Backend **backend** () const
- QStringList **checkedTagsList** () const
- void **clearSelection** ()
- QStringList **defaultFilter** () const
- int **itemsCount** () 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

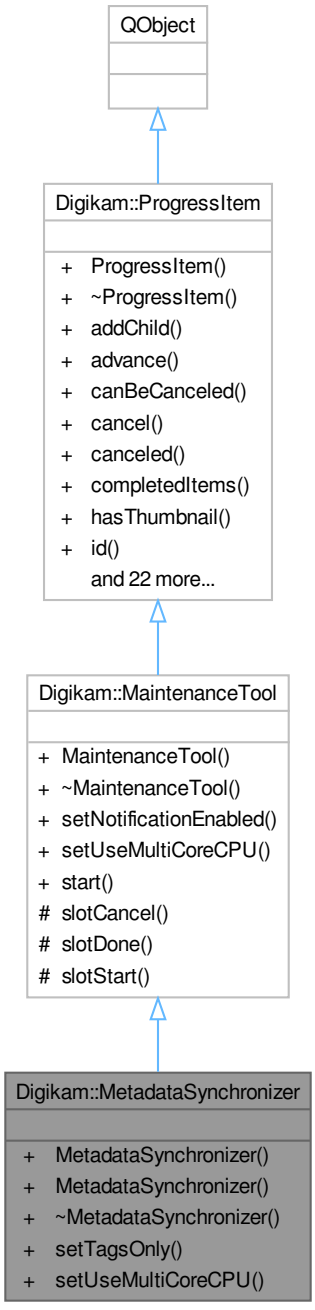
- void **slotAddedProgressItem** (ProgressItem ∗item)
- void **slotCompletedProgressItem** (ProgressItem ∗item)
- void **slotSetPendingItems** (int number)
- void **slotSettingsChanged** ()

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=WriteFromDatabaseTo↩File, 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 elpased 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

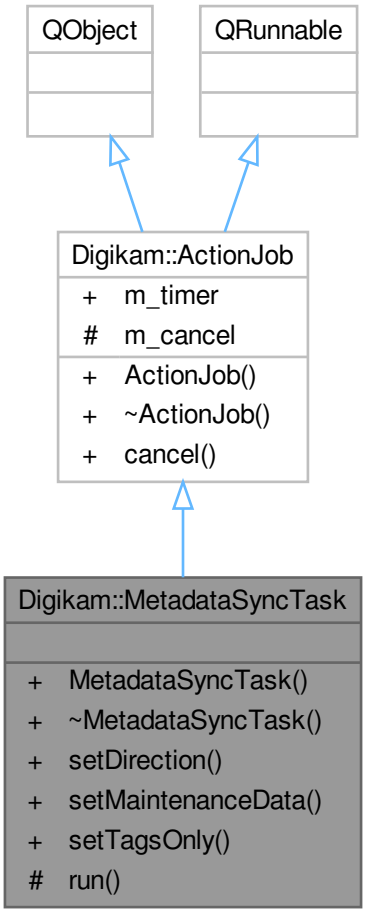
9.975.2.1 setUseMultiCoreCPU()

```
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.

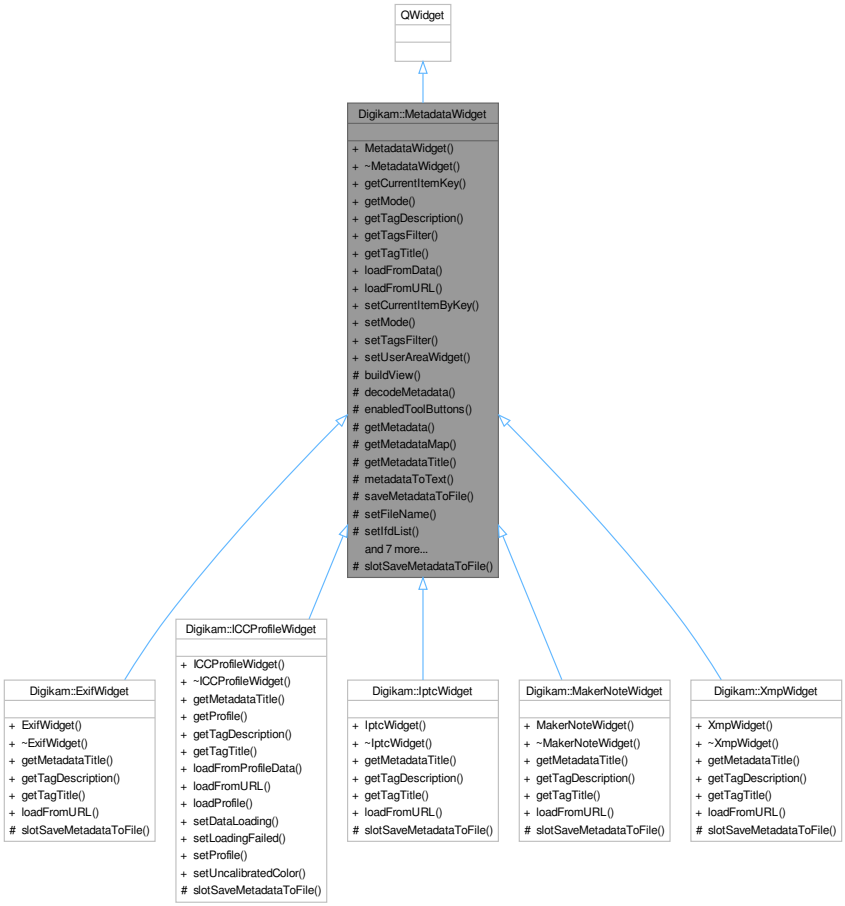
Protected Attributes inherited from Digikam::ActionJob

- 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 DMetadata &data=DMetadata())
- 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 **slotSaveMetadataToFile** ()=0

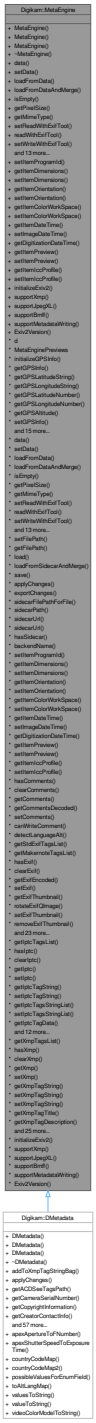
Protected Member Functions

- virtual void **buildView** ()
- virtual bool **decodeMetadata** ()=0
- void **enabledToolButtons** (bool)
- DMetadata ∗ **getMetadata** () const
- const DMetadata::MetaDataMap & **getMetadataMap** ()
- virtual QString **getMetadataTitle** () const =0
- 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

9.978 Digikam::MetaEngine Class Reference

Inheritance diagram for Digikam::MetaEngine:



Public Types

- 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

- **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 setImageDateTime (const QDateTime &dateTime, bool setDateTimeDigitized=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.

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 from "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 MetaEngine.
LibRawBackend	DMetadata only.
LibHeifBackend	DMetadata only.
ImageMagickBackend	DMetadata only.
FFMpegBackend	DMetadata only.
ExifToolBackend	DMetadata only.
VideoMergeBackend	DMetadata only.
NoBackend	No backend used (aka file cannot be read).

9.978.2.2 MetadataWritingMode

```
enum Digikam::MetaEngine::MetadataWritingMode
```

See also

MetadataWritingMode(), metadataWritingMode()

Enumerator

WRITE_TO_FILE_ONLY	Write metadata to item file only.
WRITE_TO_SIDECAR_ONLY	Write metadata to sidecar file only.
WRITE_TO_SIDECAR_AND_FILE	Write metadata to item and sidecar files.
WRITE_TO_SIDECAR_ONLY_FOR_READ_ONLY↩ _FILES	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) 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 founf, 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 'inverSelection' is false.
- not include "Iop", or "Thumbnail", or "Image", or "Photo" in the Exif tag keys if 'inverSelection' 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 'inverSelection' is false.
- not include "Envelope", or "Application2" in the Iptc tag keys if 'inverSelection' 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" // Dubling 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 'inverSelection' is false.
- not include "dc", or "xmp" in the Xmp tag keys if 'inverSelection' 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 StringList. 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 historicaly 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 ByteArray 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 *const 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 setDateTimeDigitized = false ) const
```

If 'setDateTimeDigitized' 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

mode	Metadata writing mode as defined by the MetadataWritingMode 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 of 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()

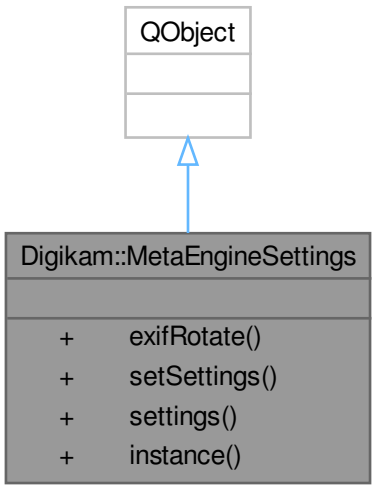
```
QList< MetaEngineRotation::TransformationAction > Digikam::MetaEngineRotation::transformations
( ) const
```

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 }

  Describes the allowed and desired operation when rotating a picture.
- typedef QFlags< RotationBehaviorFlag > **RotationBehaviorFlags**

Public Member Functions

- QStringList **defaultExifToolSearchPaths** () const
- void **readFromConfig** (const KConfigGroup &group)
- void **writeToConfig** (KConfigGroup &group) const

Public Attributes

- AlbumDateSource **albumDateFrom** = OldestItemDate
- bool **exifRotate** = true
- bool **exifSetOrientation** = true
- QString **exifToolPath**
- MetaEngine::MetadataWritingMode **metadataWritingMode** = MetaEngine::WRITE_TO_FILE_ONLY
- bool **readWithExifTool** = false
- bool **rescanImageIfModified** = false
- RotationBehaviorFlags **rotationBehavior** = RotationBehaviorFlags(RotatingFlags | RotateByLosslessRotation)
- 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

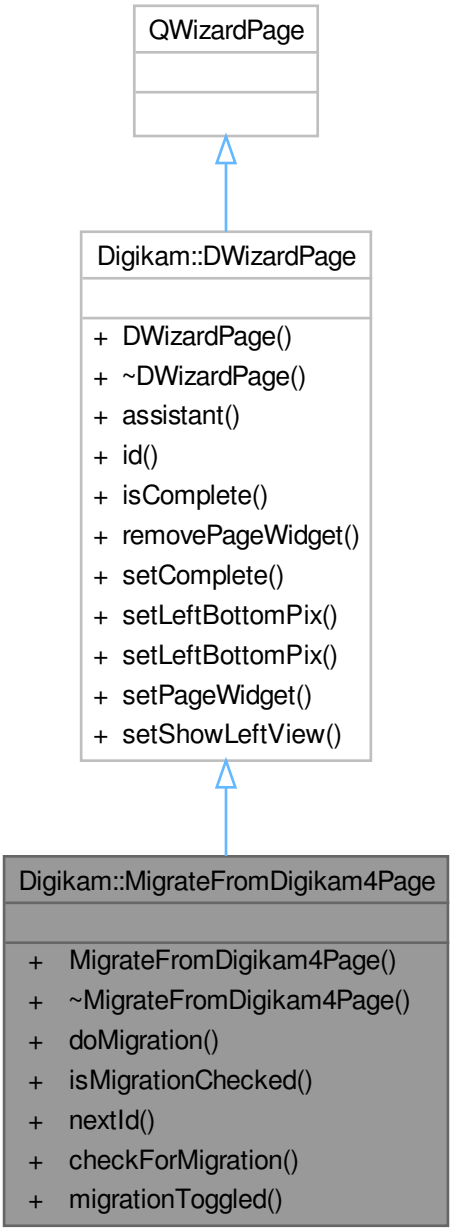
9.984.2.1 RotationBehaviorFlag

`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 by rotated by content, setting the metadata may be problematic 2) Read-Only files cannot 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)

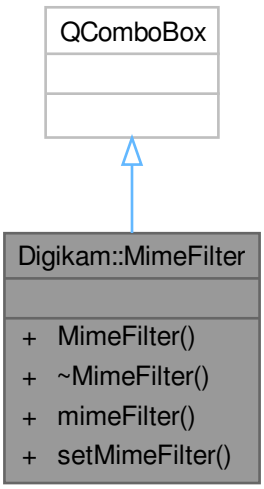
Static Public Member Functions

- 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** ,
  TIFFiles , **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 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** ()

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.

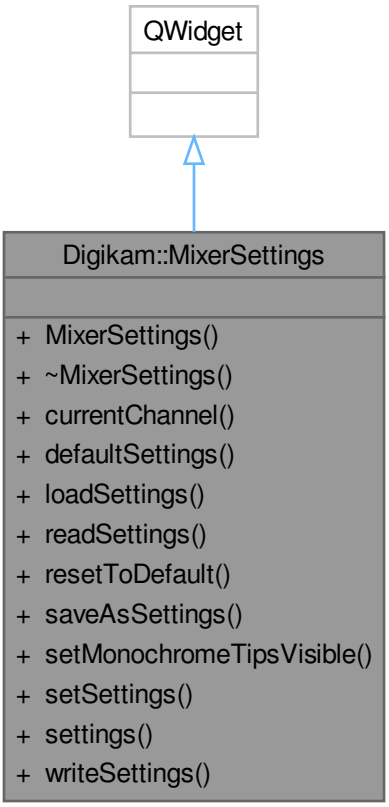
9.988.1.3 readParameters()

```
void Digikam::MixerFilter::readParameters (
            const FilterAction & action )  [override], [virtual]
```

Implements Digikam::DImgThreadedFilter.

9.989 Digikam::MixerSettings Class Reference

Inheritance diagram for Digikam::MixerSettings:



Signals

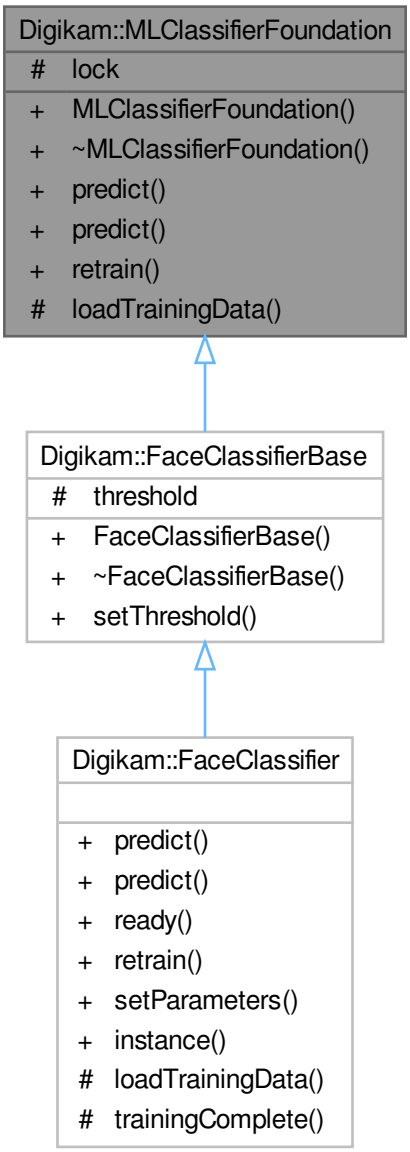
- void **signalMonochromeActived** (bool)
- void **signalOutChannelChanged** ()
- void **signalSettingsChanged** ()

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 **MLPipelinePerformance↩ Profile**
- 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

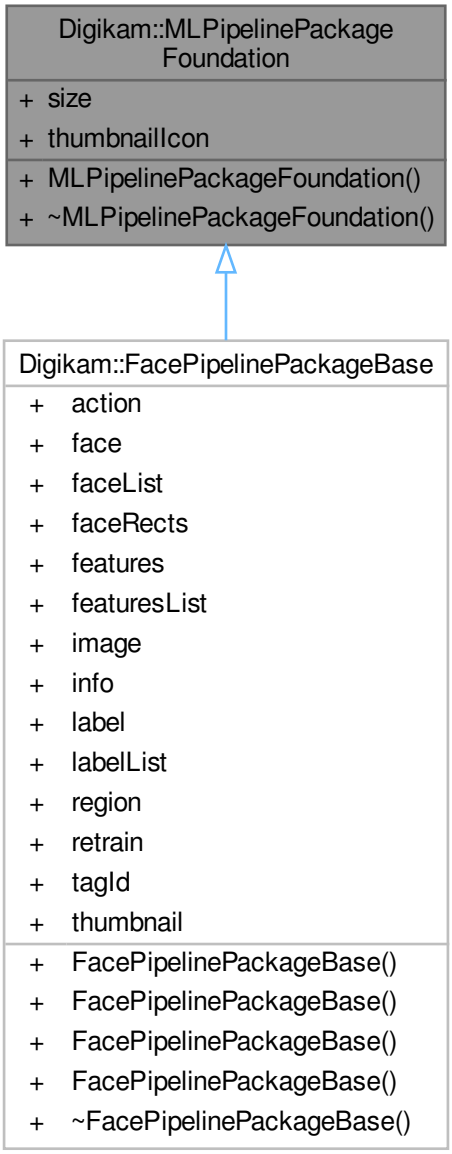
9.994 Digikam::MLPipelineFoundation::_MLPipelinePerformanceProfile Struct Reference

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:

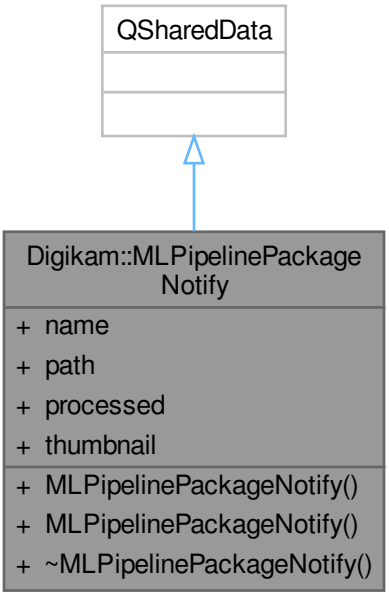


Public Attributes

- 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

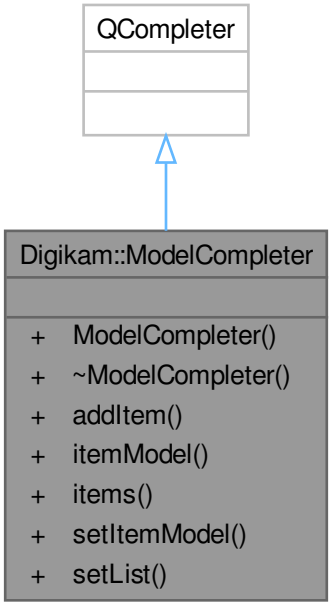
- **MLPipelinePackageNotify** (const QString &_name, const QString &_path, int _processed, const DImg &↩ _thumbnail)
- **MLPipelinePackageNotify** (const QString &_name, const QString &_path, int _processed, const QIcon &↩ _thumbnail)

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()

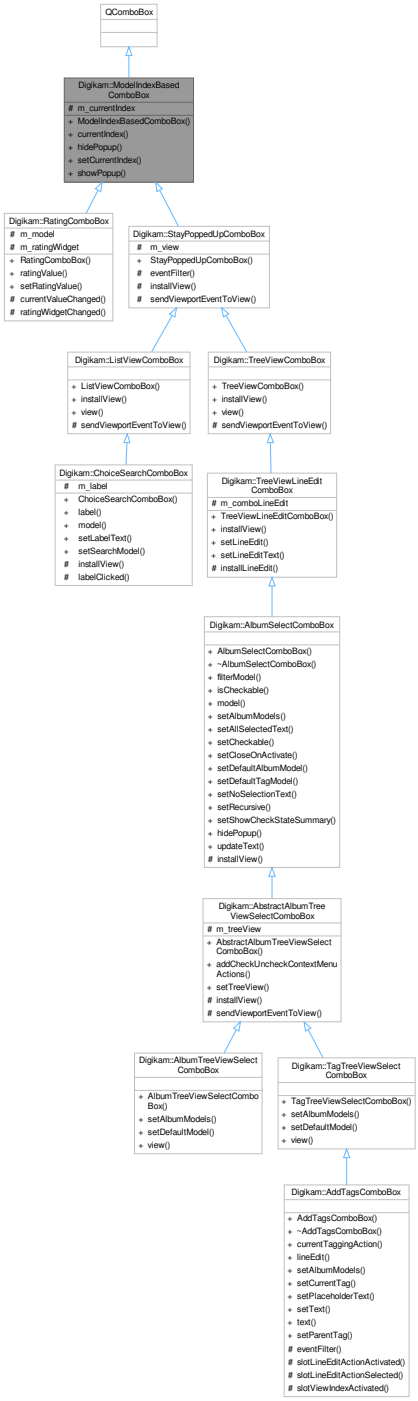
```
void Digikam::ModelCompleter::setItemModel (
            QAbstractItemModel *const model,
            int uniqueIdRole,
            int displayRole = Qt::DisplayRole )
```

Parameters

model	to fill from or null for manual mode
uniqueIdRole	a role for which the model will return a unique integer for each entry
displayRole	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 ModelIndexBasedComboBox()

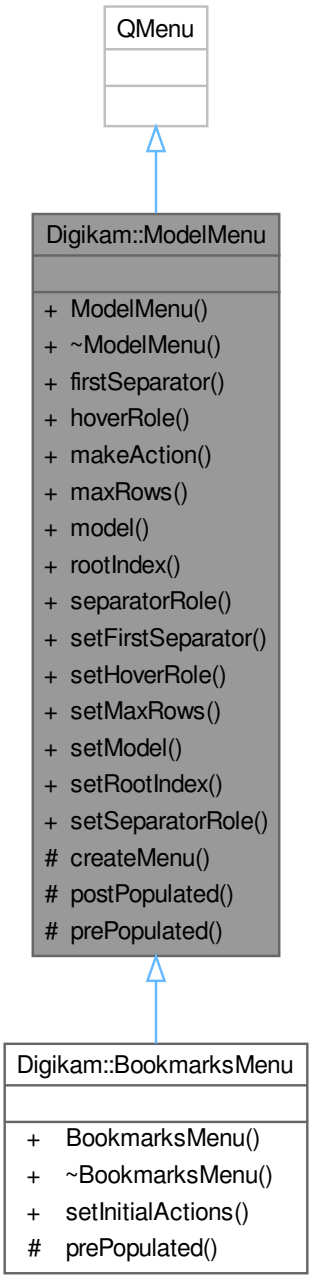
```
Digikam::ModelIndexBasedComboBox::ModelIndexBasedComboBox (
            QWidget *const parent = nullptr )  [explicit]
```

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

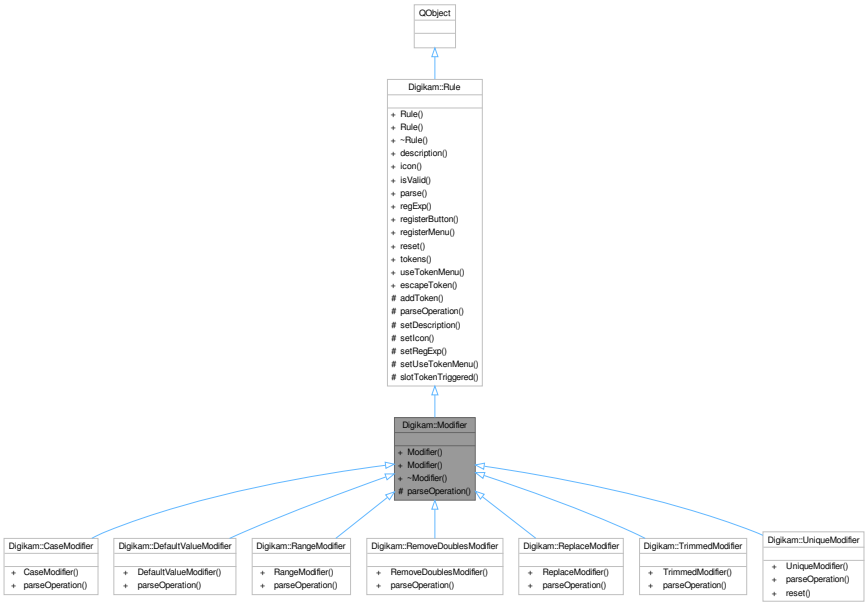
9.999.1.1 prePopulated()

```
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

settings	contains settings
match	result of the regular expression match done in Option::parse()

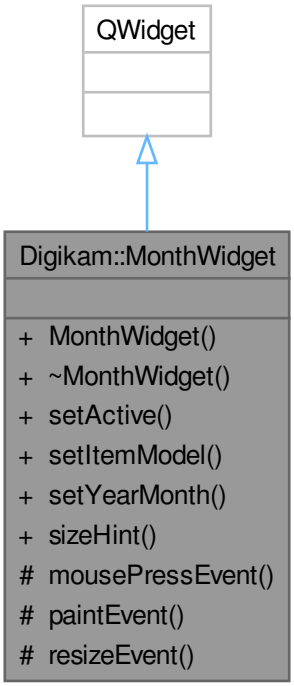
Returns

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 **versionIsRight** () const
- bool **versionIsRight** (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 **versionIsRight** () const
- bool **versionIsRight** (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 **versionIsRight** () const
- bool **versionIsRight** (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 **versionIsRight** () const
- bool **versionIsRight** (const float) const

Static Public Member Functions inherited from Digikam::DBinaryIface

- static QString **goodBaseName** (const QString &b)

Protected Member Functions inherited from Digikam::DBinaryIface

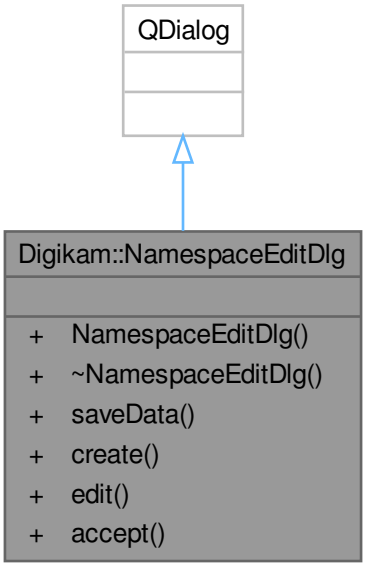
- QString **findHeader** (const QStringList &output, const QString &header) const
- virtual bool **parseHeader** (const QString &output)
- virtual QString **readConfig** ()
- void **setVersion** (QString &version)
- virtual void **writeConfig** ()

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

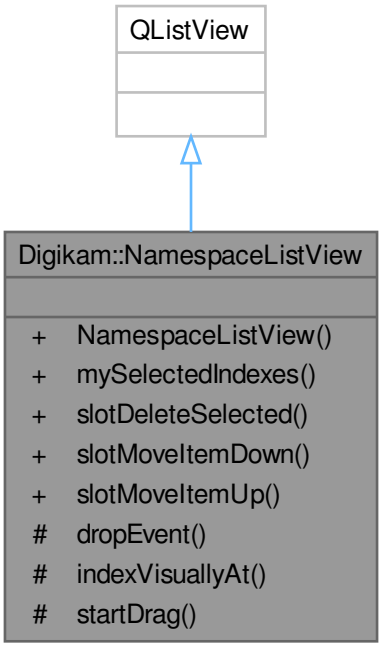
- QString **alternativeName**
- QList< int > **convertRatio**

  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** ()

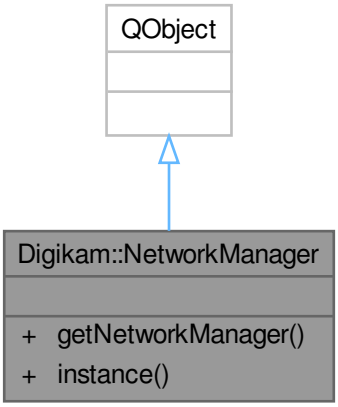
Protected Member Functions

- 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

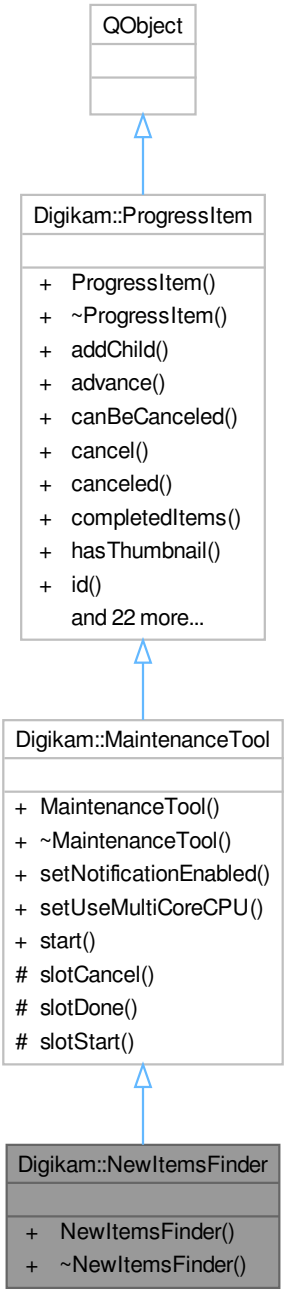
9.1009.1.1 instance()

```
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 elpased 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

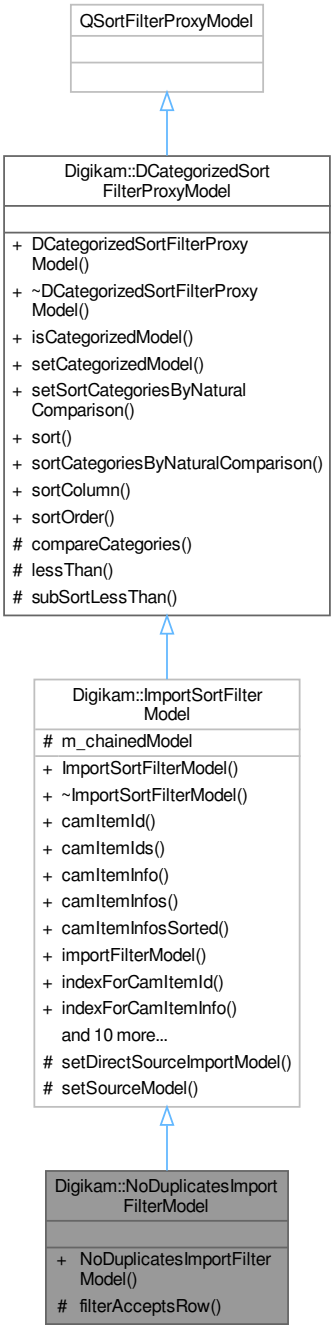
```
enum Digikam::NewItemsFinder::FinderMode
```

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

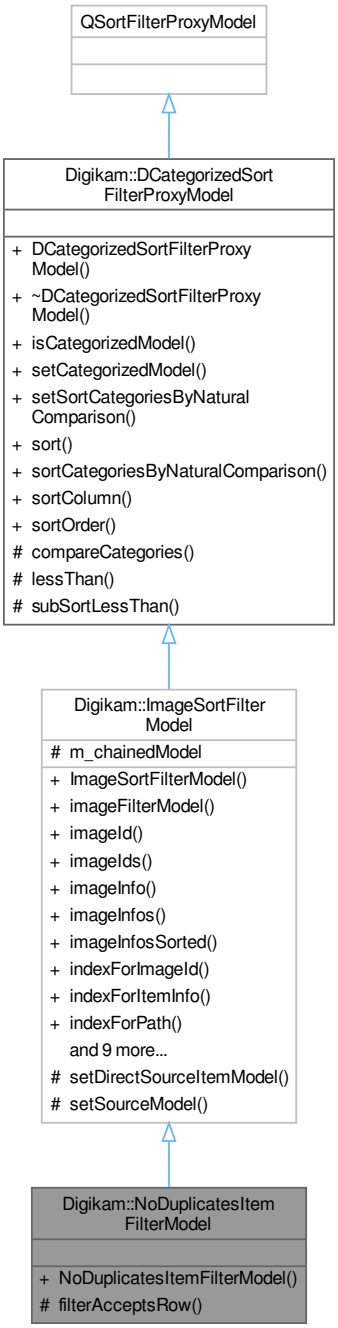
- enum AdditionalRoles { CategoryDisplayRole = 0x17CE990A , CategorySortRole = 0x27857E60 }

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 **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)

  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

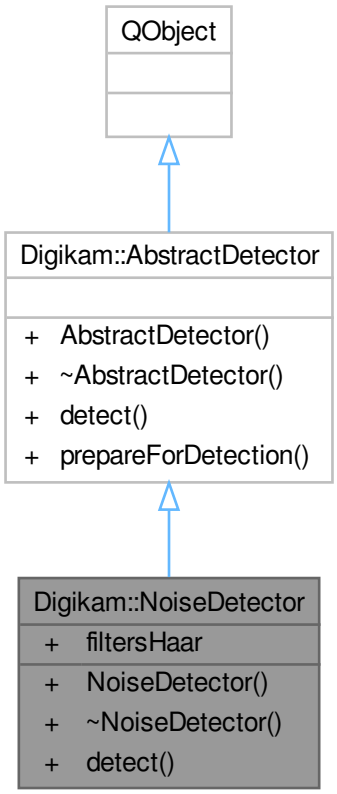
- enum AdditionalRoles { CategoryDisplayRole = 0x17CE990A , CategorySortRole = 0x27857E60 }

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

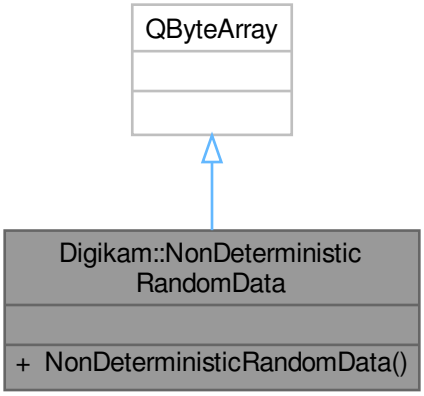
9.1013.1.1 detect()

```
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** (DImg ∗const orgImage, const DImg ∗const refImage, 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 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** ()

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()

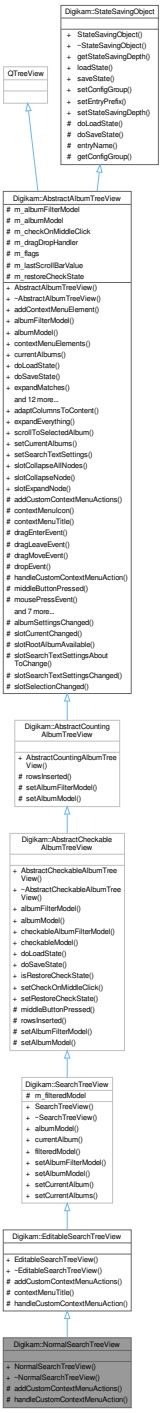
```
void Digikam::NormalizeFilter::readParameters (
            const FilterAction & action )  [override], [virtual]
```

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 of 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 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 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

parent	qt parent
searchModel	the model this view should act on
searchModificationHelper	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

album	search to copy

9.1016.3.3 editSearch

```
void Digikam::NormalSearchTreeView::editSearch (
            SAlbum * album )  [signal]
```

Parameters

album	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

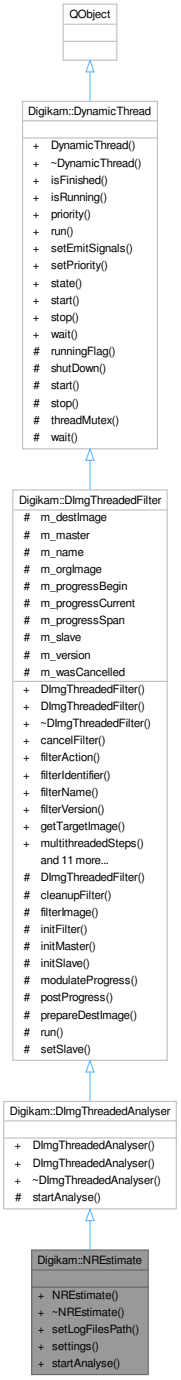
9.1017.1.1 thresholds

```
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 ananlyser 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 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.

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 setLogFilesPath()

```
void Digikam::NREstimate::setLogFilesPath (
            const QString & path )
```

If path is not set, no log files will be created.

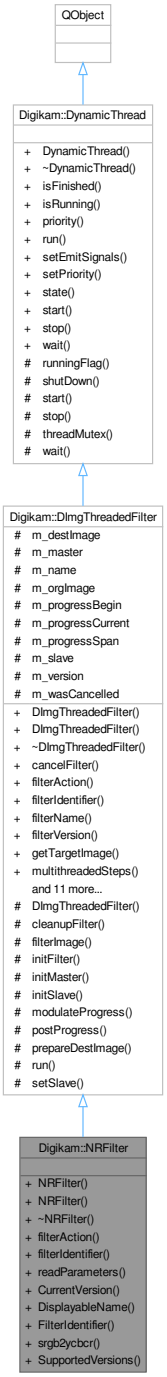
9.1018.1.2 startAnalyse()

```
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 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 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.

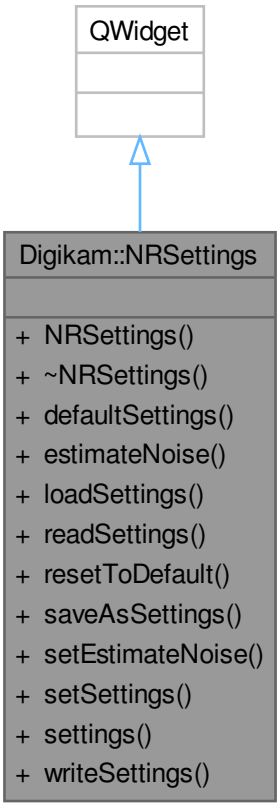
9.1019.1.3 readParameters()

```
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 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** ()

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.

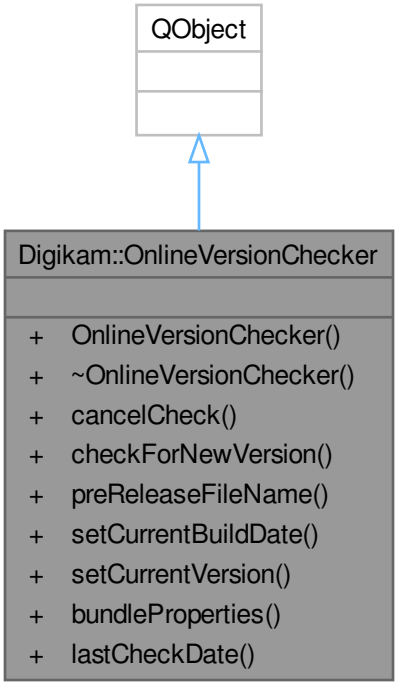
9.1021.1.3 readParameters()

```
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

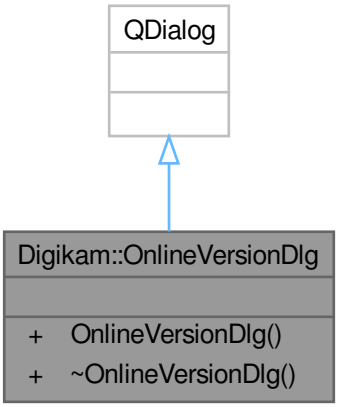
9.1022.1.1 bundleProperties()

```
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

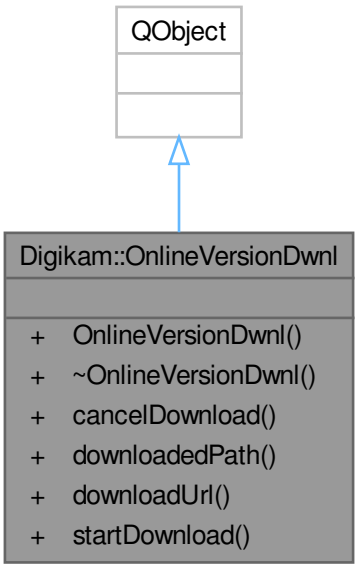
- void **signalSetupUpdate** ()

Public Member Functions

- **OnlineVersionDlg** (QWidget *const parent=nullptr, const QString &version=QLatin1String(digikam_version_short), const QDateTime &buildDt=digiKamBuildDate(), bool checkPreRelease=false, bool updateWithDebug=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 updateWith↩ Debug=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

```
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 (https://docs.opencv.org/4.x/dc/dd6/ml_intro.html#ml_intro_svm)
OpenCV_KNN	K-Nearest Neighbors (https://docs.opencv.org/4.x/dc/dd6/ml_intro.html#ml_intro_knn)
Tree	K-Nearest Neighbors Tree (https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm)
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

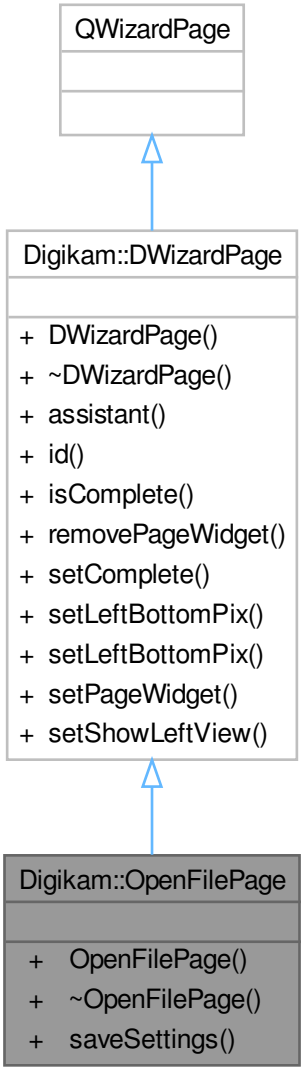
Public Member Functions

- 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** ()

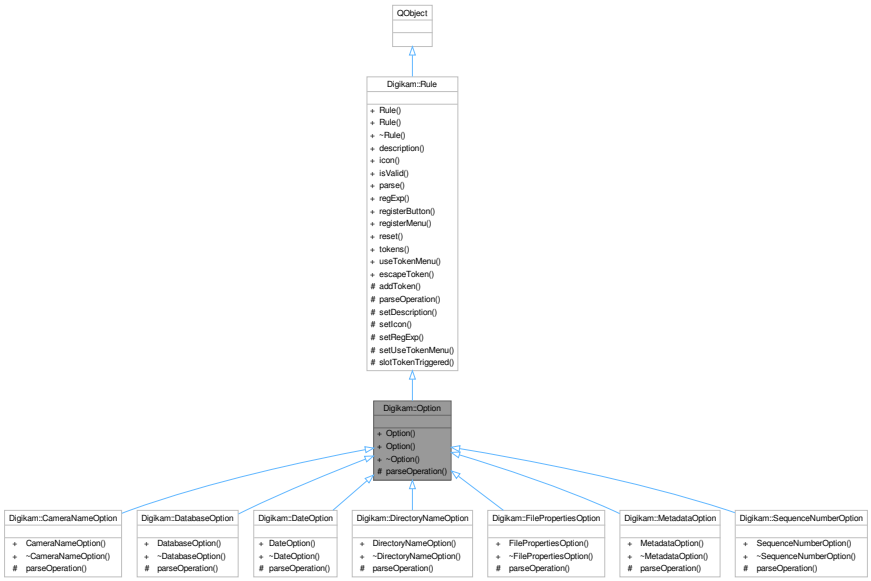
Public Member Functions inherited from Digikam::DWizardPage

- **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

  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 &)

9.1029.1 Member Function Documentation

9.1029.1.1 parseOperation()

```
QString Digikam::Option::parseOperation (
            ParseSettings & settings,
            const QRegularExpressionMatch & match )  [override], [protected], [pure virtual]
```

Parameters

settings	contains settings
match	result of the regular expression match done in Option::parse()

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

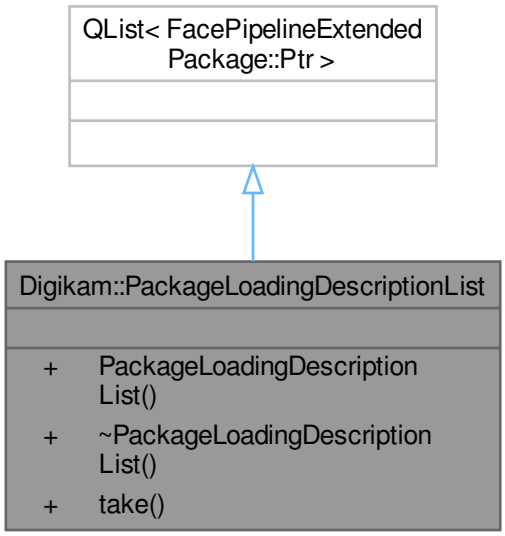
9.1030.1 Detailed Description

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:



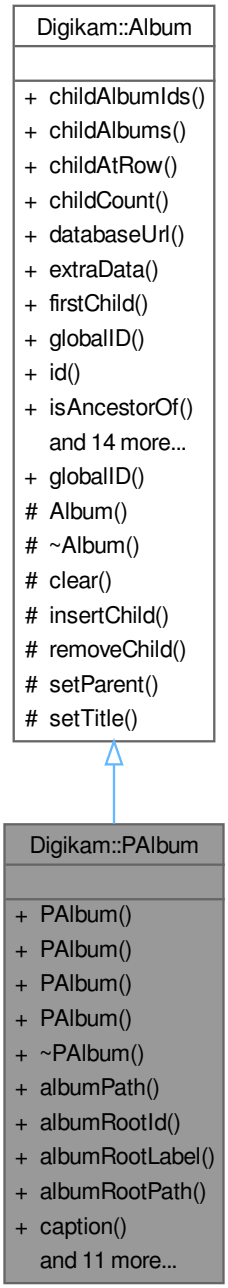
Public Member Functions

- 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

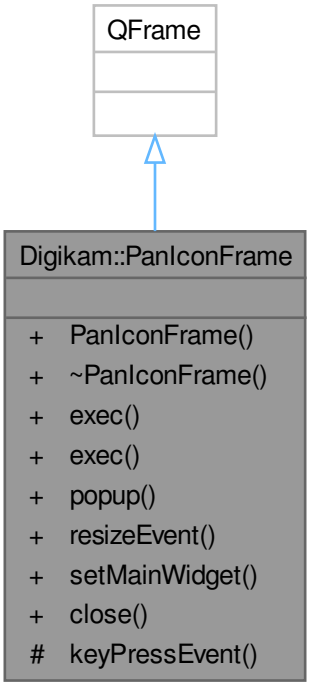
the kde url of the album

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.

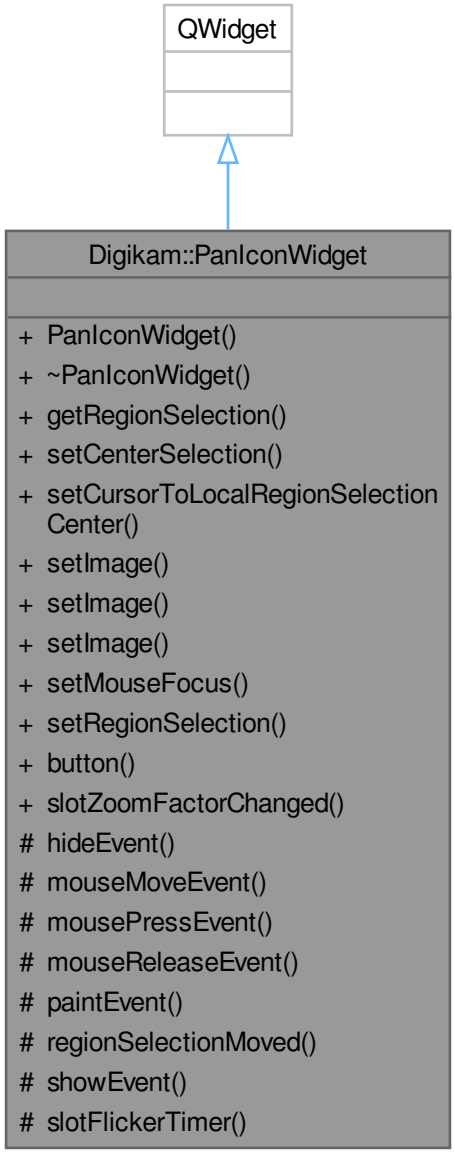
9.1033.1.3 setMainWidget()

```
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 itselfes. 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 DImg &fullOriginalImage)
- void **setImage** (int previewWidth, int previewHeight, const QImage &fullOriginalImage)
- void **setMouseFocus** ()
- void **setRegionSelection** (const QRect &regionSelection)

Static Public Member Functions

- static QToolButton ∗ **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

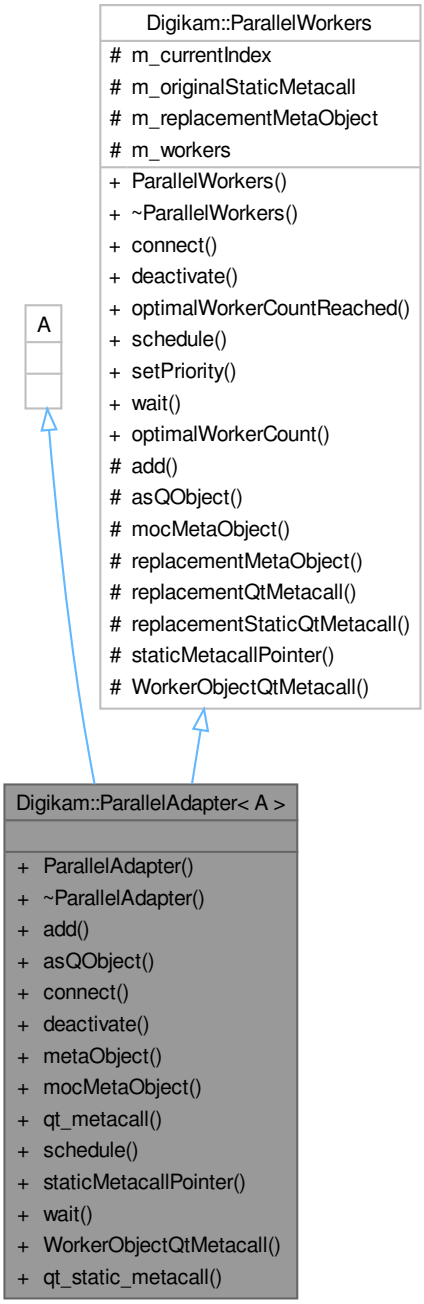
9.1034.1.1 signalSelectionMoved

```
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.

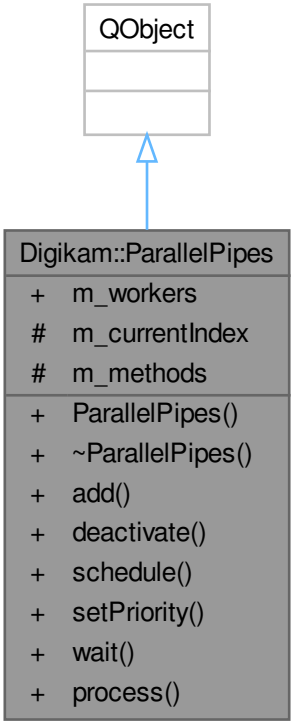
9.1035.2.4 WorkerObjectQtMetacall()

```
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

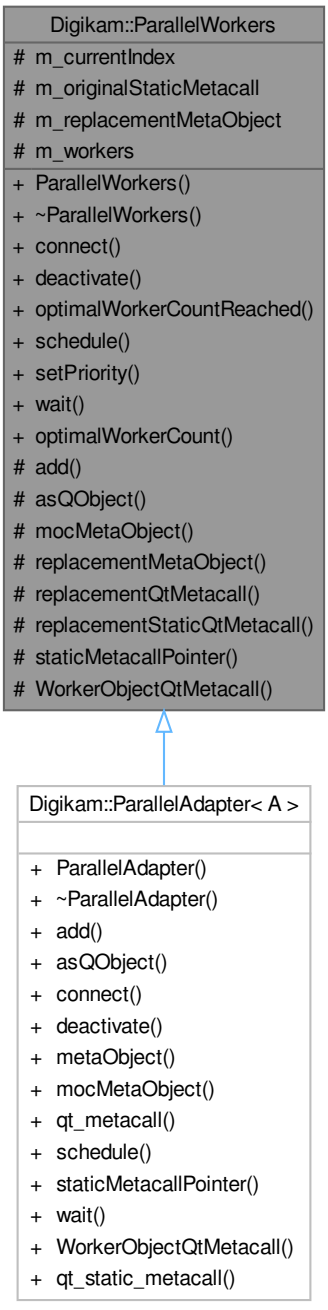
- QList< WorkerObject ∗ > **m_workers**

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 >.

