Window Maker
X Window Manager

User's Guide

Alfredo K. Kojima

Version 0.20.2

Chapter 1

Introduction

This manual describes the usage and configuration of the Window Maker window manager. It is intended for both users who never used the X Window System and for those who have experience with other window managers.

How to Read this guide

If you never have used a X window manager, you should read all this guide, as it contains detailed instructions for newbies.

Text in sans serif font, indicate instructions you must follow to accomplish a given task. If you're out of time (or patience), you should at least read text in these parts.

You can ignore the text in Extra Bindings boxes while you're getting familiar with Window Maker. Once you've got familiar with it, you can read the text in these boxes to learn more ways to accomplish tasks.

Some of the actions described here can be performed by keyboard shortcuts, but are only described in the section about configuration options.

1.1 What is a window manager?

If you come from the Windows®or MacOS™world, you might be confused about all these things like window managers, X windows etc.

In the UNIX™world, the task of providing a graphical user interface (GUI) is normally divided by 3 different components:

- the window server;
- the window manager and
- the application itself, normally by using a user interface toolkit.

The **window server** is standard and is usually the *X Window System* or some vendor provided compatible version of it. The X Window System, or X for short, is a window server. It's function is to provide a portable and high-level access to devices like keyboard, mouse and video display. It allows applications to show graphical information on the display through rectangular areas called windows.

Most user interface objects, like buttons, menus and scrollers are made of windows. The top level windows displayed by applications are named windows as well. These objects are not provided by the window server. These must be made by the application program or by the user interface toolkit.

For more information, read the manual page for `X(1)` and the documentation for Xlib.

The primary function of the **window manager** is to control the layout of top level windows on screen. Window Maker is a window manager. It provides a titlebar and a resizebar to change window layout, application menus to launch applications and execute special commands, application icons, miniwindows and an application dock. They will be explained in more detail in the following chapters.

The **user interface toolkit** is a library or collection of libraries that provide an API for application developers to program the interfaces for their applications. Toolkits generally provide controls like buttons, menus, radio-buttons etc to be used for program interaction. There is currently many of these toolkits available for X. Motif™, OpenLook™, and Athena are examples of toolkits.

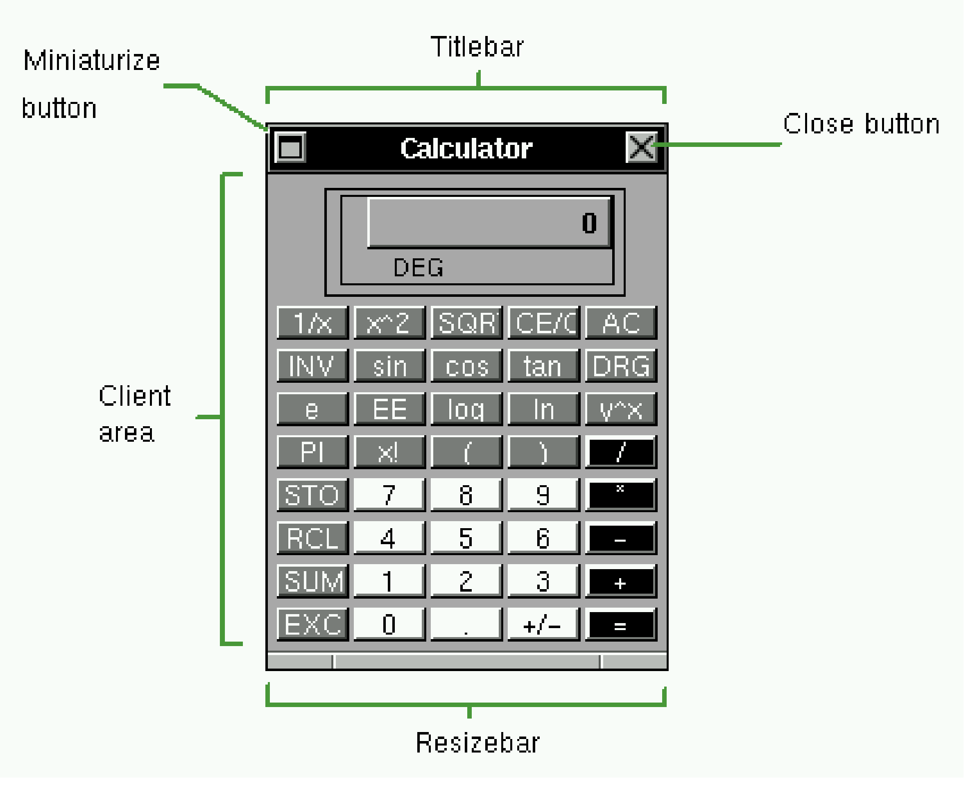
All other features normally found in other operating systems, like file managers, are implemented as separate programs and are not directly related to the window manager.