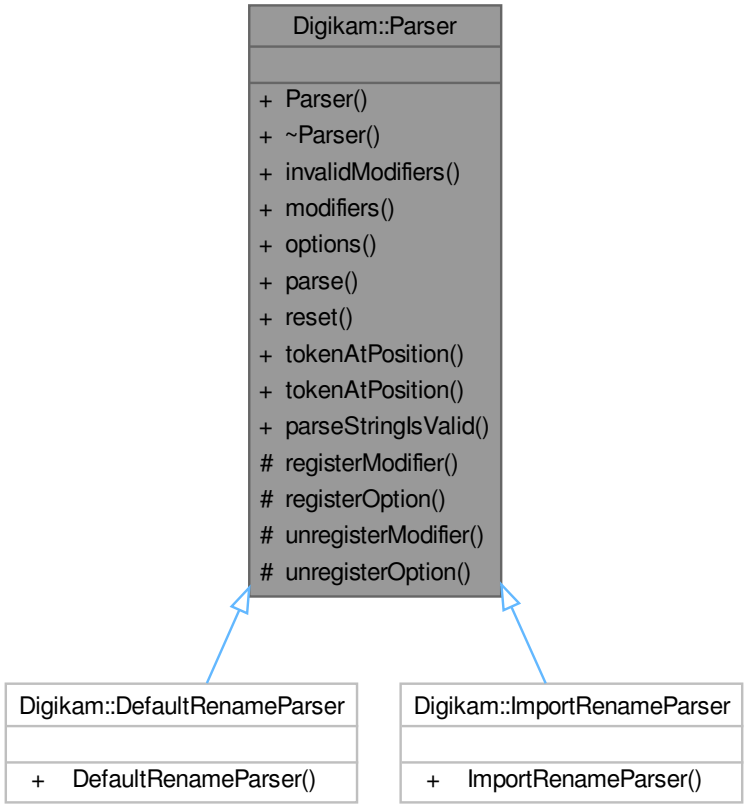
9.1037.2.3 WorkerObjectQtMetacall()

```
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

str	the parse string

Returns

true if valid / can be parsed

9.1039 Digikam::ParseResults Class Reference

Public Types

- typedef QPair< int, int > **ResultsKey**
- typedef QMultiMap< 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)

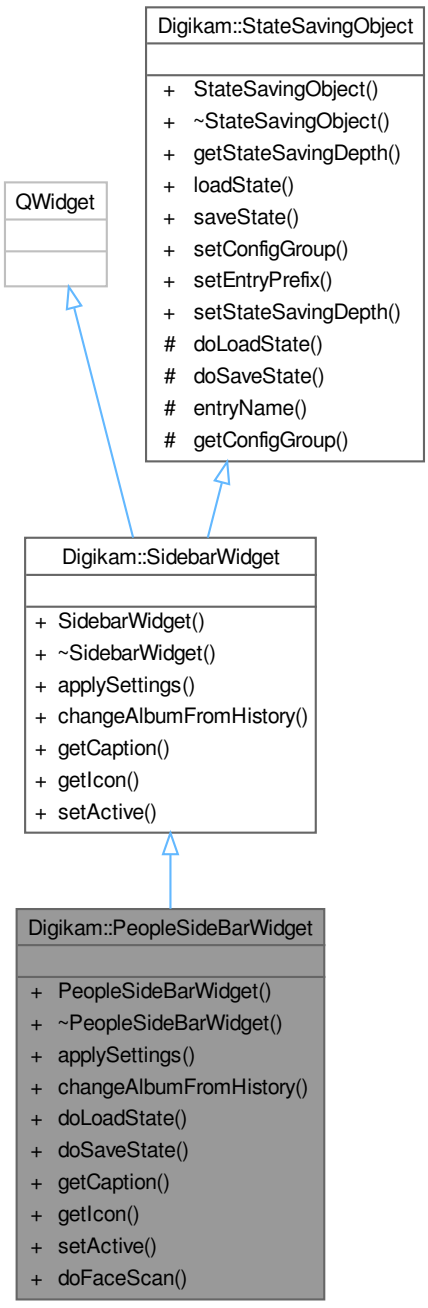
  ItemInfo constructor.
- **ParseSettings** (const QString &_parseString)
- **ParseSettings** (const QString &_parseString, const ItemInfo &info)
- bool **isValid** () const

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]
```

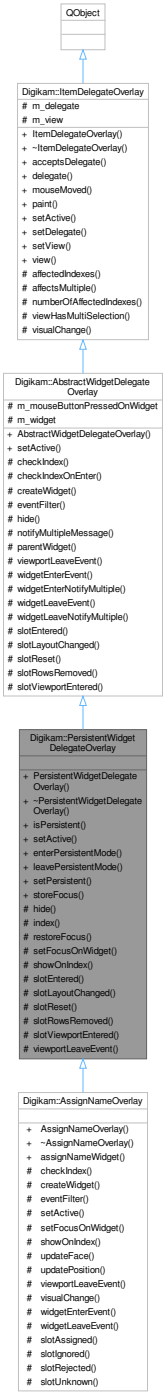
Parameters

active	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 overlayed 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

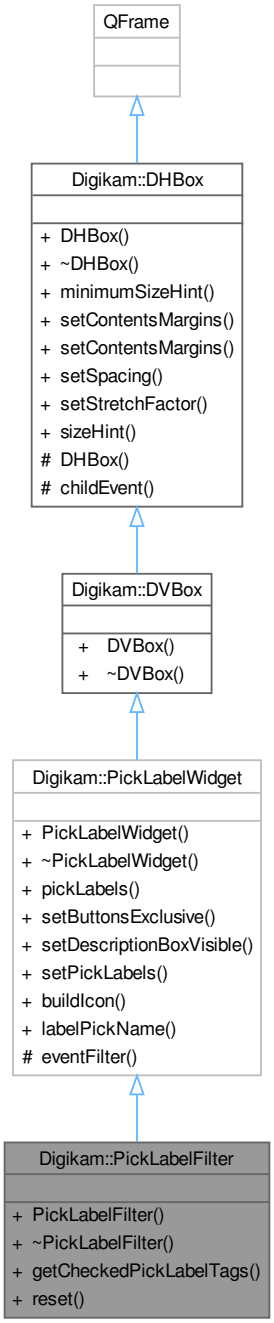
Public Attributes

- 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

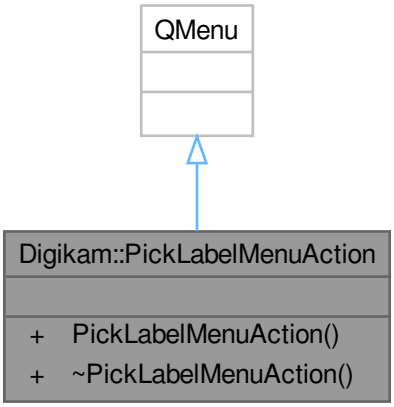
- bool **eventFilter** (QObject ∗obj, QEvent ∗ev) override

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

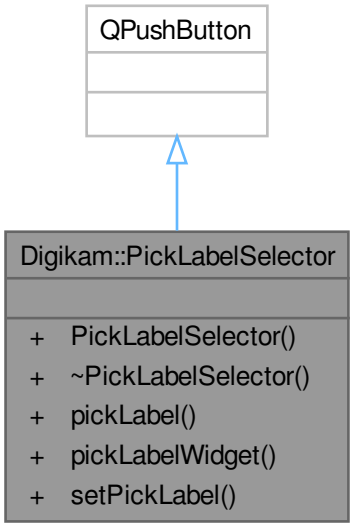
- void **signalPickLabelChanged** (int)

Public Member Functions

- **PickLabelMenuAction** (QMenu ∗const parent=nullptr)

9.1046 Digikam::PickLabelSelector Class Reference

Inheritance diagram for Digikam::PickLabelSelector:



Signals

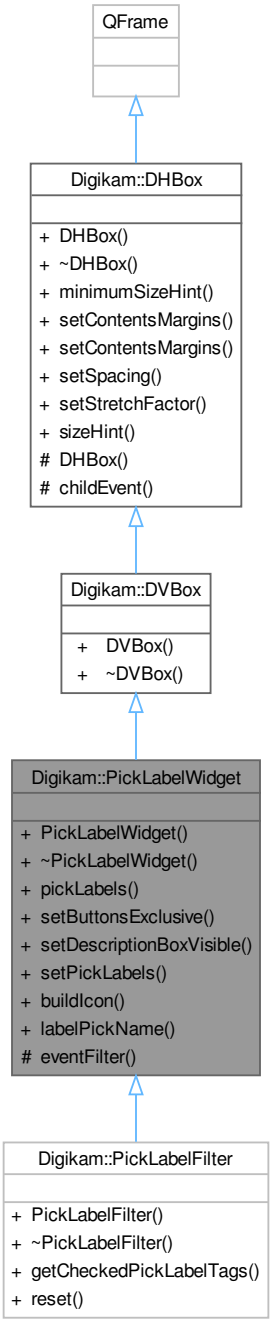
- void **signalPickLabelChanged** (int)

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.

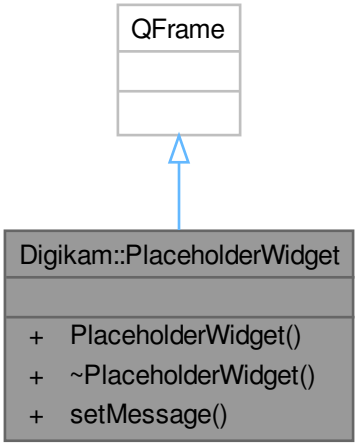
9.1047.1.2 setPickLabels()

```
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)

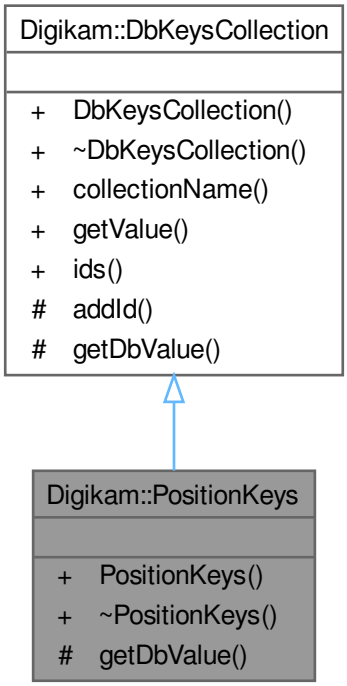
9.1049 Digikam::PointTransformAffine Class Reference

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

key	the key representing the value in the database
settings	the ParseSettings object holding all relevant information about the image.

Returns

the value of the given database key

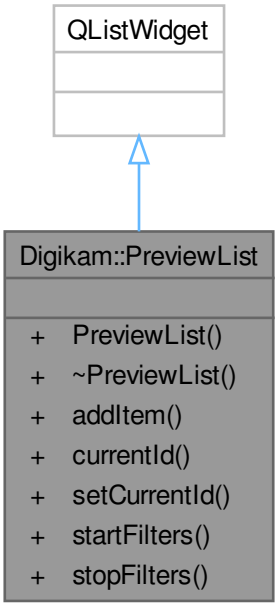
See also

DbKeysCollection::getValue()

Implements Digikam::DbKeysCollection.

9.1051 Digikam::PreviewList Class Reference

Inheritance diagram for Digikam::PreviewList:

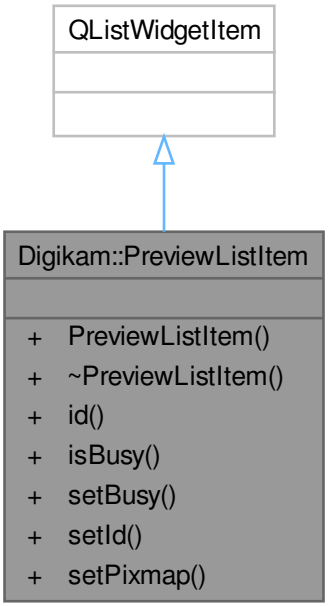


Public Member Functions

- **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:

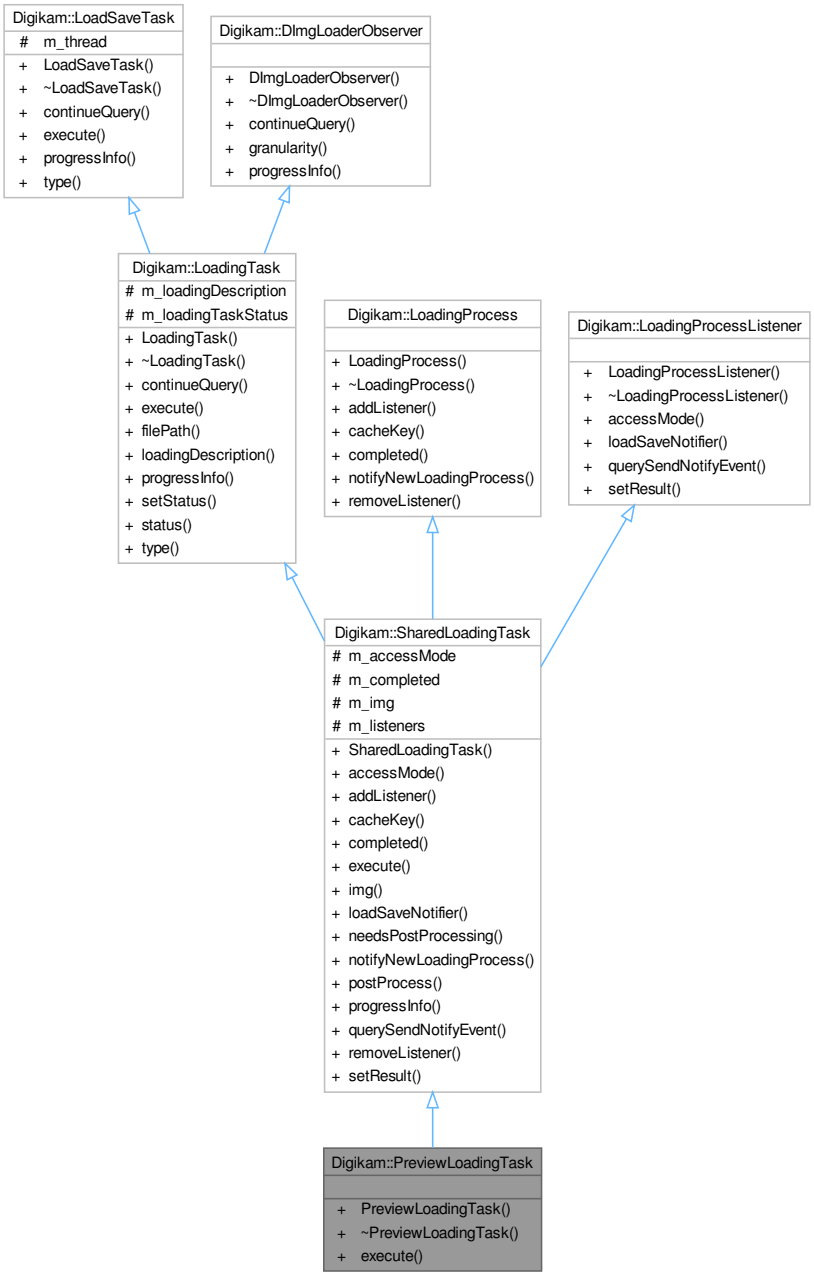


Public Member Functions

- **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

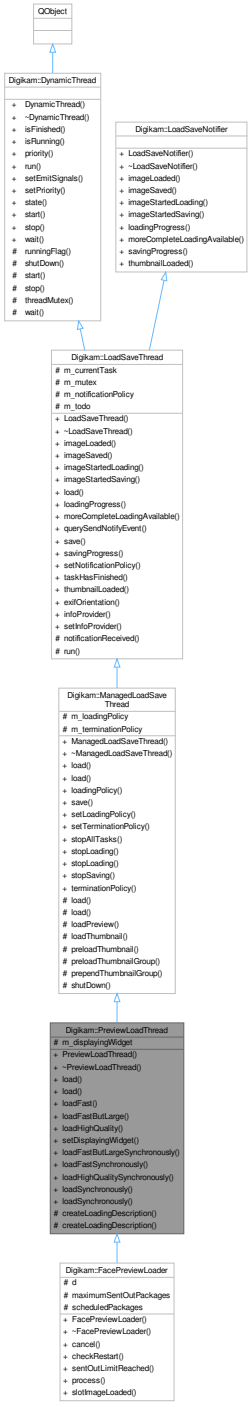
9.1053.1.1 execute()

```
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

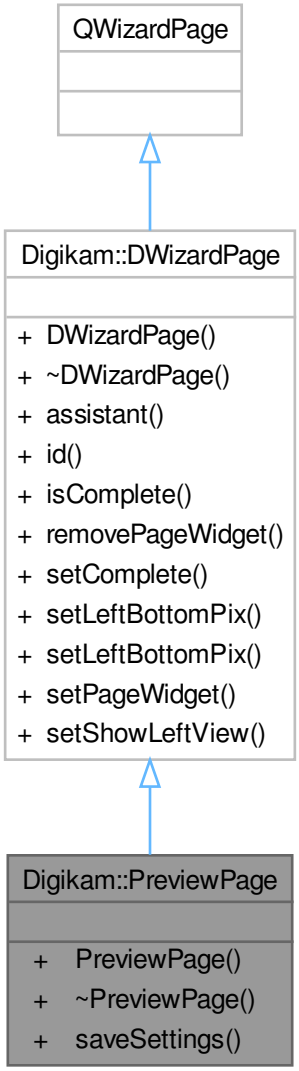
9.1054.2.6 loadHighQuality()

```
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** , **RawPreview↩ FromRawHalfSize** , **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

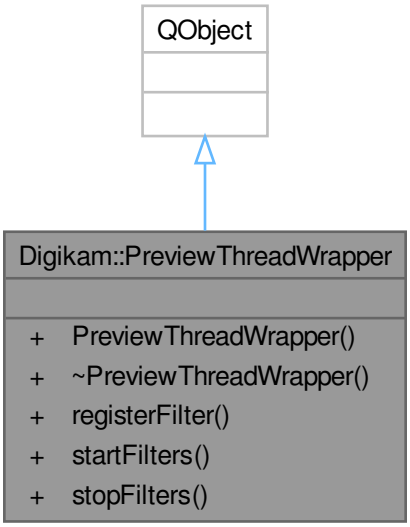
`enum Digikam::PreviewSettings::Quality`

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

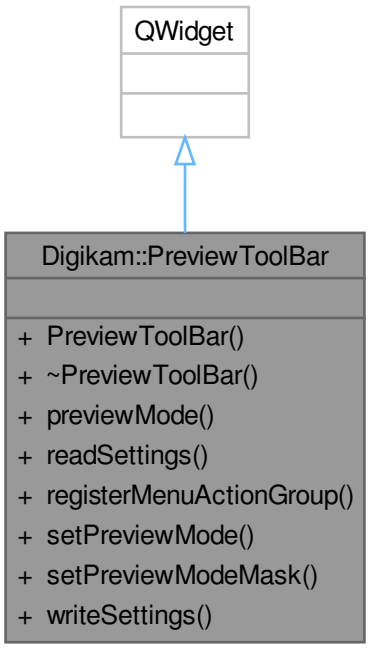
- void **signalFilterFinished** (int, const QPixmap &)
- void **signalFilterStarted** (int)

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 | Preview↩ ToggleOnMouseOver }

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

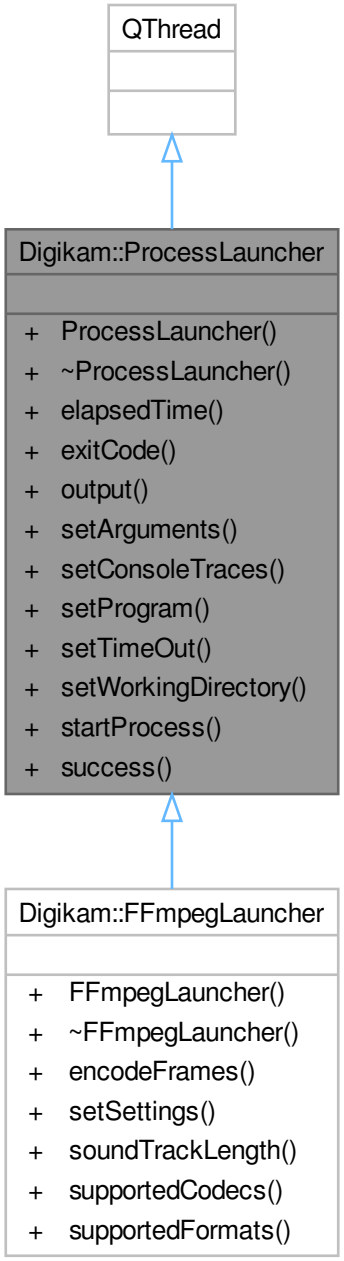
enum Digikam::PreviewToolBar::PreviewMode

Enumerator

PreviewOriginalImage	Original image only.
PreviewBothImagesHorz	Horizontal with original and target duplicated.
PreviewBothImagesVert	Vertical with original and target duplicated.
PreviewBothImagesHorzCont	Horizontal with original and target in contiguous.
PreviewBothImagesVertCont	Vertical with original and target in contiguous.
PreviewTargetImage	Target image only.
PreviewToggleOnMouseOver	Original image if mouse is over image area, else target image.
NoPreviewMode	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 outout as string.
- void **setArguments** (const QStringList &args)
- void **setConsoleTraces** (bool b)

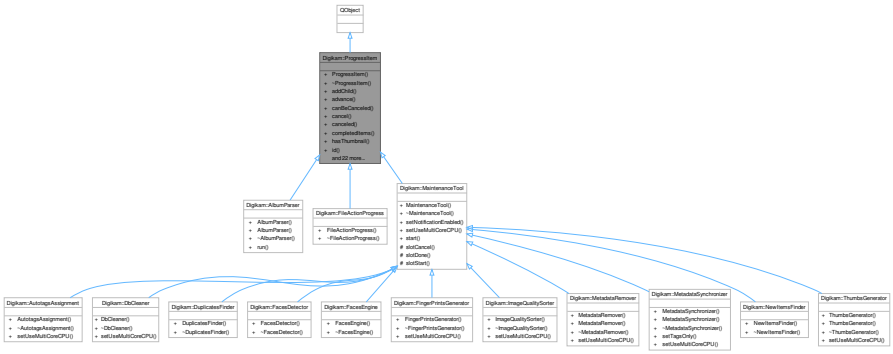
  If turned on, all traces from the process are printed on the console.
- void **setProgram** (const QString &prog)
- void **setTimeOut** (int msecs)
- void **setWorkingDirectory** (const QString &dir)
- void **startProcess** ()

  Start the process.
- bool **success** () const

  Return true if the process is startd 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

v	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

item	The ProgressItem 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

item	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

item	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

item	The updated item.
label	The new label.

9.1060.1.12 progressItemProgress

```
void Digikam::ProgressItem::progressItemProgress (
            ProgressItem * item,
            unsigned int v )  [signal]
```

Parameters

item	The item which got a new value.
v	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

item	The updated item.
mess	The new message.

9.1060.1.14 progressItemThumbnail

```
void Digikam::ProgressItem::progressItemThumbnail (
            ProgressItem * item,
            const QPixmap & thumb )  [signal]
```

Parameters

item	The updated item
thumb	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

item	The updated item
value	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

v	Set the user visible string identifying this item.

9.1060.1.18 setProgress()

```
void Digikam::ProgressItem::setProgress (
            unsigned int v )
```

Parameters

v	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

showAtStart	The flag to turn on this property.

9.1060.1.20 setStatus()

```
void Digikam::ProgressItem::setStatus (
            const QString & v )
```

Parameters

v	The status string.

9.1060.1.21 setThumbnail()

```
void Digikam::ProgressItem::setThumbnail (
            const QIcon & icon )
```

Parameters

icon	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
```

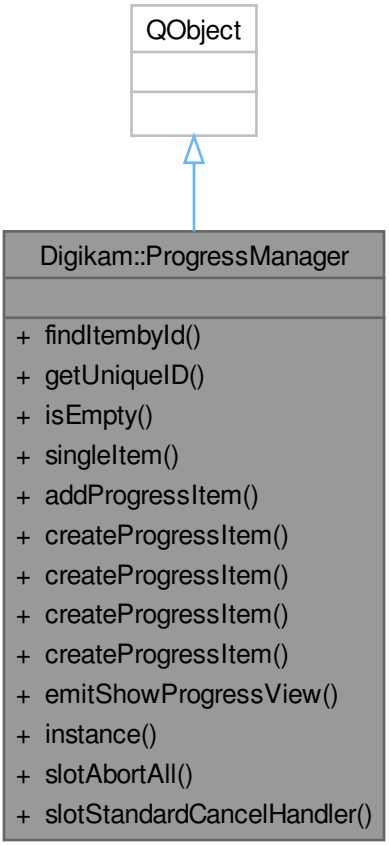
Returns

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

t	The process to add on manager.
parent	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

id	Used to identify this operation for cancel and progress info.
label	The text to be displayed by progress handlers

Parameters

status	Additional text to be displayed for the item.
canBeCanceled	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.
hasThumb	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

label	The text to be displayed by progress handlers
status	Additional text to be displayed for the item.
canBeCanceled	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.
hasThumb	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

parent	Specify an already existing item as the parent of this one.
id	Used to identify this operation for cancel and progress info.
label	The text to be displayed by progress handlers
status	Additional text to be displayed for the item.

Parameters

canBeCanceled	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.
hasThumb	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

parent	Specify an already existing item as the parent of this one.
id	Used to identify this operation for cancel and progress info.
label	The text to be displayed by progress handlers
status	Additional text to be displayed for the item.
canBeCanceled	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.
hasThumb	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

id	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]
```

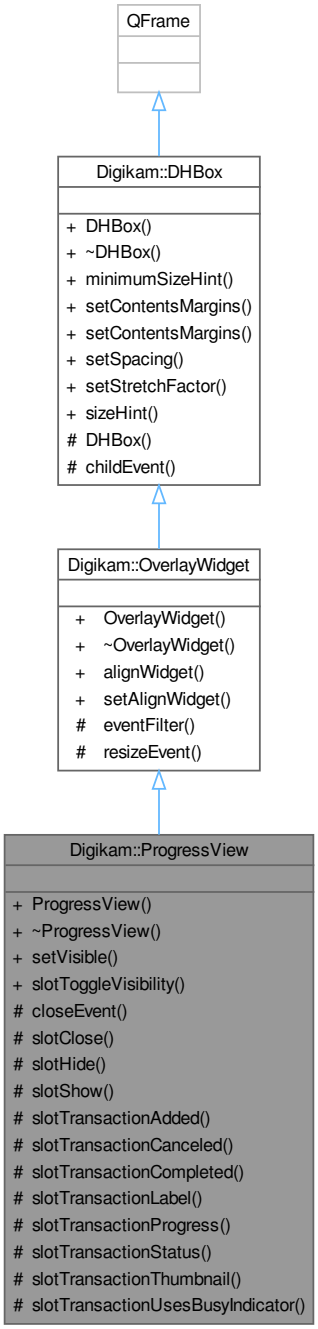
Provided for convenience.

Parameters

item	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()

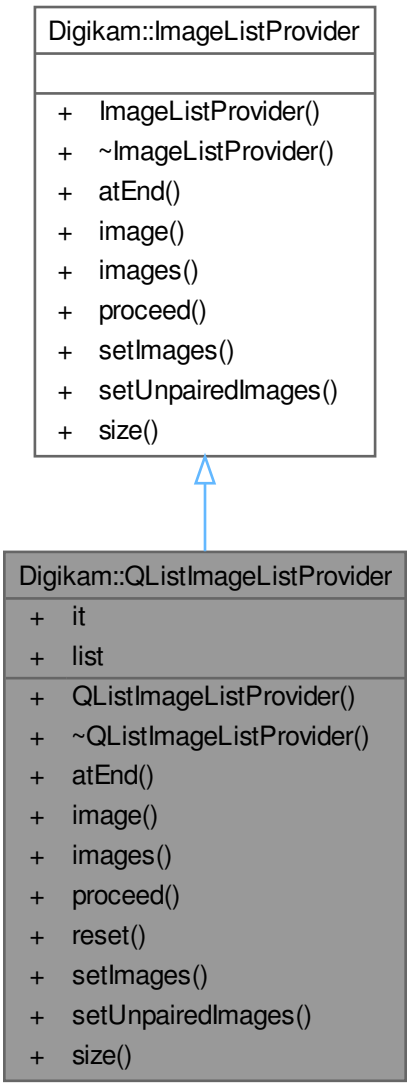
```
Digikam::ProxyLineEdit::ProxyLineEdit (
            QWidget *const parent = nullptr )  [explicit]
```

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()

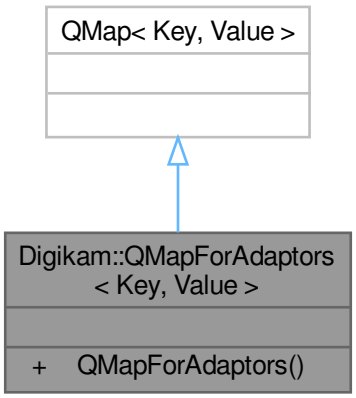
```
int Digikam::QListImageListProvider::size ( ) const  [override], [virtual]
```

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 >:

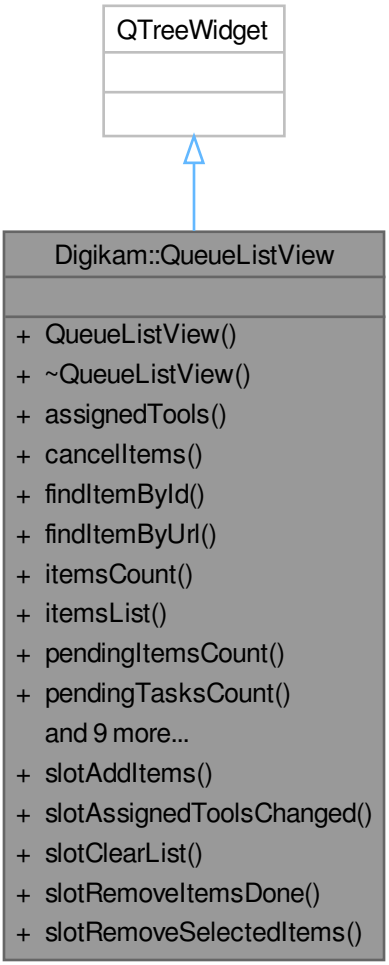


Public Types

- 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 **cancelItems** ()
- QueueListViewItem ∗ **findItemById** (qlonglong 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** (qlonglong id)
- void **removeItemByInfo** (const ItemInfo &info)
- void **setAssignedTools** (const AssignedBatchTools &tools)
- void **setEnableToolTips** (bool val)
- void **setItemBusy** (qlonglong id)
- void **setSettings** (const QueueSettings &settings)
- QueueSettings **settings** () const

9.1067.1 Member Enumeration Documentation

9.1067.1.1 ItemListType

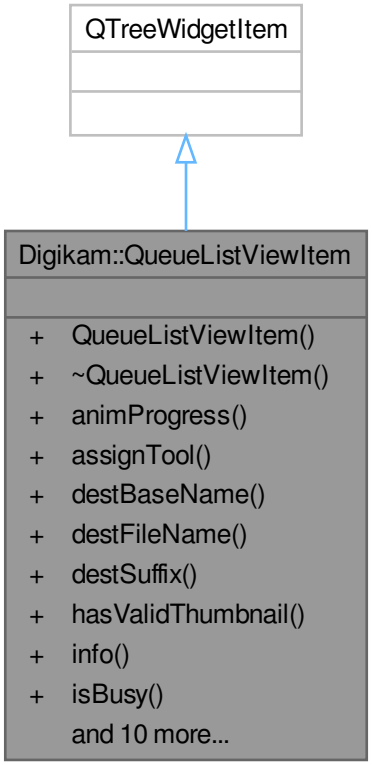
`enum Digikam::QueueListView::ItemListType`

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 * **infoIface** (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 infoIface()

```
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** ()

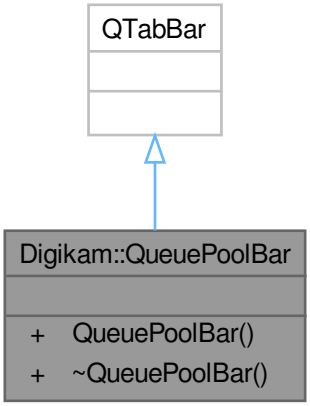
  Apply settings changes to all queues settings container when something have been changed in digiKam setup dialog.
- bool **assignedBatchToolsListsAreValid** () const
- QueueListView * **currentQueue** () const
- QString **currentTitle** () const
- bool **customRenamingRulesAreValid** () const
- QueueListView * **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** (qlonglong 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

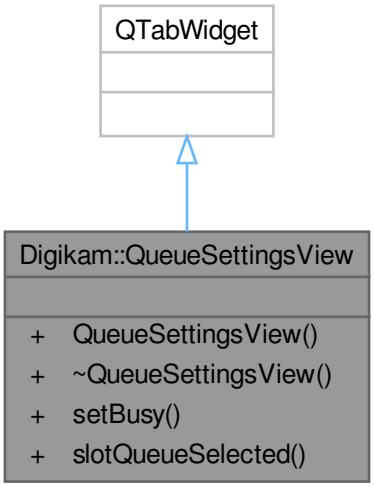
- FileSaveConflictBox::ConflictRule **conflictRule** = FileSaveConflictBox::DIFFNAME
- bool **exifSetOrientation** = true

  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 **toolTipIsEmpty** () 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 **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

9.1075.1 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().

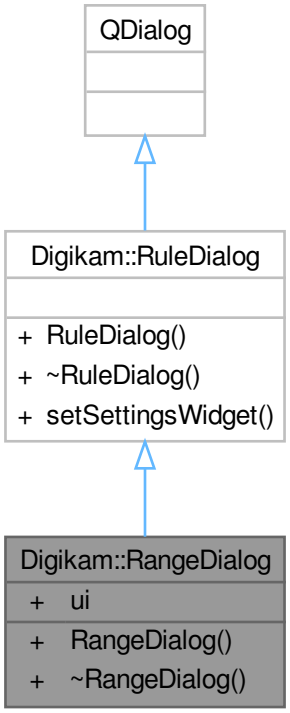
9.1076.3.6 seedNonDeterministic()

```
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

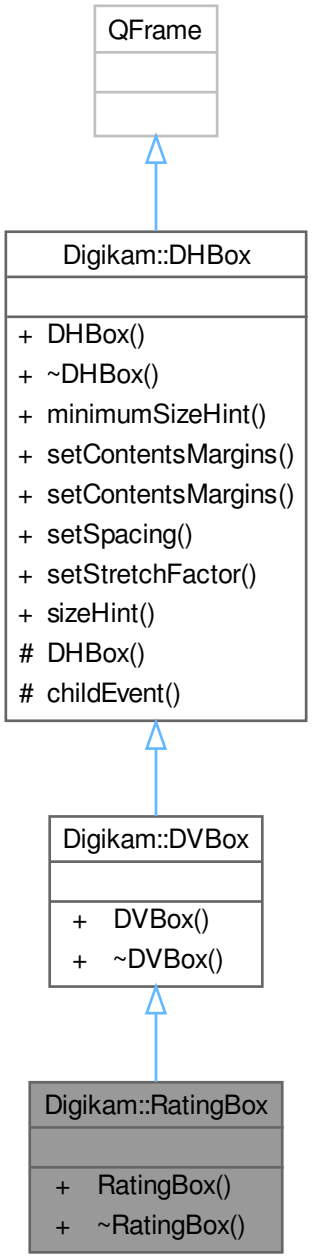
settings	contains settings
match	result of the regular expression match done in Option::parse()

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 **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

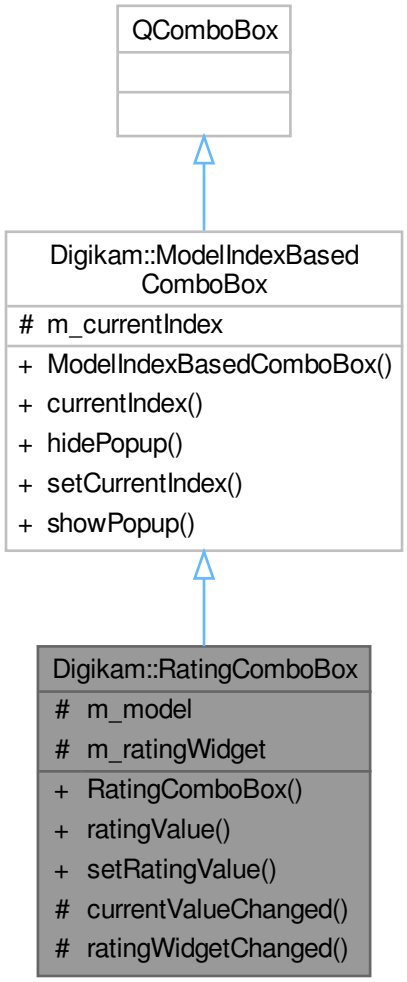
Additional Inherited Members

Protected Member Functions inherited from Digikam::DHBox

- **DHBox** (bool vertical, QWidget ∗const parent)
- void **childEvent** (QChildEvent ∗e) override

9.1080 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

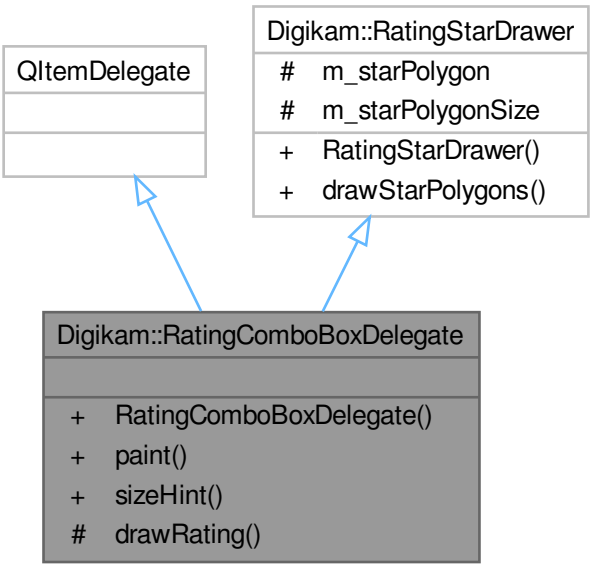
```
enum Digikam::RatingComboBox::RatingValue
```

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

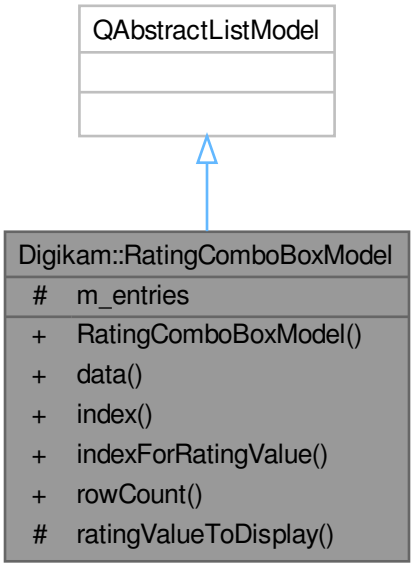
Additional Inherited Members

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

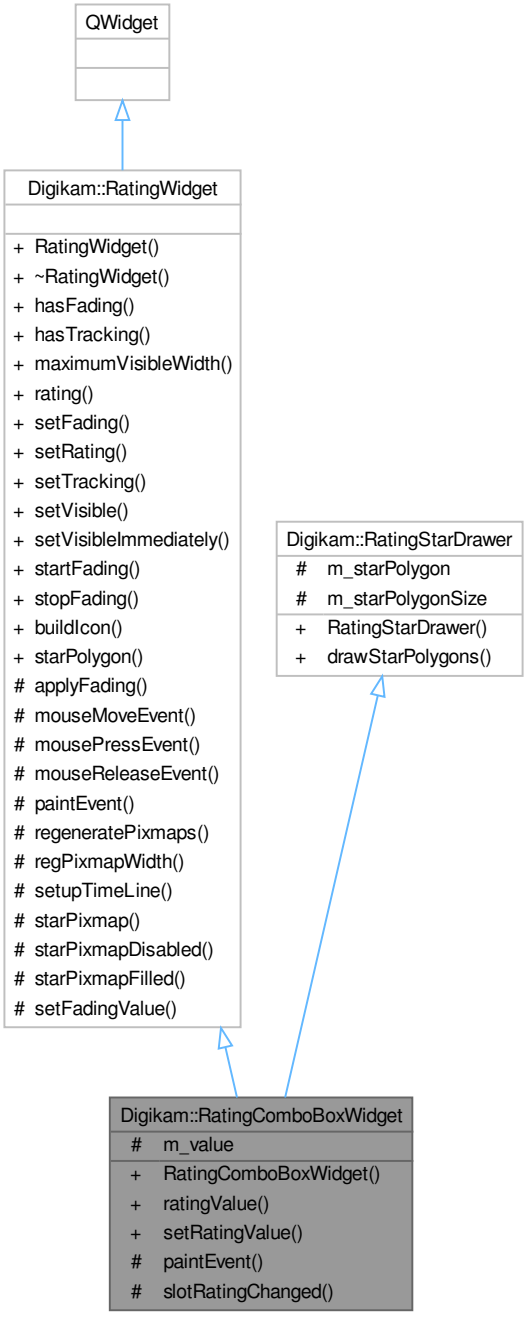
- QVariant **ratingValueToDisplay** (RatingComboBox::RatingValue value) const

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 **regeneratePixmaps** ()
- 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

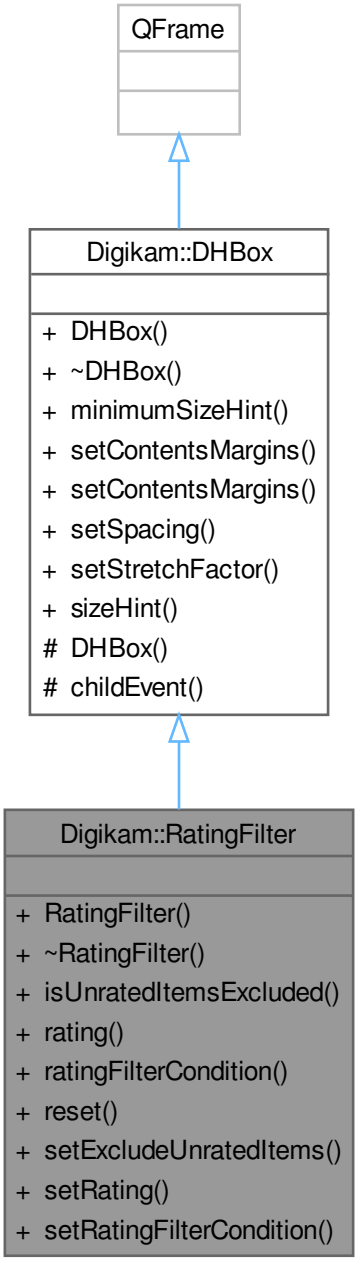
Static Public Member Functions inherited from Digikam::RatingWidget

- 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 **regeneratePixmaps** ()
- 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** ()

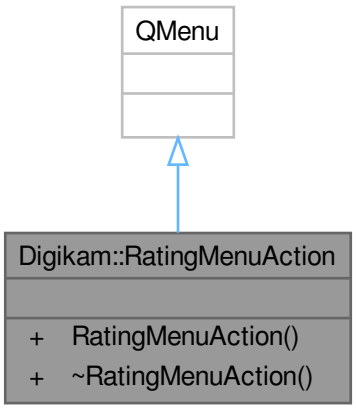
  Pre-computed star polygon for a 15x15 pixmap.

Protected Slots inherited from Digikam::RatingWidget

- void **setFadingValue** (int value)

9.1086 Digikam::RatingMenuAction Class Reference

Inheritance diagram for Digikam::RatingMenuAction:



Signals

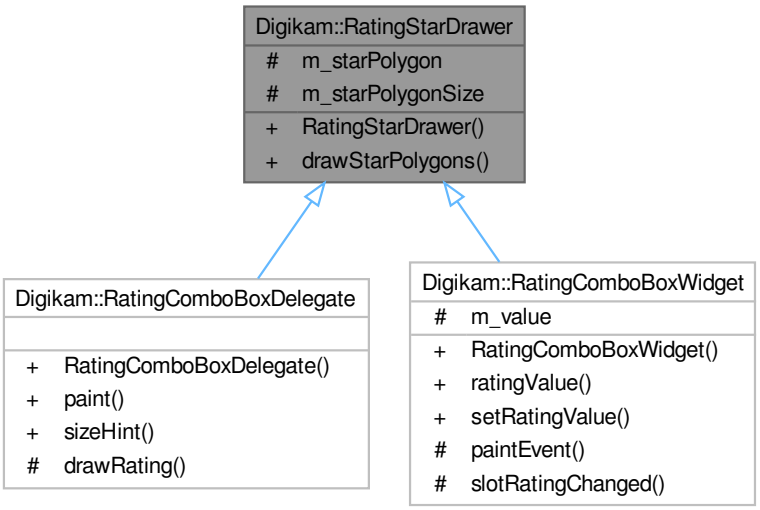
- void **signalRatingChanged** (int)

Public Member Functions

- **RatingMenuAction** (QMenu ∗const parent=nullptr)

9.1087 Digikam::RatingStarDrawer Class Reference

Inheritance diagram for Digikam::RatingStarDrawer:



Public Member Functions

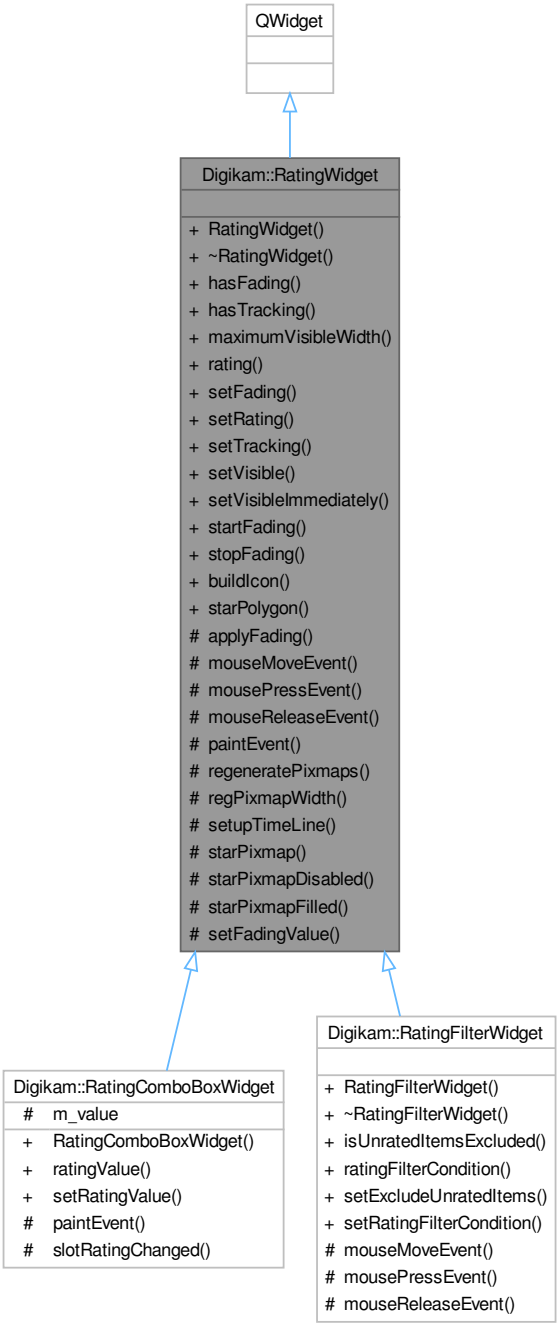
- QRect **drawStarPolygons** (QPainter ∗p, int numberOfStars) const

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 **regeneratePixmaps** ()
- 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

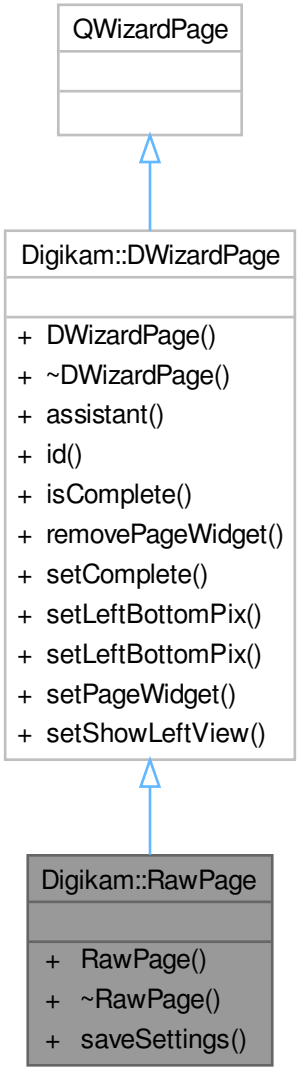
- **RawCameraDlg** (QWidget ∗const parent)

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

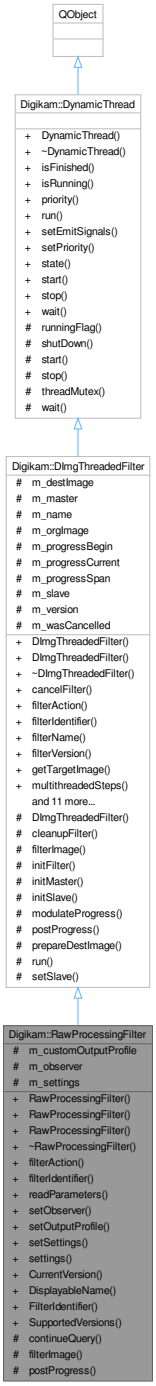
- **DWizardPage** (QWizard ∗const dlg, const QString &title)
- QWizard ∗ **assistant** () const

- 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. It's 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

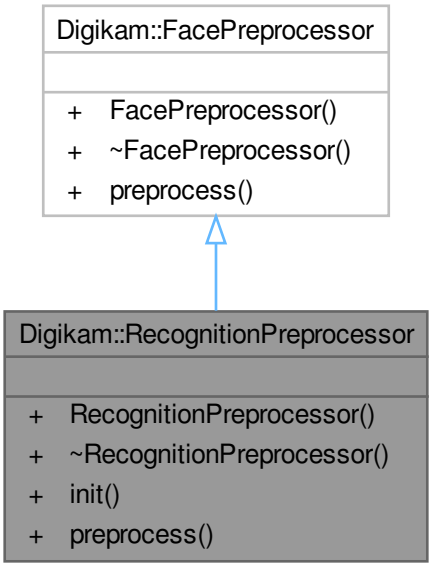
9.1093 Digikam::RecognitionBenchmarker::Statistics Class Reference

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()

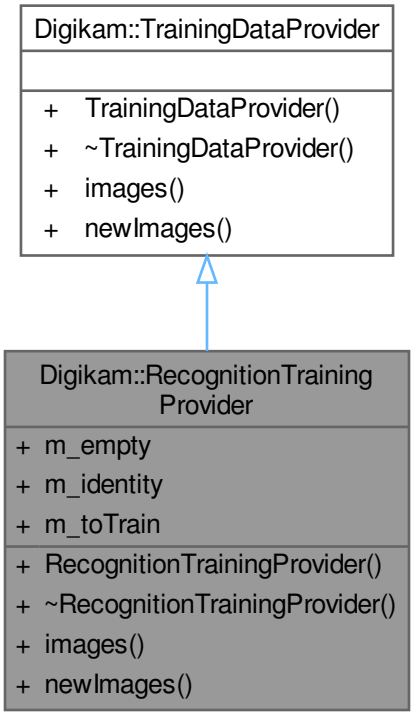
```
cv::Mat Digikam::RecognitionPreprocessor::preprocess (
            const cv::Mat & image ) const  [override], [virtual]
```

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
- FaceItemRetriever **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

feature_pixel_values	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	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 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** ()

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.

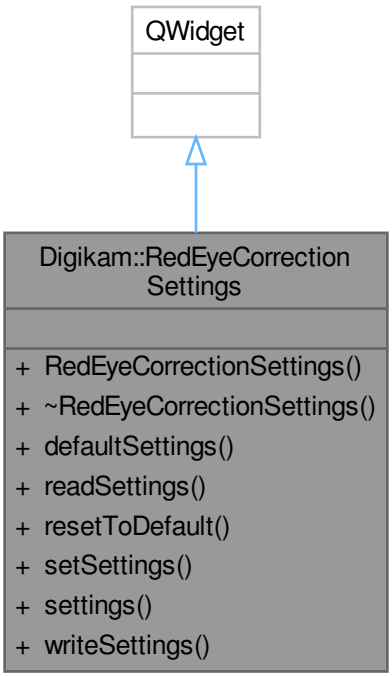
9.1102.1.2 filterIdentifier()

```
QString Digikam::RedEyeCorrectionFilter::filterIdentifier ( ) const  [inline], [override],
[virtual]
```

Implements Digikam::DImgThreadedFilter.

9.1103 Digikam::RedEyeCorrectionSettings Class Reference

Inheritance diagram for Digikam::RedEyeCorrectionSettings:



Signals

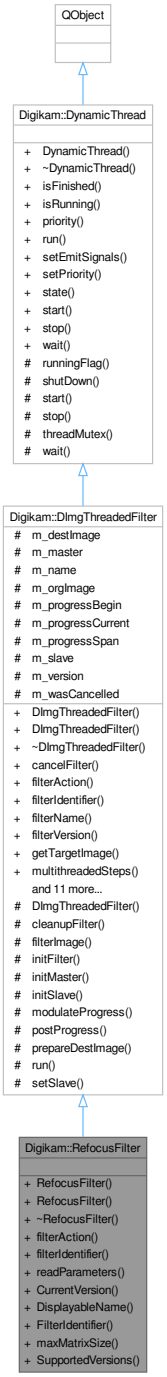
- void **signalSettingsChanged** ()

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** (DImg *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 **DisplayableName** ()
- 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.

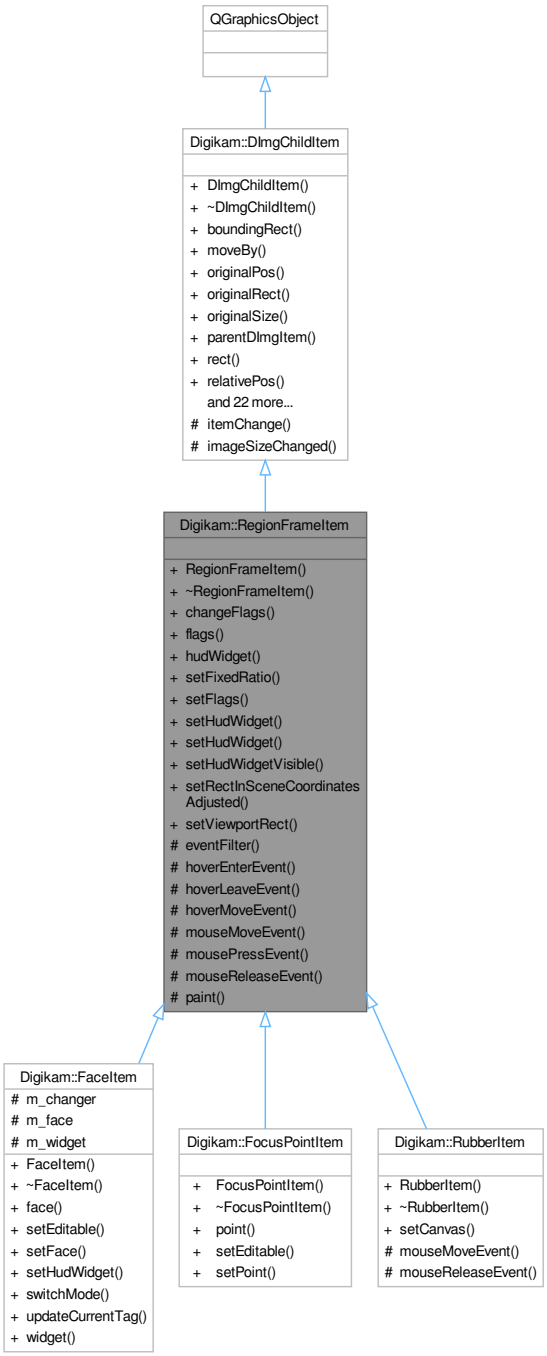
9.1105 Digikam::RefocusMatrix Class Reference

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.

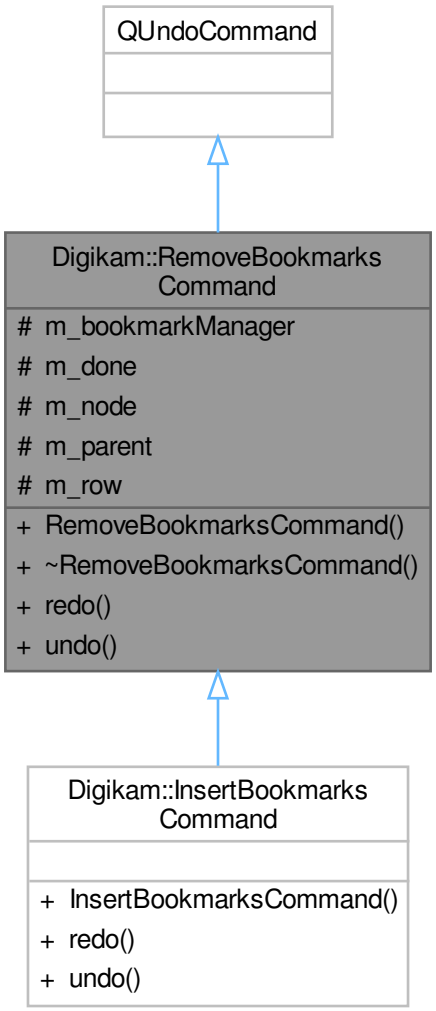
9.1106.1.2 setViewportRect

```
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

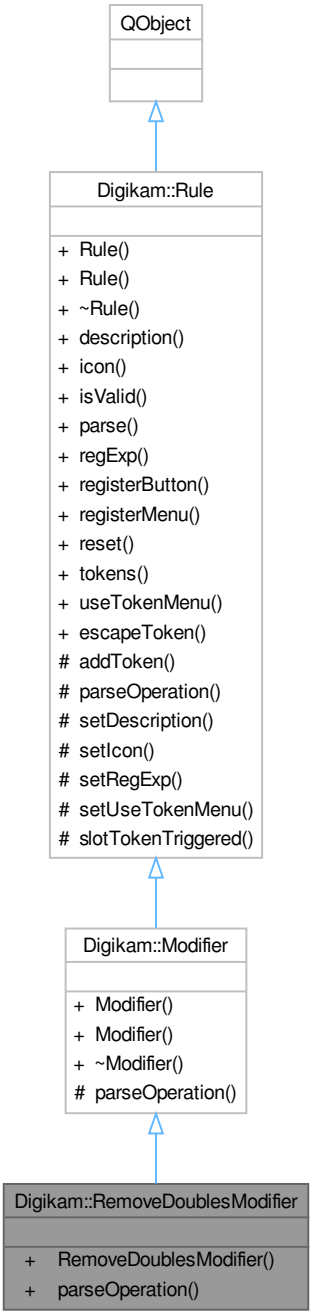
- **RemoveBookmarksCommand** (BookmarksManager ∗const mngr, BookmarkNode ∗const parent, int row)
- void **redo** () override
- void **undo** () override

Protected Attributes

- BookmarksManager ∗ **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

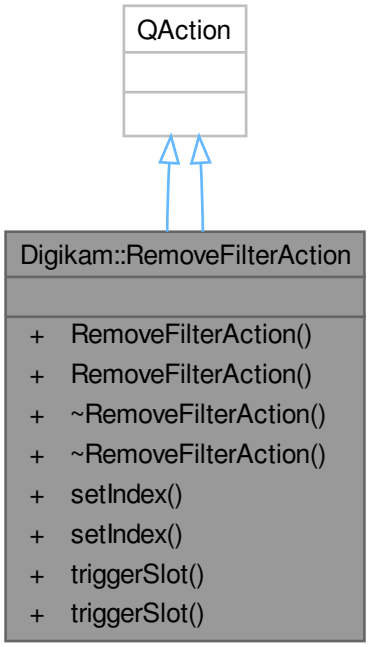
settings	contains settings
match	result of the regular expression match done in Option::parse()

Returns

Implements Digikam::Modifier.

9.1109 Digikam::RemoveFilterAction Class Reference

Inheritance diagram for Digikam::RemoveFilterAction:



Public Slots

- void **triggerSlot** ()
- void **triggerSlot** ()

Signals

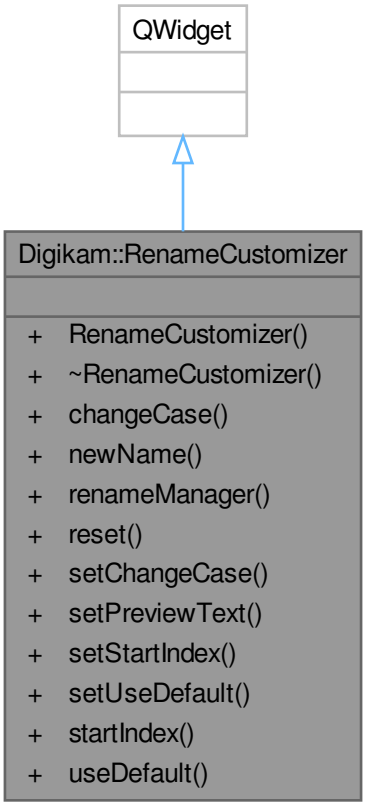
- void **actionTriggered** (QModelIndex index)
- void **actionTriggered** (QModelIndex index)

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

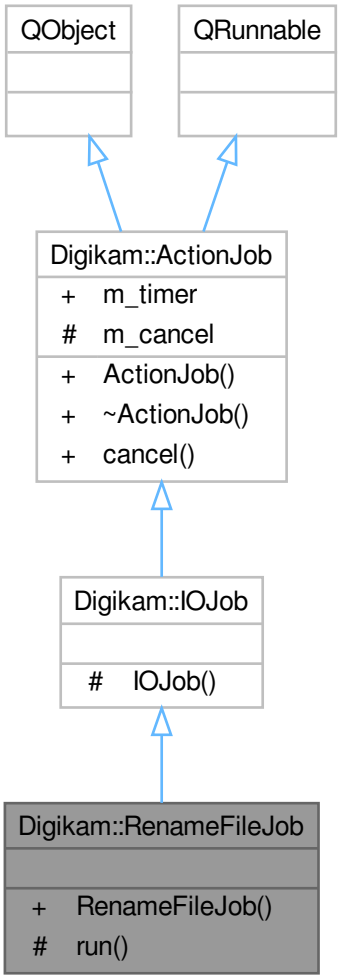
- void **signalChanged** ()

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 **setUseDefault** (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 **signalOneProccessed** (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.

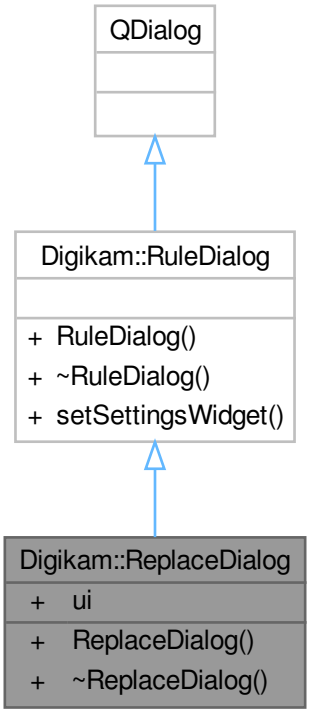
Protected Attributes inherited from Digikam::ActionJob

- 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

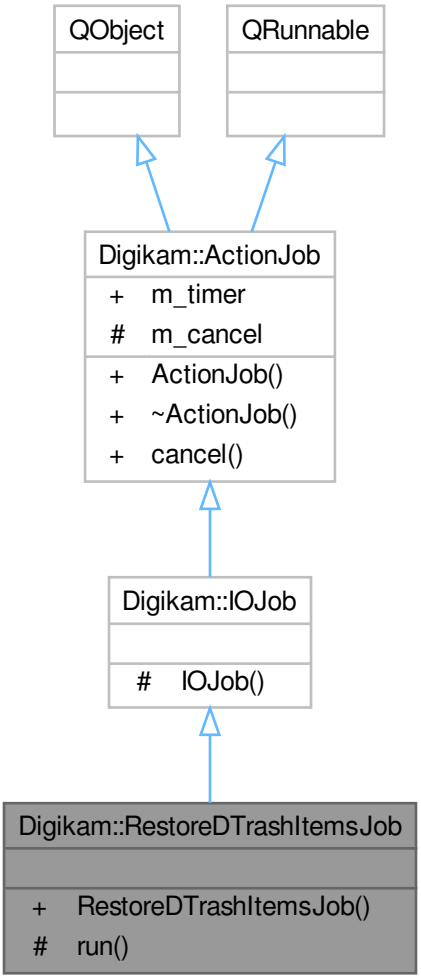
settings	contains settings
match	result of the regular expression match done in Option::parse()

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 **signalOneProccessed** (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 callRGBackend()

```
virtual void Digikam::RGBackend::callRGBackend (
            const QList< RGInfo > & ,
            const QString &  )  [pure virtual]
```

Implemented in Digikam::BackendGeonamesRG, Digikam::BackendGeonamesUSRG, and Digikam::BackendOsmRG.

9.1115.1.3 getErrorMessage()

```
QString Digikam::RGBackend::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.

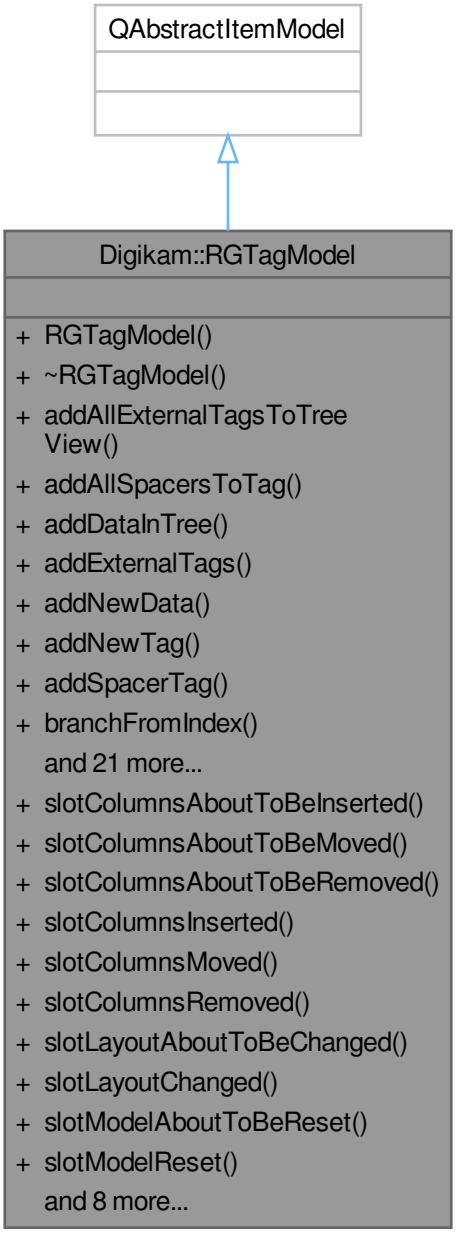
Public Attributes

- 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 readdNewTags (const QList< QList< TagData > > &tagAddressList)

  Takes each tag contained in tagAddressList and adds it to the tag tree.
- void readdTag (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 helds 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

externalTagModel	The tag model found in the host application.
parent	The parent object

9.1117.3 Member Function Documentation

9.1117.3.1 addDataInTree()

```
void Digikam::RGTagModel::addDataInTree (
            TreeBranch * 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 address↩ Elements list. If it's found, a new tag is added.

Parameters

currentBranch	The branch from where the scan starts.
currentRow	The row of the current branch.
addressElements	A list containing address elements. Example: {Country}, {City}...
elementsData	A list containing the name of each address element found in elements. Example: France, Paris...

9.1117.3.2 addExternalTags()

```
void Digikam::RGTagModel::addExternalTags (
            TreeBranch * parentBranch,
            int currentRow )
```

Parameters

parentBranch	The branch that will be parent for the old tag.
currentRow	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

elements	A list containing address elements. Example: {Country}, {City}...
resultedData	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

parent	The index of the parent.
newTagName	The name of the new tag.
newElement	The new element of the tag.

9.1117.3.5 addSpacerTag()

```
void Digikam::RGTagModel::addSpacerTag (
            const QModelIndex & parent,
            const QString & spacerName )
```

Parameters

parent	The index of the parent. If parent == QModelIndex(), then the spacer is added to top-level
spacerName	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

index	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

currentBranch	The branch from where the search starts.

9.1117.3.8 deleteAllSpacersOrNewTags()

```
void Digikam::RGTagModel::deleteAllSpacersOrNewTags (
            const QModelIndex & currentIndex,
            Type whatShouldRemove )
```

Parameters

currentIndex	If whatShouldRemove represents a spacer, the function will remove all spacers below currentIndex.If whatShouldRemove represents a new tag, the function will delete all new tags.
whatShouldRemove	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

currentIndex	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

currentBranch	The tree branch from where the scan starts.
currentRow	The row of current branch.
whatShouldRemove	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

currentBranch	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

index	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

tagAddressList	A list containing new tags.

9.1117.3.16 readdTag()

```
void Digikam::RGTagModel::readdTag (
            TreeBranch *& currentBranch,
            int currentRow,
            const QList< TagData > & tagAddressElements,
            int currentAddressElementIndex )
```

Parameters

currentBranch	The branch from where the scan starts.
currentRow	The row of the currentBranch.
tagAddressElements	A list containing address elements. Example: {Country}, {City}...
currentAddressElementIndex	The current element in the tag adress list.

Note

tagAddressElements contains address tag: Places,Spain,Barcelona readdTag 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
```

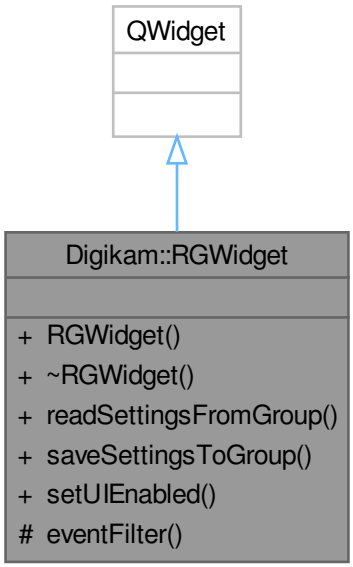
Returns

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 &cancel↩Slot)
- void signalUndoCommand (GPSUndoCommand *undoCommand)

  Sends the needed data to Undo/Redo Widget.

Public Member Functions

- RGWidget (GPSItemModel *const imageModel, QItemSelectionModel *const selectionModel, QAbstract↩ItemModel *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

imageModel	The image model
selectionModel	The image selection model
externTagModel	The tag model
parent	The parent object

9.1118.2 Member Function Documentation

9.1118.2.1 readSettingsFromGroup()

```
void Digikam::RGWidget::readSettingsFromGroup (
            const KConfigGroup *const group )
```

Parameters

group	Here are stored the settings.

9.1118.2.2 saveSettingsToGroup()

```
void Digikam::RGWidget::saveSettingsToGroup (
            KConfigGroup *const group )
```

Parameters

group	Here are stored the settings.

9.1118.2.3 setUIEnabled()

```
void Digikam::RGWidget::setUIEnabled (
```

```
            const bool state )
```

Parameters

state	If true, the controls are enabled.

9.1118.2.4 signalProgressChanged

```
void Digikam::RGWidget::signalProgressChanged (
            const int currentProgress )  [signal]
```

Parameters

currentProgress	The number of processed images.

9.1118.2.5 signalSetUIEnabled

```
void Digikam::RGWidget::signalSetUIEnabled (
            const bool enabledState )  [signal]
```

Parameters

enabledState	If true, the containing widgets will be enabled. Else, they will be disabled.

9.1118.2.6 signalUndoCommand

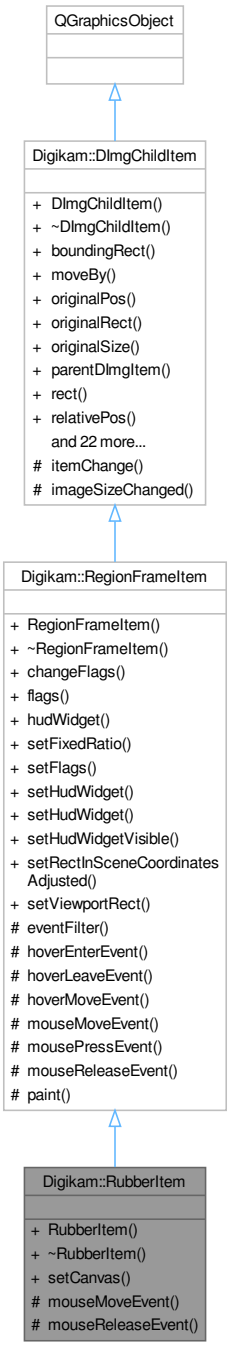
```
void Digikam::RGWidget::signalUndoCommand (
            GPSUndoCommand * undoCommand )  [signal]
```

Parameters

undoCommand	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 ()

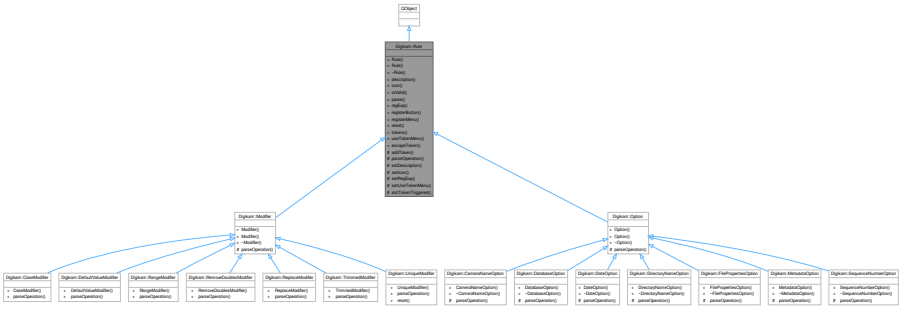
  These signals are emitted when the geometry, relative to the original image, of this item has changed.
- void **sizeChanged** ()
- void **sizeOnImageChanged** ()

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

id	the token id string (used for parsing)
description	the description of the token (used for example in the tooltip)
actionName	[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

token	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

settings	contains settings
match	result of the regular expression match done in Option::parse()

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

parent	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

parent	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

value	boolean parameter to set token menu usage

9.1120.1.10 tokens()

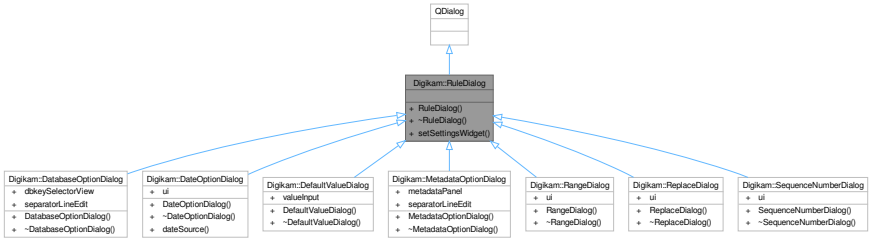
```
TokenList & Digikam::Rule::tokens ( ) const
```

Returns

a list of all registered tokens

9.1121 Digikam::RuleDialog Class Reference

Inheritance diagram for Digikam::RuleDialog:

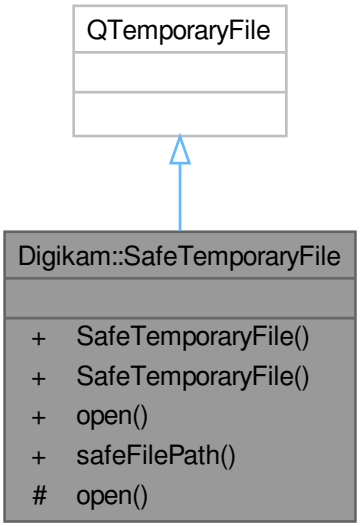


Public Member Functions

- **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

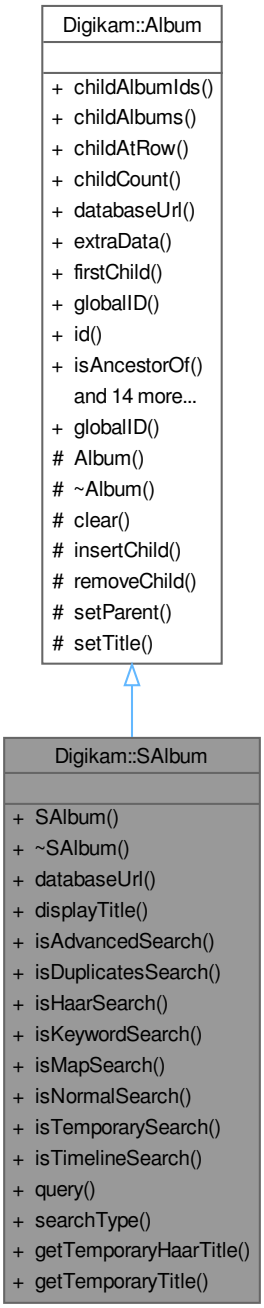
Protected Member Functions

- 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 haar↩ Type=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

haarType	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

type	the type of the search to get the temporary title for
haarType	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

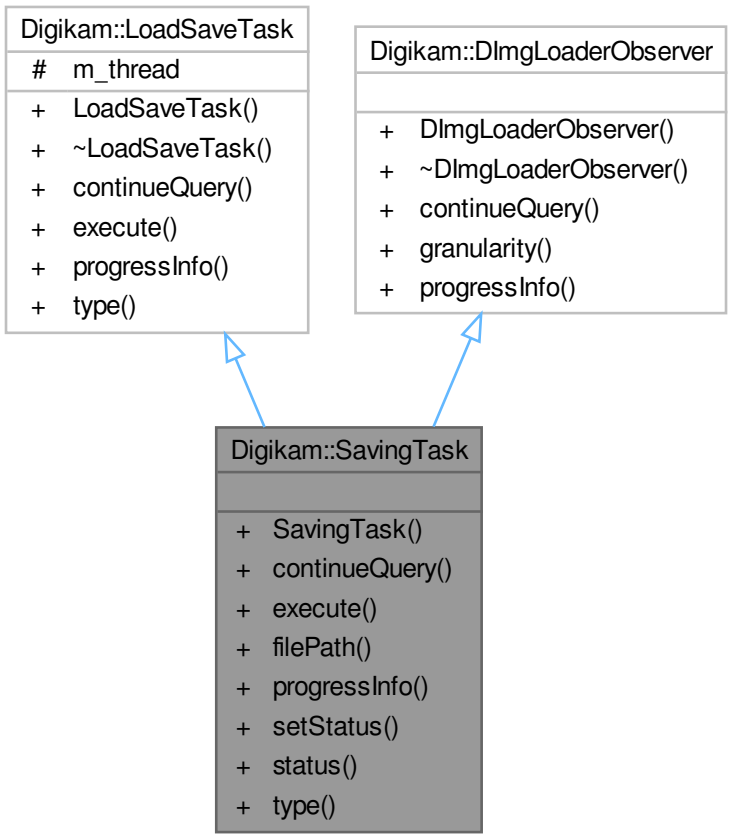
- enum **SavingState** { **SavingStateNone** , **SavingStateSave** , **SavingStateSaveAs** , **SavingStateVersion** }
- enum **SynchronizingState** { **NormalSaving** , **SynchronousSaving** }

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.

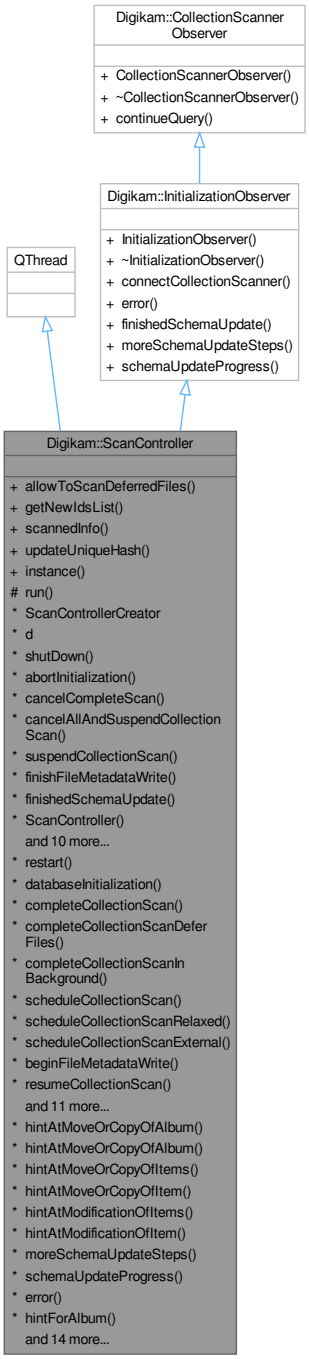
9.1126.1.4 type()

```
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 PAlbum *const album, const PAlbum *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 PAlbum *const album, const QString &dstPath, const QString &newAlbumName=QString())
- void hintAtMoveOrCopyOfItems (const QList< qlonglong > &ids, const PAlbum *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 PAlbum *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 allowToScan↩ DeferredFiles() 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

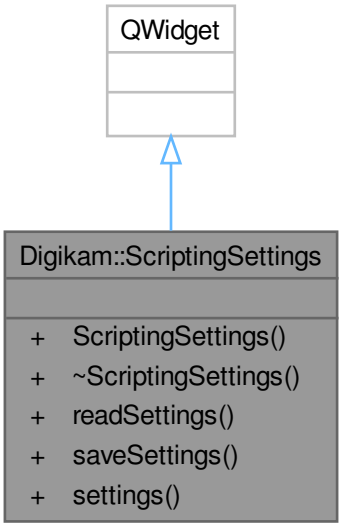
9.1129.1.1 run()

```
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 > &&imageIds, bool isAlbumUpdate, HaarIface::RefImageSelMethod referenceSelectionMethod, QSet< qlonglong > &&refImageIds)

- const QSet< qlonglong > & **imageIds** () const
- bool **isAlbumUpdate** () const
- bool **isDuplicatesJob** () const
- double **maxThreshold** () const
- double **minThreshold** () const
- const QSet< qlonglong > & **refImageIds** () const
- HaarIface::RefImageSelMethod **refImageSelectionMethod** () 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** ()

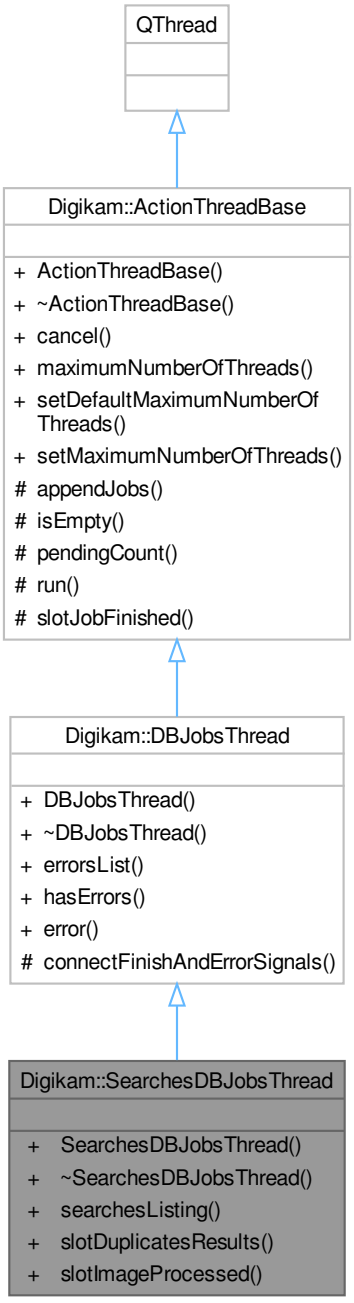
Public Attributes

- bool **m_albumUpdate** = false
- bool **m_duplicates** = false
- QSet< qlonglong > **m_imageIds**
- double **m_maxThreshold** = 1.0
- double **m_minThreshold** = 0.4
- QSet< qlonglong > **m_refImageIds**

  Image ids of the reference images if duplicates are available.
- HaarIface::RefImageSelMethod **m_refImageSelectionMethod** = HaarIface::RefImageSelMethod::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 HaarIface::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()

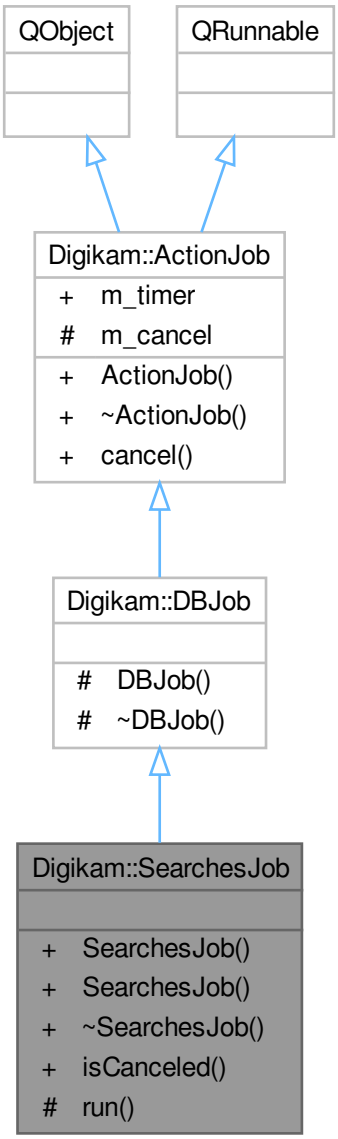
```
void Digikam::SearchesDBJobsThread::searchesListing (
            const SearchesDBJobInfo & info )
```

Parameters

info	represents the searches job info

9.1134 Digikam::SearchesJob Class Reference

Inheritance diagram for Digikam::SearchesJob:



Signals

- void **signalDuplicatesResults** (const HaarIface::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.

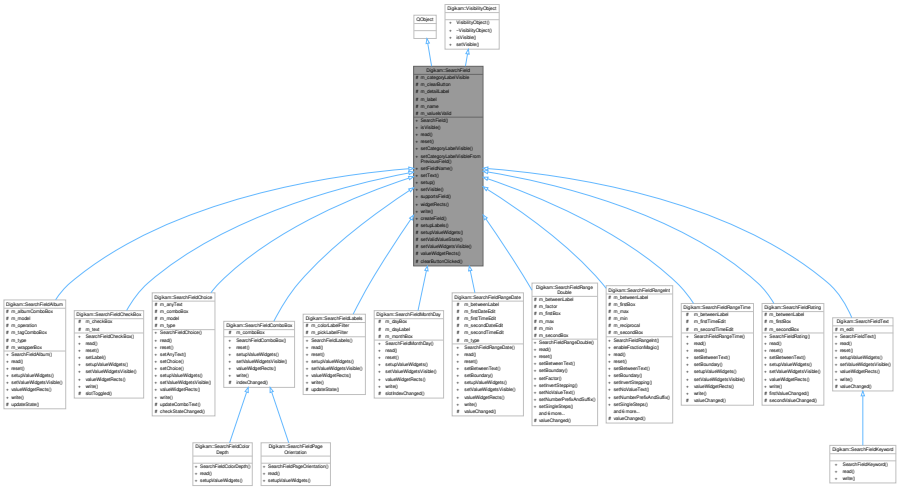
Protected Attributes inherited from Digikam::ActionJob

- 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_valueIsValid** = 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.

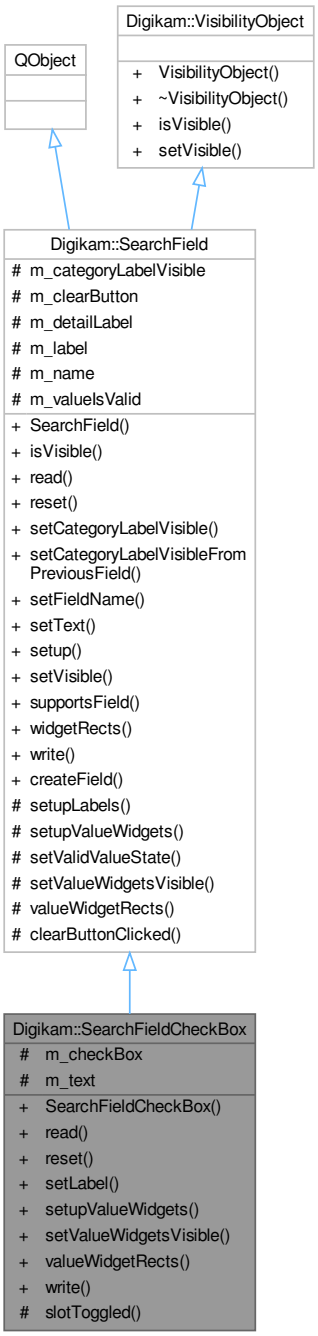
9.1136.1.6 write()

```
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_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)

9.1137.1 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 setupValueWidgets()

```
void Digikam::SearchFieldCheckBox::setupValueWidgets (
            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 valueIsValid)

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_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)

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 setupValueWidgets (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_valueIsValid** = 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_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)

9.1140.1 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

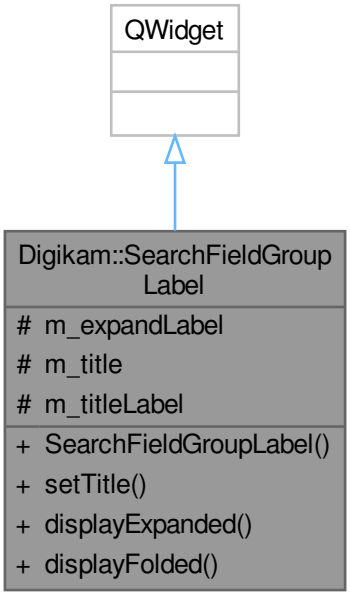
- void **slotLabelClicked** ()

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 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

- 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 setupValueWidgets()

```
void Digikam::SearchFieldLabels::setupValueWidgets (
            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.

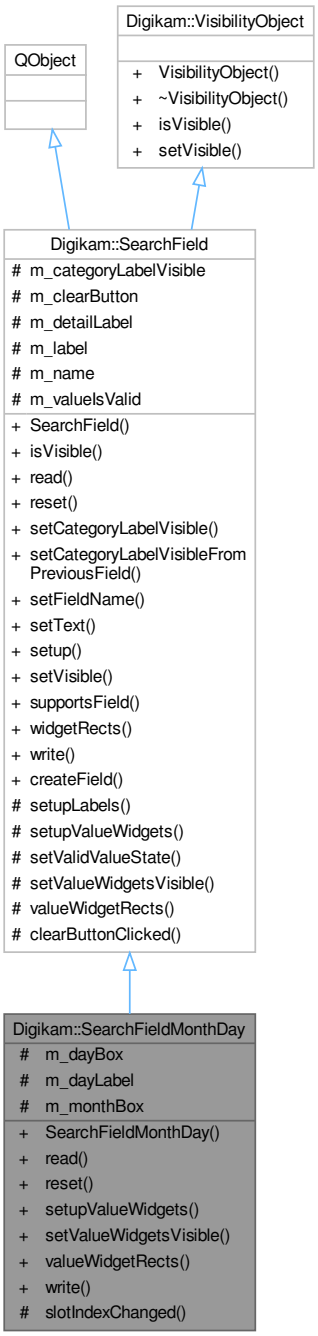
9.1144.1.6 write()

```
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 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 **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_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)

9.1145.1 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.

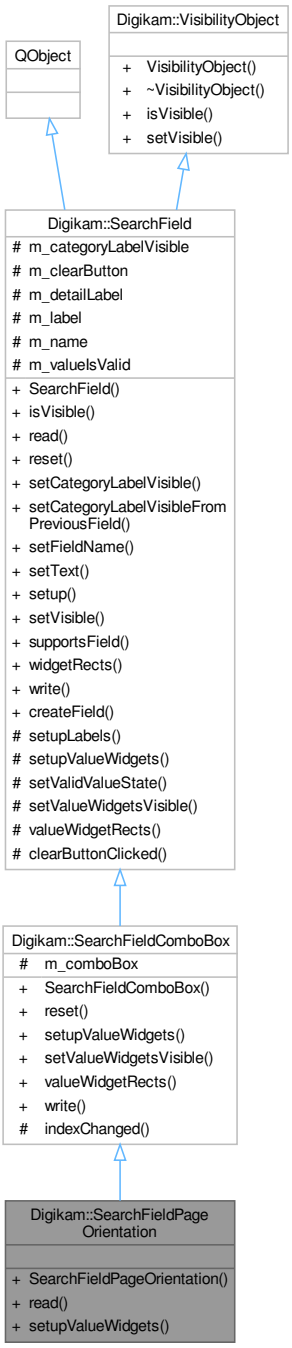
9.1145.1.6 write()

```
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 setupValueWidgets (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_valueIsValid** = 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_valueIsValid** = 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)

9.1147.1 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_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)

9.1148.1 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_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)

9.1149.1 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_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)

9.1150.1 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_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)

9.1151.1 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 setupValueWidgets()

```
void Digikam::SearchFieldRating::setupValueWidgets (
            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 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** (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_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)

9.1152.1 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()

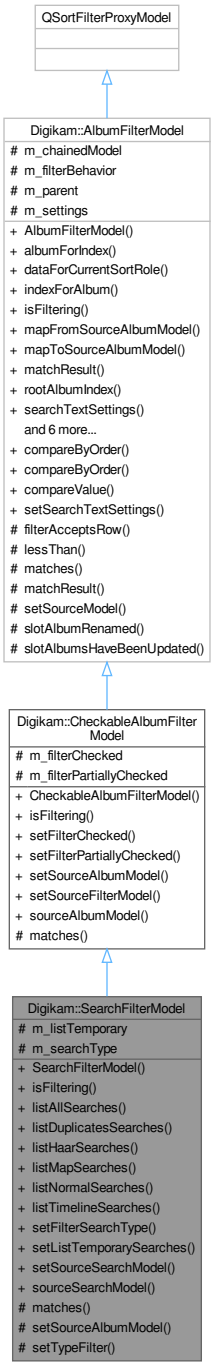
```
void Digikam::SearchFieldText::write (
            SearchXmlWriter & writer )  [override], [virtual]
```

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.

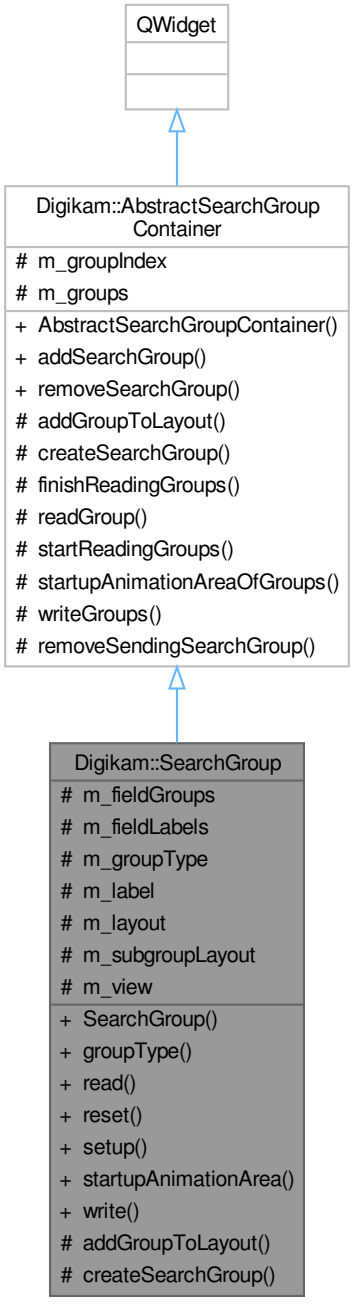
Parameters

album	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.

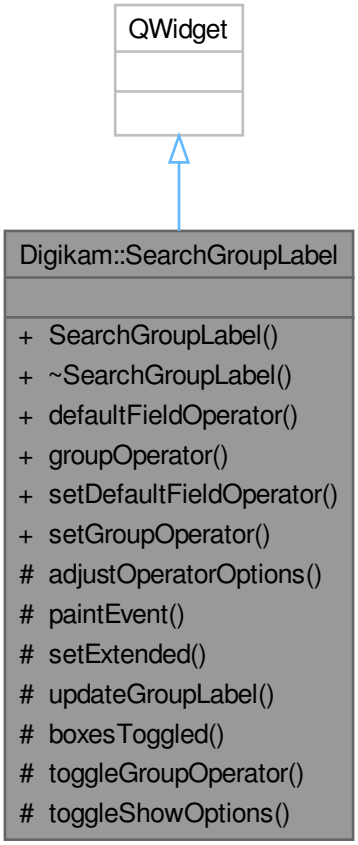
9.1154.1.2 createSearchGroup()

```
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 root↩ Behavior=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()

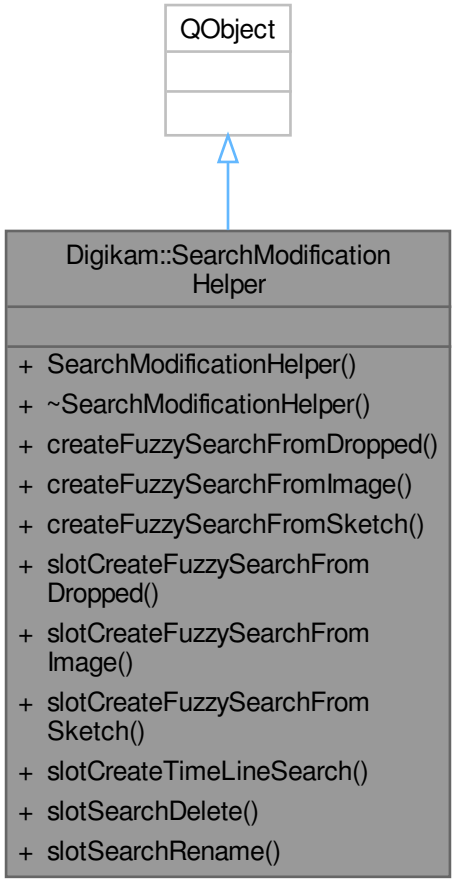
```
void Digikam::SearchModel::setReplaceNames (
            const QHash< QString, QString > & replaceNames )
```

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 overwriteIfExisting=false)
- SAlbum * createFuzzySearchFromImage (const QString &name, const ItemInfo &image, float threshold, float maxThreshold, const QList< int > &targetAlbums, bool overwriteIfExisting=false)
- SAlbum * createFuzzySearchFromSketch (const QString &name, SketchWidget *sketchWidget, unsigned int numberOfResults, const QList< int > &targetAlbums, bool overwriteIfExisting=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

parent	the parent for qt parent child mechanism
dialogParent	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 overwriteIfExisting = 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 overwriteIfExisting = 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 overwriteIfExisting = 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 overwriteIfExisting )  [slot]
```

Parameters

name	of the new search
filePath	path of the image to base this search on
threshold	minimum threshold for image search
maxThreshold	maximum threshold for image search
targetAlbums	The image must be in one of these albums
overwriteIfExisting	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 overwriteIfExisting = false )  [slot]
```

Parameters

name	of the new search
image	the image to base this search on
threshold	the threshold for image search, 0 <= threshold <= 1
maxThreshold	the maximum threshold of similarity.
targetAlbums	The image must be in one of these albums
overwriteIfExisting	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 overwriteIfExisting = false )  [slot]
```

Parameters

name	the name of the new sketch search
sketchWidget	the widget containing the sketch of the user
numberOfResults	max number of results to display
targetAlbums	The image must be in one of these albums
overwriteIfExisting	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 overwriteIfExisting = false )  [slot]
```

Parameters

desiredName	desired name for the search. If this name already exists and overwriteIfExisting is false, then the user will be prompted for a new name
dateRanges	date ranges to contain in this timeline search. If this is empty, no search will be created.
overwriteIfExisting	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

searchAlbum	search to delete

9.1158.3.9 slotSearchRename

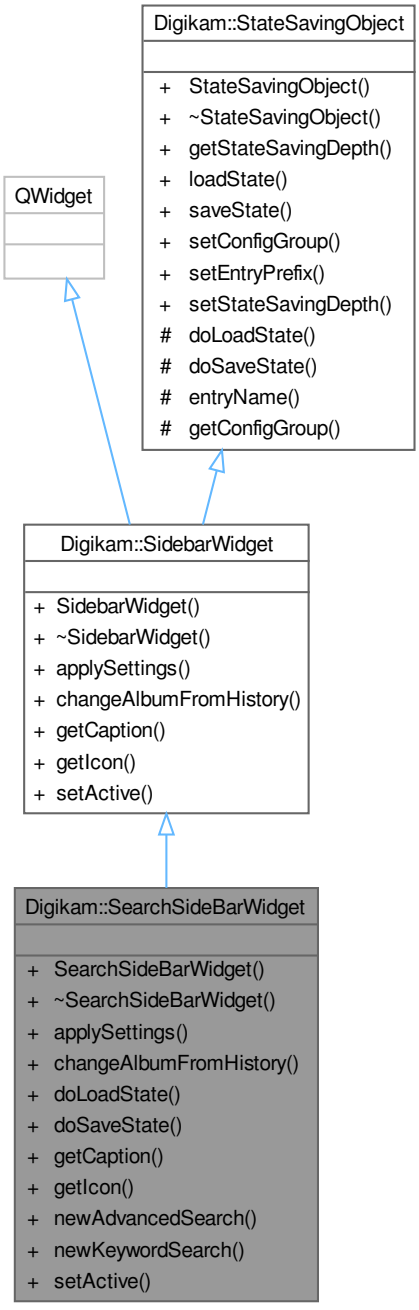
```
void Digikam::SearchModificationHelper::slotSearchRename (
            SAlbum * searchAlbum )  [slot]
```

Parameters

searchAlbum	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]
```

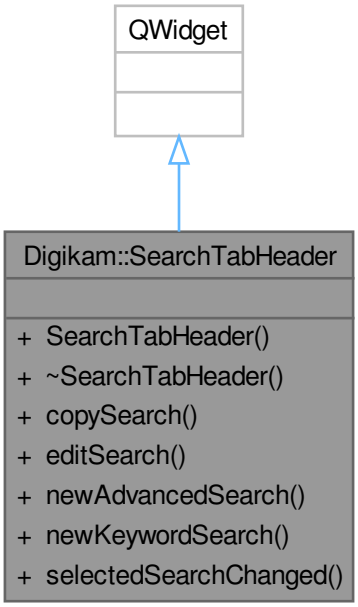
Parameters

active	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)

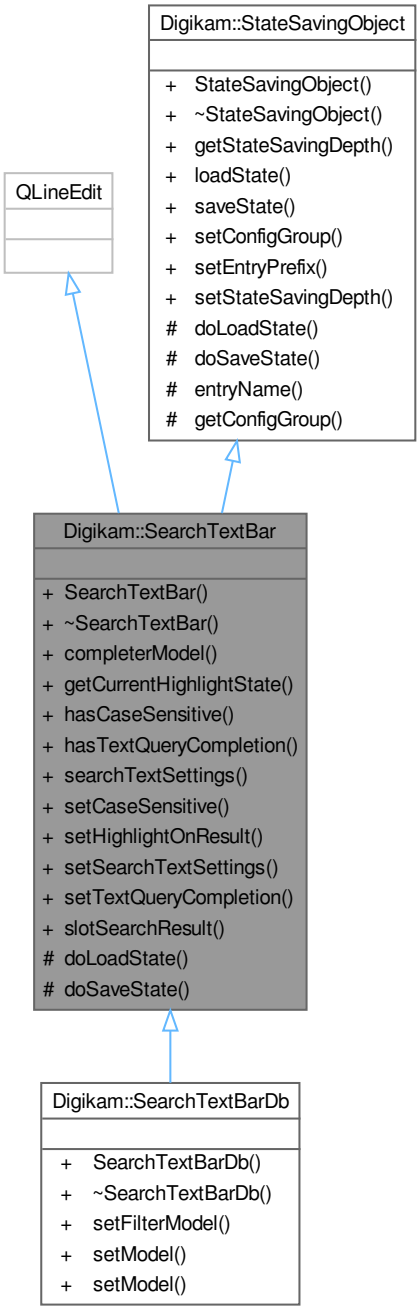
Public Member Functions

- **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

b	if `true` the user can decide the toggle between case sensitivity, on `false` every search is case- insensitive

9.1161.3.5 setHighlightOnResult()