Chapter 2

Windows

2.1 Anatomy of a Window

Generally an application window will have the following layout:



Titlebar. The titlebar presents the name of the application, document or window. It's color indicates the keyboard focus state and type of the window. You can use it to move, activate, raise, lower and access the window commands menu.

Miniaturize button. You can click on the miniaturize button to miniaturize/iconify a window or click it with the `Control` key pressed to hide the application.

Close button. The close button can be used to close a window or kill the application by clicking it with the `Control` key pressed.

Resizebar. You use the resizebar to (surprise!) resize a window.

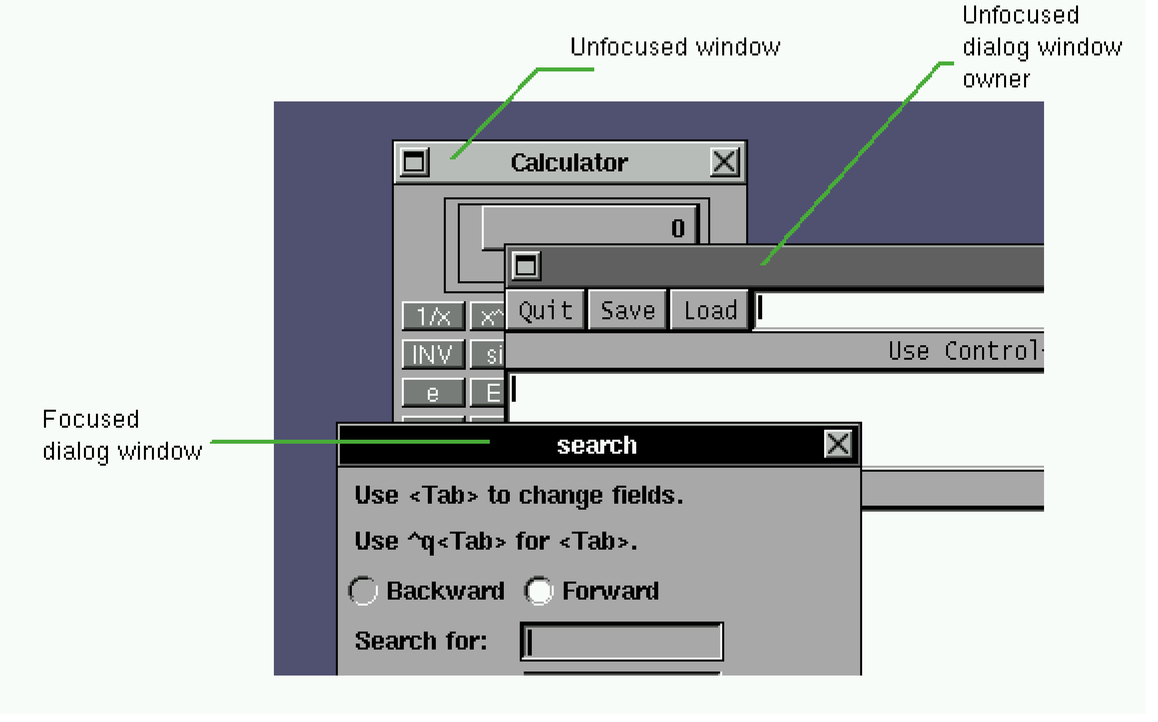
Client Area. The client area is where the application show it's information. If the window is inactive, you can click on it to activate it.

2.2 Working with Windows

2.2.1 Focusing a Window

Windows can be in two states: *focused*, or *unfocused*. The focused window (also called the key or active window) has a titlebar with a differentiated color and is the window that receives the keystrokes you type in the keyboard. Usually it's the window where you work on. Only one window may be focused at a time. Unfocused windows have a light gray titlebar.

Applications may have a special type of window, called dialog windows, transient windows or panels. When these windows are focused, the window that owns them (the main window) get a dark gray titlebar. As soon as the dialog window is closed, the focus is returned to the owner window. **??** shows an active Open File panel and it's owner window.