```
void Digikam::SearchTextBar::setHighlightOnResult (
            bool highlight )
```

Default behavior has highlighting enabled.

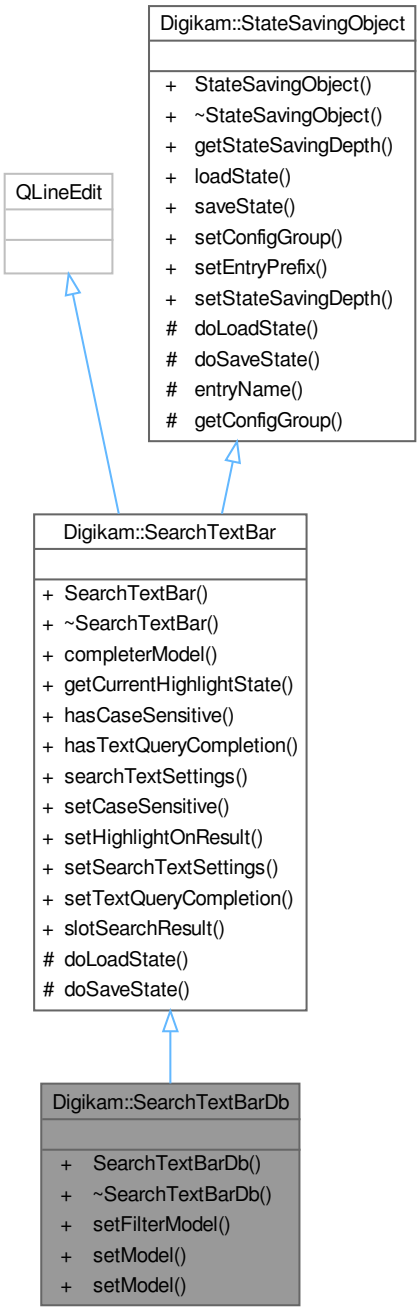
Parameters

highlight	`true` activates green and red highlighting, with `false` 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

filterModel	filter model to use for filtering. `null` means there is no model to use and external connections need to be created with signalSearchTextSettings and slotSearchResult

9.1162.2.2 setModel() [1/2]

```
void Digikam::SearchTextBarDb::setModel (
            AbstractAlbumModel *const model )
```

Parameters

model	album model to use for filtering. `null` means there is no model to use and external connections need to be created with signalSearchTextSettings and slotSearchResult

9.1162.2.3 setModel() [2/2]

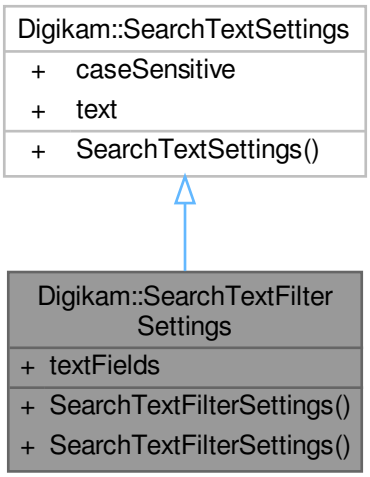
```
void Digikam::SearchTextBarDb::setModel (
            QAbstractItemModel * model,
            int uniqueIdRole,
            int displayRole = Qt::DisplayRole )
```

Parameters

model	to fill from or null for manual mode
uniqueIdRole	a role for which the model will return a unique integer for each entry
displayRole	the role to retrieve the text for completion, default is Qt::DisplayRole.

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 | Image↩
  PixelSize }

Public Member Functions

- **SearchTextFilterSettings** (const SearchTextSettings &settings)

Public Attributes

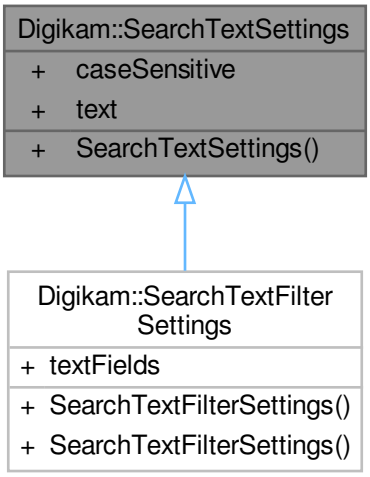
- TextFilterFields **textFields** = None

Public Attributes inherited from Digikam::SearchTextSettings

- Qt::CaseSensitivity **caseSensitive** = Qt::CaseInsensitive
- QString **text**

9.1164 Digikam::SearchTextSettings Class Reference

Inheritance diagram for Digikam::SearchTextSettings:

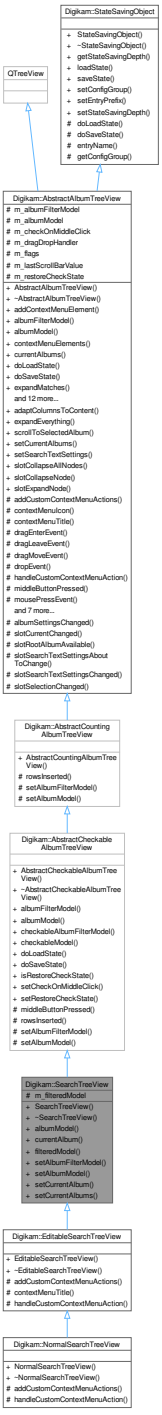


Public Attributes

- 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 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

- 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 | 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.

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.

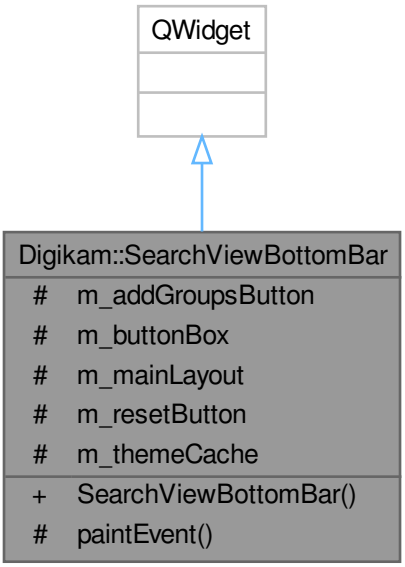
9.1166.1.4 groupLabelPixmap()

```
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

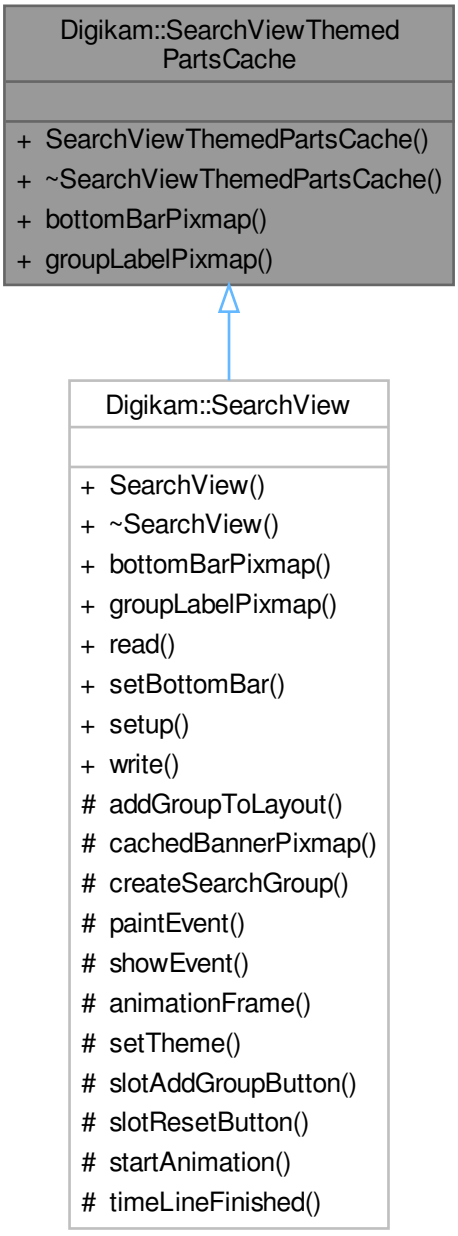
- void **paintEvent** (QPaintEvent ∗) override

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.

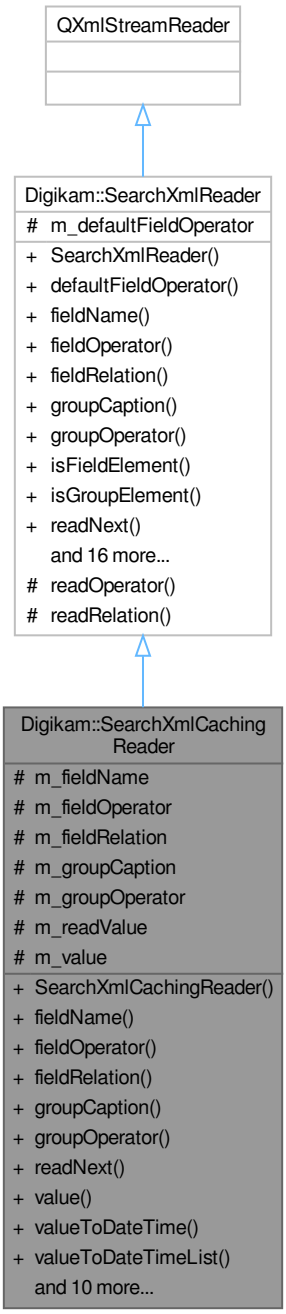
9.1169.1.3 searchEdited

```
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 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** ()

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**

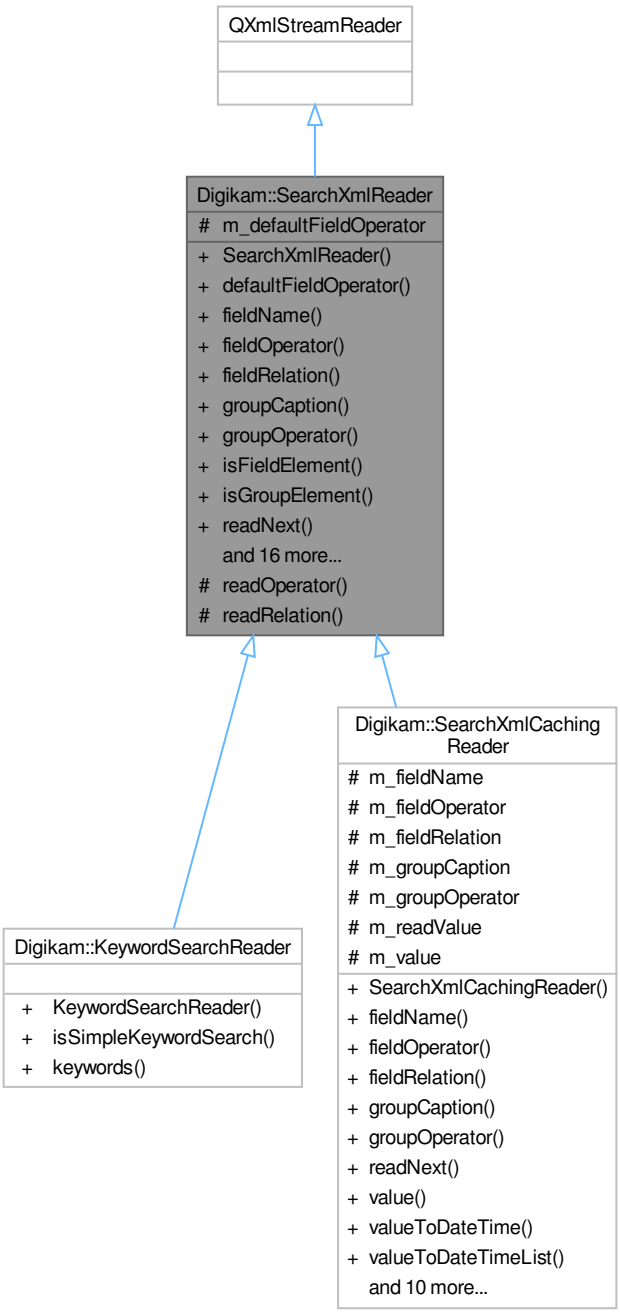
Additional Inherited Members

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 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** ()

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.

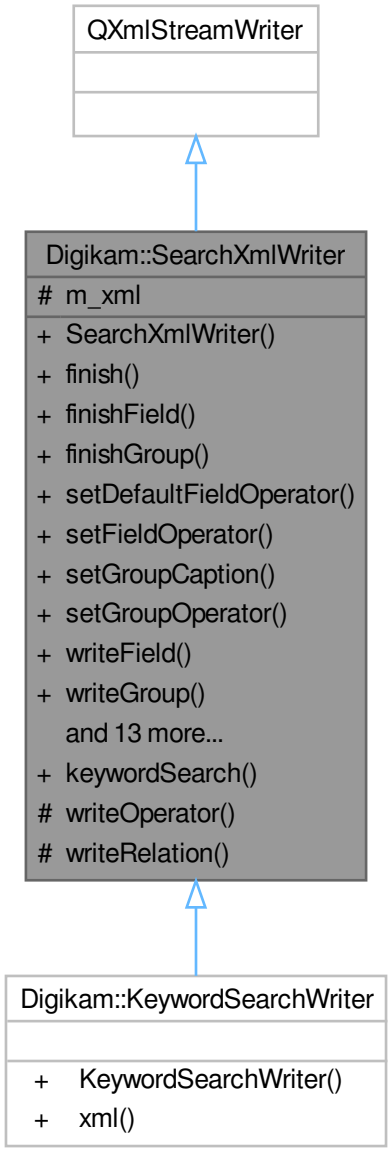
9.1171.1.7 value()

```
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.

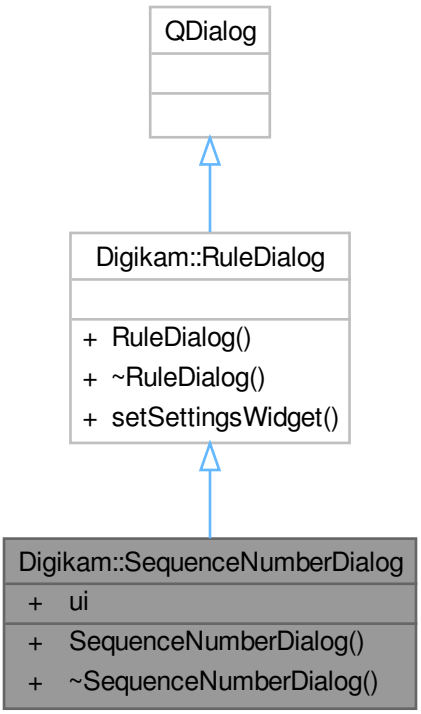
9.1172.1.8 xml()

```
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

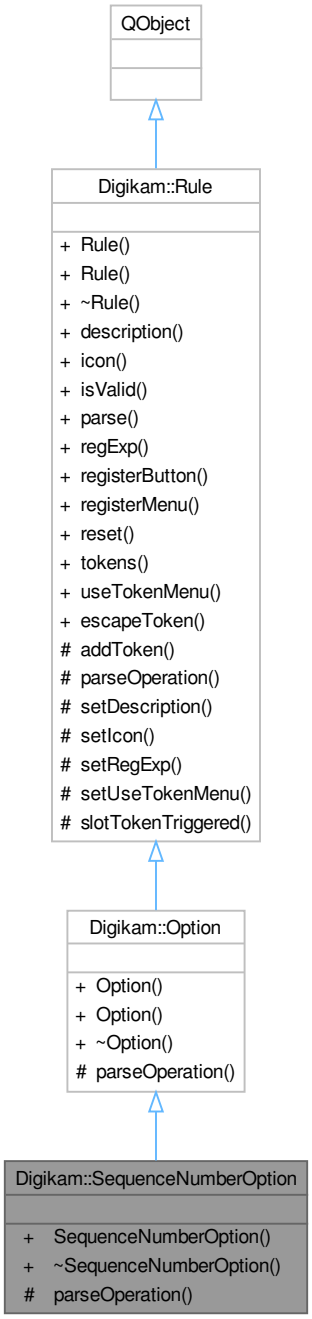
- **RuleDialog** (Rule ∗const parent)
- void **setSettingsWidget** (QWidget ∗const settingsWidget)

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

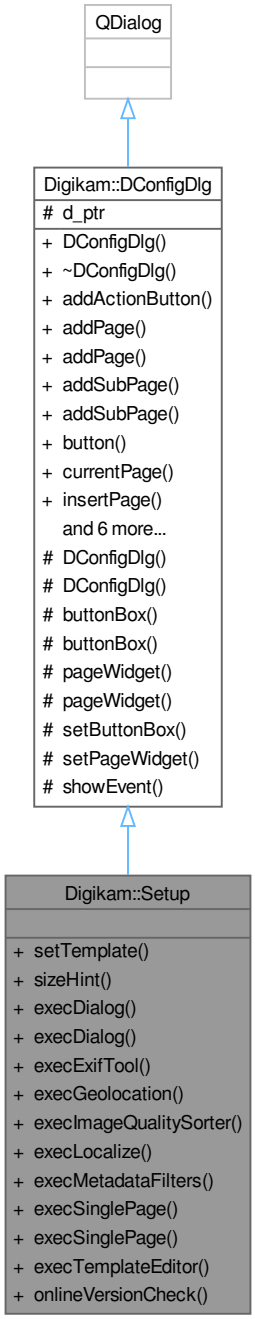
settings	contains settings
match	result of the regular expression match done in Option::parse()

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.

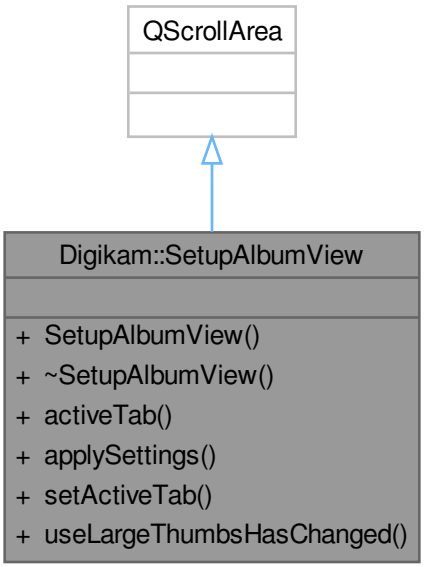
9.1175.1.2 execSinglePage()

```
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

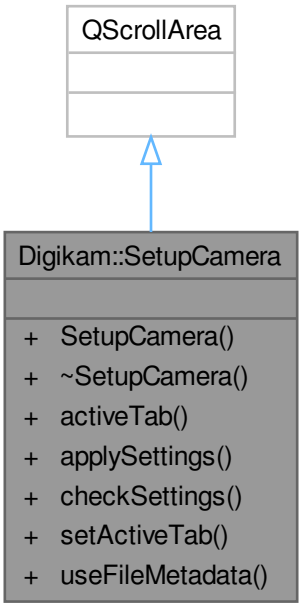
- enum **AlbumTab** {
  IconView = 0 , **FolderView** , **Preview** , **FullScreen** ,
  MimeType , **Category** }

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

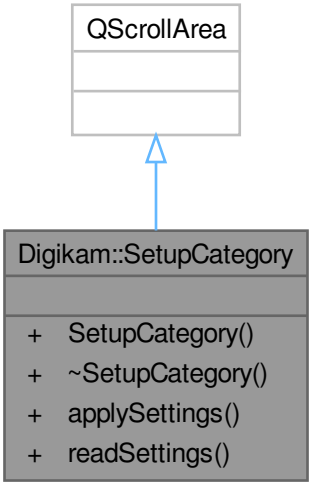
- void **signalUseFileMetadataChanged** (bool)

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:

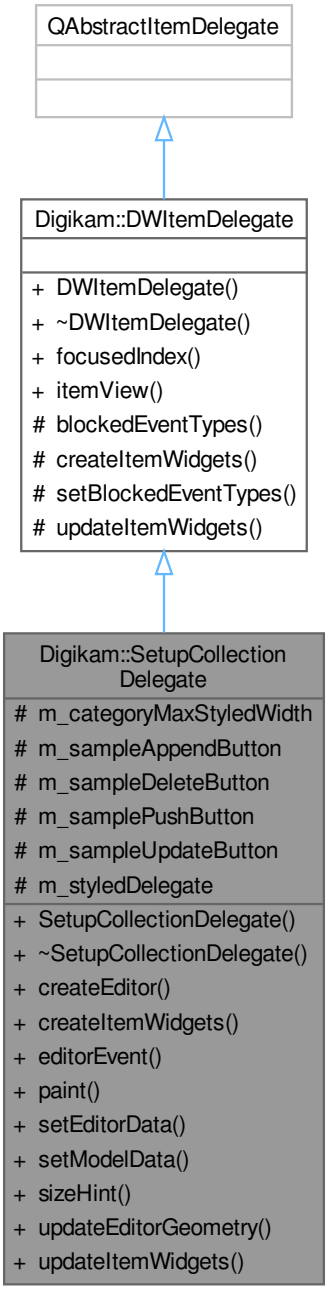


Public Member Functions

- **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
- QStyledItemDelegate ∗ **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.

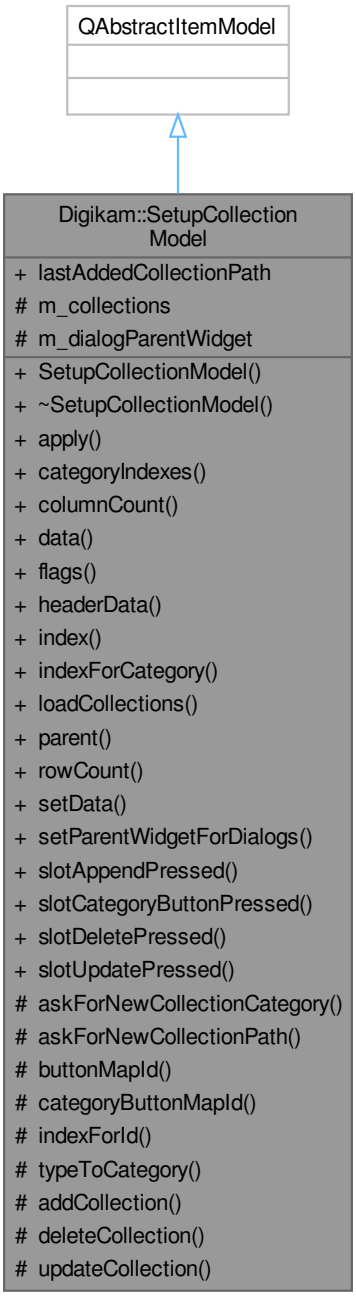
Parameters

widgets	the widgets to update
option	the current set of style options for the view.
index	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 , **NumberOf↩ Categories** }
- 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::User↩ Role + 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 new↩ Label)
- 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

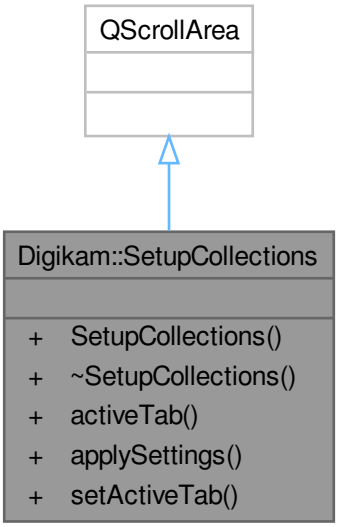
- **Item** (const CollectionLocation &location)
- **Item** (const QString &path, const QString &label, SetupCollectionModel::Category category)

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

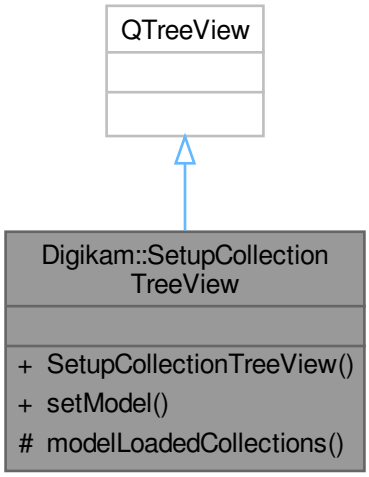
- enum **CollectionsTab** { **Collections** = 0 , **IgnoreDirs** }

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

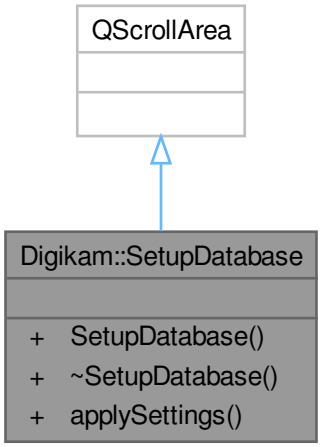
- **SetupCollectionTreeView** (QWidget ∗const parent=nullptr)
- void **setModel** (SetupCollectionModel ∗model)

Protected Slots

- void **modelLoadedCollections** ()

9.1184 Digikam::SetupDatabase Class Reference

Inheritance diagram for Digikam::SetupDatabase:

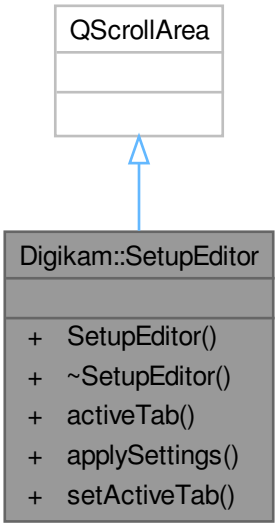


Public Member Functions

- **SetupDatabase** (QWidget ∗const parent=nullptr)
- void **applySettings** ()

9.1185 Digikam::SetupEditor Class Reference

Inheritance diagram for Digikam::SetupEditor:



Public Types

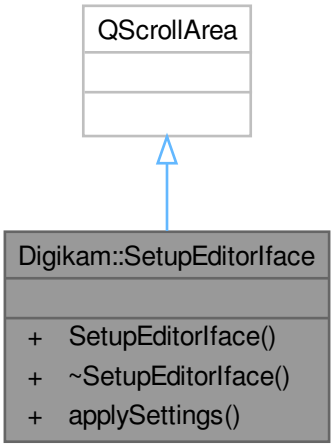
- enum **EditorTab** {
  EditorWindow = 0 , **Versioning** , **SaveSettings** , **RAWBehavior** ,
  RAWDefaultSettings }

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:

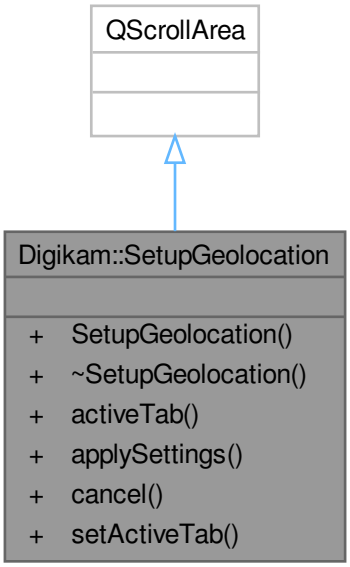


Public Member Functions

- **SetupEditorIface** (QWidget ∗const parent=nullptr)
- void **applySettings** ()

9.1187 Digikam::SetupGeolocation Class Reference

Inheritance diagram for Digikam::SetupGeolocation:



Public Types

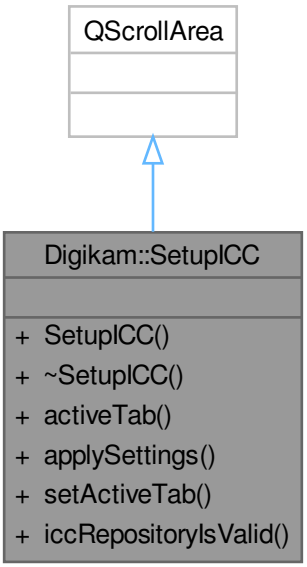
- enum **GeolocationTab** { **MarbleView** = 0 , **MarblePlugins** , **GoogleMaps** }

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** ()

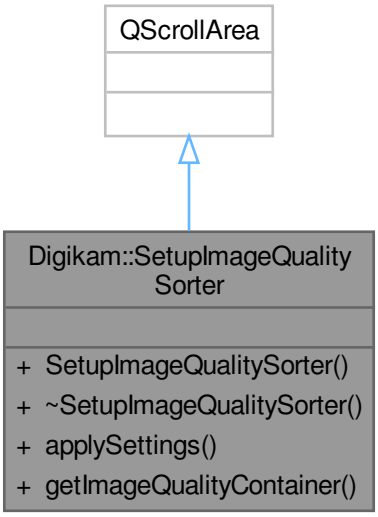
9.1188.1 Constructor & Destructor Documentation

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:

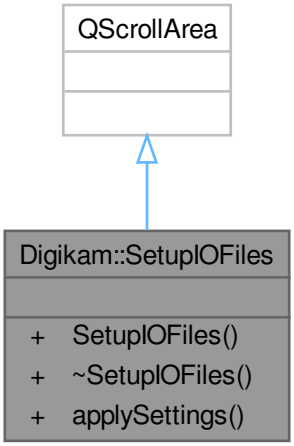


Public Member Functions

- **SetupImageQualitySorter** (QWidget ∗const parent=nullptr)
- void **applySettings** ()
- ImageQualityContainer **getImageQualityContainer** () const

9.1190 Digikam::SetupIOFiles Class Reference

Inheritance diagram for Digikam::SetupIOFiles:

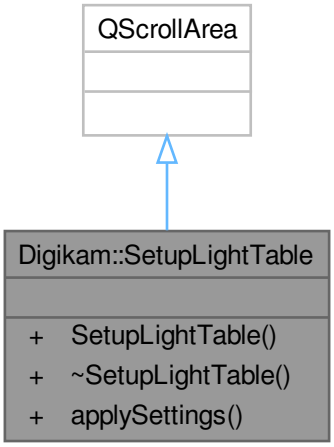


Public Member Functions

- **SetupIOFiles** (QWidget ∗const parent=nullptr)
- void **applySettings** ()

9.1191 Digikam::SetupLightTable Class Reference

Inheritance diagram for Digikam::SetupLightTable:

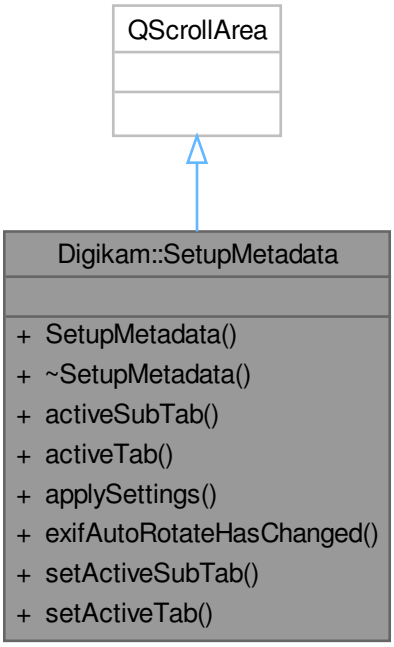


Public Member Functions

- **SetupLightTable** (QWidget ∗const parent=nullptr)
- void **applySettings** ()

9.1192 Digikam::SetupMetadata Class Reference

Inheritance diagram for Digikam::SetupMetadata:



Public Types

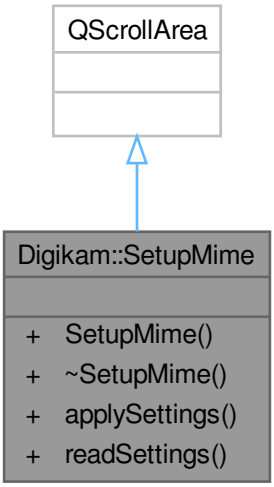
- enum **MetadataSubTab** {
  ExifViewer = 0 , **MakernotesViewer** , **IPTCViewer** , **XMPViewer** ,
  ExifToolViewer }
- enum **MetadataTab** {
  Behavior = 0 , **Sidecars** , **Rotation** , **Display** ,
  ExifTool , **Baloo** , **AdvancedConfig** }

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:

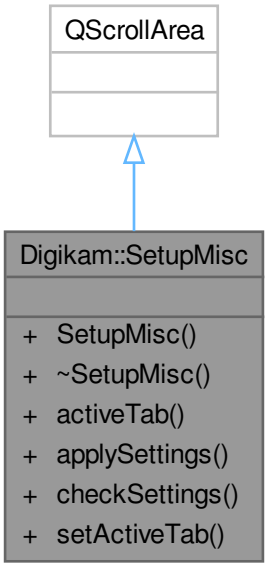


Public Member Functions

- **SetupMime** (QWidget ∗const parent=nullptr)
- void **applySettings** ()
- void **readSettings** ()

9.1194 Digikam::SetupMisc Class Reference

Inheritance diagram for Digikam::SetupMisc:



Public Types

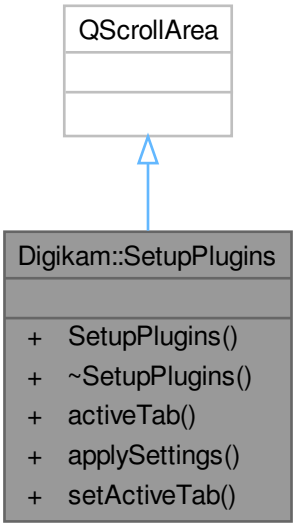
- enum **MiscTab** {
  Behaviour = 0 , **Appearance** , **Grouping** , **SpellCheck** ,
  Localize , **System** }

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

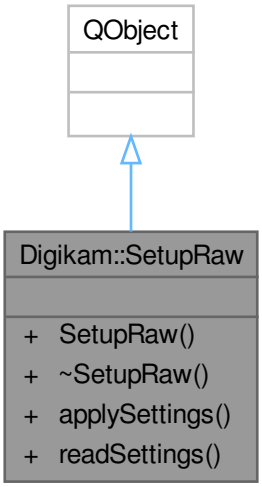
- enum **PluginTab** { **Generic** = 0 , **Editor** , **Bqm** , **Loaders** }

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:

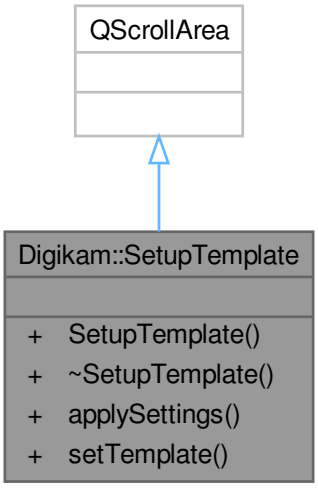


Public Member Functions

- **SetupRaw** (QTabWidget ∗const tab)
- void **applySettings** ()
- void **readSettings** ()

9.1197 Digikam::SetupTemplate Class Reference

Inheritance diagram for Digikam::SetupTemplate:

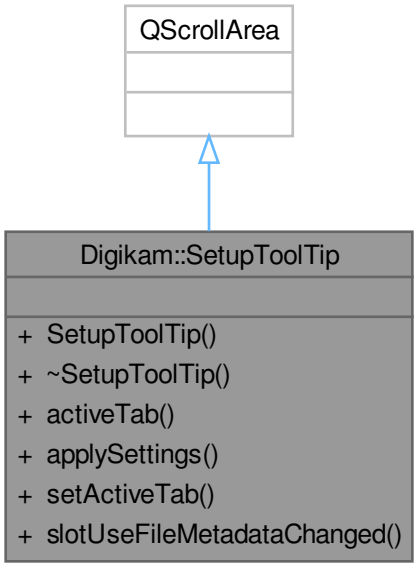


Public Member Functions

- **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

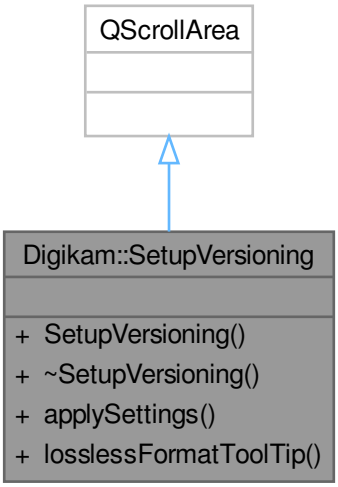
- void **slotUseFileMetadataChanged** (bool)

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

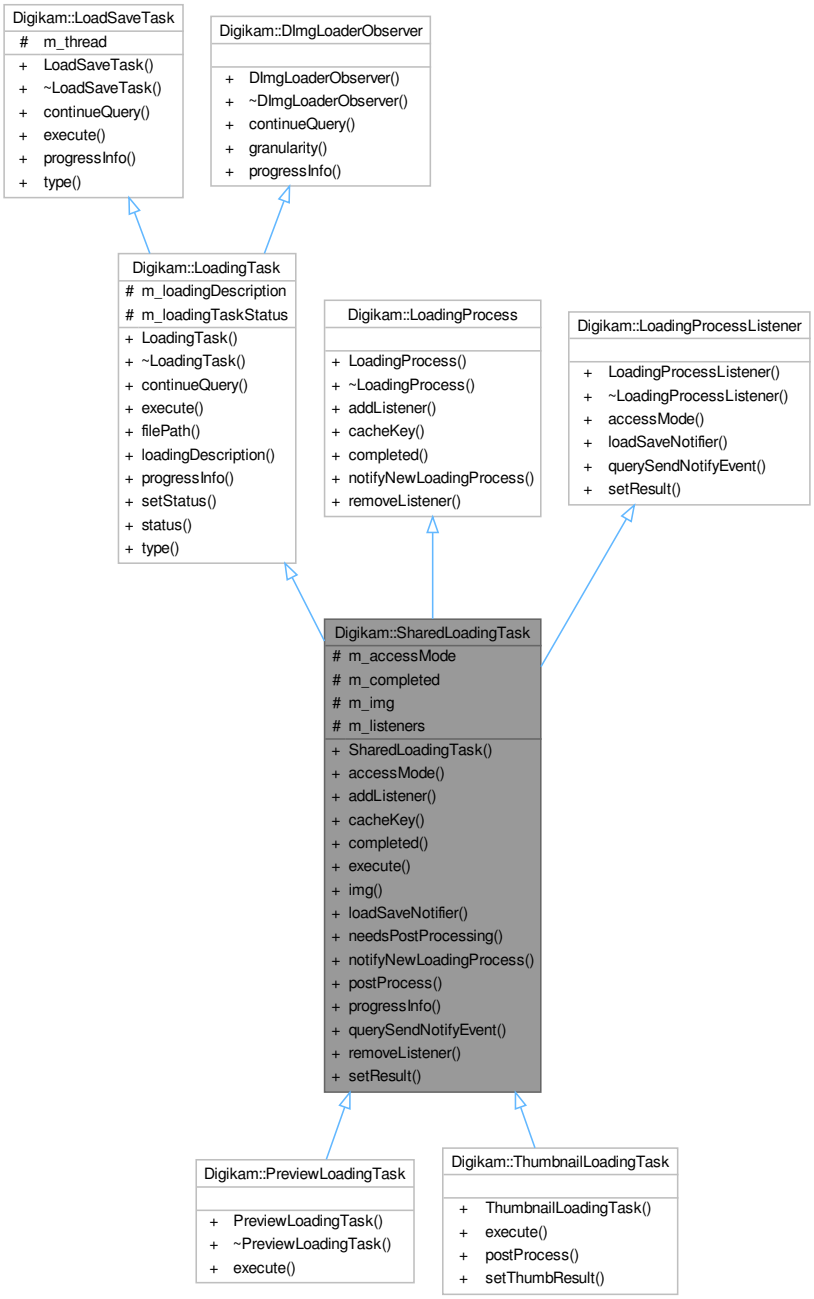
- **SetupVersioning** (QWidget ∗const parent=nullptr)
- void **applySettings** ()

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.

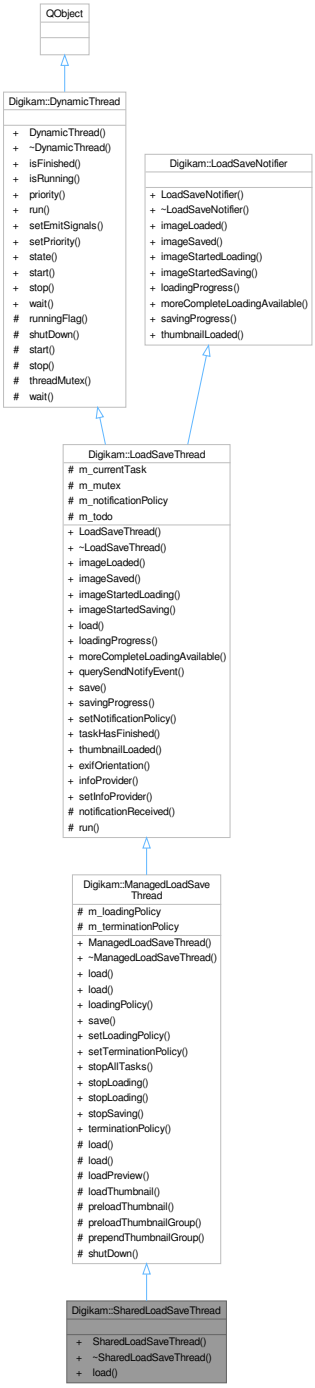
9.1200.1.11 setResult()

```
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 **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

9.1204.1 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.

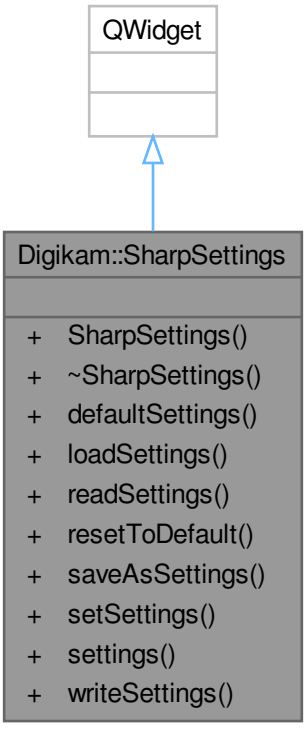
9.1204.1.3 readParameters()

```
void Digikam::SharpenFilter::readParameters (
            const FilterAction & action )  [override], [virtual]
```

Implements Digikam::DImgThreadedFilter.

9.1205 Digikam::SharpSettings Class Reference

Inheritance diagram for Digikam::SharpSettings:



Signals

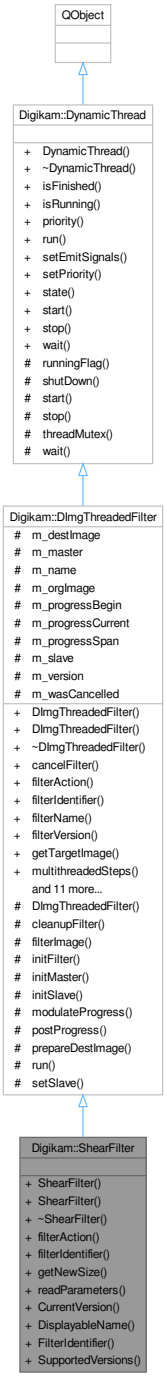
- void **signalSettingsChanged** ()

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** (DImg *const orgImage, QObject *const parent=nullptr, float hAngle=0.0, float vAngle=0.0, bool antialiasing=true, const QColor &backgroundColor=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 **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

9.1206.1 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 overlayed 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]
```

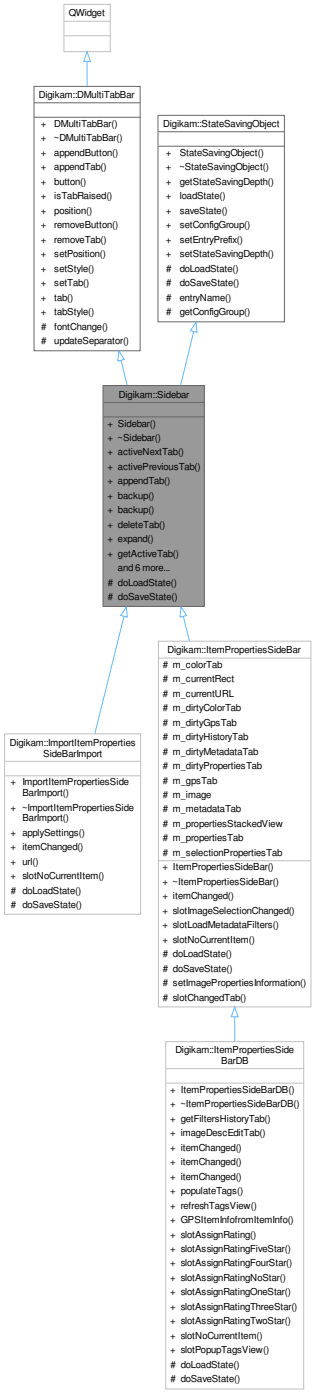
Reposition your button here, adjust and store state.

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

parent	sidebar's parent
sp	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.
side	where the sidebar should be displayed. At the left or right border. Use Qt::LeftEdge or Qt::RightEdge.
minimizedDefault	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

w	widget which is activated by this tab
pic	icon which is shown in this tab
title	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.

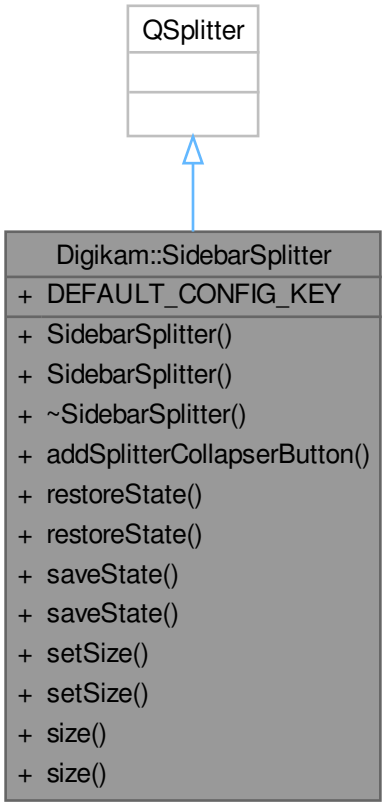
9.1208.3.7 restore()

```
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.

9.1209.1.4 saveState() [2/2]

```
void Digikam::SidebarSplitter::saveState (
            KConfigGroup & group,
            const QString & key )
```

This version uses a specified key in the config group.

9.1209.1.5 setSize()

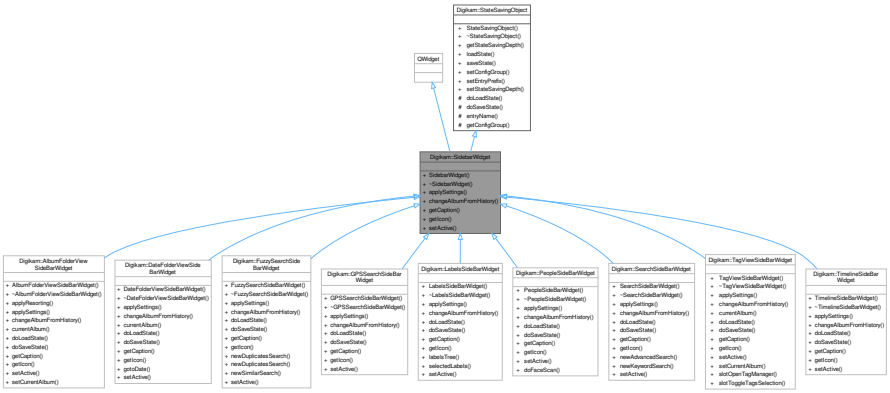
```
void Digikam::SidebarSplitter::setSize (
            Sidebar *const bar,
            int size )
```

Special value -1: Sets the minimum size hint of the widget.

9.1210 Digikam::SidebarWidget Class Reference

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

Inheritance diagram for Digikam::SidebarWidget:



Signals

- 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

- 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

parent	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::FuzzySearchSideBarW
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::FuzzySearchSideBarW
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::FuzzySearchSideBarW
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::FuzzySearchSideBarW
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

active	if true, this widget is the new active widget, if false another widget is active

Implemented in Digikam::AlbumFolderViewSideBarWidget, Digikam::DateFolderViewSideBarWidget, Digikam::FuzzySearchSideBarW
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=Fuzzy↩Algorithm::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 (qlonglong 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< qlonglong > registeredImageIds () const

  This method returns all image ids that are present in the similarity db tables.
- void removeImageFingerprint (qlonglong imageID, FuzzyAlgorithm algorithm=FuzzyAlgorithm::Haar)

  This method removes the fingerprint entry for the given imageId and algorithm.
- void removeImageSimilarity (qlonglong imageID, FuzzyAlgorithm algorithm=FuzzyAlgorithm::Haar)

  This method removes the image similarity entries for the imageID and algorithm.
- void removeImageSimilarity (qlonglong imageID1, qlonglong imageID2, FuzzyAlgorithm algorithm=Fuzzy↩ Algorithm::Haar)

  This method removes the image similarity entry for the imageIDs and algorithm.
- void **setImageSimilarity** (qlonglong imageID1, qlonglong 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

algorithm	The algorithm.

9.1212.1.2 getDirtyOrMissingFingerprints()

```
QList< qlonglong > Digikam::SimilarityDb::getDirtyOrMissingFingerprints (
```

```
            const QList< ItemInfo > & imageInfos,
            FuzzyAlgorithm algorithm = FuzzyAlgorithm::Haar )
```

Parameters

imageInfos	The image info objects representing the items.
algorithm	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

imageInfos	The image info objects representing the items.
algorithm	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

imageID1	The first image id.
imageID2	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

keyword	The setting entry name.

Returns

The setting value.

9.1212.1.7 getSetting()

```
QString Digikam::SimilarityDb::getSetting (
            const QString & keyword )
```

Parameters

keyword	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

imageInfo	The image info object representing the item.
algorithm	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

imageId	The Id of the image to check.
algorithm	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

algorithm	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< qlonglong > Digikam::SimilarityDb::registeredImageIds ( ) const
```

Returns

a set of all present image ids.

9.1212.1.14 removeImageFingerprint()

```
void Digikam::SimilarityDb::removeImageFingerprint (
            qlonglong imageID,
            FuzzyAlgorithm algorithm = FuzzyAlgorithm::Haar )
```

Also, this automatically removes the entries in the ImageSimilarities table for the given algorithm and image id.

Parameters

imageID	The image id.
algorithm	The algorithm.

9.1212.1.15 removeImageSimilarity() [1/2]

```
void Digikam::SimilarityDb::removeImageSimilarity (
            qlonglong imageID,
            FuzzyAlgorithm algorithm = FuzzyAlgorithm::Haar )
```

Parameters

imageID	The image id.
algorithm	The algorithm.

9.1212.1.16 removeImageSimilarity() [2/2]

```
void Digikam::SimilarityDb::removeImageSimilarity (
            qlonglong imageID1,
            qlonglong imageID2,
            FuzzyAlgorithm algorithm = FuzzyAlgorithm::Haar )
```

Parameters

imageID1	The first image id.
imageID2	The second image id.
algorithm	The algorithm.

9.1212.1.17 setSetting()

```
bool Digikam::SimilarityDb::setSetting (
            const QString & keyword,
            const QString & value )
```

Parameters

keyword	The keyword, i.e. setting name.
value	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

observer	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

errorhandler	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()

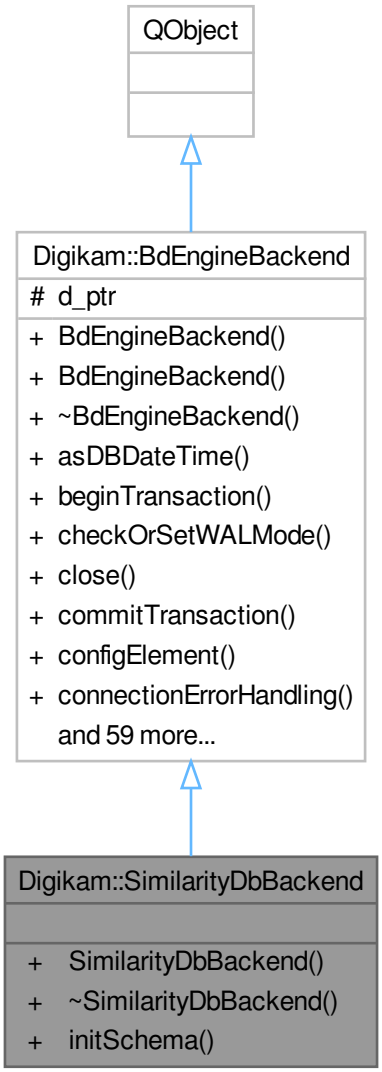
```
void Digikam::SimilarityDbAccess::setParameters (
            const DbEngineParameters & parameters )  [static]
```

Parameters

parameters	The db parameters.

9.1214 Digikam::SimilarityDbBackend Class Reference

Inheritance diagram for Digikam::SimilarityDbBackend:



Public Member Functions

- **SimilarityDbBackend** (DbEngineLocking *const locking, const QString &backendName=QLatin1↩ String("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 &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 lastInsert↩Id=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 last↩InsertId=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 &field↩Names, 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

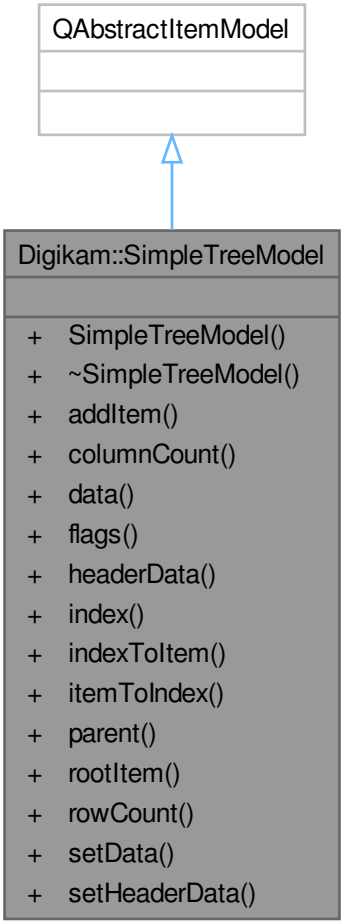
- **SimilarityDbSchemaUpdater** (SimilarityDbAccess ∗const dbAccess)
- void **setObserver** (InitializationObserver ∗const observer)
- bool **update** ()

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

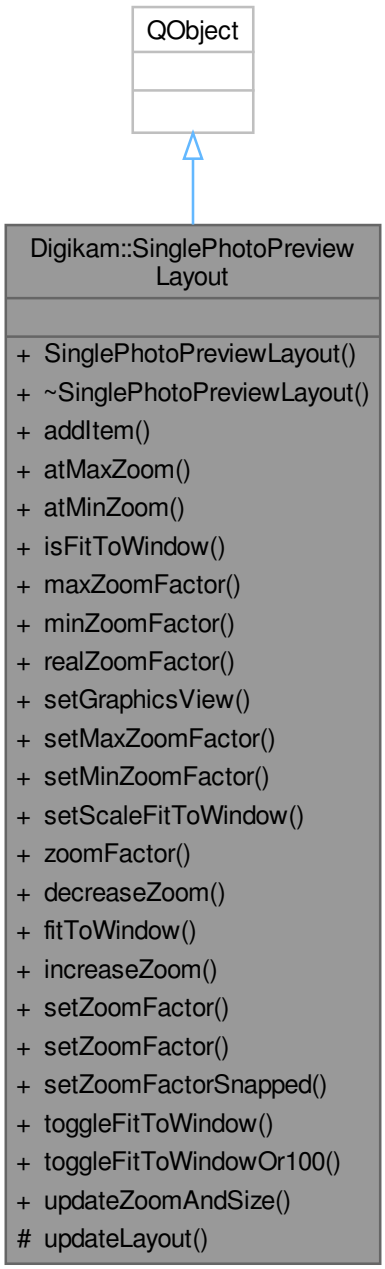
- QString **data**

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=JustSetFactor)
- 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 SAlbum 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

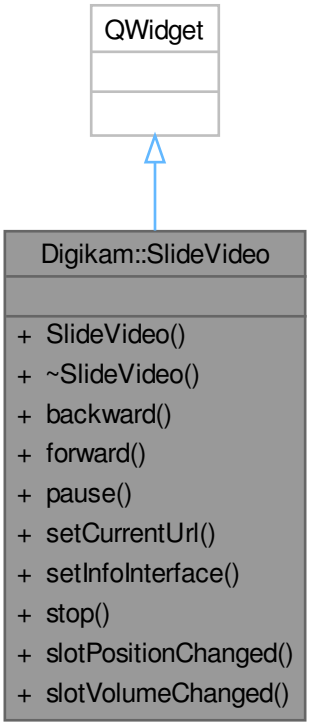
9.1219.1.1 setSketchImageFromXML()

```
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

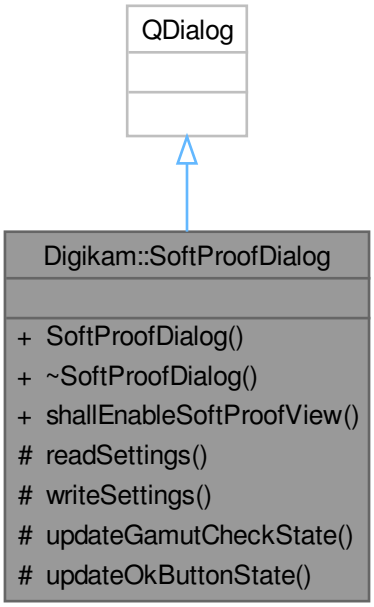
- void **signalVideoDuration** (qint64)
- void **signalVideoFinished** ()
- void **signalVideoLoaded** (bool)
- void **signalVideoPosition** (qint64)
- void **signalVideoVolume** (int)

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

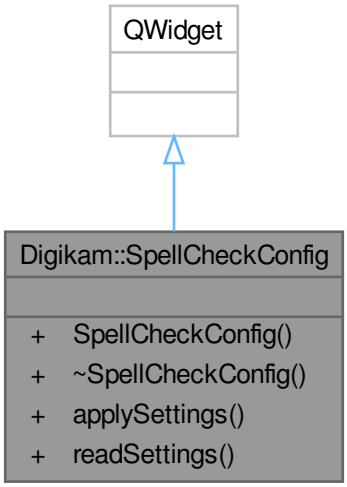
- **SolidHardwareDlg** (QWidget ∗const parent)

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:



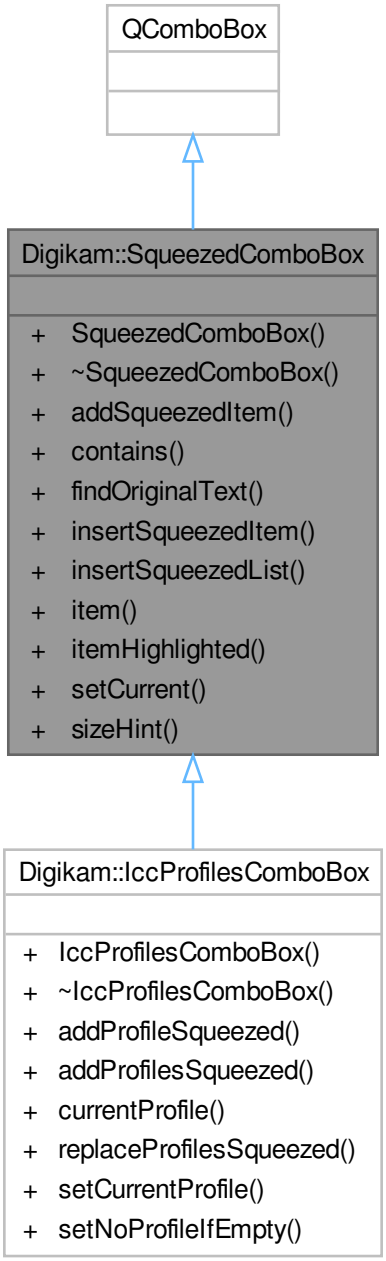
Public Member Functions

- **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

parent	the parent widget
name	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

newItem	the original (long version) of the item which needs to be added to the combobox
userData	custom meta-data assigned to new item.

9.1224.3.2 contains()

```
bool Digikam::SqueezedComboBox::contains (
            const QString & text ) const
```

Parameters

text	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

text	the original (not-squeezed) text to find for
cs	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

newItem	the original (long version) of the item which needs to be added to the combobox
index	the position in the widget.
userData	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

newItems	the originals (long version) of the items which needs to be added to the combobox
index	the position in the widget.

9.1224.3.6 item()

```
QString Digikam::SqueezedComboBox::item (
            int index ) const
```

Parameters

index	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()

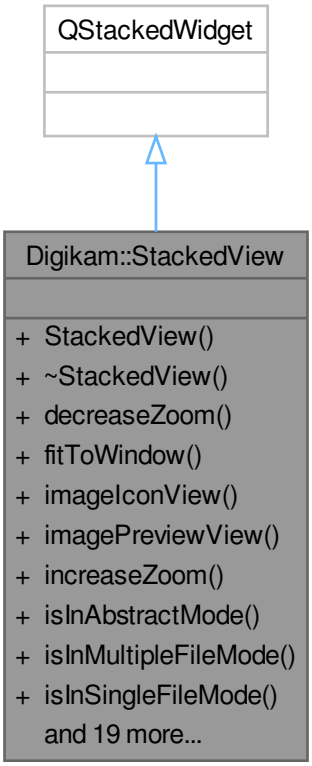
```
void Digikam::SqueezedComboBox::setCurrent (
            const QString & itemText )
```

Parameters

itemText	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)

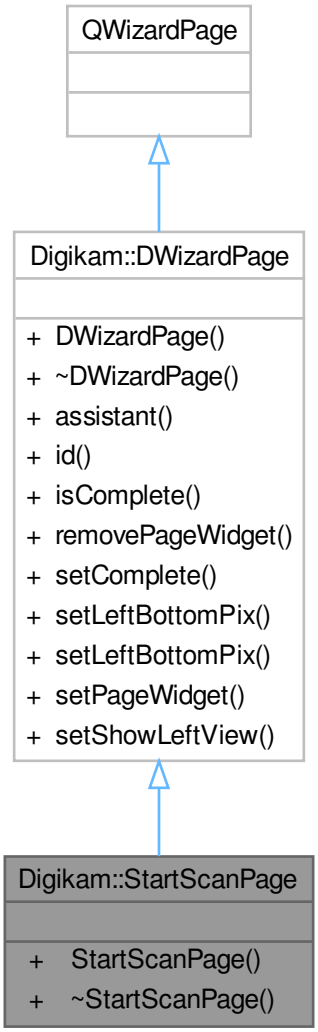
Public Member Functions

- **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

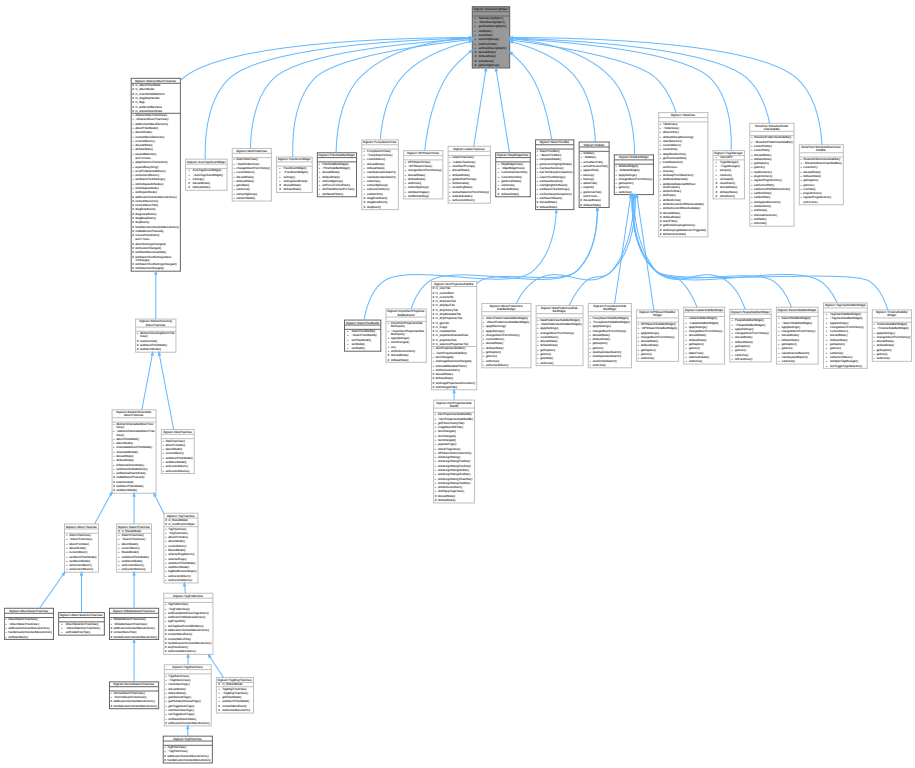
- **DWizardPage** (QWizard *const dlg, const QString &title)
- QWizard * **assistant** () const
- int **id** () const
- bool **isComplete** () const override

- 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 StateSavingObject 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

host	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

base	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

group	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

prefix	the prefix to use for the config entries

9.1227.4.8 setStateSavingDepth()

```
void Digikam::StateSavingObject::setStateSavingDepth (
            const StateSavingDepth depth )
```

Parameters

depth	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

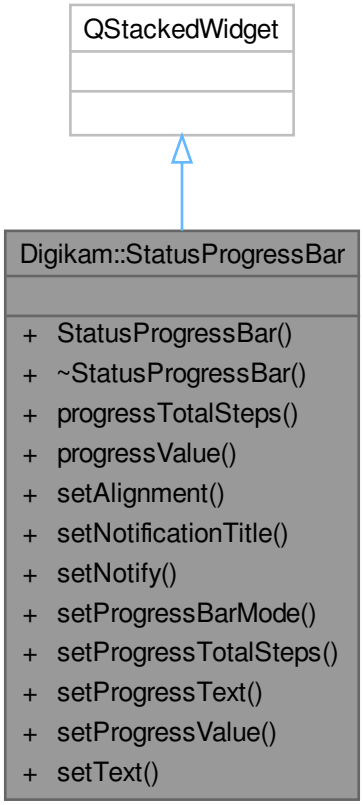
- void **slotBusyIndicator** ()
- void **slotProgressViewVisible** (bool)
- void **slotShowItemDelayed** ()
- void **updateBusyMode** ()

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

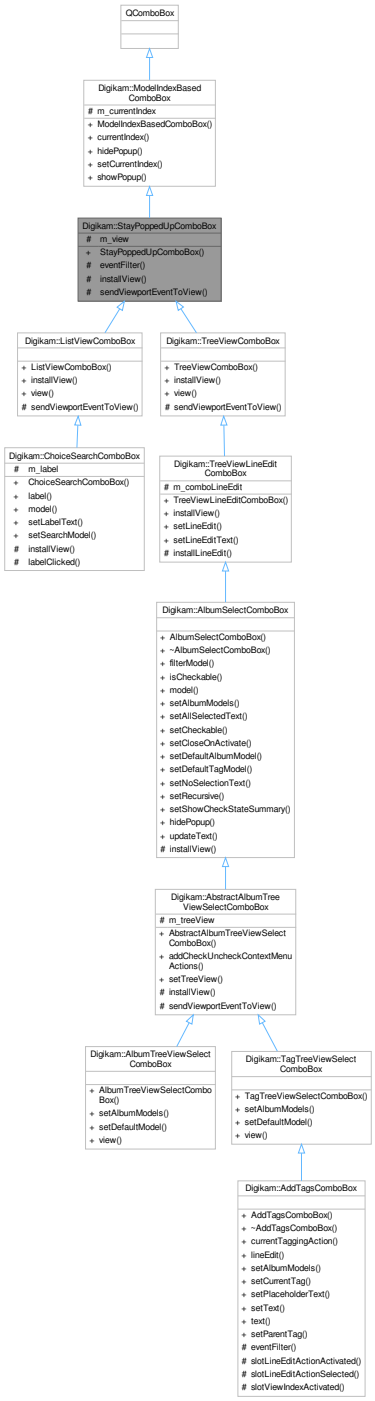
- void **signalCancelButtonPressed** ()

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** (DImg ∗const orgImage, const DImg ∗const refImage, 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 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** ()

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.

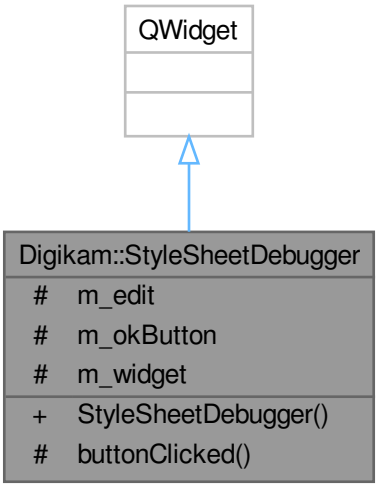
9.1231.1.3 readParameters()

```
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(my↩Widget); 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**

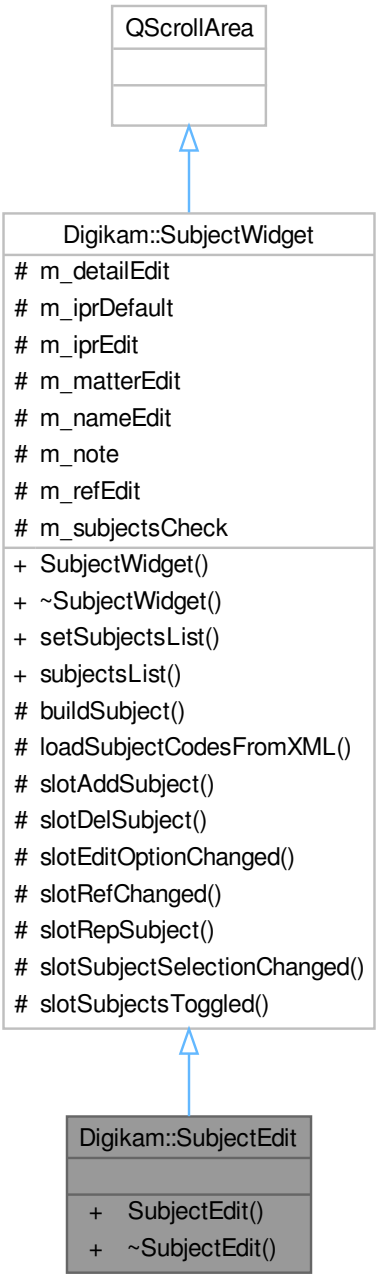
  English and Detail Name of subject.
- QString **matter**

  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

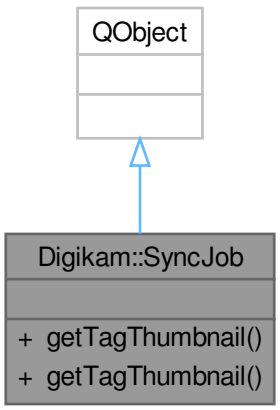
- virtual QString **buildSubject** () const
- virtual bool **loadSubjectCodesFromXML** (const QUrl &url)

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 **enableHWTConv** = 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

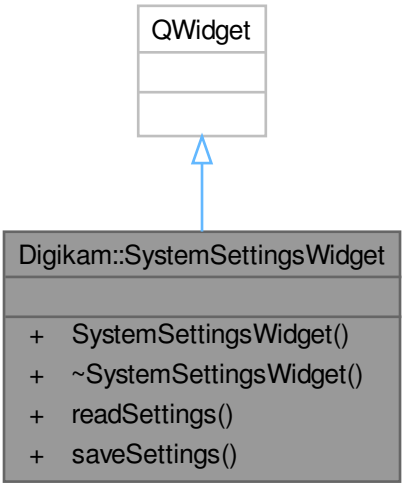
`enum Digikam::SystemSettings::ProxyType`

Enumerator

HttpProxy	Uses an Http proxy.
Socks5Proxy	Uses a Socks5 proxy.

9.1238 Digikam::SystemSettingsWidget Class Reference

Inheritance diagram for Digikam::SystemSettingsWidget:

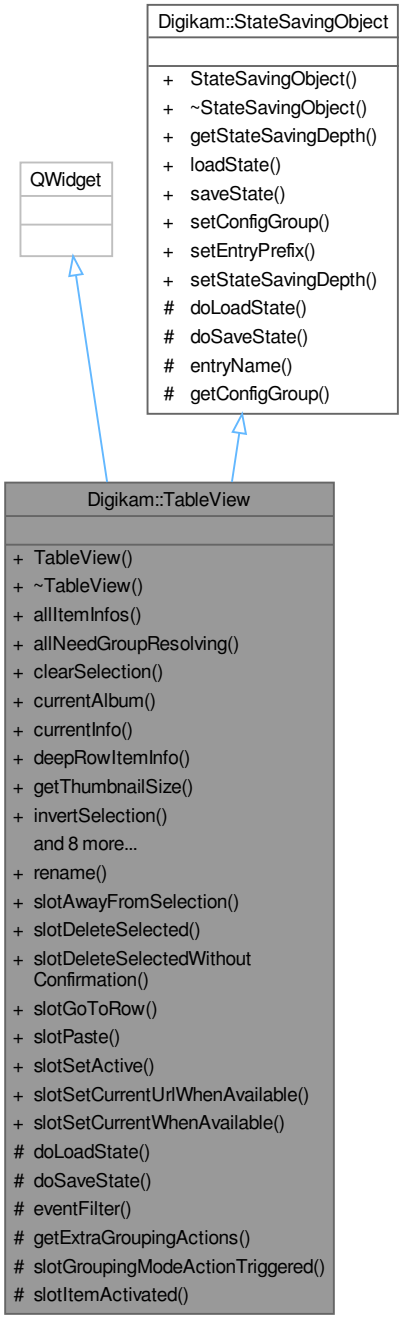


Public Member Functions

- **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::DeleteUse↩ Trash)
- void **slotDeleteSelectedWithoutConfirmation** (const ItemViewUtilities::DeleteMode deleteMode=Item↩ ViewUtilities::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

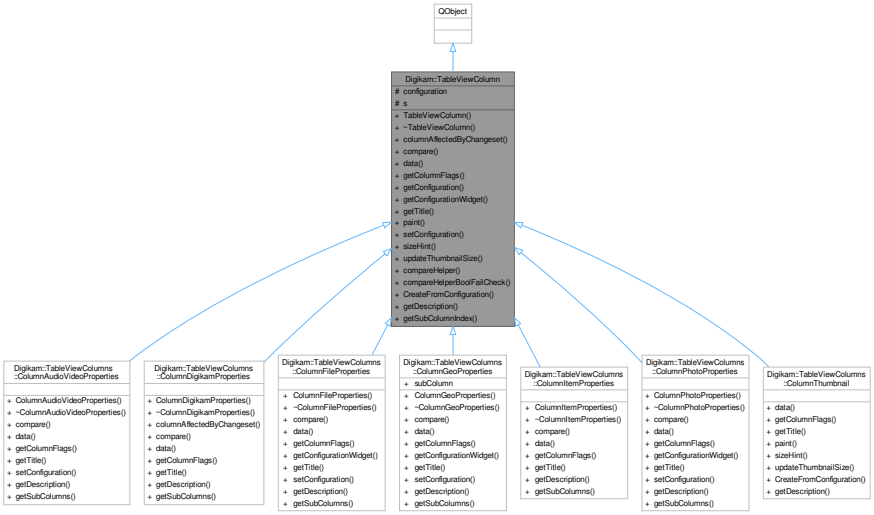
```
void Digikam::TableView::slotDeleteSelected (
            const ItemViewUtilities::DeleteMode deleteMode = ItemViewUtilities::DeleteUseTrash
)  [slot]
```

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::ColumnDigikamPropert Digikam::TableViewColumns::ColumnFileProperties, Digikam::TableViewColumns::ColumnGeoProperties, Digikam::TableViewColum 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::ColumnItemProperties, 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

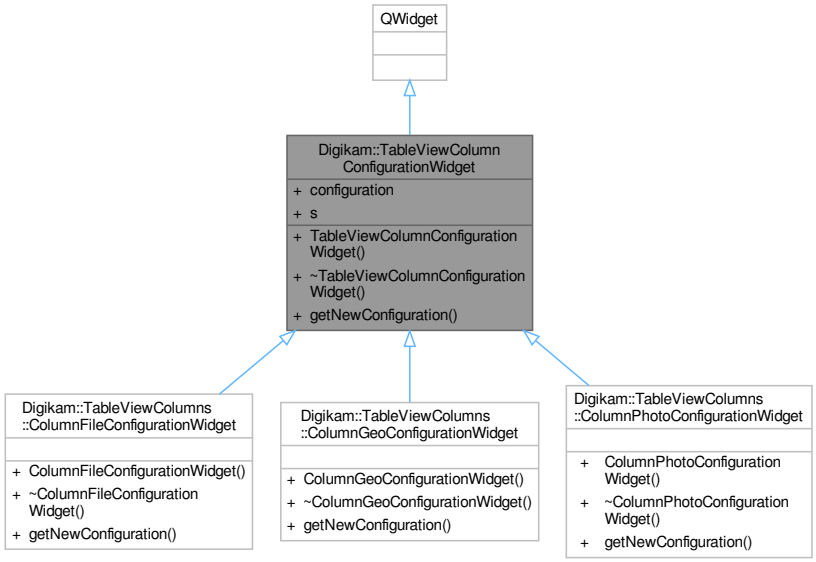
- **TableViewColumnConfiguration** (const QString &id=QString())
- QString **getSetting** (const QString &key, const QString &defaultValue=QString()) const
- void **loadSettings** (const KConfigGroup &configGroup)
- void **saveSettings** (KConfigGroup &configGroup) const

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 &settingKey=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

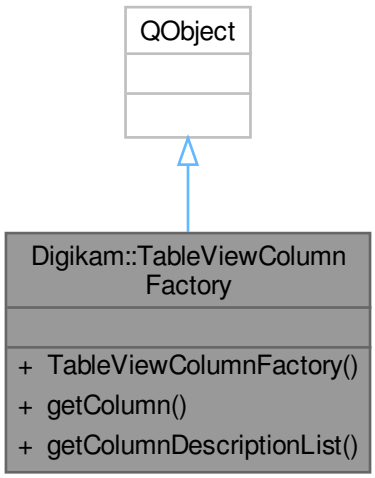
- static bool **FindInListById** (const TableViewColumnDescription::List &listToSearch, const QString &targetId, TableViewColumnDescription ∗const resultDescription)

Public Attributes

- QString **columnIcon**
- QString **columnId**
- QHash< QString, QString > **columnSettings**
- QString **columnTitle**
- 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 &p↩ Configuration, 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 &p↩ Configuration, 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]
```

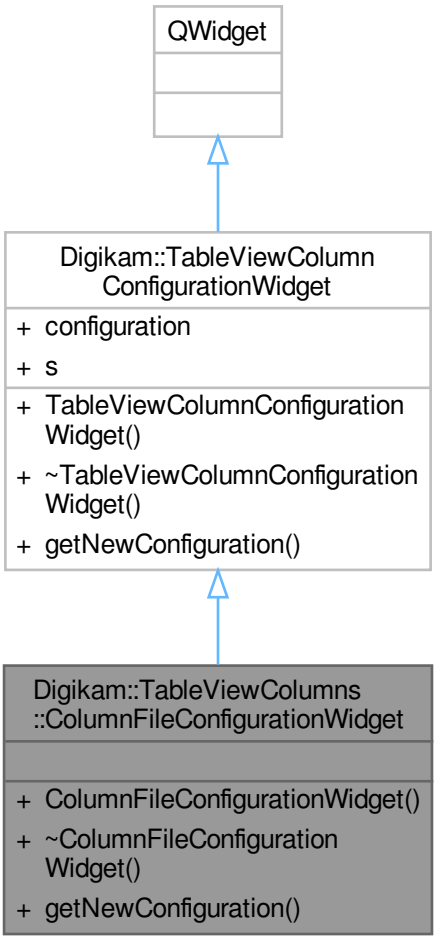
9.1247.1.6 getTitle()

```
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

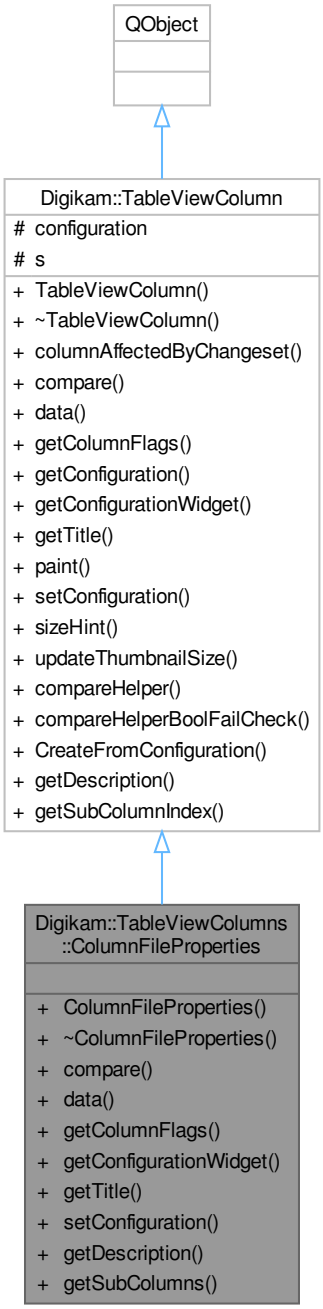
9.1248.1.1 getNewConfiguration()

```
TableViewColumnConfiguration Digikam::TableViewColumns::ColumnFileConfigurationWidget::get↩
NewConfiguration ( )  [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 , **Sub↩ ColumnLastModified** = 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 &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 imageId)

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.

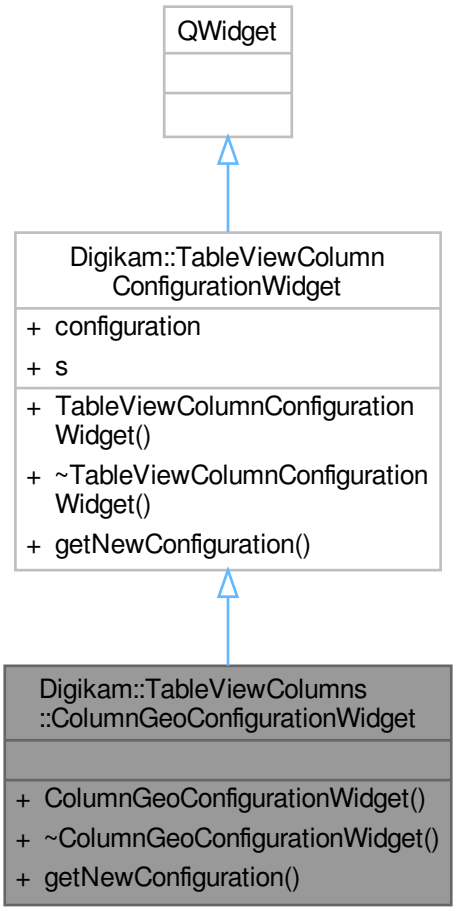
9.1249.1.6 setConfiguration()

```
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

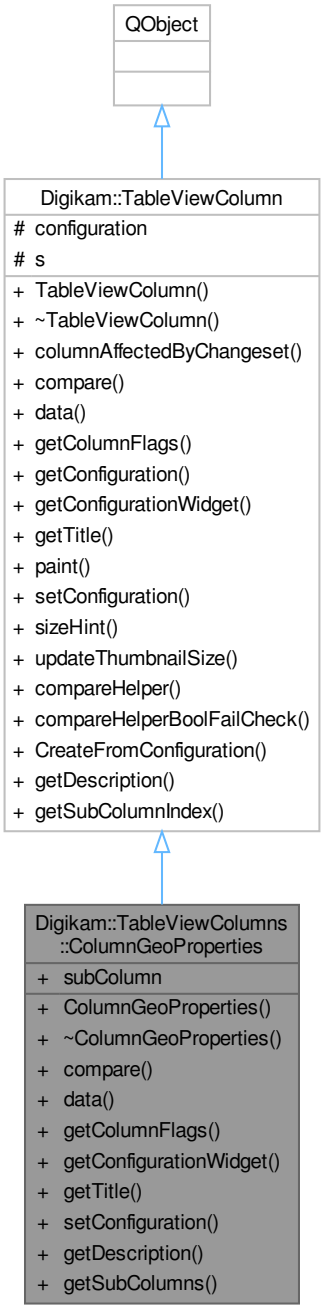
9.1250.1.1 getNewConfiguration()

```
TableViewColumnConfiguration Digikam::TableViewColumns::ColumnGeoConfigurationWidget::getNew↩
Configuration ( )  [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 &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)

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.

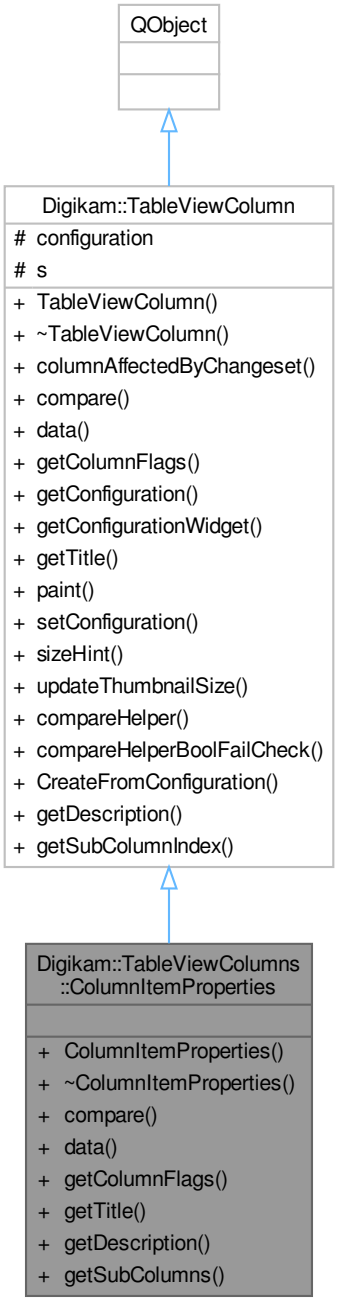
9.1251.1.6 setConfiguration()

```
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 , **SubColumnCreation↩ 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 &p↩ Configuration, 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.

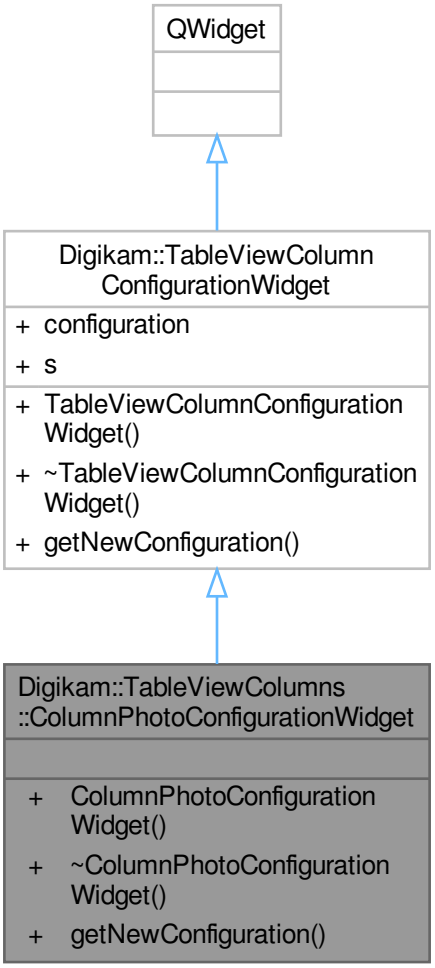
9.1252.1.4 getTitle()

```
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

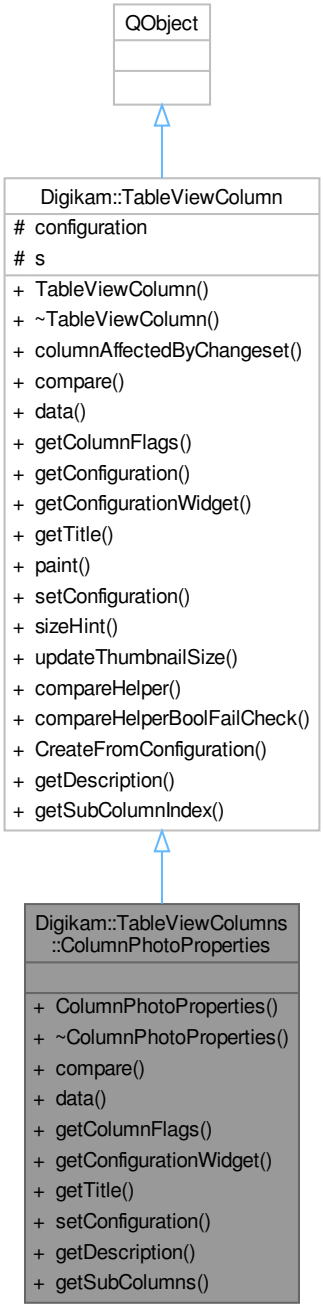
9.1253.1.1 getNewConfiguration()

```
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 , **SubColumn↩ Aperture** = 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 imageId)

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.

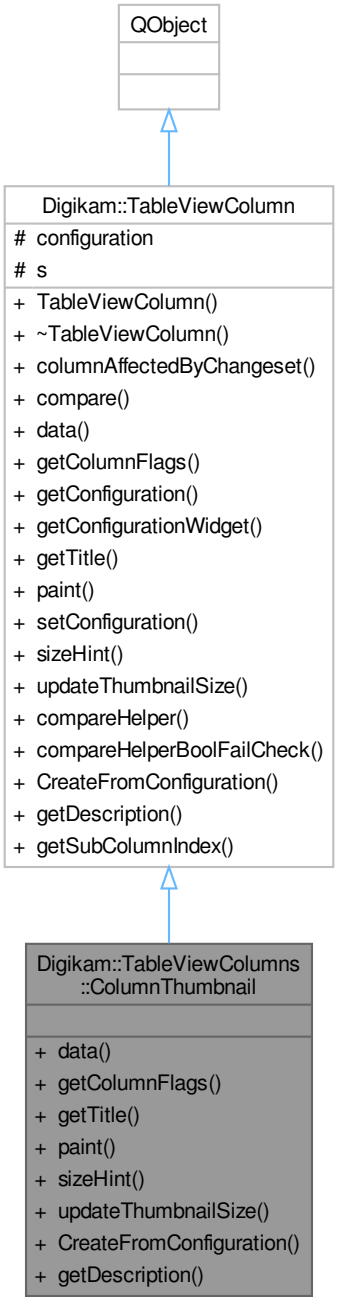
9.1254.1.6 setConfiguration()

```
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 &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 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.

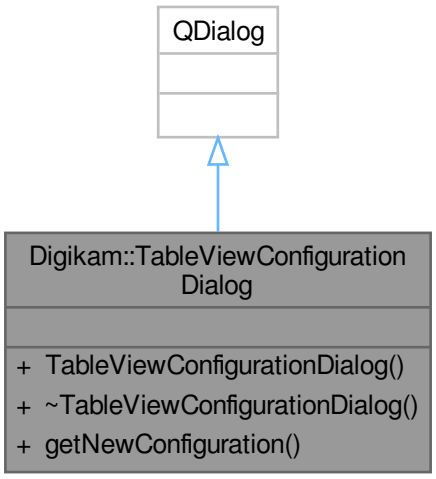
9.1255.1.6 updateThumbnailSize()

```
void Digikam::TableViewColumns::ColumnThumbnail::updateThumbnailSize ( )  [override], [virtual]
```

Reimplemented from Digikam::TableViewColumn.

9.1256 Digikam::TableViewConfigurationDialog Class Reference

Inheritance diagram for Digikam::TableViewConfigurationDialog:

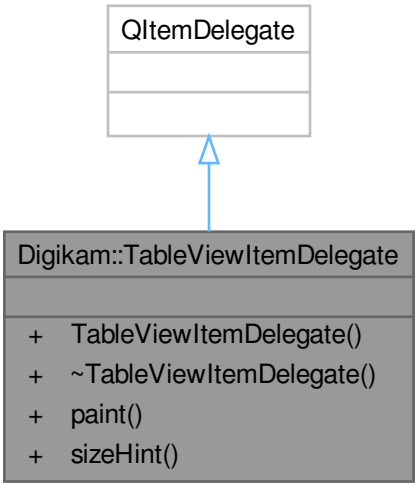


Public Member Functions

- **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 &tableViewModel↩ Index) const override
- QSize sizeHint (const QStyleOptionViewItem &option, const QModelIndex &tableViewModelIndex) const override

9.1257.1 Member Function Documentation

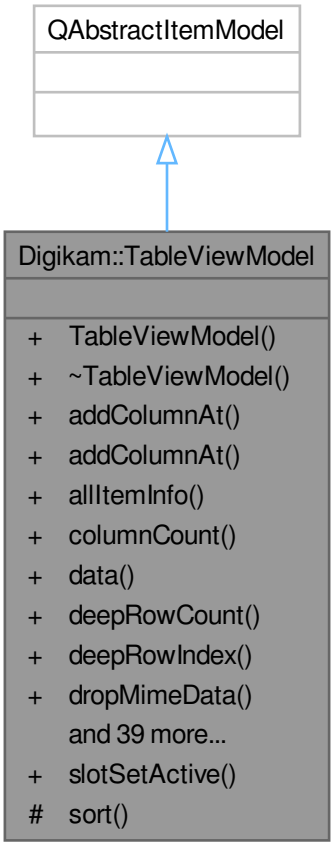
9.1257.1.1 sizeHint()

```
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 **toCol0** (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

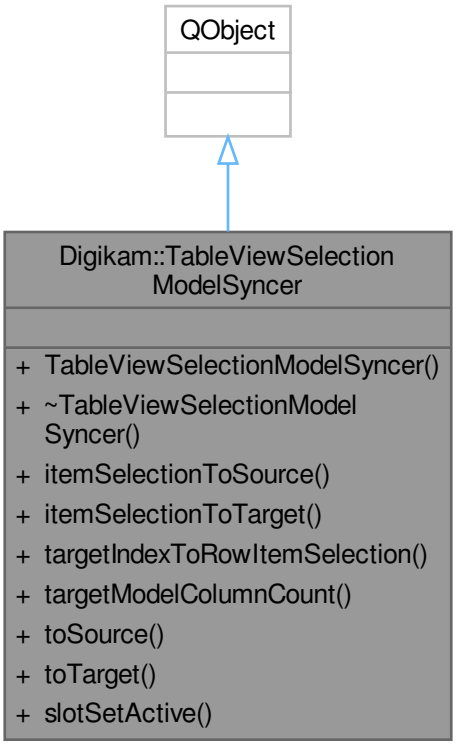
- void **addChild** (Item ∗const newChild)
- Item ∗ **findChildWithImageId** (const qlonglong searchImageId)
- void **insertChild** (const int pos, Item ∗const newChild)
- void **takeChild** (Item ∗const oldChild)

Public Attributes

- QList< Item ∗ > **children**
- qlonglong **imageId** = 0
- Item ∗ **parent** = nullptr

9.1260 Digikam::TableViewSelectionModelSyncer Class Reference

Inheritance diagram for Digikam::TableViewSelectionModelSyncer:



Public Slots

- void **slotSetActive** (const bool isActive)

Public Member Functions

- TableViewSelectionModelSyncer (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 TableViewSelectionModelSyncer()

```
Digikam::TableViewSelectionModelSyncer::TableViewSelectionModelSyncer (
            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
- TableViewSelectionModelSyncer * **tableViewSelectionModelSyncer** = 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 **decodeIsCutSelection** (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 **encodeIsCutSelection** (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

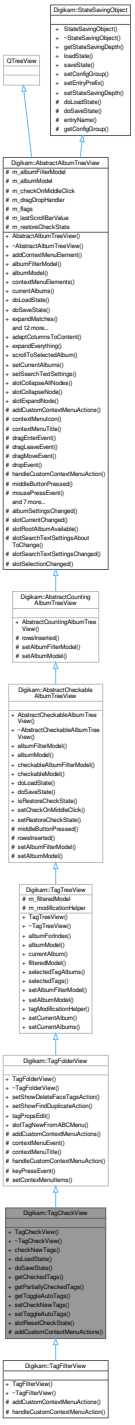
```
enum Digikam::TagChangeset::Operation
```

Enumerator

PropertiesChanged	ImageTagProperties Table.

9.1264 Digikam::TagCheckView Class Reference

Inheritance diagram for Digikam::TagCheckView:



Public Types

- enum **ToggleAutoTags** { **NoToggleAuto** = 0 , **Children** , **Parents** , **ChildrenAndParents** }

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

- 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 * > &excluded↩ Tags)

  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 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 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 setContexMenuItems (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

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

cmh	helper object to create the context menu
album	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

includedTags	a list of selected tag ids processed.
excludedTags	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()

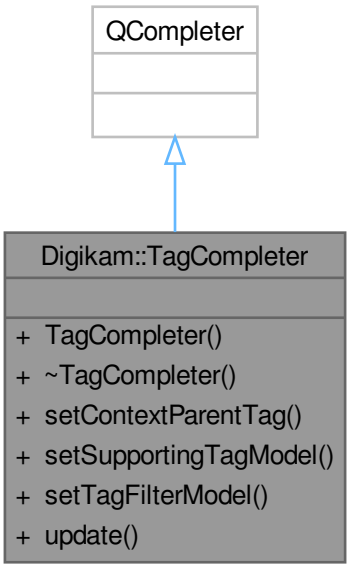
```
void Digikam::TagCheckView::setCheckNewTags (
            bool checkNewTags )
```

Parameters

checkNewTags	is switched on, a tag that is created from *within* 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.

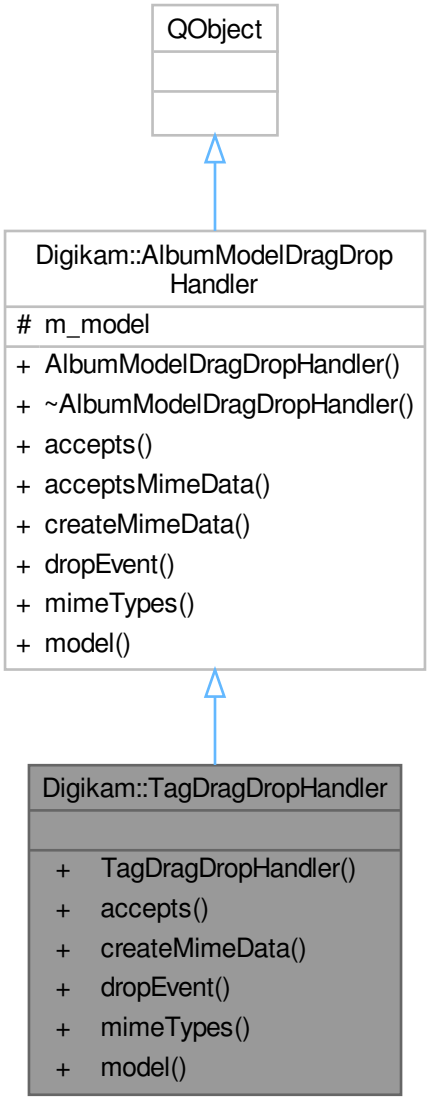
9.1266 Digikam::TagData Struct Reference

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 mimeTypes()

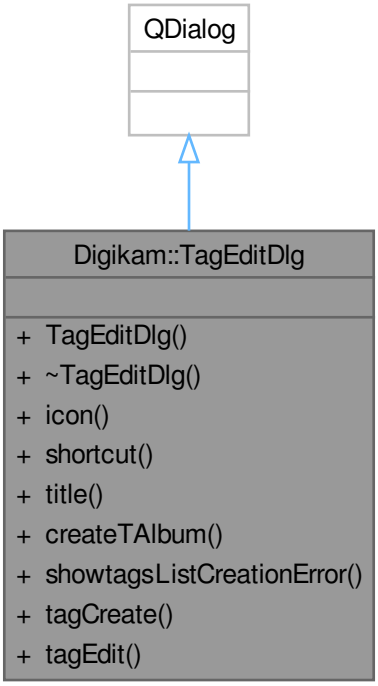
```
QStringList Digikam::TagDragDropHandler::mimeTypes ( ) const  [override], [virtual]
```

Called by the default implementation of model's mimeTypes().

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()

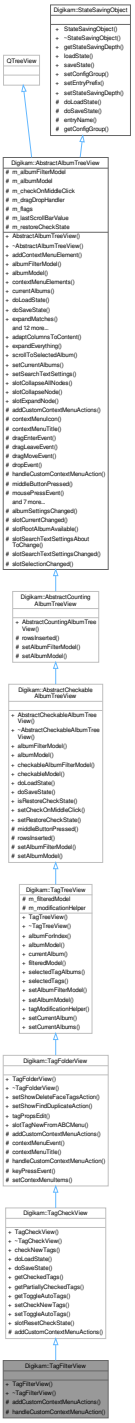
```
AlbumList Digikam::TagEditDlg::createTAlbum (
            TAlbum *const mainRootAlbum,
            const QString & tagStr,
            const QString & icon,
            const QKeySequence & ks,
            QMap< QString, QString > & errMap )  [static]
```

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.

Inheritance diagram for Digikam::TagFilterView:



Public Member Functions

- TagFilterView (QWidget ∗const parent, TagModel ∗const tagFilterModel)

  Constructor.
- ∼**TagFilterView** () override

  Destructor.

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 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 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 setContexMenuItems (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 {
  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::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 * > &excluded↩ Tags)

  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

parent	the parent for qt parent child mechanism
tagFilterModel	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

cmh	helper object to create the context menu
album	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]
```

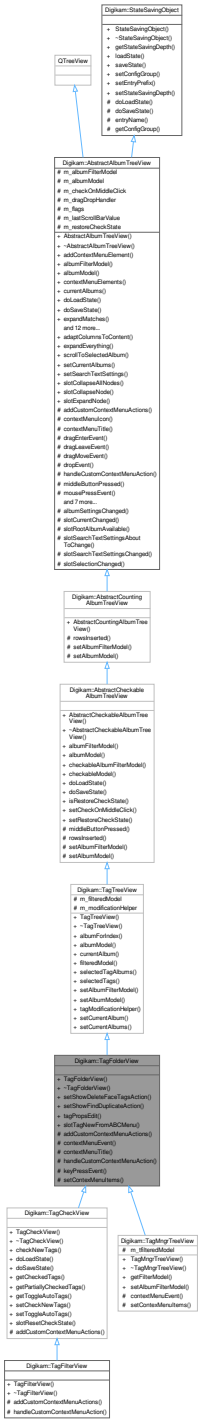
Parameters

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

Reimplemented from Digikam::AbstractAlbumTreeView.

9.1270 Digikam::TagFolderView Class Reference

Inheritance diagram for Digikam::TagFolderView:



Public Slots

- 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 **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 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 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 setContexMenuItems (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 | 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 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

parent	the parent for Qt's parent child mechanism
model	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

cmh	helper object to create the context menu
album	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

event	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

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

Reimplemented from Digikam::AbstractAlbumTreeView.