There are three styles of window focusing:

Click-to-Focus, or manual focus mode. In click-to-focus mode, you explicitly choose the window that should be focused. This is the default mode.

Focus-Follow-Mouse, or auto-focus mode. In this mode, the focused window is chosen based on the position of the mouse pointer. The window below the mouse pointer is always the focused window.

Sloppy-Focus mode. This is similar to the focus-follow-mouse mode, but if you move the pointer from a window to the root window, the window will not loose focus.

You can choose between these modes with the `FocusMode` option.

To focus a window in click-to-focus mode:

- Click on the titlebar, resizebar or in the client area of the window with the left or right mouse button.
  OR
- Click on the titlebar with the middle mouse button. This will focus the window without bringing it to the front.
  OR
- Open the window list menu and select the window to focus.

When you click in the client area of an inactive window to set the focus, the click is normally processed by the application. If you find this behaviour a little confusing, you can make the application ignore this click by using the `IgnoreFocusClick` option.

To focus a window in focus-follow-mouse mode:

- Move the pointer over the window you want to focus.

2.2.2 Reordering Overlapping Windows

Windows can overlap other windows, making some windows be over or in front of others.

To bring a window to the front:

- Click on the titlebar or resizebar of the desired window with the left mouse button.
  OR
- Select the desired window from the Window List menu.

Dialog/transient windows are always placed over their owner windows, unless the `OnTopTransients` option is disabled. Some windows have a special attribute that allow them be permanently over normal windows. You can make specific windows have this attribute use the `KeepOnTop` window option or set it in the Attributes Panel.

Extra Bindings

Action	Effect
Hold the `Meta` key and click on the window titlebar with the left mouse button	Sends the window to the back of all other windows.
Hold the `Meta` key and click on the Client Area of the window with the left mouse button	Brings the window to the front and focuses it.
Hold the `Meta` key and press the Up Arrow key	Brings the current focused window to the front.
Hold the `Meta` key and press the Down Arrow key	Sends the current focused window to the back.

2.2.3 Moving a Window

To move the window around the screen, drag the window through it's titlebar with the left mouse button pressed. This will also bring the window to the front and focus the window.

To move a window:

- Click on the titlebar of the window you want to move with the left mouse button and drag it with the button pressed.

While you move the window, a little box will appear in the screen, indicating the current window position in pixels, relative to the top left corner of the screen. You can change the location of this position box by hitting the `Shift` key during the move operation.

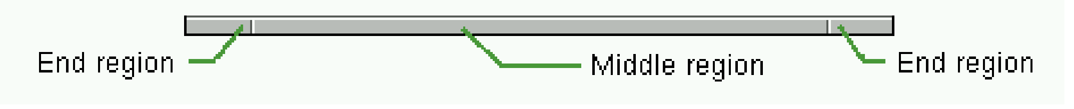
In some rare occasions, it is possible for a window to be placed off screen. This can happen with some buggy applications. To bring a window back to the visible screen area, select the window in the Window List menu. You can prevent windows from doing that with the dontmoveoff_at window attribute.

Extra Bindings

Action	Effect
Drag the titlebar with the middle mouse button	Move the window without changing it's stacking order.
Drag the titlebar while holding the `Control` key	Move the window without focusing it.
Drag the client area or resizebar while holding the `Meta` key	Move the window.

2.2.4 Resizing a Window

The size of a window can be adjusted by dragging the resizebar.



Depending on the place you click to drag the resizebar, the resize operation is constrained to a direction.

To resize a window:

- To change the window's height, click in the middle region of the resizebar and drag it vertically.
- To change the window's width, click in either end regions of the resizebar and drag it horizontally or drag it with the `Shift` key pressed.
- To change both height and width at the same time, click in either end regions of the resizebar and drag it diagonally.

While you resize the window, a little box will appear in the screen, indicating the current window size. You can change the location of this size box or change it into a different format by hitting the `Shift` key during the resize operation.

If a window gets too big to fit on the screen and you loose access to it's titlebar or resizebar, you can move the window through the client are, holding the `Meta` key and then resize it.

Extra Bindings

Action	**Effect**
Drag the window in the client area with the Right mouse button while holding the `Meta` key	Resizes the window.
Drag the resizebar with the middle mouse button	Resize the window without bringing it to the front
Drag the resizebar while holding the `Control` key	Resize the window without focusing it.