9.1270.2.5 setContexMenuItems()

```
void Digikam::TagFolderView::setContexMenuItems (
            ContextMenuHelper & cmh,
            const QList< TAlbum * > & albums )  [protected], [virtual]
```

If only one element is selected, only AddCustomContextMenuActions is called

Parameters

cmh	- helper object to create context menu
albums	- 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

show	if `true` the action to delete people tags in the tag album is displayed

9.1270.2.7 setShowFindDuplicateAction()

```
void Digikam::TagFolderView::setShowFindDuplicateAction (
            bool show )
```

Parameters

show	if `true` 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

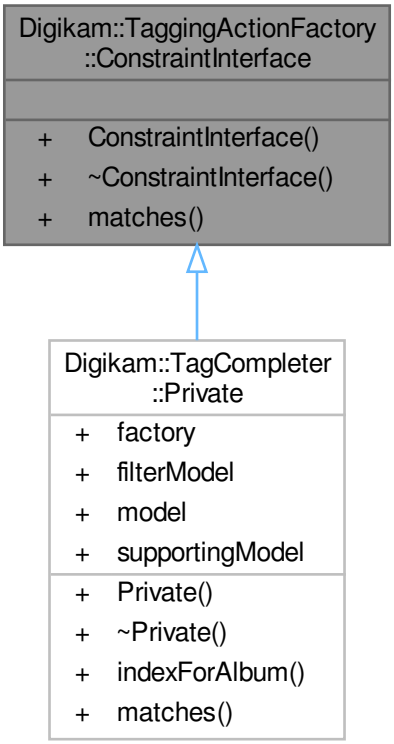
9.1272.2.1 setConstraintInterface()

```
void Digikam::TaggingActionFactory::setConstraintInterface (
            ConstraintInterface *const iface )
```

Pass an implementation of ConstraintInterface (reamins 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

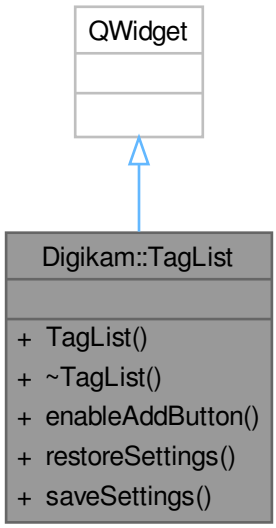
- bool **isNull** () const
- bool **operator**< (const TagInfo &info) const

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 ( )
```

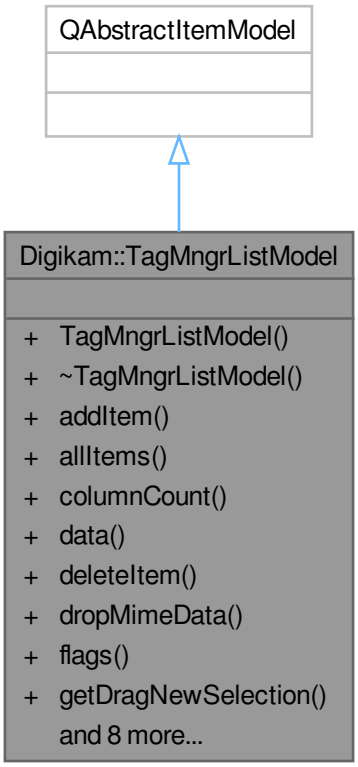
If config is empty add generic All Tags

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

- **TagMngrListModel** (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 subcassing QAbstractItemModel.
- 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::TagMngrListModel::addItem (
            QList< QVariant > values )
```

Parameters

values	- 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.

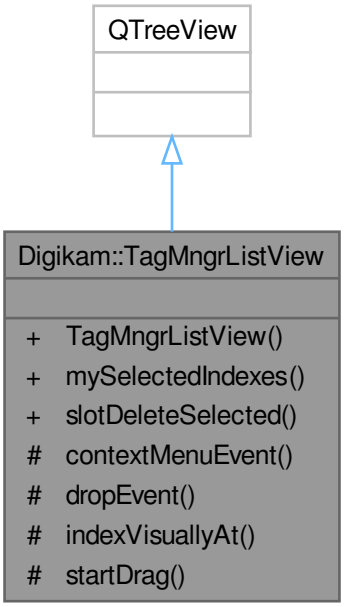
9.1276.1.2 dropMimeData()

```
bool Digikam::TagMngrListModel::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 TagsMngrListView 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

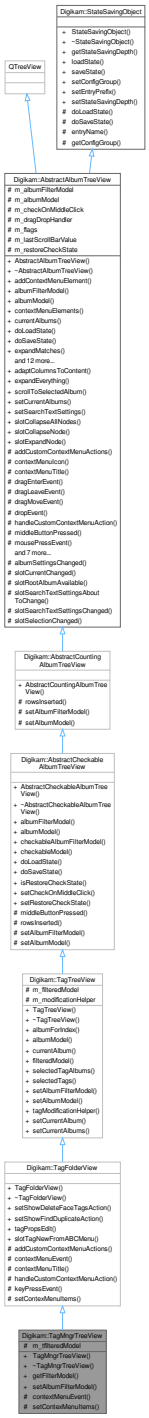
- void **contextMenuEvent** (QContextMenuEvent ∗event) override

  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 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 Member Functions

- void contextMenuEvent (QContextMenuEvent ∗event) override

  contextMenuEvent Reimplement contextMenuEvent from AbstractAlbumTree to support multiple selection
- void setContexMenuItems (ContextMenuHelper &cmh, const QList< TAlbum ∗ > &albums) override

  setContexMenuItems 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 | 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::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

event	context menu event triggered by right click

Append root tag if no nodes are selected

9.1278.1.2 setContexMenuItems()

```
void Digikam::TagMngrTreeView::setContexMenuItems (
            ContextMenuHelper & cmh,
            const QList< TAlbum * > & albums )  [override], [protected], [virtual]
```

Will set custom actions for Tags Manager. Some actions are also available in toolbar

Parameters

cmh	ContextMenuHelper class to help setting some basic actions
albums	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()

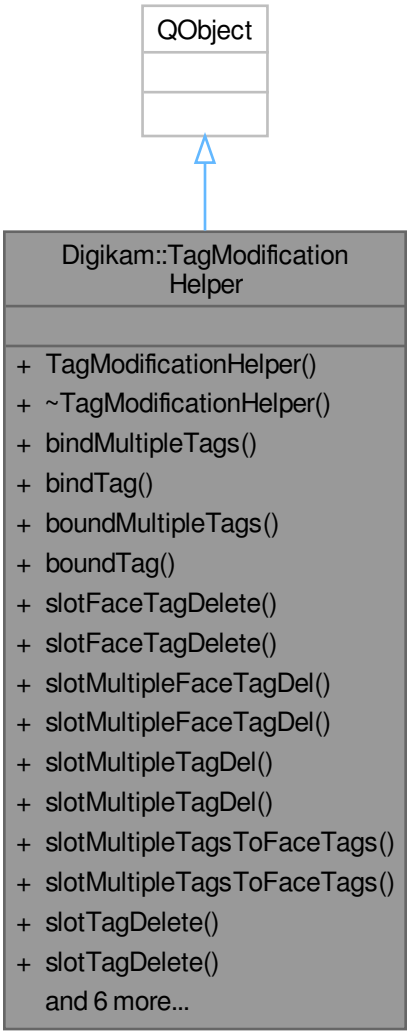
```
QVariant Digikam::TagModel::fontRoleData (
            Album * a ) const  [override], [protected], [virtual]
```

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

parent	the parent for qt parent child mechanism
dialogParent	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

action	- action to store pointer
tags	- 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

tag	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

tags	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

tags	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 QMultiMap 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

tags	the tags to mark.

9.1280.3.9 slotTagDelete

```
void Digikam::TagModificationHelper::slotTagDelete (
            TAlbum * tag )  [slot]
```

Parameters

tag	the tag to delete, must not be the root tag album

9.1280.3.10 slotTagEdit

```
void Digikam::TagModificationHelper::slotTagEdit (
            TAlbum * tag )  [slot]
```

Parameters

tag	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

parent	the parent tag album under which to create the new tags. May be 0 to use the root album
title	if this isn't an empty string, then this tag name is suggested
iconName	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

tag	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)
- QMultiMap< 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()

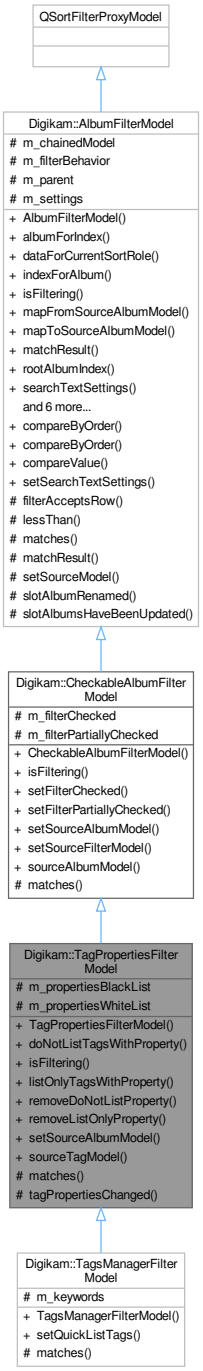
```
QString Digikam::TagProperties::value (
            const QString & key ) const
```

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

album	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**

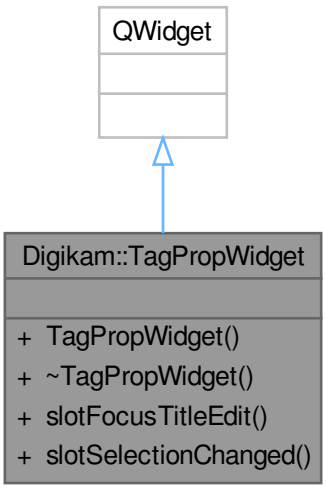
9.1284 Digikam::TagPropertyName Class Reference

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.

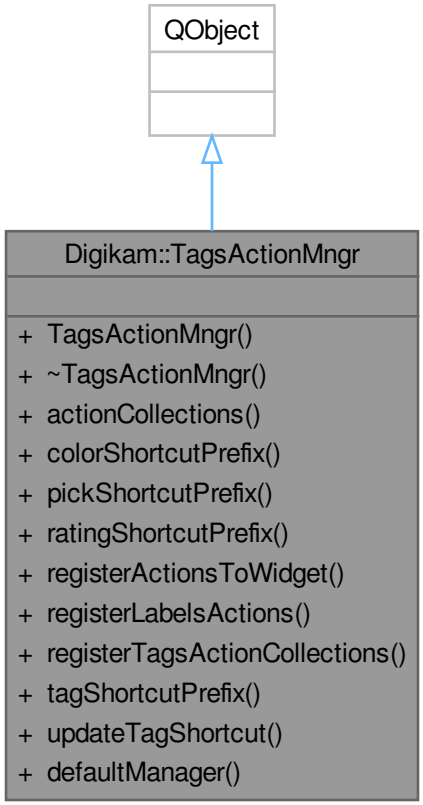
9.1286.2.5 toVariant()

```
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::create↩ GUI(), 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.

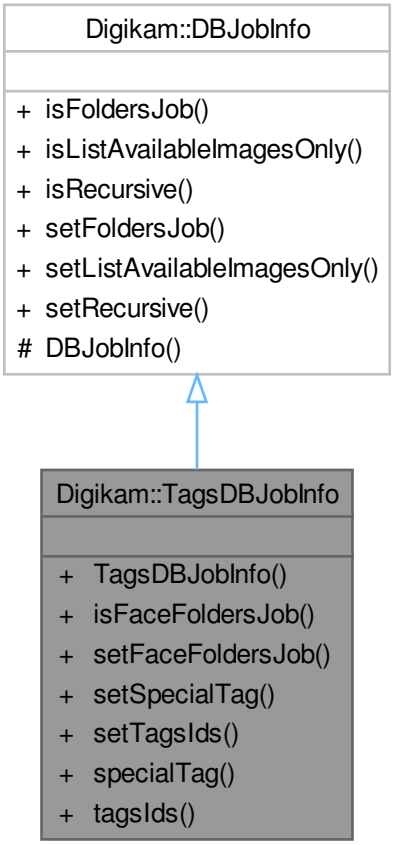
9.1288.2.23 tagsWithPropertyCached()

```
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

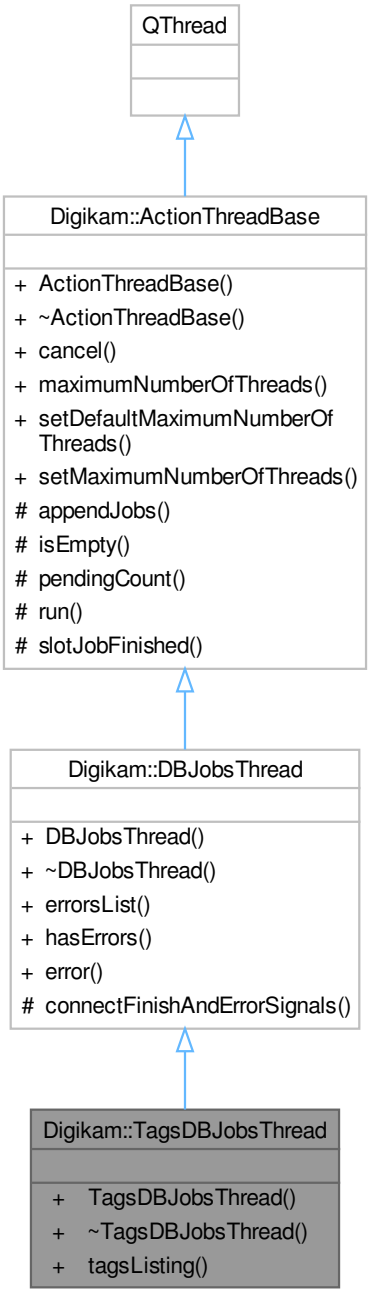
- bool **isFaceFoldersJob** () const
- void **setFaceFoldersJob** ()
- void **setSpecialTag** (const QString &tag)
- void **setTagsIds** (const QList< int > &tagsIds)
- QString **specialTag** () const
- QList< int > **tagsIds** () const

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()

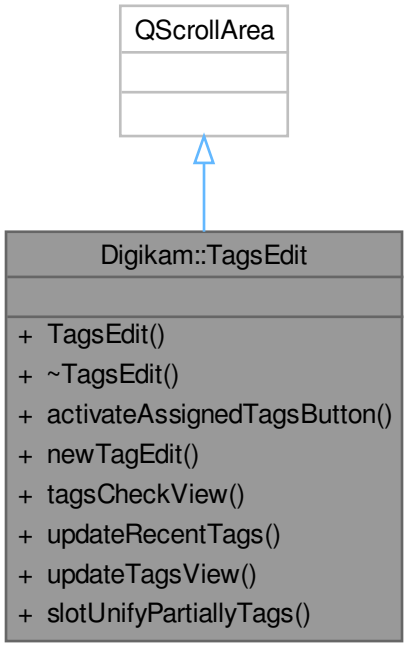
```
void Digikam::TagsDBJobsThread::tagsListing (
            const TagsDBJobInfo & info )
```

Parameters

info	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** (qlonglong 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

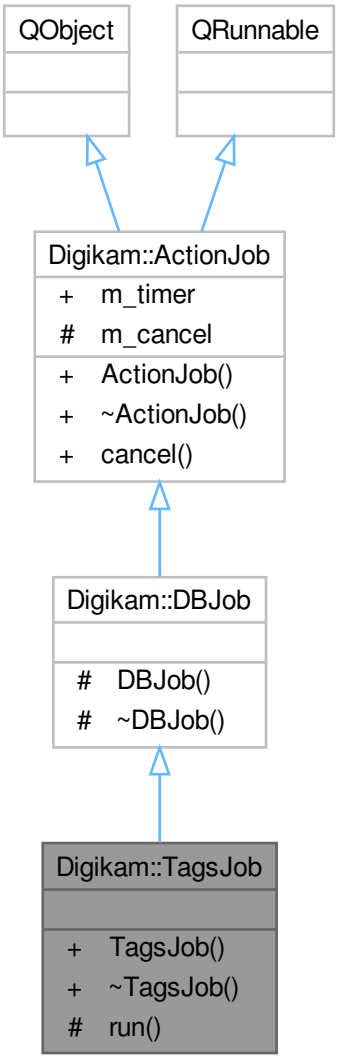
- bool **isNull** () const

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.

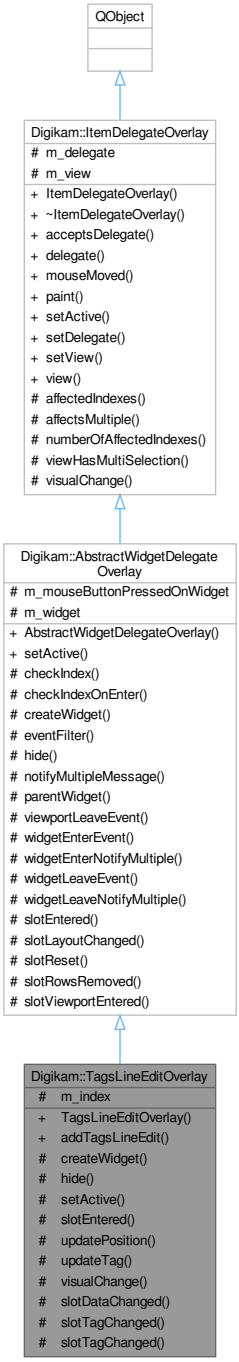
Protected Attributes inherited from Digikam::ActionJob

- 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 overlayed 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.

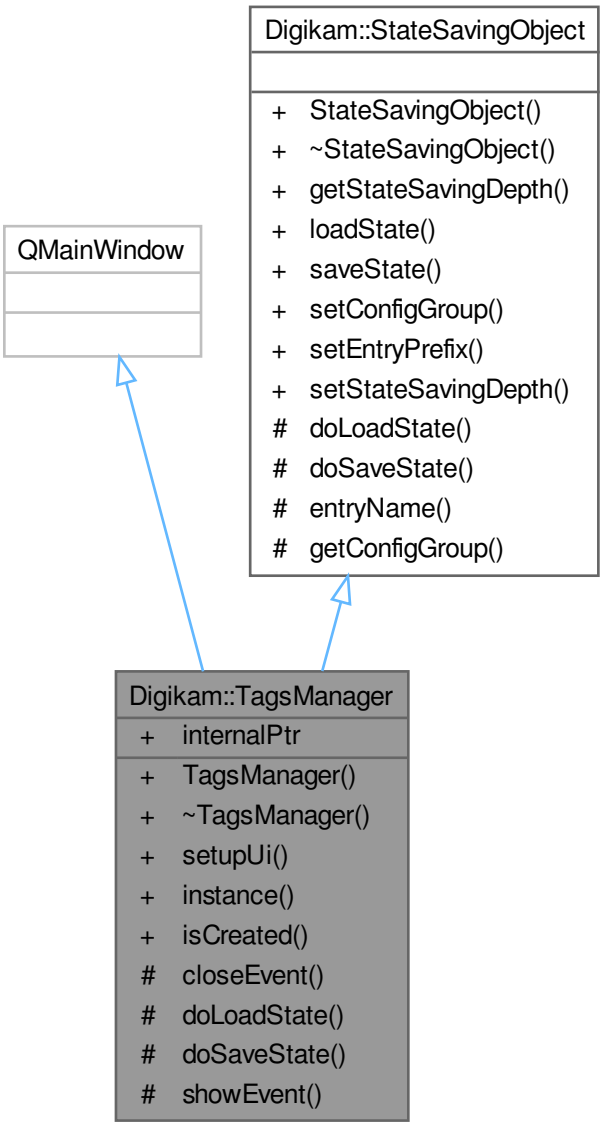
9.1294.1.5 visualChange()

```
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** (TAlbum *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()

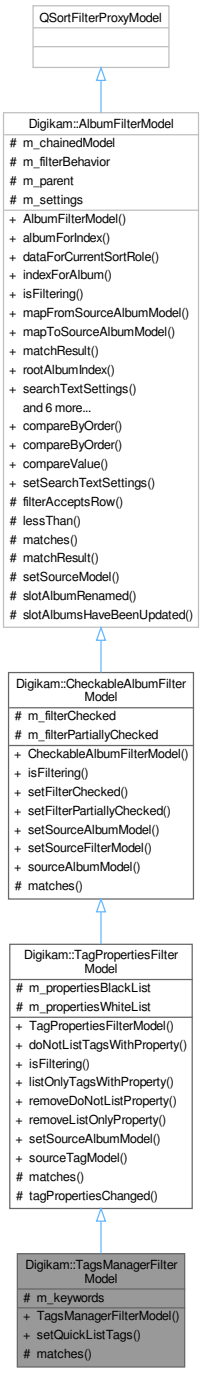
```
void Digikam::TagsManager::doSaveState ( )  [override], [protected], [virtual]
```

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.

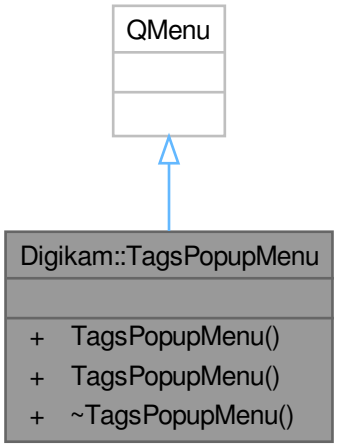
Parameters

album	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 **signalPopupTagsView** ()
- 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

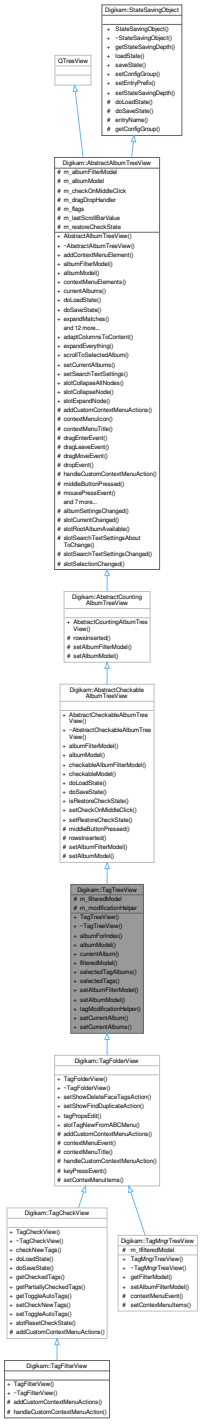
```
enum Digikam::TagsPopupMenu::Mode
```

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

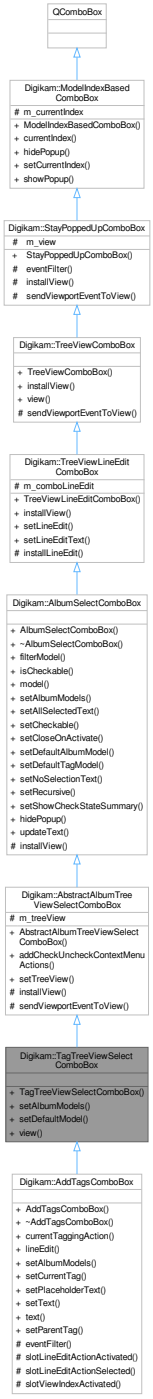
- QString entryName (const QString &base) const

  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

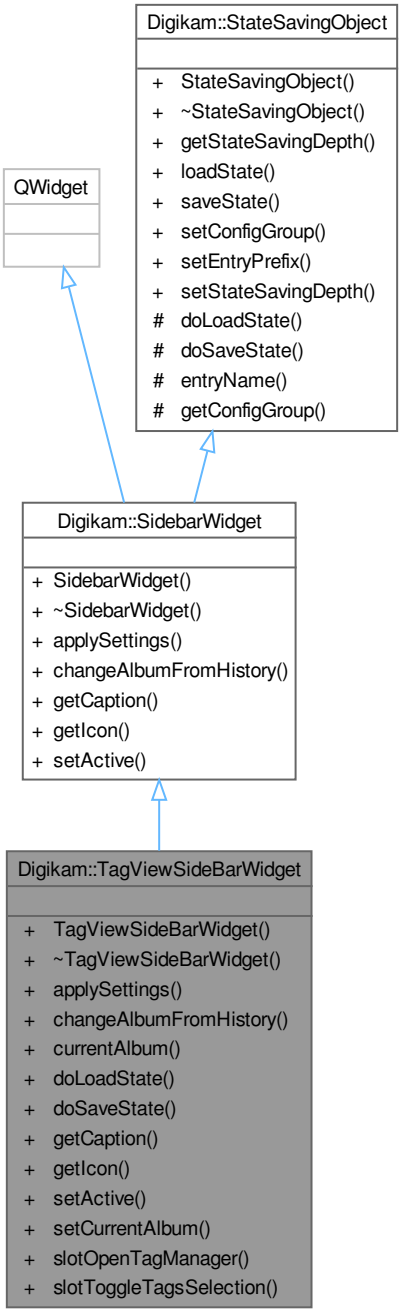
- QAbstractItemView ∗ **m_view** = nullptr

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

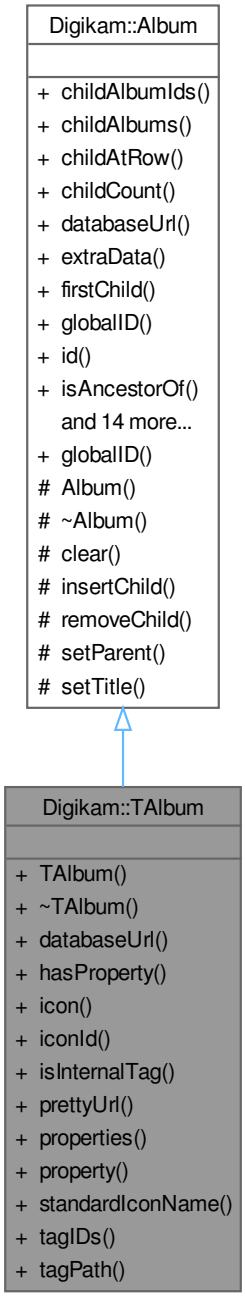
active	if true, this widget is the new active widget, if false another widget is active

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

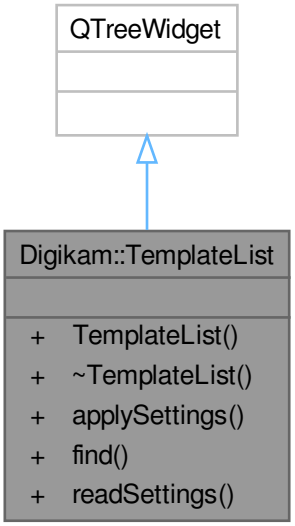
9.1302.1.1 m_templateTitle

```
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:

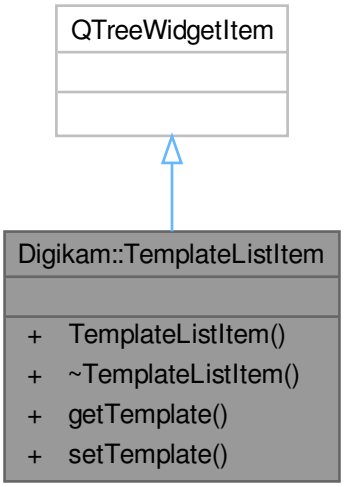


Public Member Functions

- **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:

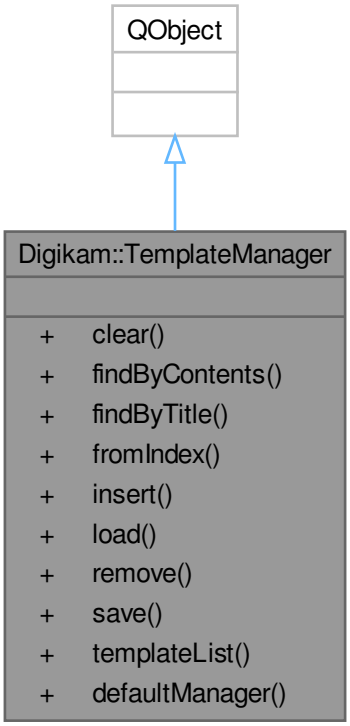


Public Member Functions

- **TemplateListItem** (QTreeWidget ∗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

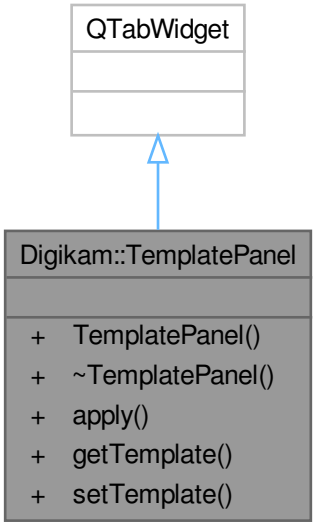
- static TemplateManager * **defaultManager** ()

Friends

- class **TemplateManagerCreator**

9.1306 Digikam::TemplatePanel Class Reference

Inheritance diagram for Digikam::TemplatePanel:



Public Types

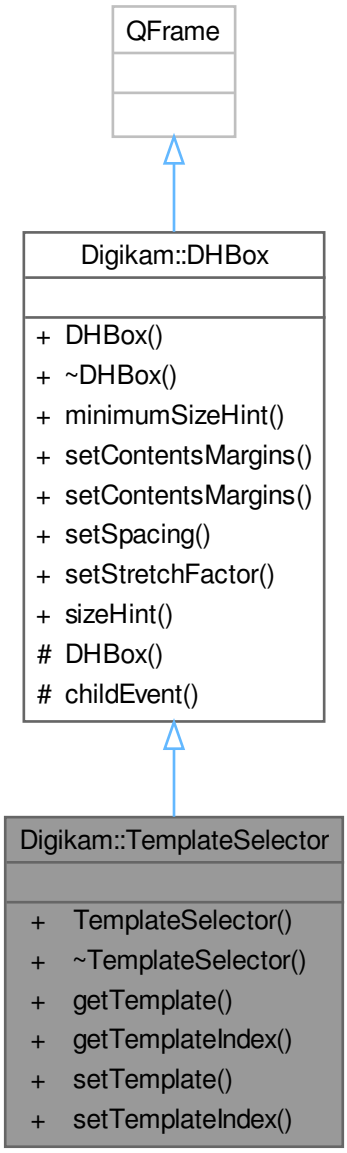
- enum **TemplateTab** { **RIGHTS** =0 , **LOCATION** , **CONTACT** , **SUBJECTS** }

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

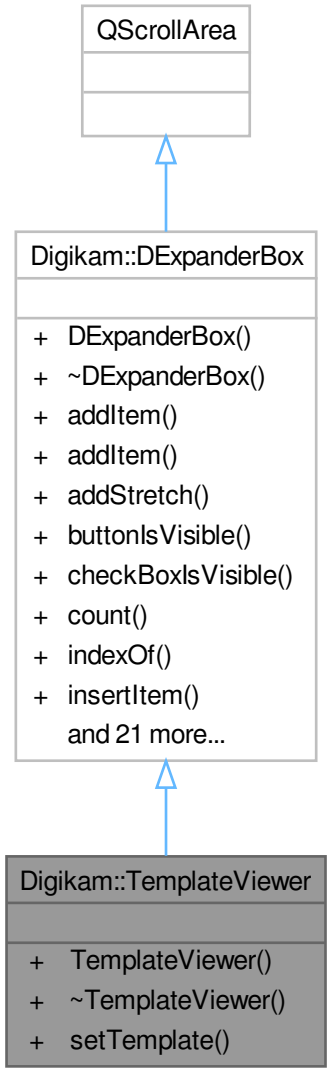
Additional Inherited Members

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 expand↩ Bydefault)
- 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)

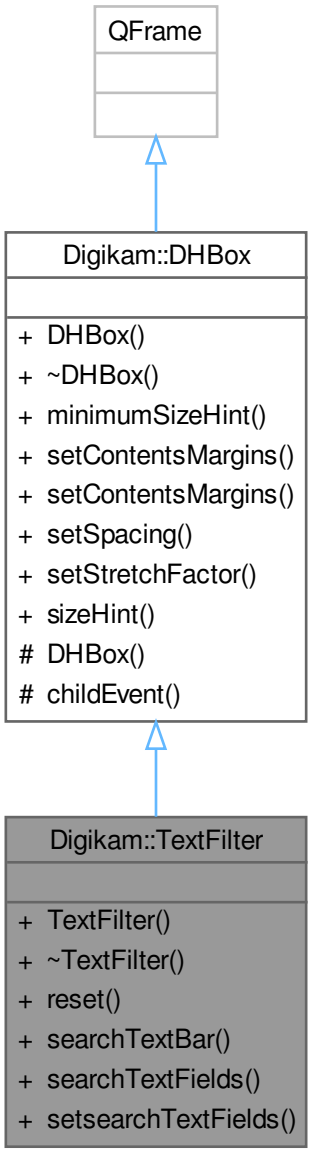
Additional Inherited Members

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 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** ()

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.

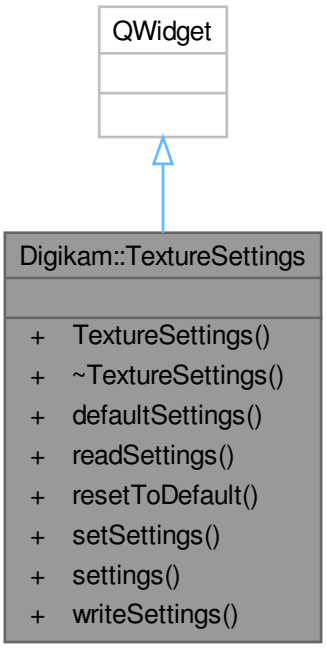
9.1311.1.3 readParameters()

```
void Digikam::TextureFilter::readParameters (
            const FilterAction & action )  [override], [virtual]
```

Implements Digikam::DImgThreadedFilter.

9.1312 Digikam::TextureSettings Class Reference

Inheritance diagram for Digikam::TextureSettings:



Signals

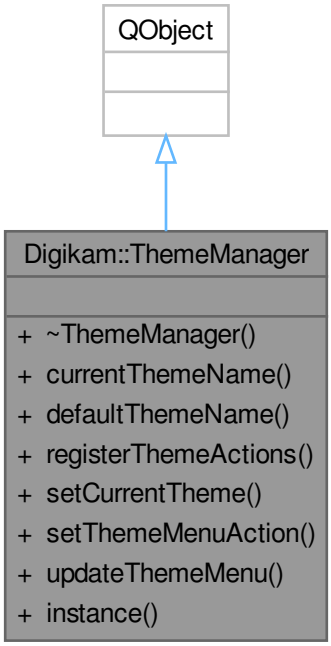
- void **signalSettingsChanged** ()

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

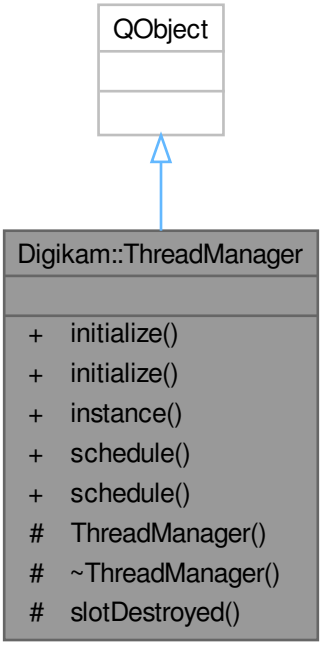
- static ThemeManager ∗ **instance** ()

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)

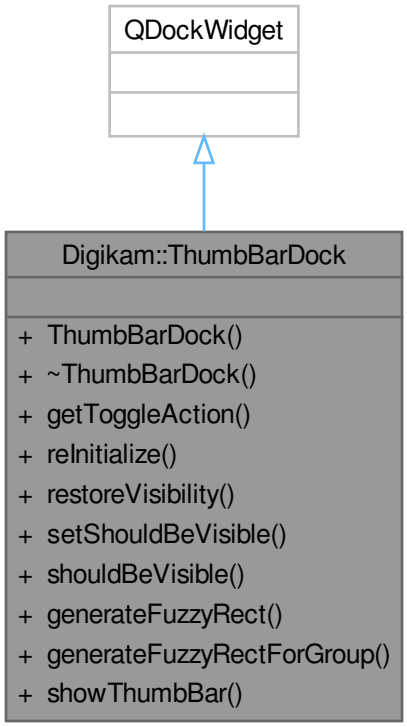
Friends

- 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.

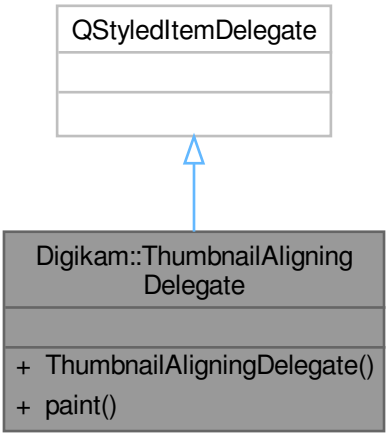
9.1315.2.2 shouldBeVisible()

```
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().

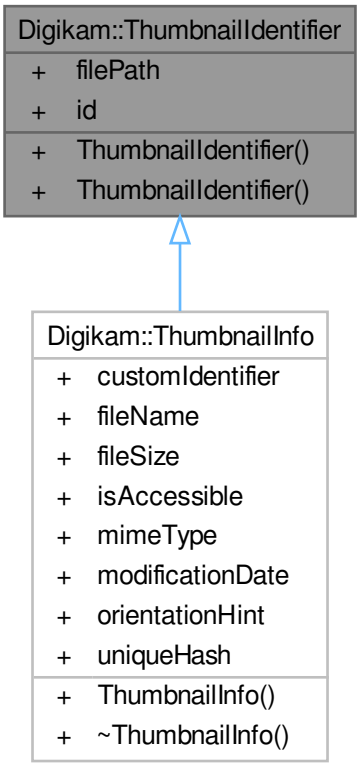
9.1317.2.10 storedSize()

```
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

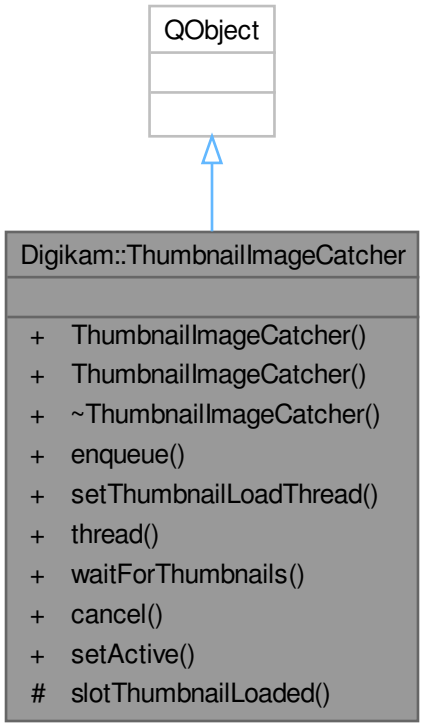
- QString **filePath**

  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.

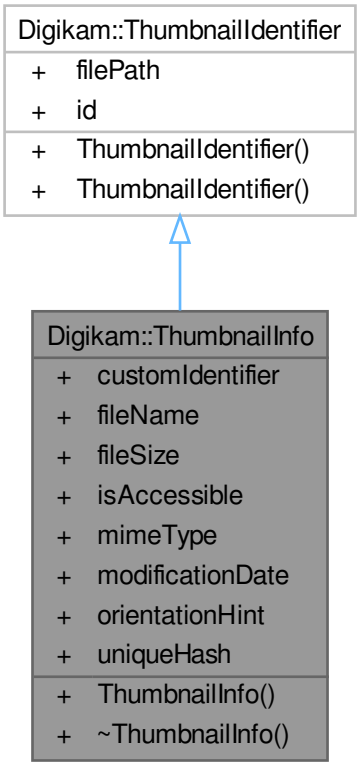
9.1319.2.3 setActive

```
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 = DMetadata::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.

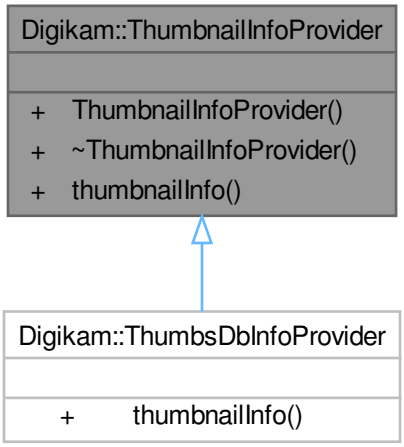
9.1320.1.4 orientationHint

```
int Digikam::ThumbnailInfo::orientationHint = DMetadata::ORIENTATION_UNSPECIFIED
```

This can be used to supersede the Exif information in the file. Will not be used if DMetadata::ORIENTATION_↩ UNSPECIFIED (default value)

9.1321 Digikam::ThumbnailInfoProvider Class Reference

Inheritance diagram for Digikam::ThumbnailInfoProvider:

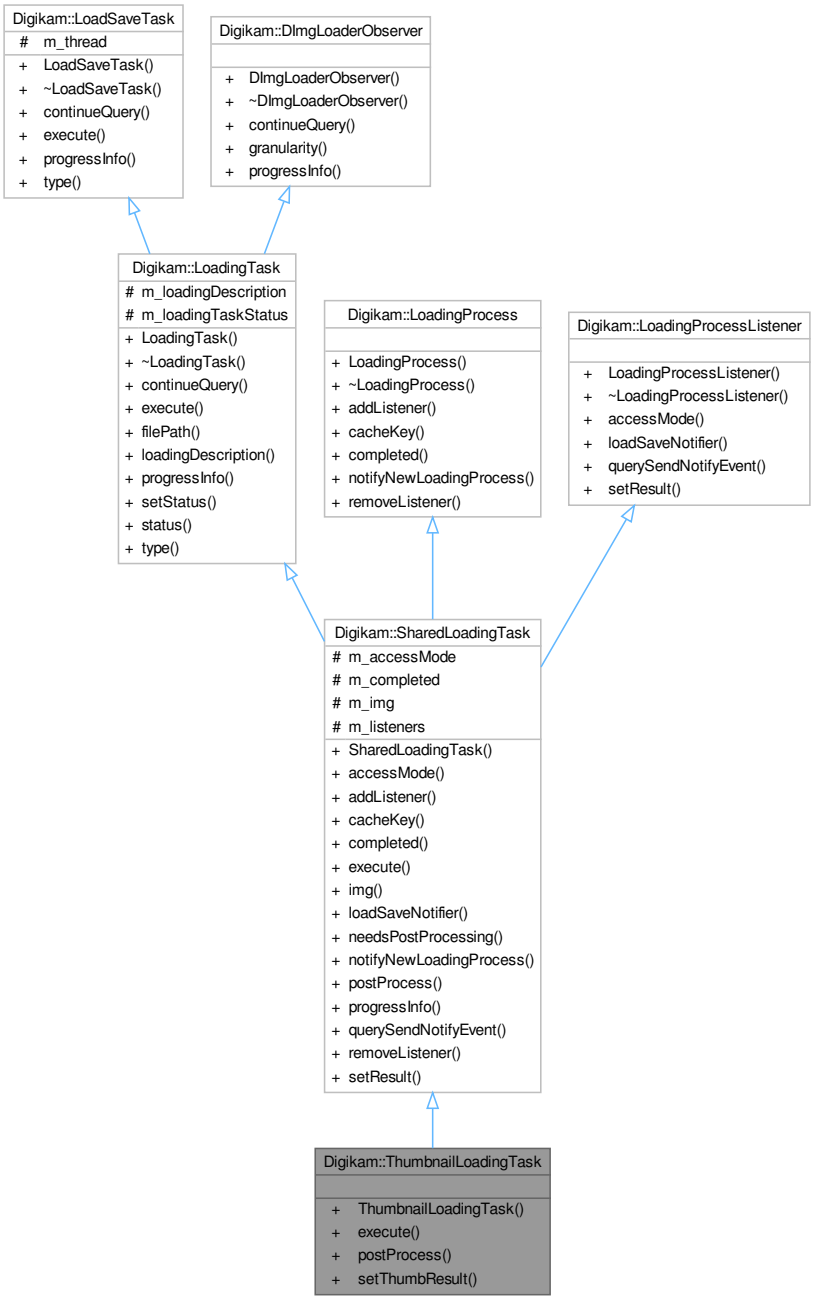


Public Member Functions

- 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.

9.1323 Digikam::ThumbnailLoadThread Class Reference

Inheritance diagram for Digikam::ThumbnailLoadThread:

Signals

- void signalThumbnailLoaded (const LoadingDescription &loadingDescription, const QPixmap &pix)
- void **ThumbnailLoaded** (const LoadingDescription &, const QImage &)
- void thumbnailsAvailable ()

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 lastInsert↩ Id=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

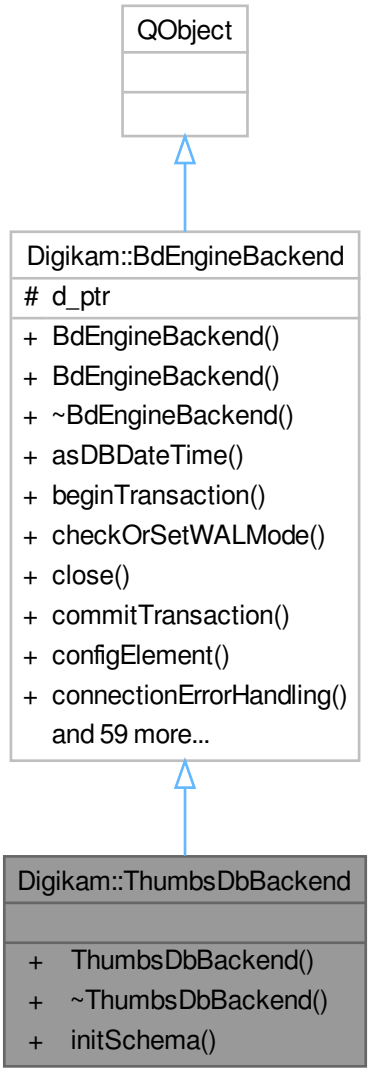
9.1326.2.1 setLastError()

```
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)
- 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 lastInsert↩Id=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 last↩InsertId=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 &field↩Names, 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().

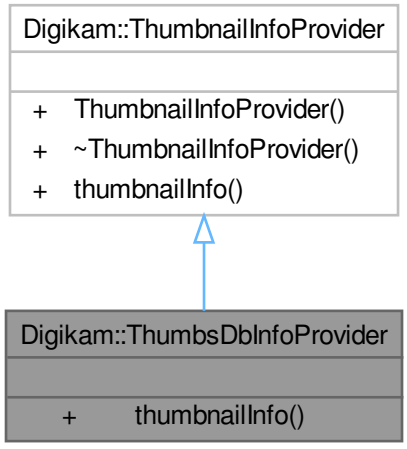
9.1328 Digikam::ThumbsDbInfo Class Reference

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

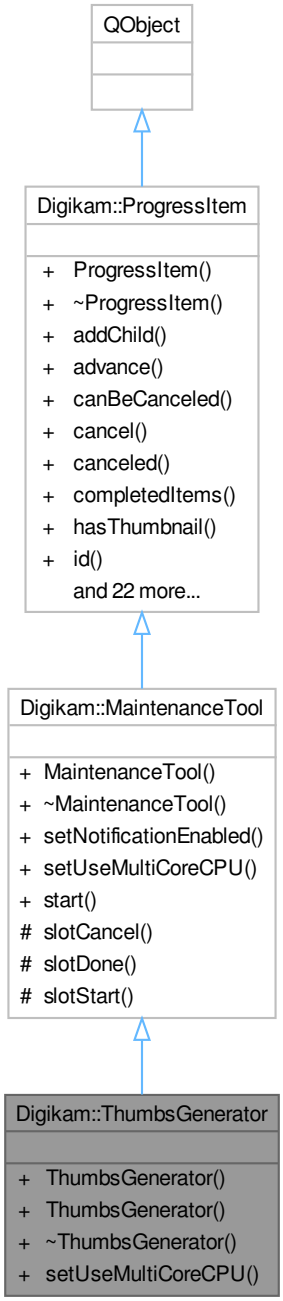
- **ThumbsDbSchemaUpdater** (ThumbsDbAccess ∗const dbAccess)
- void **setObserver** (InitializationObserver ∗const observer)
- bool **update** ()

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 elpased 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

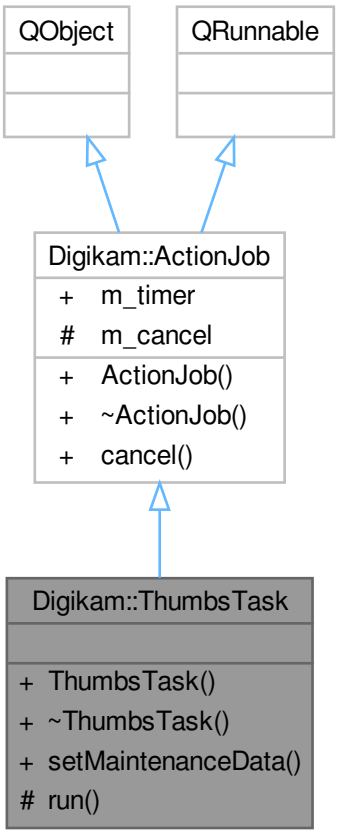
9.1331.2.1 setUseMultiCoreCPU()

```
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.

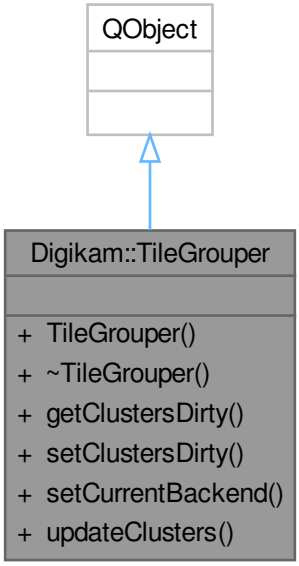
Protected Attributes inherited from Digikam::ActionJob

- 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< GeoIfaceSharedData > &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
- QIntList **toIntList** () const

Static Public Member Functions

- static TileIndex **fromCoordinates** (const Digikam::GeoCoordinates &coordinate, const int getLevel)
- static TileIndex **fromIntList** (const QIntList &intList)
- static bool **indicesEqual** (const TileIndex &a, const TileIndex &b, const int upToLevel)
- static QList< QIntList > **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 **UseDateSource** {
  APPDATE = 0 , **FILENAME** , **FILEDATE** , **METADATADATE** ,
  CUSTOMDATE }
- enum **UseFileDateType** { **FILELASTMOD** = 0 , **FILECREATED** }
- enum **UseMetaDateType** {
  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 **getDateTimeFromString** (const QString &dateStr) const
- QMap< QString, bool > **getDateTimeTagsMap** () const

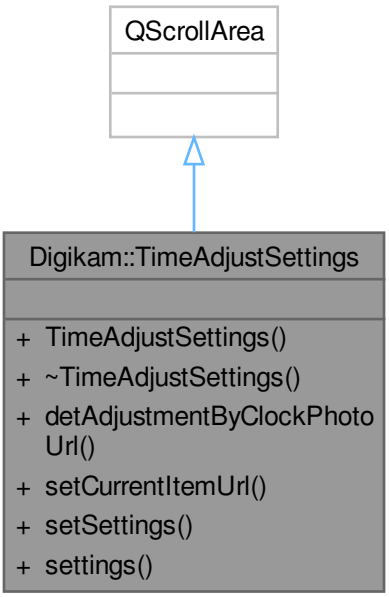
Public Attributes

- 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 **updIfAvailable** = 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

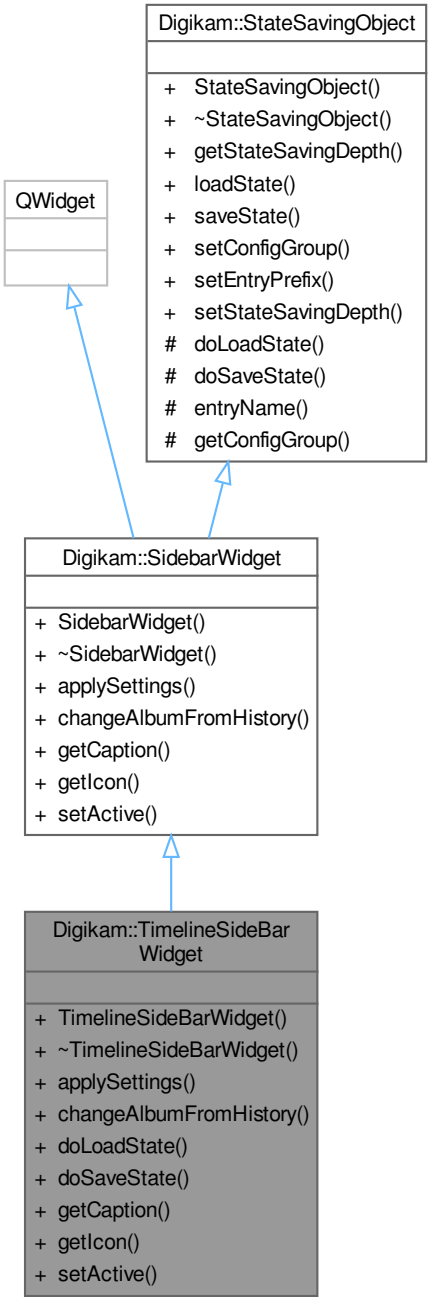
9.1336.1.1 detAdjustmentByClockPhotoUrl()

```
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]
```

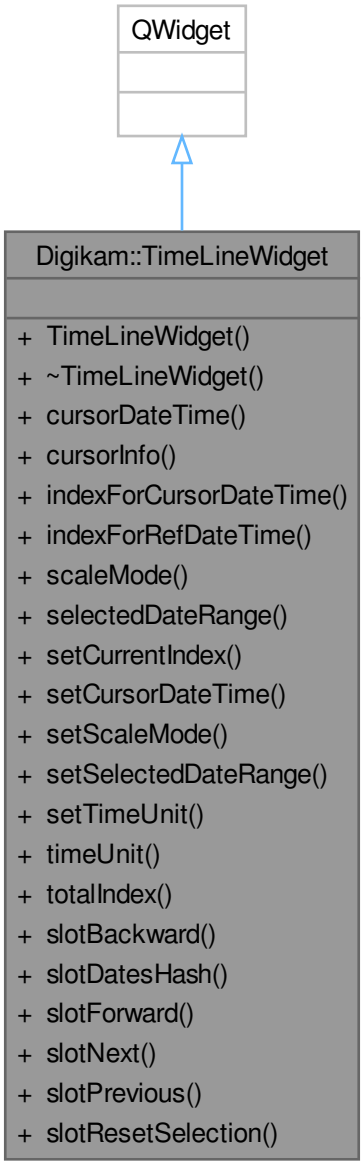
Parameters

active	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

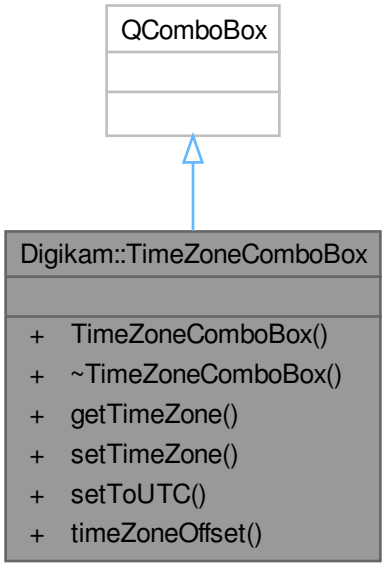
`enum Digikam::TimeLineWidget::SelectionMode`

Enumerator

Unselected	No selection.
FuzzySelection	Partially selected.
Selected	Fully selected.

9.1339 Digikam::TimeZoneComboBox Class Reference

Inheritance diagram for Digikam::TimeZoneComboBox:



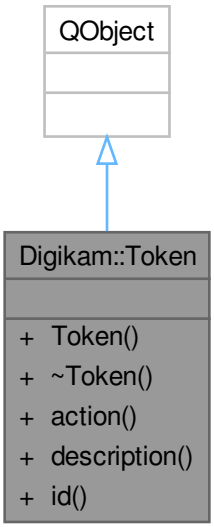
Public Member Functions

- **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 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** ()

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.

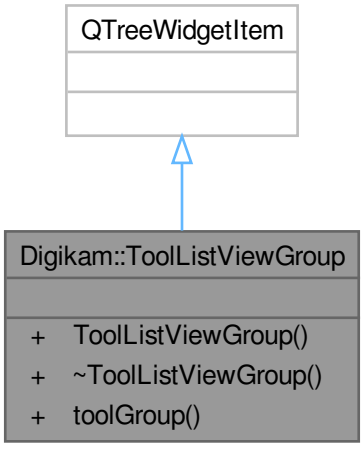
9.1342.1.3 readParameters()

```
void Digikam::TonalityFilter::readParameters (
            const FilterAction & action )  [override], [virtual]
```

Implements Digikam::DImgThreadedFilter.

9.1343 Digikam::ToolListViewGroup Class Reference

Inheritance diagram for Digikam::ToolListViewGroup:

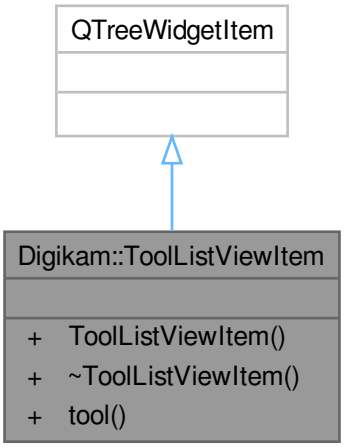


Public Member Functions

- **ToolListViewGroup** (QTreeWidget *const parent, BatchTool::BatchToolGroup group)
- BatchTool::BatchToolGroup **toolGroup** () const

9.1344 Digikam::ToolListViewItem Class Reference

Inheritance diagram for Digikam::ToolListViewItem:

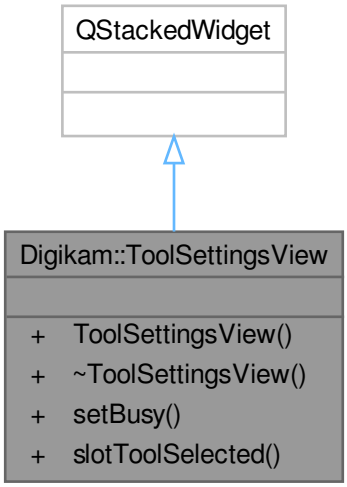


Public Member Functions

- **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

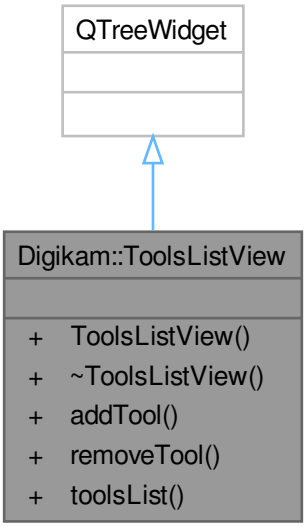
- void **signalSettingsChanged** (const BatchToolSet &)

Public Member Functions

- **ToolSettingsView** (QWidget ∗const parent=nullptr)
- void **setBusy** (bool b)

9.1346 Digikam::ToolsListView Class Reference

Inheritance diagram for Digikam::ToolsListView:



Signals

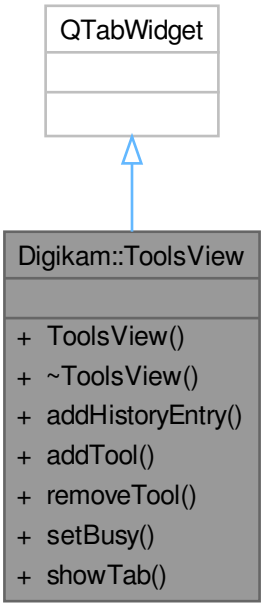
- void **signalAssignTools** (const QMultiMap< int, QString > &)

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 QMultiMap< 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 item↩ Id=-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

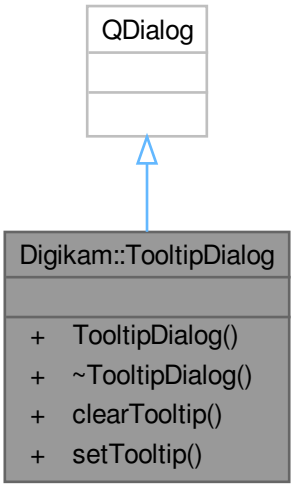
- QString **tooltip** (const Parser ∗const parser)

Static Public Member Functions

- static TooltipCreator & **getInstance** ()

9.1349 Digikam::TooltipDialog Class Reference

Inheritance diagram for Digikam::TooltipDialog:

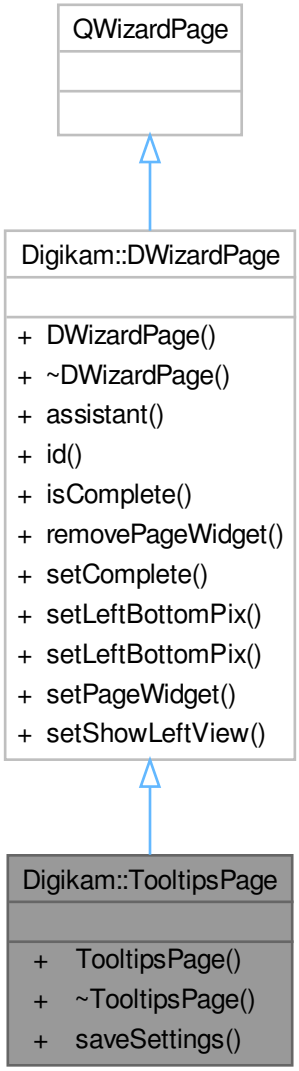


Public Member Functions

- **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** ()

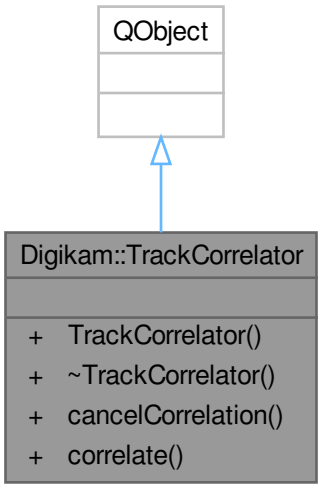
Public Member Functions inherited from Digikam::DWizardPage

- **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**

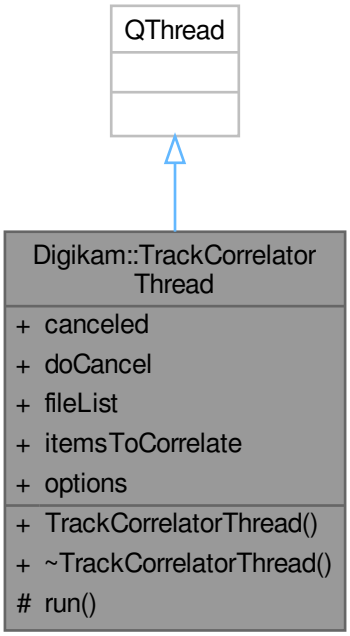
9.1353 Digikam::TrackCorrelator::CorrelationOptions Class Reference

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

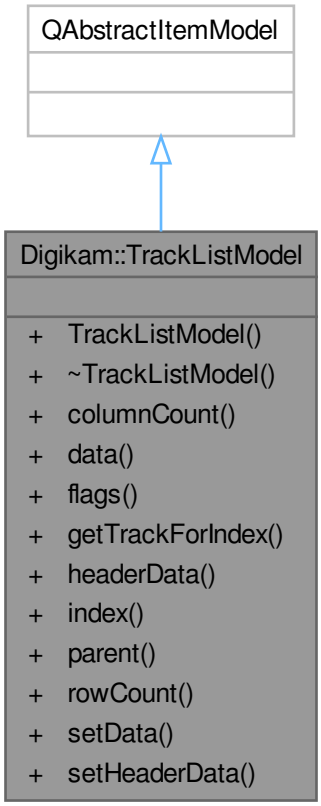
- bool **canceled** = false
- bool **doCancel** = false
- TrackManager::Track::List **fileList**
- TrackCorrelator::Correlation::List **itemsToCorrelate**
- TrackCorrelator::CorrelationOptions **options**

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()

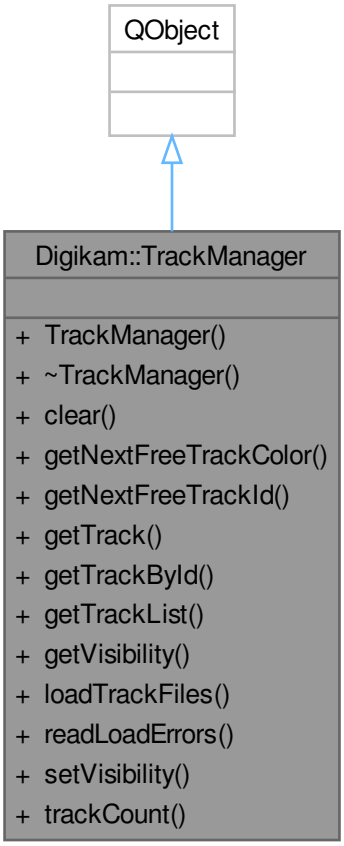
```
QVariant Digikam::TrackListModel::headerData (
            int section,
            Qt::Orientation orientation,
            int role ) const  [override]
```

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

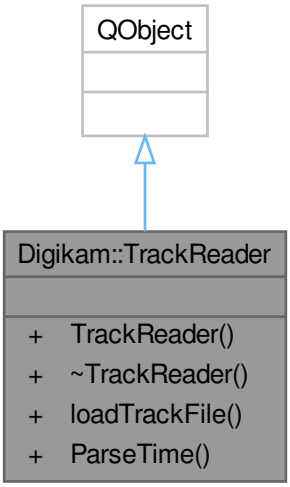
- static bool **EarlierThan** (const TrackPoint &a, const TrackPoint &b)

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
- FaceItemRetriever **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]
```

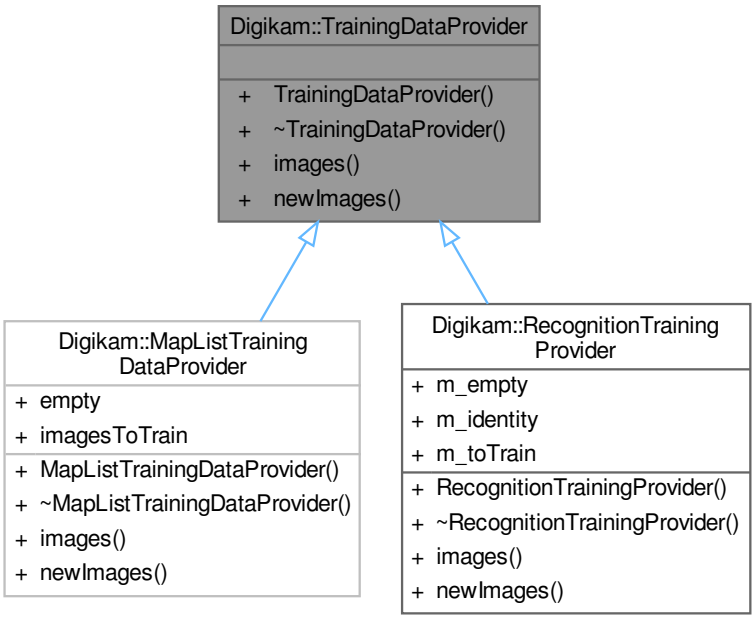
You can stop any extra controlled threads here. Immediately afterwards, an event will be sent to the working thread which will cause the event loop to quit. (aboutToQuitLoop())

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()

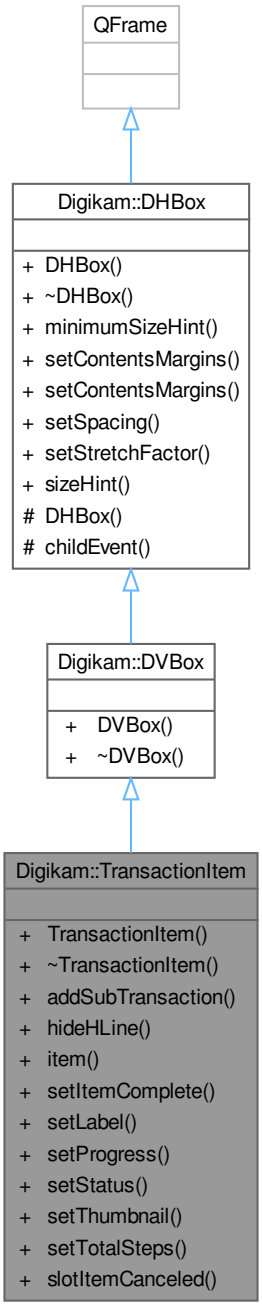
```
virtual ImageListProvider * Digikam::TrainingDataProvider::newImages (
            const Identity & identity )  [pure virtual]
```

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 **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

9.1363.1 Member Function Documentation

9.1363.1.1 setStatus()

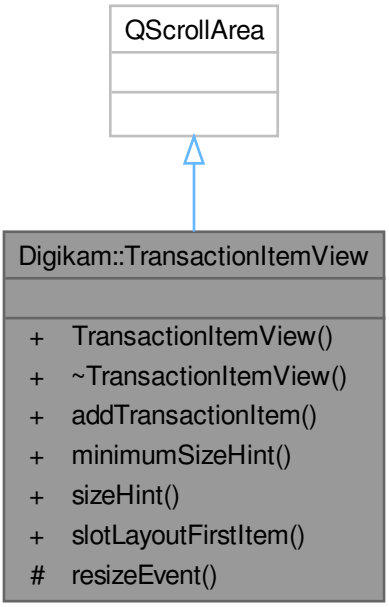
```
void Digikam::TransactionItem::setStatus (
            const QString & status )
```

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 **signalTransactionViewIsEmpty** ()

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

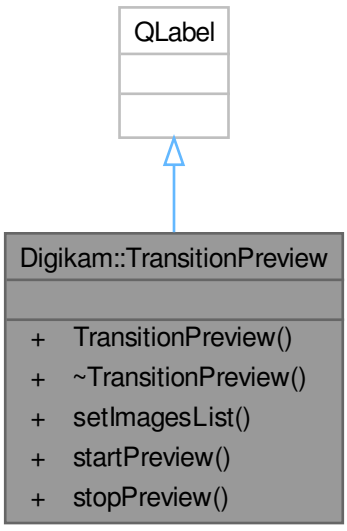
- QImage **currentFrame** (int &tmout)
- void **setInImage** (const QImage &iimg)
- void **setOutImage** (const QImage &oimg)
- void **setOutputSize** (const QSize &size)
- void **setTransition** (TransType type)

Static Public Member Functions

- static QMap< TransType, QString > **transitionNames** ()

9.1366 Digikam::TransitionPreview Class Reference

Inheritance diagram for Digikam::TransitionPreview:

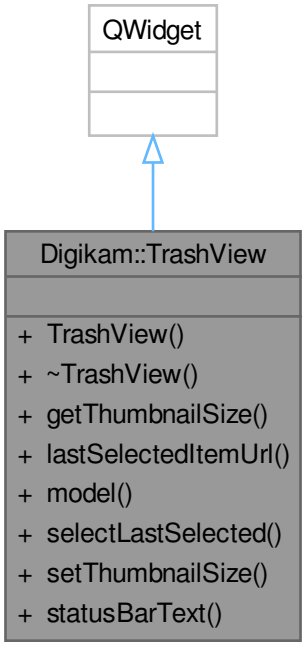


Public Member Functions

- **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

thumbSize	size to set

9.1367.1.5 statusBarText()

```
QString Digikam::TrashView::statusBarText ( ) const
```

Returns

text for the main status bar

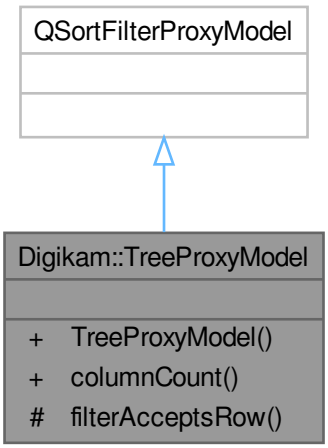
9.1368 Digikam::TreeBranch Class Reference

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

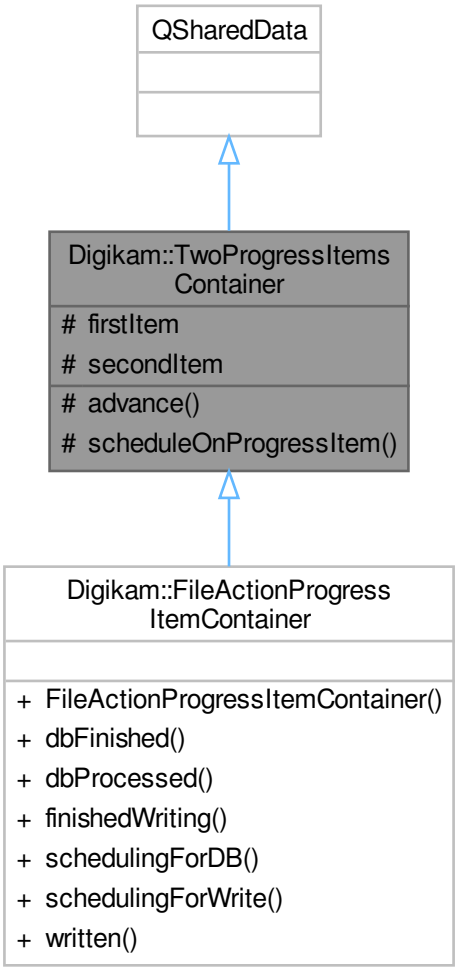
settings	contains settings
match	result of the regular expression match done in Option::parse()

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)

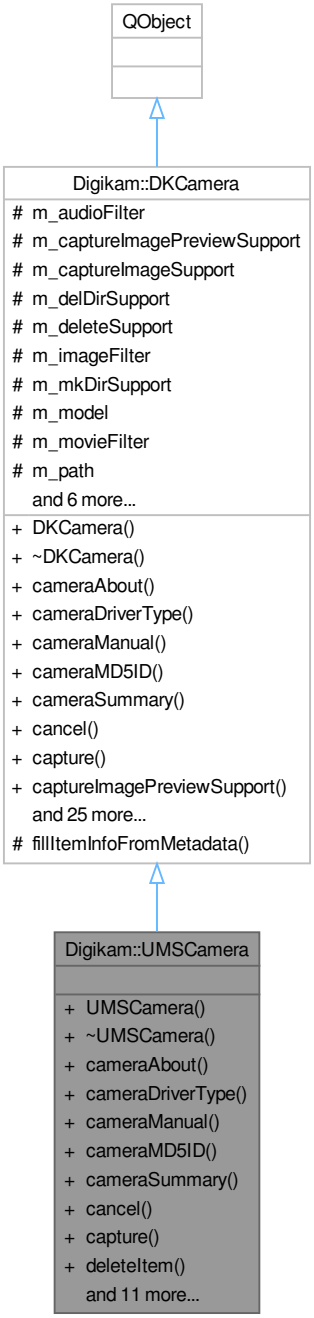
Protected Attributes

- 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 **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

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.

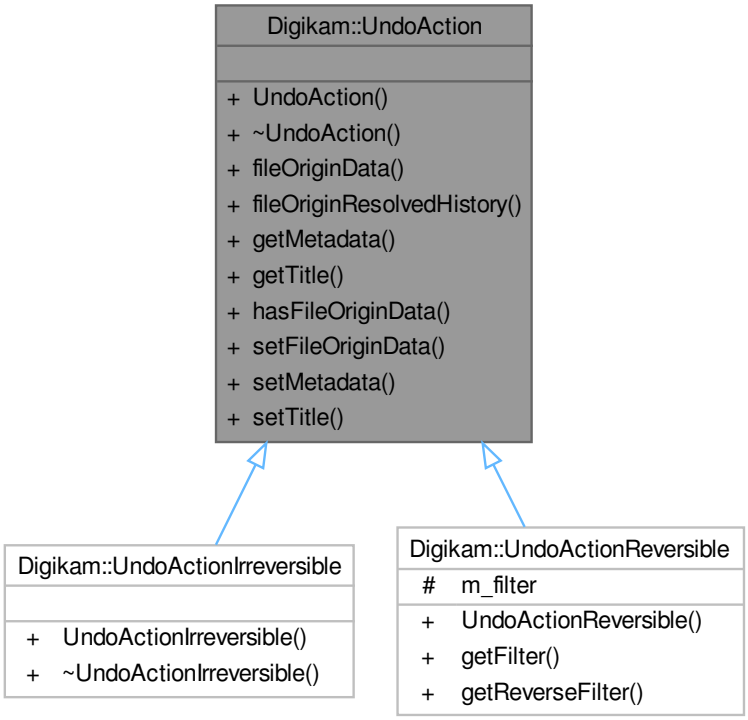
9.1374.1.19 uploadItem()

```
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:

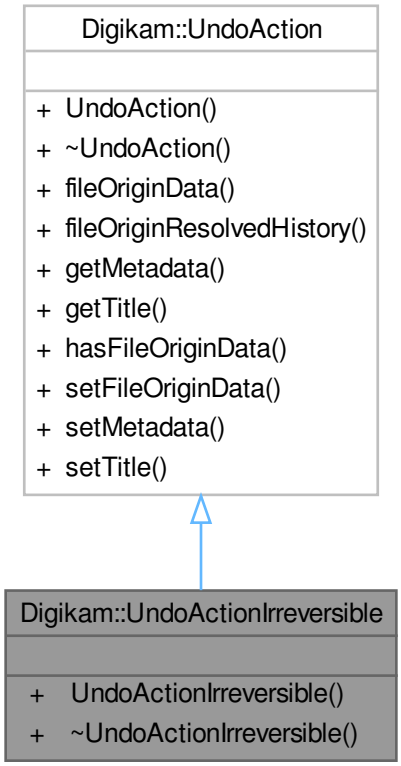


Public Member Functions

- **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

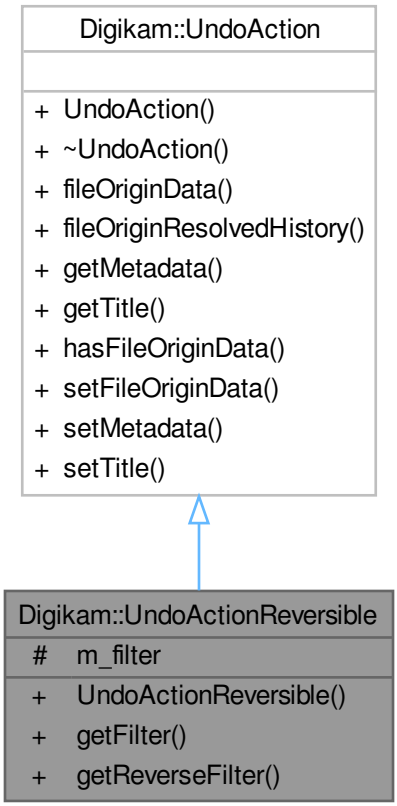
- **UndoActionIrreversible** (const EditorCore *const core, const QString &title=QString())

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 DImgBuiltinFilter &reversibleFilter)
- DImgBuiltinFilter **getFilter** () const
- DImgBuiltinFilter **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

  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.
- 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

settings	contains settings
match	result of the regular expression match done in Option::parse()

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** (DImg ∗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 **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

9.1383.1 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

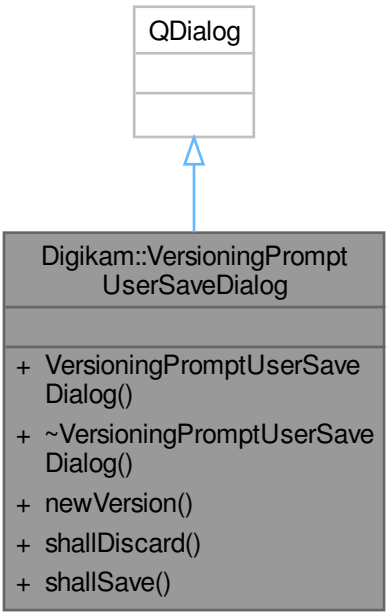
9.1385.2.1 VersionFileOperation()

```
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< qlonglong > &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

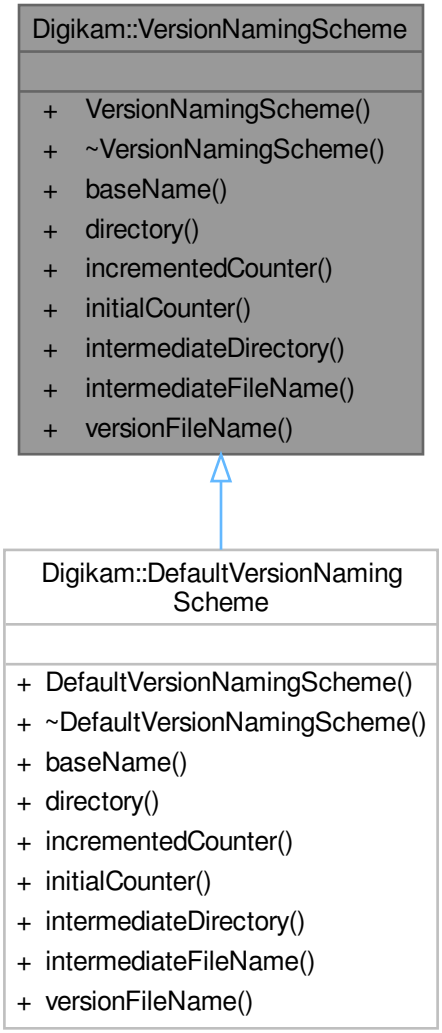
Public Attributes

- 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()

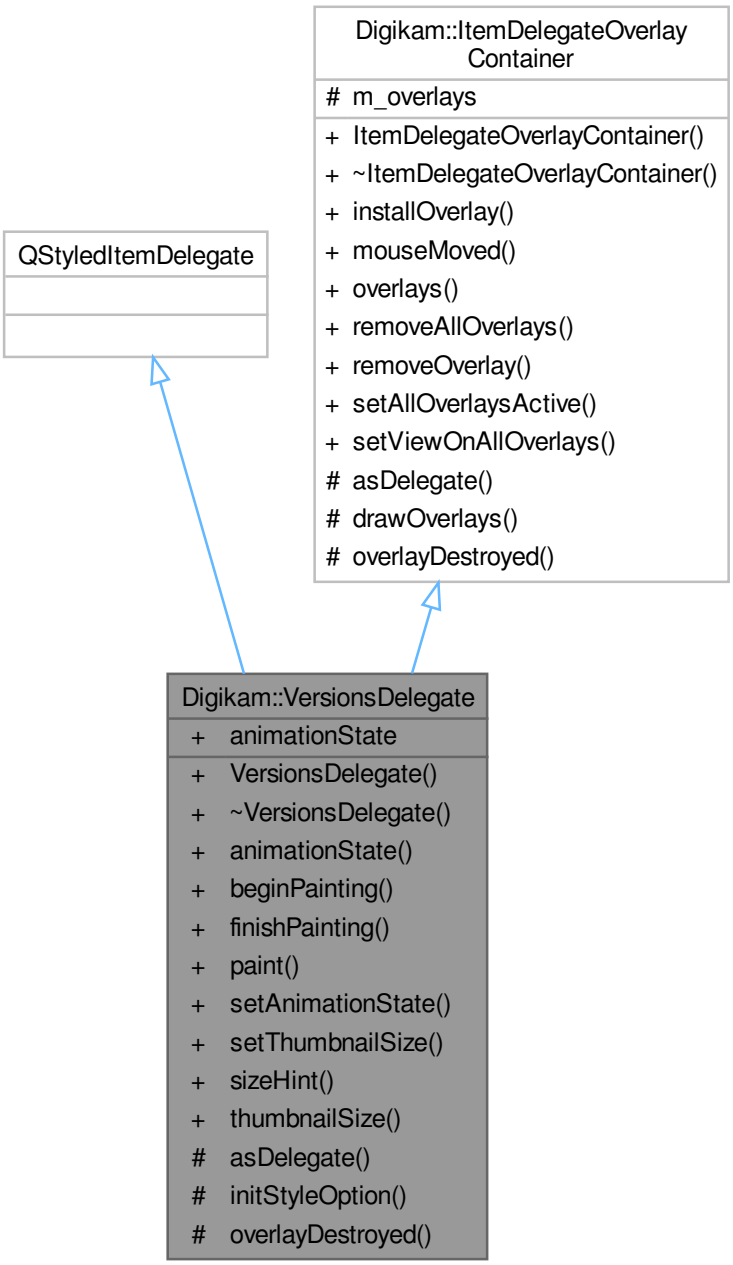
```
virtual QString Digikam::VersionNamingScheme::versionFileName (
            const QString & path,
            const QString & baseName,
            const QVariant & counter )  [pure virtual]
```

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

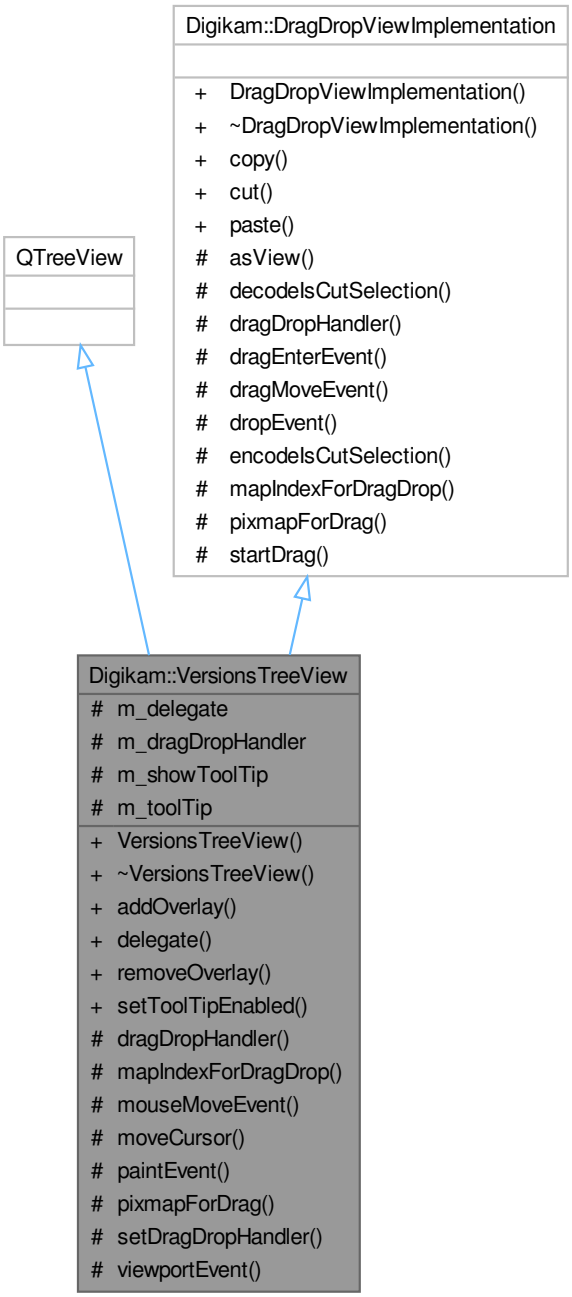
```
void Digikam::VersionsDelegate::requestNotification (
            const QModelIndex & index,
            const QString & message )  [signal]
```

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 **decodeIsCutSelection** (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 **encodeIsCutSelection** (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.

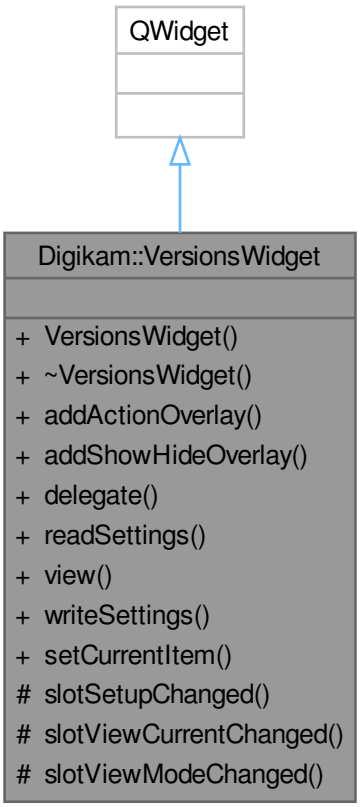
9.1392.2.3 pixmapForDrag()

```
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)

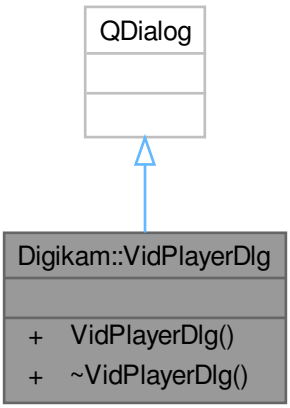
9.1400 Digikam::VideoThumbWriter Class Reference

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/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	https://en.wikipedia.org/wiki/X264
MPEG4	https://en.wikipedia.org/wiki/MPEG-4
MPEG2	https://en.wikipedia.org/wiki/MPEG-2
MJPEG	https://en.wikipedia.org/wiki/Motion_JPEG
FLASH	https://en.wikipedia.org/wiki/Adobe_Flash
WEBMVP8	https://en.wikipedia.org/wiki/VP8
THEORA	https://en.wikipedia.org/wiki/Theora
WMV7	https://en.wikipedia.org/wiki/Windows_Media_Video
WMV8	https://en.wikipedia.org/wiki/Windows_Media_Video
WMV9	https://en.wikipedia.org/wiki/Windows_Media_Video

9.1402.1.3 VidFormat

enum Digikam::VidSlideSettings::VidFormat

Enumerator

AVI	https://en.wikipedia.org/wiki/Audio_Video_Interleave
MKV	https://en.wikipedia.org/wiki/Matroska
MP4	https://en.wikipedia.org/wiki/MPEG-4_Part_14
MPG	https://en.wikipedia.org/wiki/MPEG-2

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

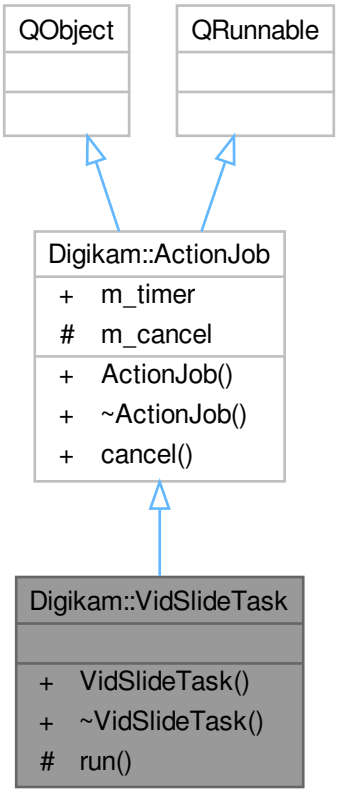
QVGA	320 x 180 - 16:9 - Computer Graphics
VCD1	352 x 240 - 7:5 - Digital TV
VCD2	352 x 288 - 6:5 - Digital TV
CVD1	352 x 480 - 11:15 - Digital TV
CVD2	352 x 576 - 11:18 - Digital TV
HVGA	480 x 270 - 16:9 - Computer Graphics
SVCD1	480 x 480 - 1:1 - Digital TV
SDTV1	528 x 480 - 11:10 - Digital TV
SDTV2	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.

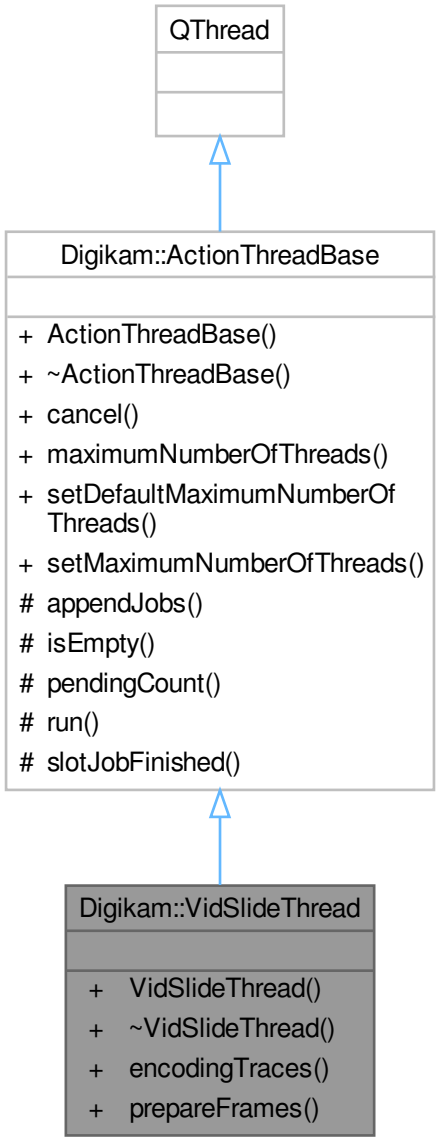
Protected Attributes inherited from Digikam::ActionJob

- 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

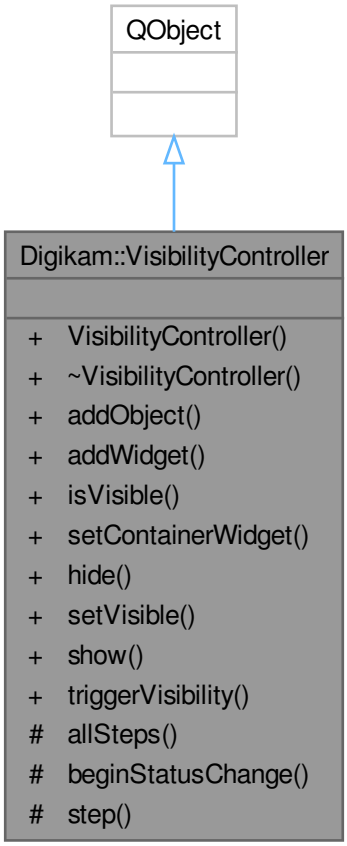
  Return true if list of pending jobs to process is empty.
- int **pendingCount** () const

  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

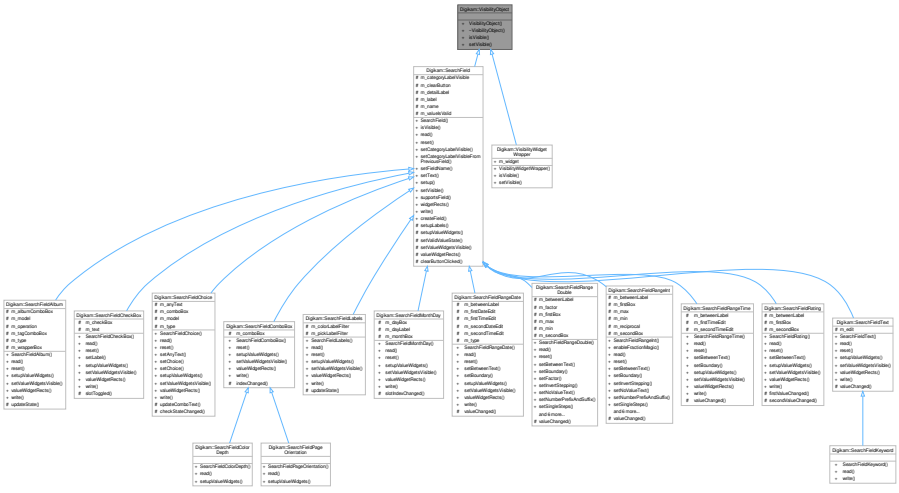
9.1405.1.1 addObject()

```
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 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 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.

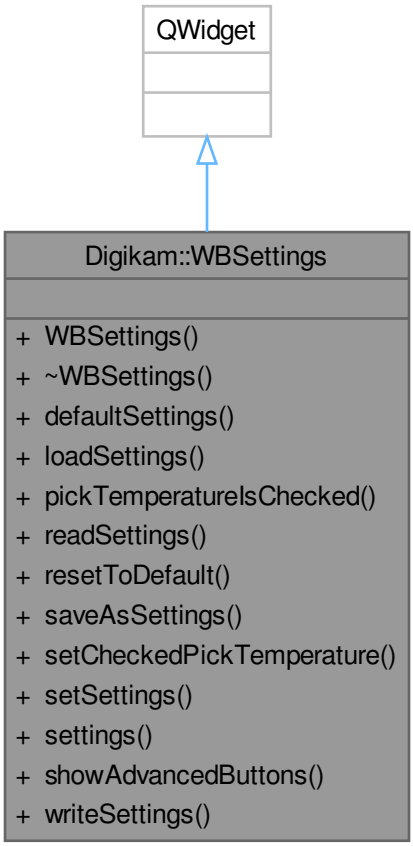
9.1408.1.5 readParameters()

```
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 **signalPickerColorButtonActived** ()
- void **signalSettingsChanged** ()

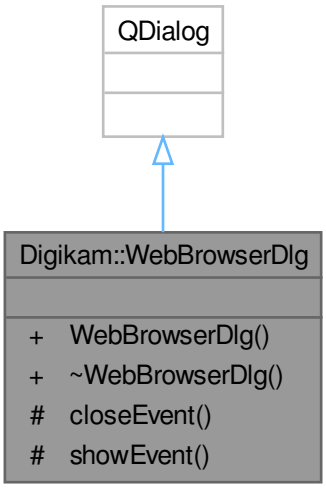
Public Member Functions

- **WBSettings** (QWidget ∗const parent)
- WBContainer **defaultSettings** () const
- void **loadSettings** ()
- bool **pickTemperatureIsChecked** ()
- 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

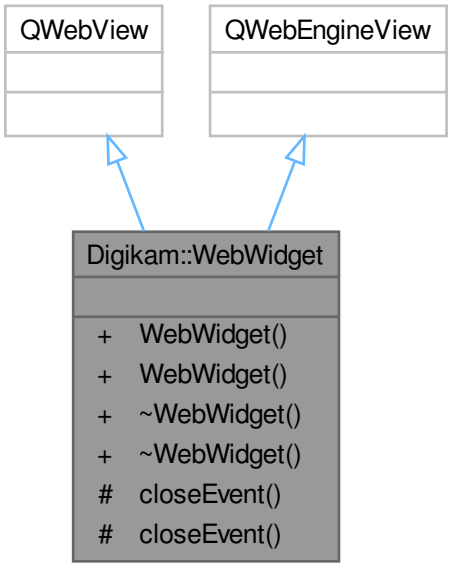
- **WebBrowserDlg** (const QUrl &url, QWidget ∗const parent, bool hideDeskBrowser=false)

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

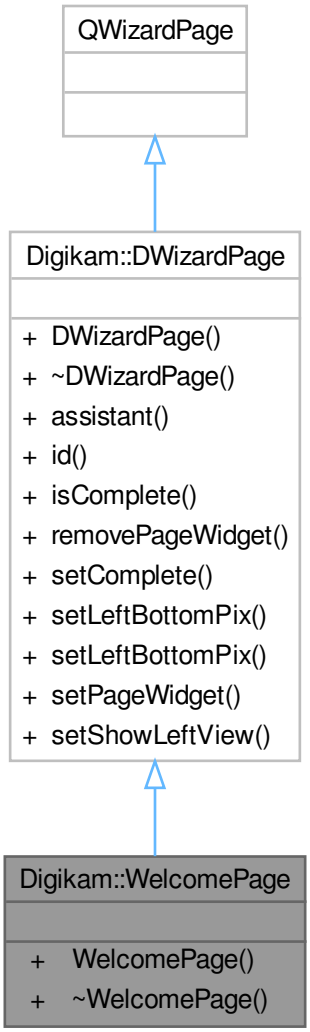
- **WebWidget** (QWidget ∗const parent=nullptr)
- **WebWidget** (QWidget ∗const parent=nullptr)

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)

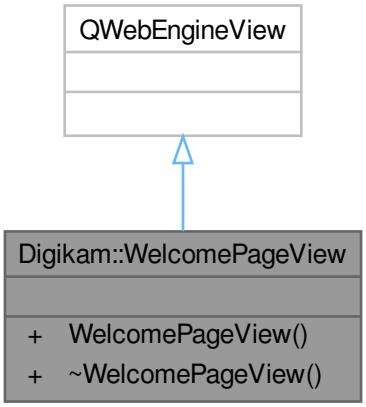
Public Member Functions inherited from Digikam::DWizardPage

- **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:

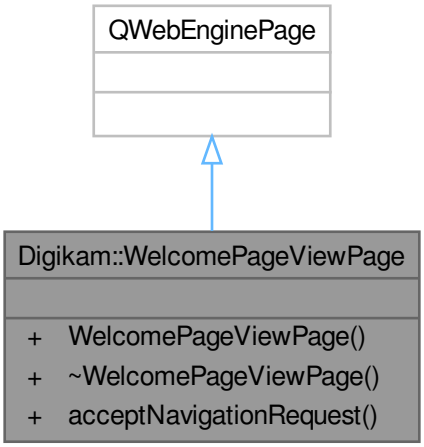


Public Member Functions

- **WelcomePageView** (QWidget ∗const parent)

9.1414 Digikam::WelcomePageViewPage Class Reference

Inheritance diagram for Digikam::WelcomePageViewPage:



Signals

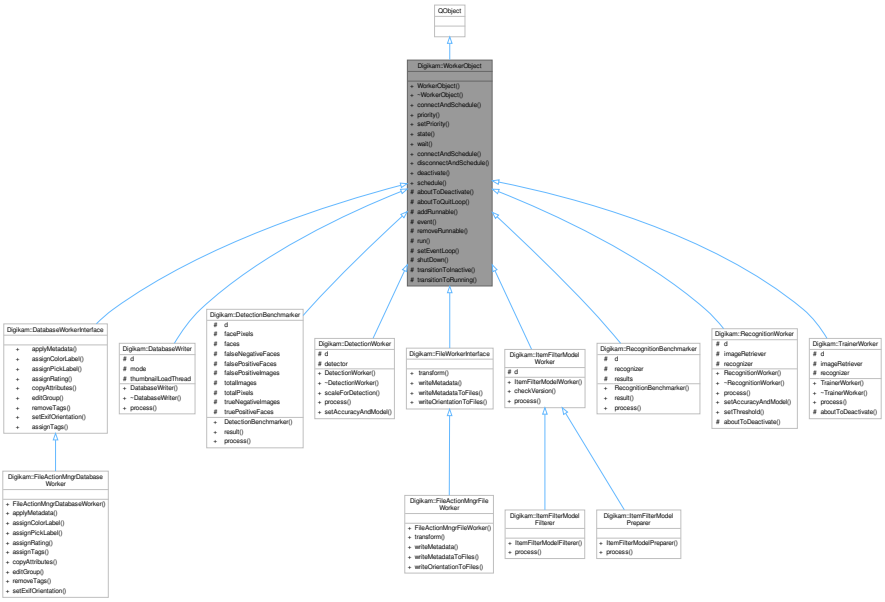
- void **linkClicked** (const QUrl &)

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

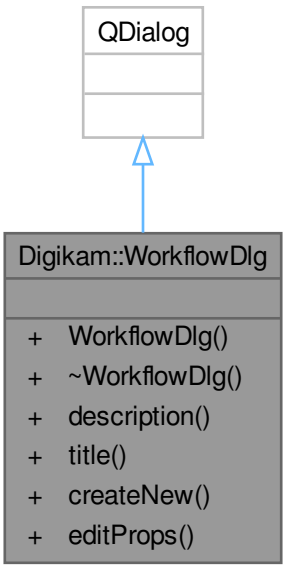
This container group all queue common settings plus all assigned batch tools.

Public Attributes

- BatchSetList **aTools**
- QString **desc**
- QueueSettings **qSettings**
- QString **title**

9.1417 Digikam::WorkflowDlg Class Reference

Inheritance diagram for Digikam::WorkflowDlg:



Public Member Functions

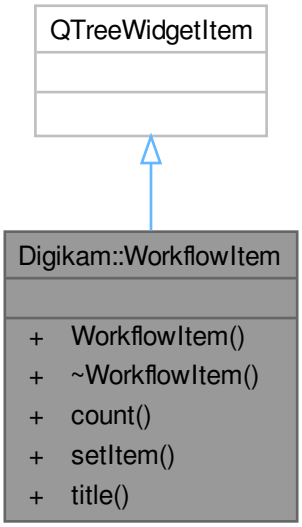
- **WorkflowDlg** (const Workflow &wf, bool create=false)
- QString **description** () const
- QString **title** () const

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:

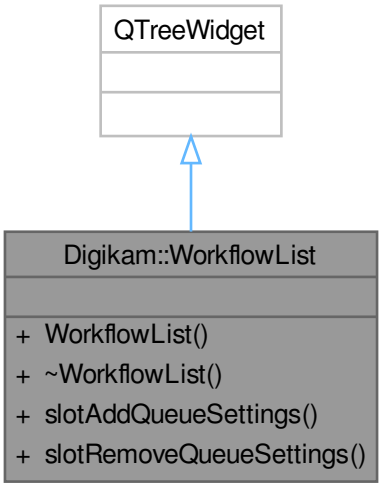


Public Member Functions

- **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

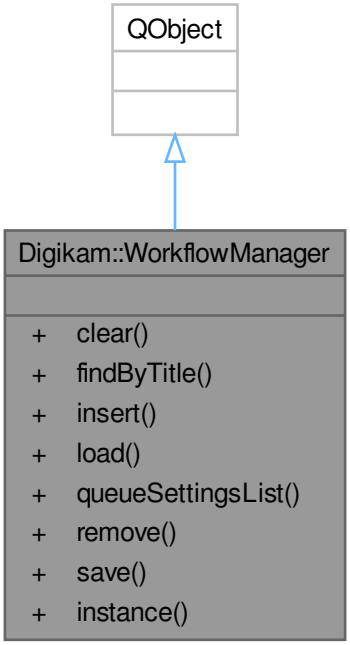
- void **signalAssignQueueSettings** (const QString &)
- void **signalUpdateQueueSettings** (const QString &)

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

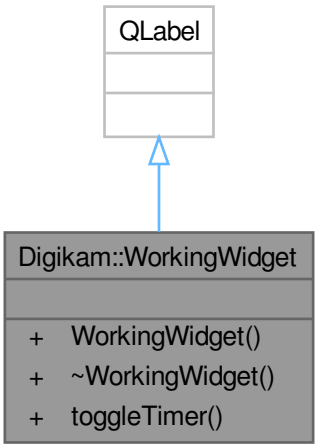
9.1420.1.1 load()

```
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)

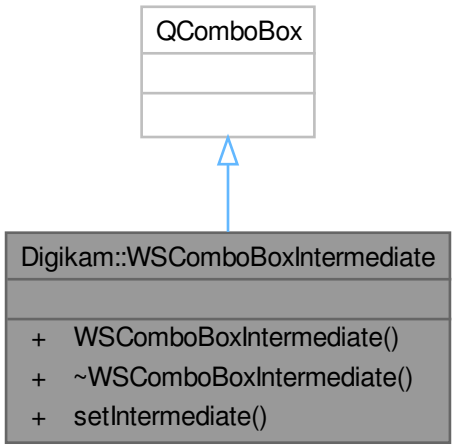
  This method is used by derived class of WSAblum, to set the attributes inherited from WSAlbum, knowing a WSAlbum.

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

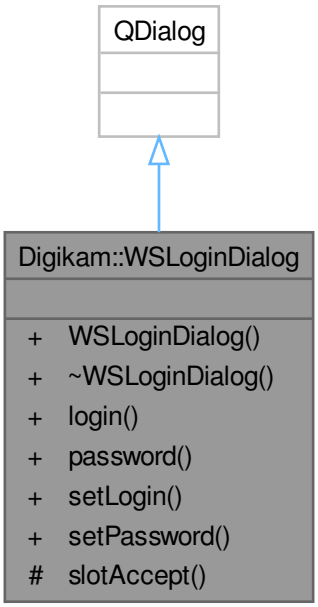
9.1423.1.1 setIntermediate()

```
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

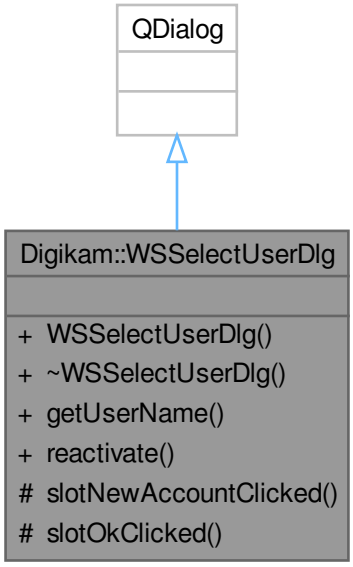
- **WSNewAlbumDialog** (QWidget *const parent, const QString &toolName)
- void **addToMainLayout** (QWidget *const widget)
- QGroupBox * **getAlbumBox** () const
- QDialogButtonBox * **getButtonBox** () const
- QDateTimeEdit * **getDateTimeEdit** () const
- DTextEdit * **getDescEdit** () const
- DTextEdit * **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

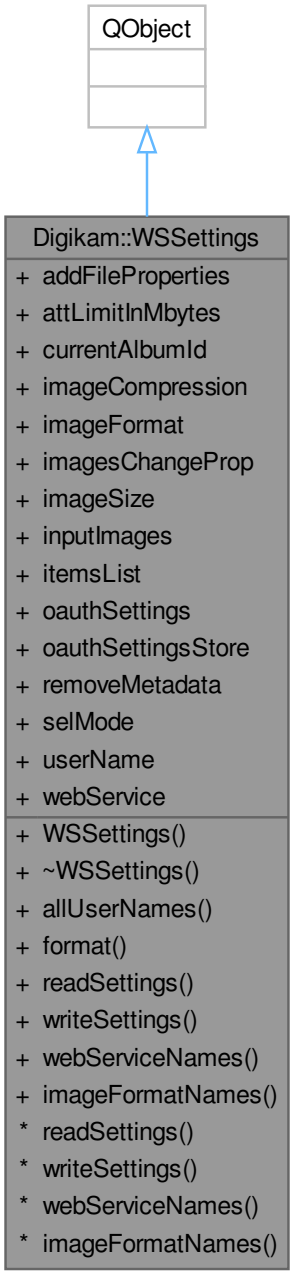
- **WSSelectUserDlg** (QWidget *const parent, const QString &serviceName)
- QString **getUserName** () const
- void **reactivate** ()

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)
- QStringList **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**

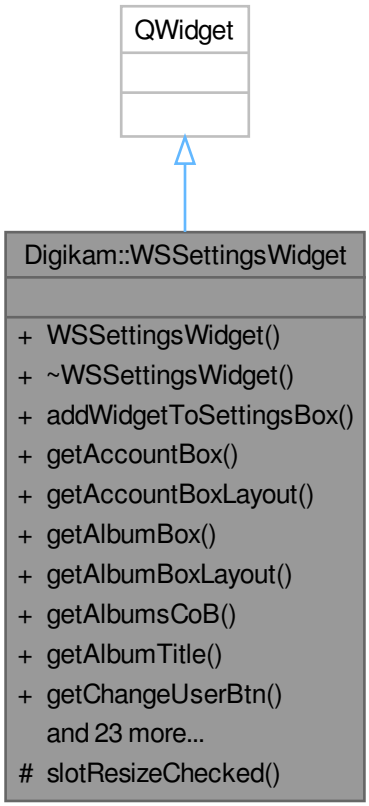
  Selected items to upload.
- QMap< QUrl, QUrl > **itemsList**

  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

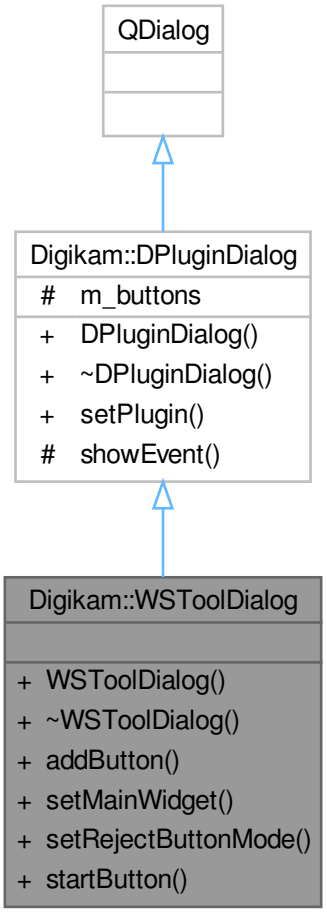
- QGridLayout * **getOptionsBoxLayout** () const
- QCheckBox * **getOriginalCheckBox** () const
- QCheckBox * **getPhotoIdCheckBox** () const
- QPushButton * **getReloadBtn** () const
- QCheckBox * **getResizeCheckBox** () const
- QWidget * **getSettingsBox** () const
- QVBoxLayout * **getSettingsBoxLayout** () const
- QGroupBox * **getSizeBox** () const
- QVBoxLayout * **getSizeBoxLayout** () const
- QGroupBox * **getUploadBox** () const
- QVBoxLayout * **getUploadBoxLayout** () const
- QLabel * **getUserNameLabel** () const
- DItemsList * **imagesList** () const
- DProgressWdg * **progressBar** () const
- void **replaceImageList** (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

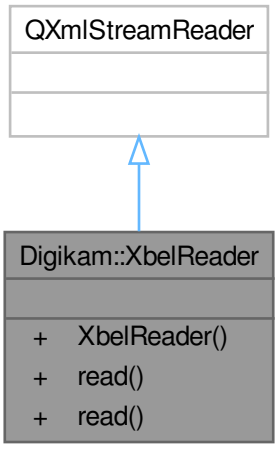
Static Public Member Functions

- 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:

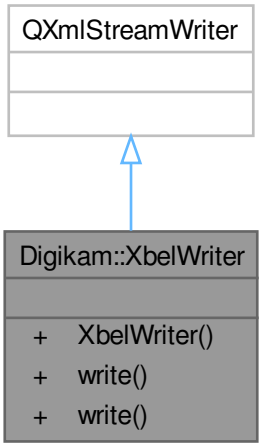


Public Member Functions

- BookmarkNode ∗ **read** (const QString &fileName)
- BookmarkNode ∗ **read** (QIODevice ∗const device, bool addRootFolder=false)

9.1432 Digikam::XbelWriter Class Reference

Inheritance diagram for Digikam::XbelWriter:

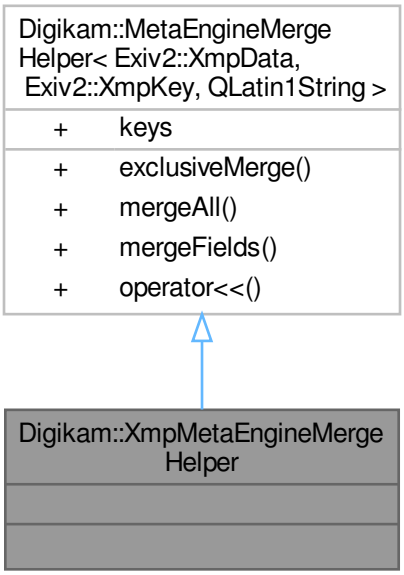


Public Member Functions

- 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)

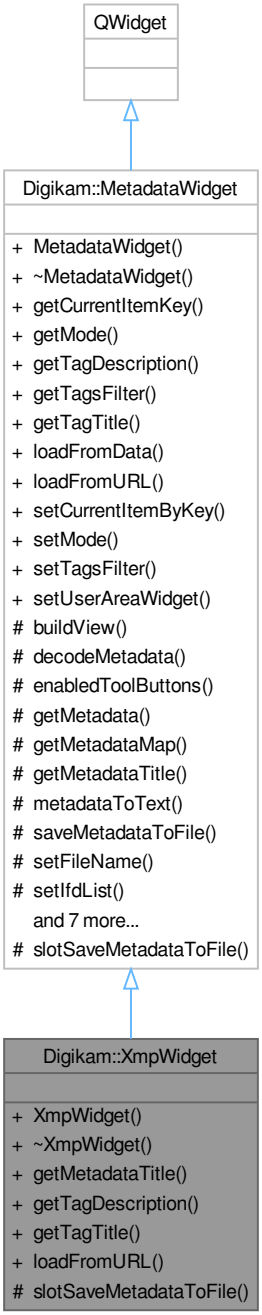
  Merge two (Exif,IPTC,Xmp) Data packages, the result is stored in dest.
- MetaEngineMergeHelper & **operator<<** (const QLatin1String &key)

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 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

9.1434.1 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.

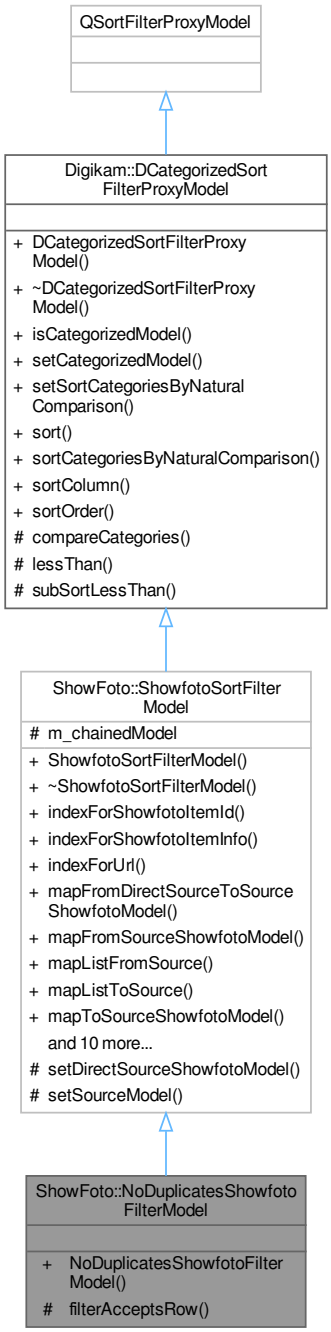
9.1434.1.4 loadFromURL()

```
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** (qlonglong 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

- 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

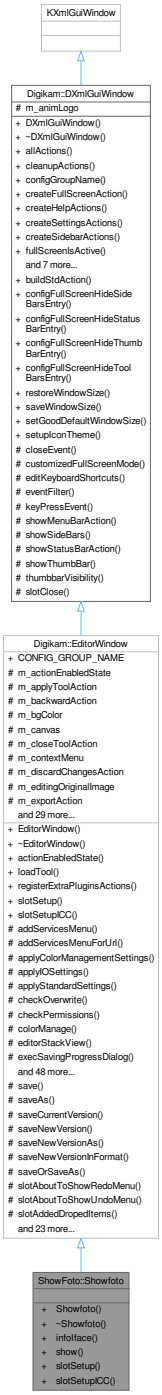
- enum AdditionalRoles { CategoryDisplayRole = 0x17CE990A , CategorySortRole = 0x27857E60 }

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 * infoIface (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 **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.
- 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 **slotAddedDropedItems** (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 infoIface()

```
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** (qlonglong 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 **decodeIsCutSelection** (const QMimeData ∗mimeData)
- void **encodeIsCutSelection** (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 **decodeIsCutSelection** (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 **encodeIsCutSelection** (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()

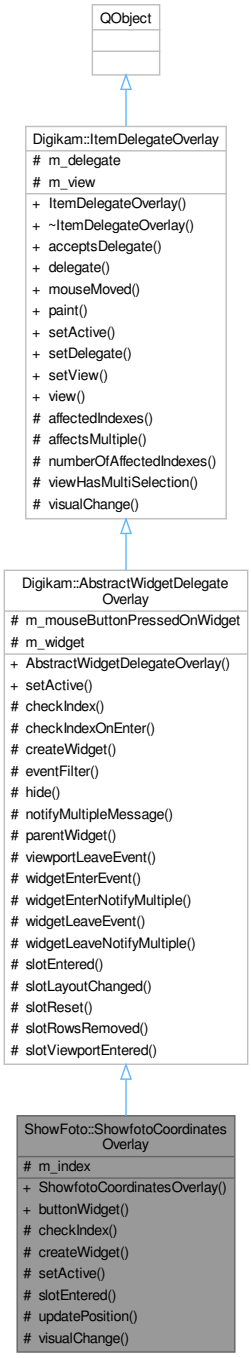
9.1437.1.11 showfotoItemInfoActivated

```
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 overlayed 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.

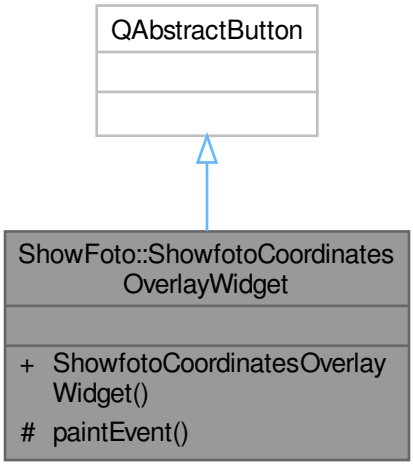
9.1438.1.5 visualChange()

```
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

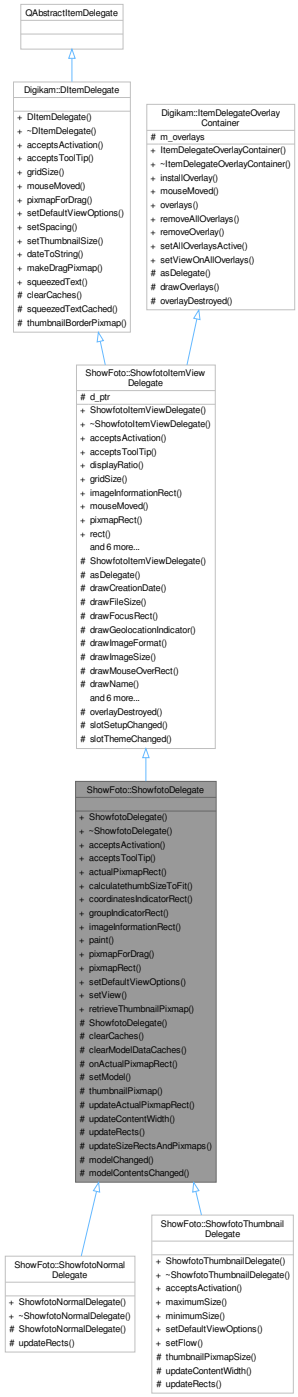
- **ShowfotoCoordinatesOverlayWidget** (QWidget ∗const parent=nullptr)

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.

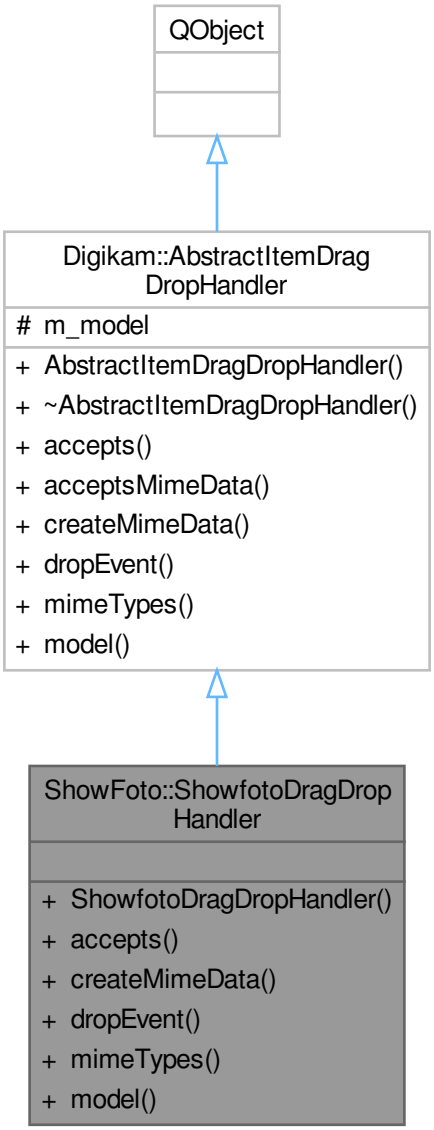
9.1440.1.10 updateSizeRectsAndPixmaps()

```
void ShowFoto::ShowfotoDelegate::updateSizeRectsAndPixmaps ( )  [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 mimeTypes()

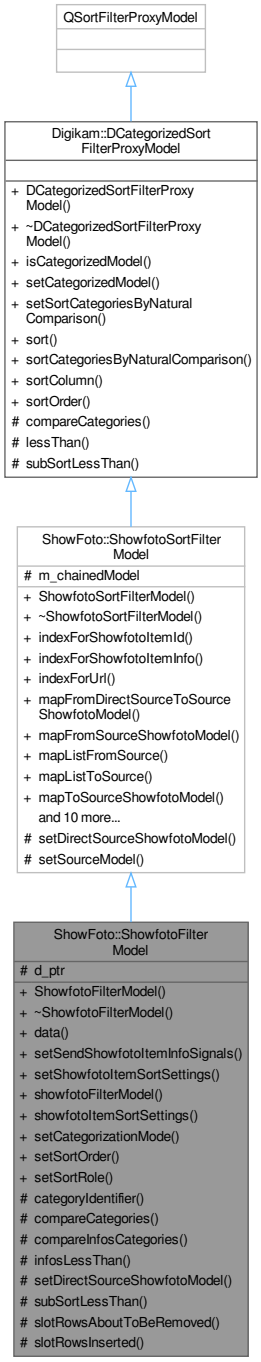
```
QStringList ShowFoto::ShowfotoDragDropHandler::mimeTypes ( ) const  [override], [virtual]
```

Called by the default implementation of model's mimeTypes().

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** (qlonglong 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)
- 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 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: GroupIsOpenRole = 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 qlonglong (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.

9.1442.2.3 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.

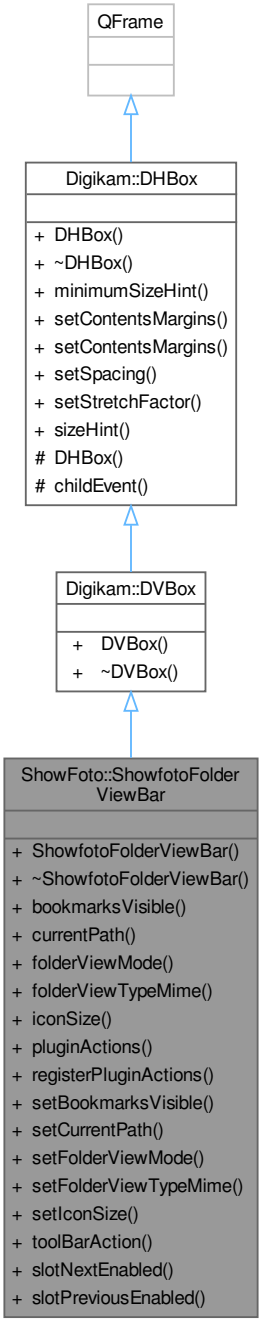
Returns

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

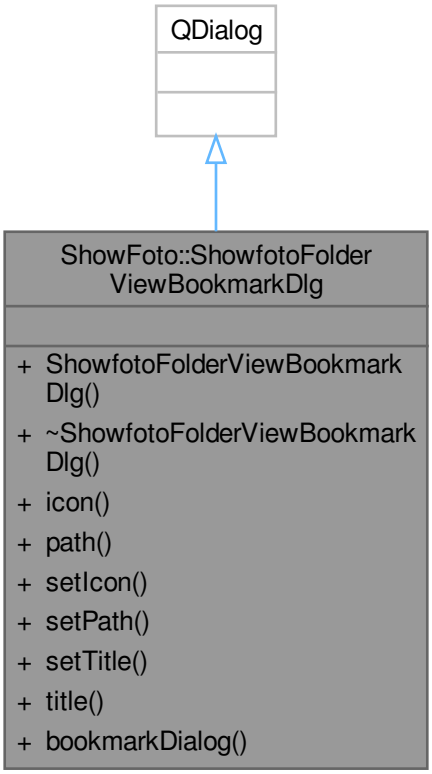
Additional Inherited Members

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

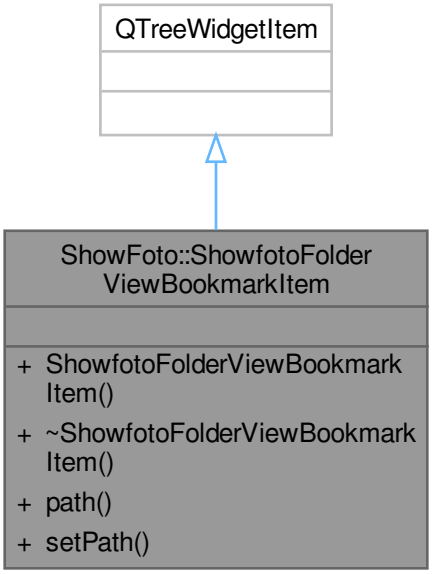
- **ShowfotoFolderViewBookmarkDlg** (ShowfotoFolderViewBookmarkList ∗const parent, bool create=false)
- QString **icon** () const
- QString **path** () const
- void **setIcon** (const QString &icon)
- void **setPath** (const QString &path)
- void **setTitle** (const QString &title)
- QString **title** () 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:

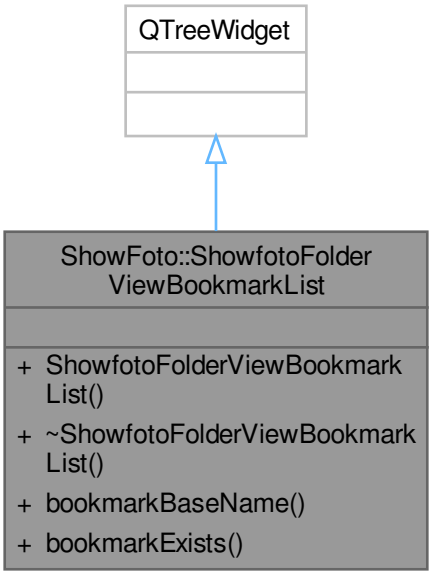


Public Member Functions

- **ShowfotoFolderViewBookmarkItem** (QTreeWidgetItem ∗const parent)
- QString **path** () const
- void **setPath** (const QString &)

9.1446 ShowFoto::ShowfotoFolderViewBookmarkList Class Reference

Inheritance diagram for ShowFoto::ShowfotoFolderViewBookmarkList:



Signals

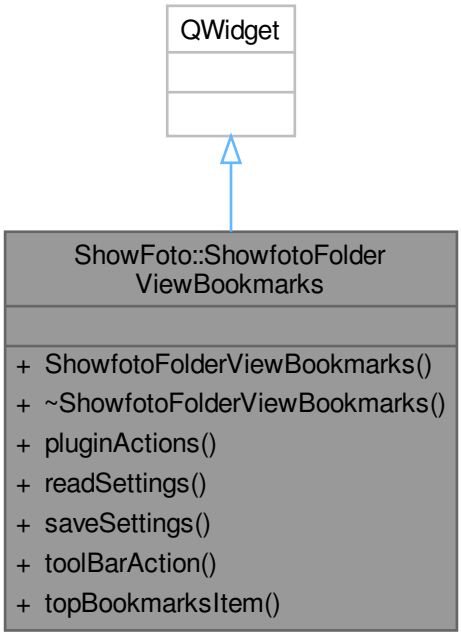
- void **signalAddBookmark** (const QString &path)
- void **signalLoadContents** (const QString &path)

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

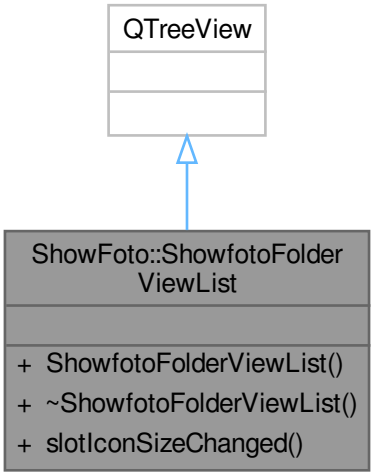
- void **signalLoadContents** ()

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 , **DetailledView** }
- 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

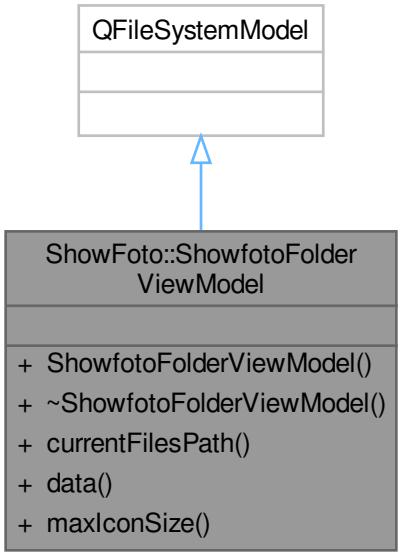
```
enum ShowFoto::ShowfotoFolderViewList::FolderViewRole
```

Enumerator

FileDate	Modifier date.

9.1449 ShowFoto::ShowfotoFolderViewModel Class Reference

Inheritance diagram for ShowFoto::ShowfotoFolderViewModel:



Public Member Functions

- **ShowfotoFolderViewModel** (ShowfotoFolderViewList ∗const view)
- QStringList **currentFilesPath** () const

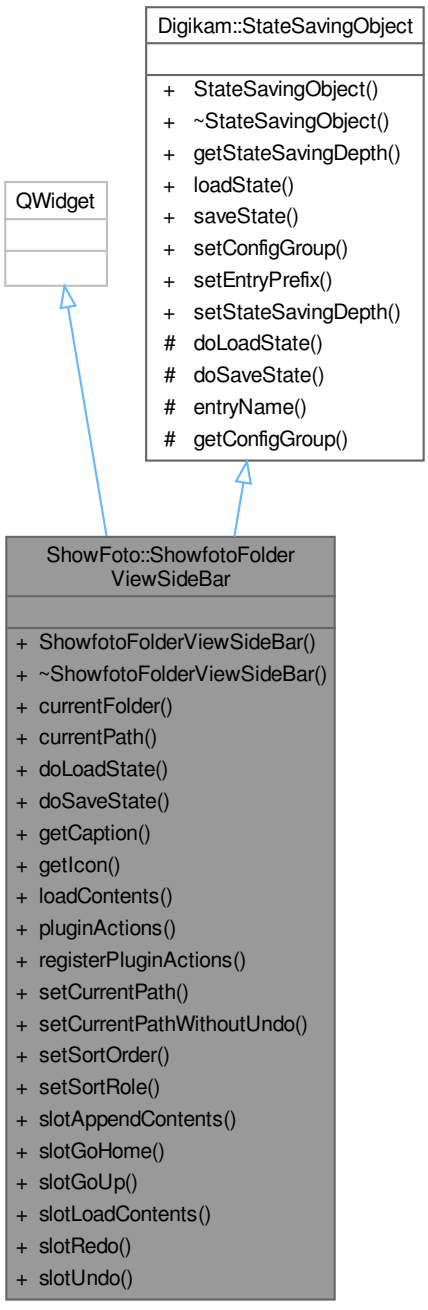
  List all file paths from the current model root index selected in the view.
- QVariant **data** (const QModelIndex &index, int role) const override

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)

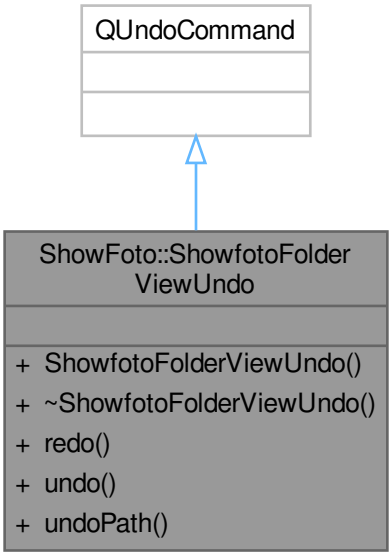
Additional Inherited Members

Protected Member Functions inherited from Digikam::DItemToolTip

- bool **event** (QEvent ∗) override
- void **paintEvent** (QPaintEvent ∗) override
- void **renderArrows** ()
- void **reposition** ()
- void **resizeEvent** (QResizeEvent ∗) override
- bool **toolTipIsEmpty** () 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::ShowfotoInfoIface Class Reference

Inheritance diagram for ShowFoto::ShowfotoInfoIface:

Public Member Functions

- **ShowfotoInfoIface** (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::DMetaInfoIface

- **DMetaInfoIface** (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 > **albumsItems** (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 prepocessor 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 **removeIndexs** (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 ShowfotoItemInfoId 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 **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
- 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.

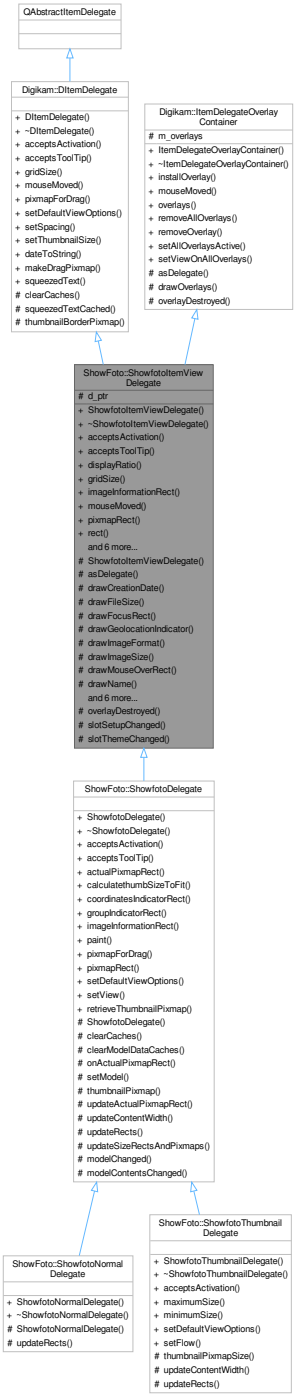
9.1456.2.4 lessThan() [2/2]

```
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()

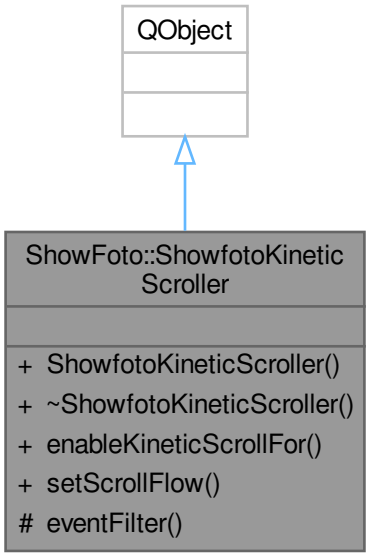
```
void ShowFoto::ShowfotoItemViewDelegate::setThumbnailSize (
            const ThumbnailSize & thumbSize )  [override], [virtual]
```

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()

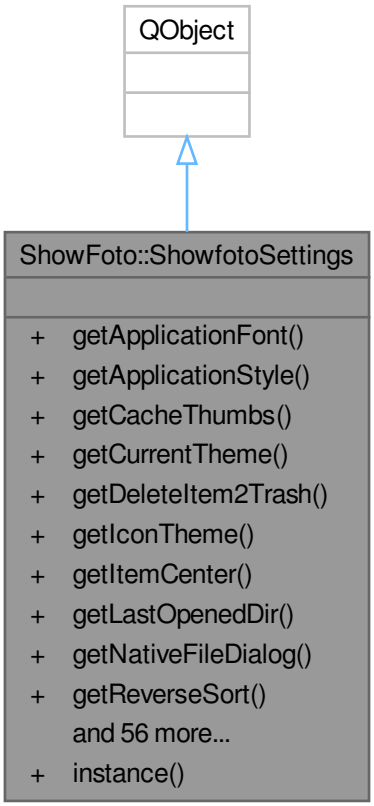
```
void ShowFoto::ShowfotoNormalDelegate::updateRects ( )  [override], [protected], [virtual]
```

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

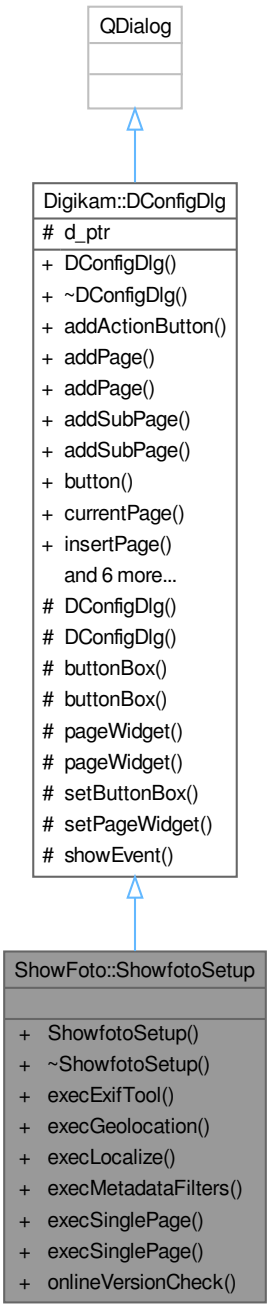
- static ShowfotoSettings * **instance** ()

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

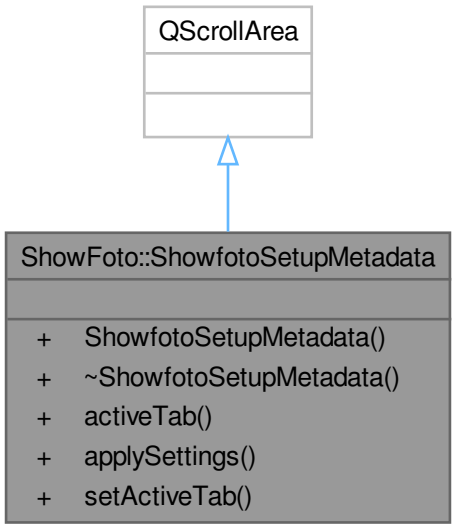
9.1461.1.1 execSinglePage()

```
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

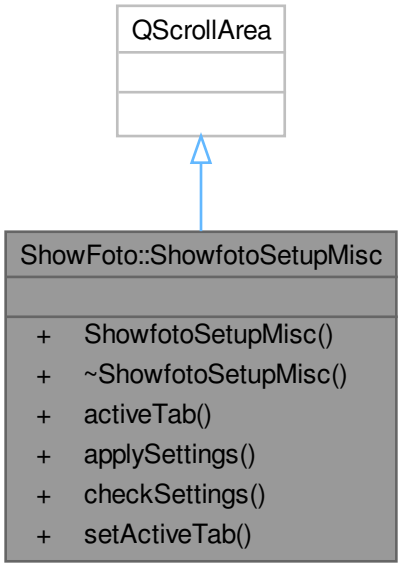
- enum **MetadataTab** {
  Behavior = 0 , **ExifViewer** , **MakernotesViewer** , **IptcViewer** ,
  XmpViewer , **ExifTool** }

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

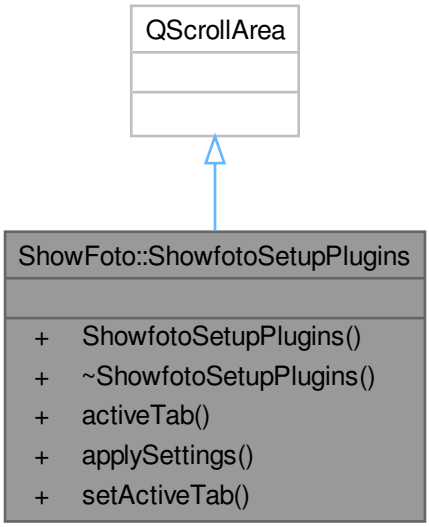
- enum **MiscTab** {
  Behaviour = 0 , **Appearance** , **SpellCheck** , **Localize** ,
  System }
- enum **SortOrder** { **SortByDate** = 0 , **SortByName** , **SortByFileSize** }

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

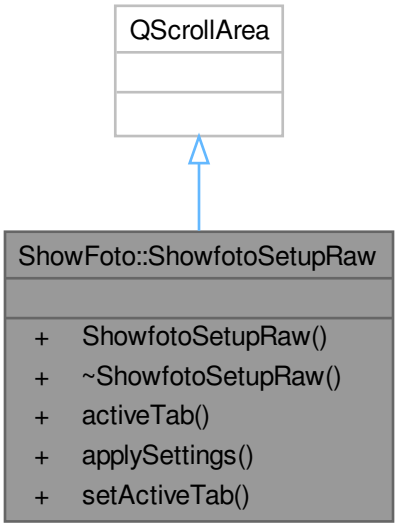
- enum **PluginTab** { **Generic** = 0 , **Editor** , **Loaders** }

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

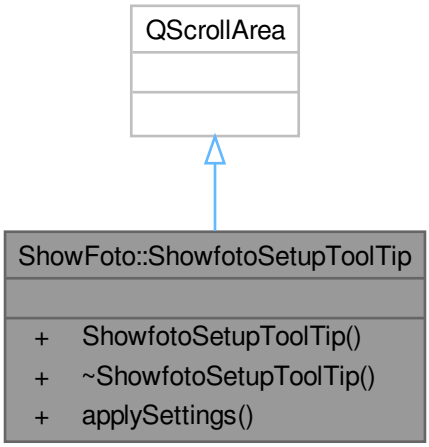
- enum **RAWTab** { **RAWBehavior** = 0 , **RAWDefaultSettings** }

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:

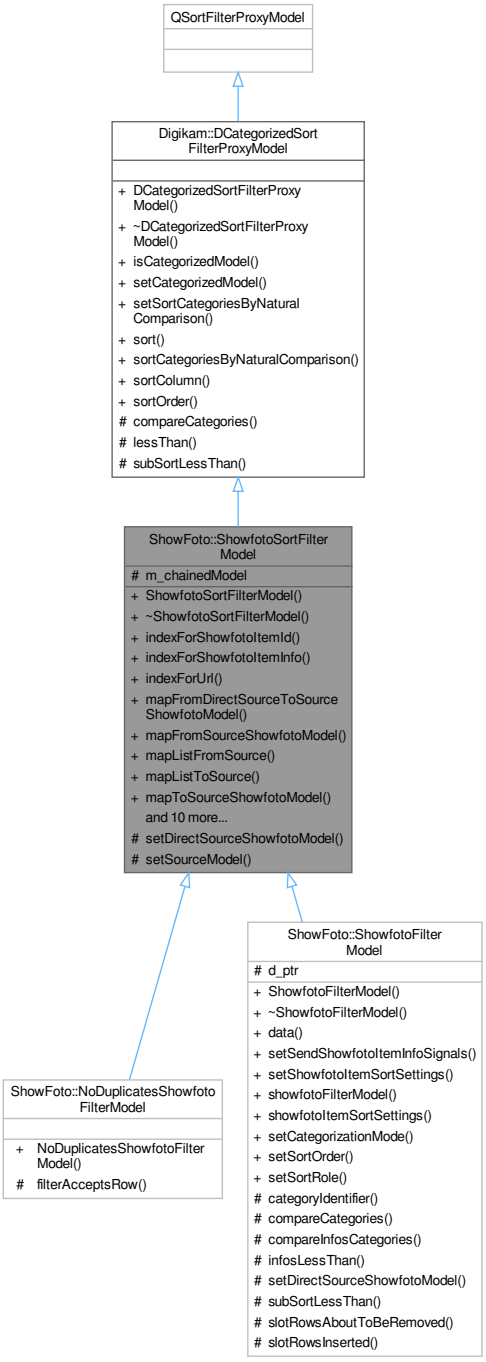


Public Member Functions

- **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** (qlonglong 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.

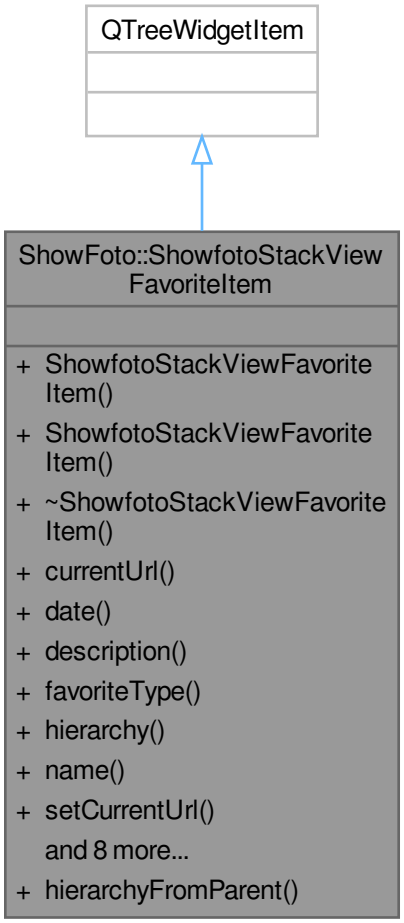
9.1467.1.4 showfotoItemInfosSorted()

```
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** (QTreeWidget *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

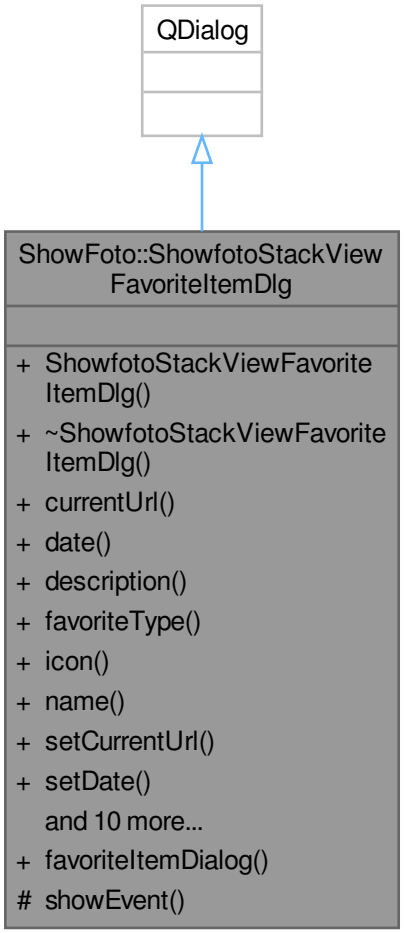
9.1468.2.1 hierarchyFromParent()

```
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

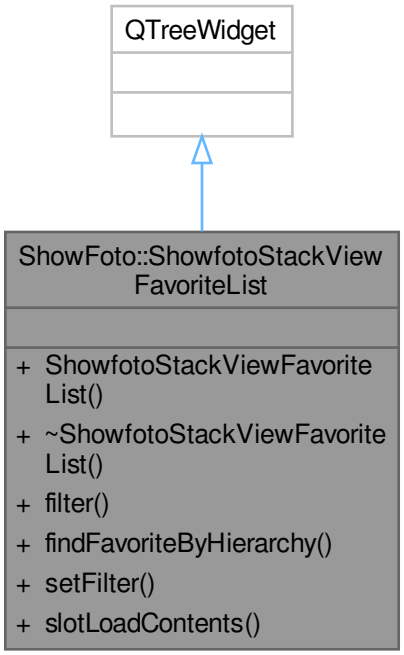
- static bool **favoriteItemDialog** (ShowfotoStackViewFavoriteList *const list, QString &name, int &favoriteType, QString &desc, QDate &date, QString &icon, QList< QUrl > &urls, QUrl &current, int iconSize, int sortOrder, int sortRole, ShowfotoStackViewFavoriteItem *const pitem, bool create=false)

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.

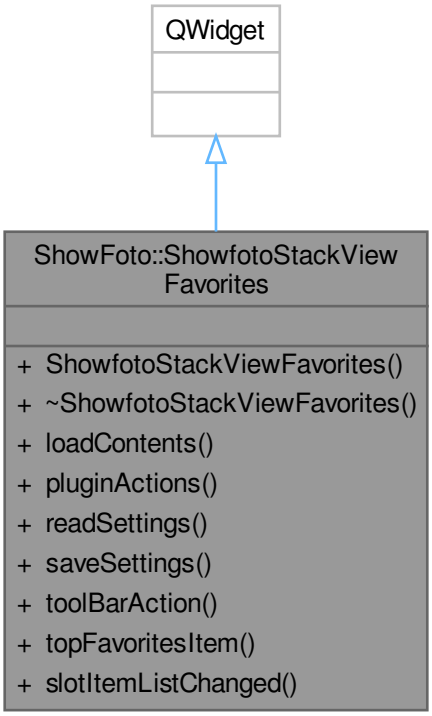
9.1470.1.2 signalSearchResult

```
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

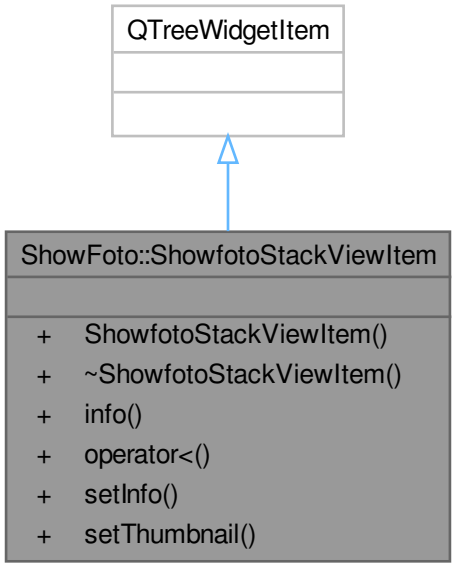
- void **signalLoadContents** ()
- void **signalLoadContentsFromFiles** (const QStringList &files, const QString &current)

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:

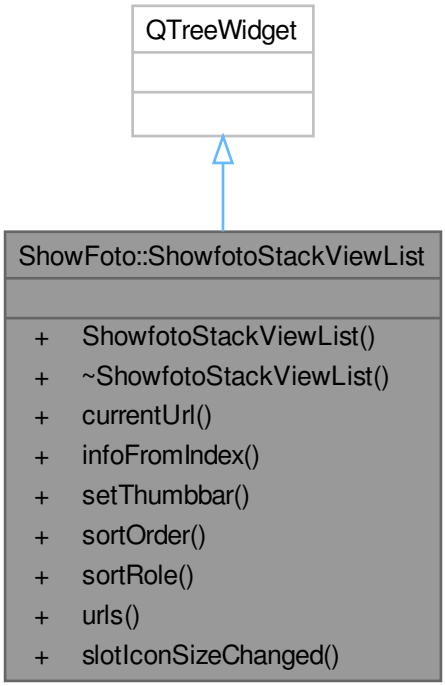


Public Member Functions

- **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 **signalItemListChanged** (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

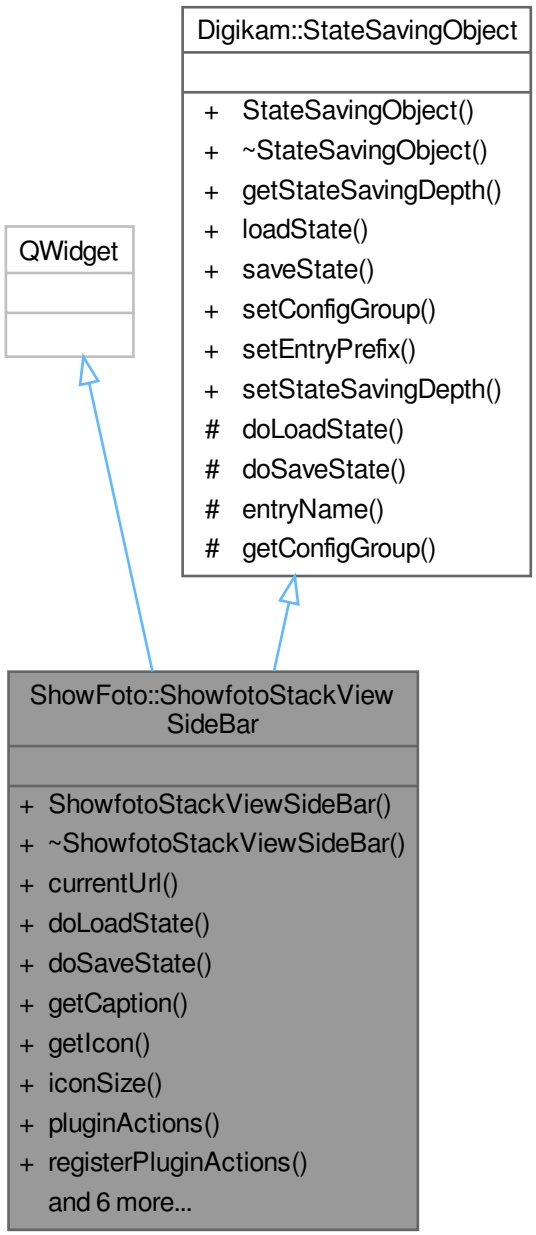
`enum ShowFoto::ShowfotoStackViewList::StackViewRole`

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()

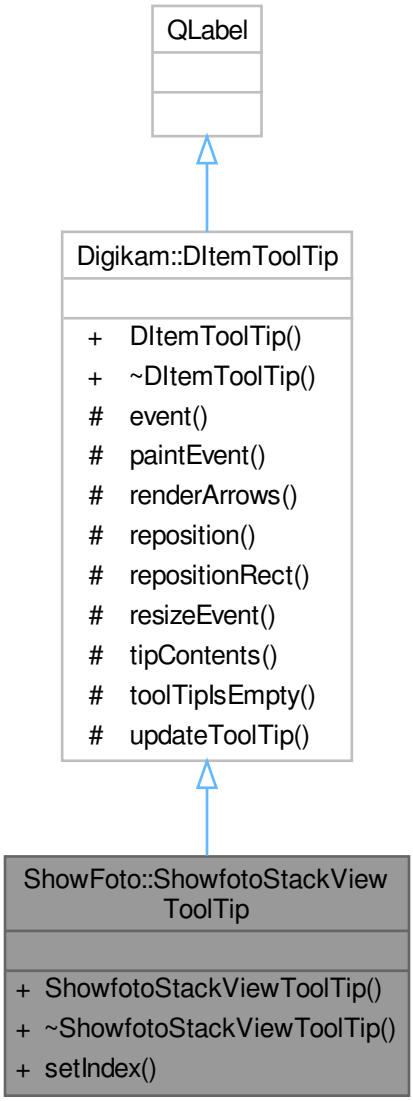
```
void ShowFoto::ShowfotoStackViewSideBar::doSaveState ( )  [override], [virtual]
```

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 **toolTipIsEmpty** () 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** (qlonglong 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 **decodeIsCutSelection** (const QMimeData ∗mimeData)
- void **encodeIsCutSelection** (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 **decodeIsCutSelection** (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 **encodeIsCutSelection** (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 **rowsInsertedArtifficial** (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 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**

9.1477.1 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 **pixmapForItem** (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 prepocessor 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 **removeIndexs** (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.

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 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.
- 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 setKeepsFilePatindexesForPathhCache 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