2.2.5 Miniaturizing a Window

If you want to temporarily get rid of a window, you can miniaturize it. When miniaturizing a window, it will shrink into a miniwindow with a icon and a title that is placed at the bottom of the screen.

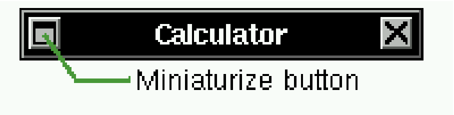


Figure 2.1: Miniwindow

You can move the miniwindow around the screen by dragging it. Unlike application icons, miniwindows cannot be docked.

To restore a window from it's miniwindow, double click the miniwindow. The window will be restored in the current workspace, with the same position, size and contents as it had before miniaturization.

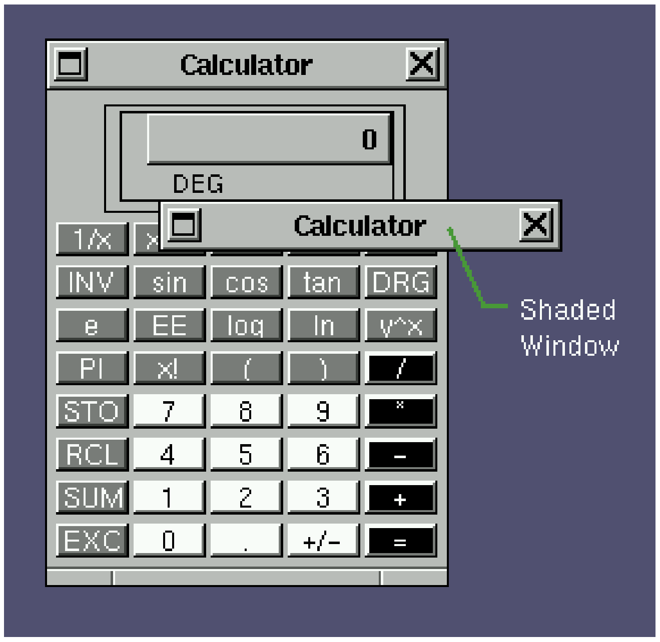
To miniaturize a window:

- Click on the miniaturize button.
  OR
- Hit the shortcut assigned to this action, which is `Meta+M` in the default configuration.

To restore a miniaturized window:

- Double click in the miniwindow.

You can also restore all miniaturized and hidden windows of an application at once by double clicking in it's application icon with the middle mouse button.



2.2.6 Shading a Window

If you want to temporarily get rid of a window, an option for it's miniaturization is to *shade* it. When you shade a window, the window rolls up to it's titlebar. You can do almost everything you do with a normal window with shaded windows, like miniaturizing or closing it.

To shade a window:

- Double Click on the titlebar of the window.

2.2.7 Closing a Window

After finishing work in a window, you can close it to completely get rid of it. When you close a window, it is removed from the screen and can no longer be restored. So, before closing a window, be sure you have saved any work you were doing on it.

Some windows will have a different close button. These windows can't be closed normally and the only ways to get rid of them is by exiting or killing the application. You should try exiting from inside the application (through it's menus or buttons) whenever possible. Otherwise you can force Window Maker to "kill" the application.

To force the closure of a window (by killing the application):

- Hold the `Control` key and click in the close button.
  OR
- Double click the close button.

It is also possible to kill applications that can be normally closed by clicking the close button while holding the `Control` key.

2.2.8 Maximizing a Window

If you want to resize a window to occupy the whole screen, you can maximize the window. When you unmaximize it, the window will be restored to the same position and size it was before maximized.

To maximize a window:

- Hold the `Control` key and double click on the window titlebar to resize the window's height to full screen.
- Hold the `Shift` key and double click on the window titlebar to resize the window's width to full screen.
- Hold both the `Control` and `Shift` keys and double click on the window titlebar to resize both window's height and width to full screen.

To restore the size of a maximized window:

- Hold the `Control` OR `Shift` key and double click on the window titlebar.

You can select whether the window should be maximized to the whole screen or if the position of the Dock and icons should be accounted for with the `NoWindowOverIcons` option and by making the dock float over normal windows (by enabling the "Keep Dock On Top" option in the dock menu).

2.2.9 The Window Commands Menu

Clicking on the titlebar of a window with the right mouse button will open a menu containing commands that will apply to that window. The menu can also be opened through the keyboard with the `Control+Escape` key, by default.

The available commands are:

(Un)Maximize will maximize the window both vertically and horizontally. If the window is already maximized, it will make the window have the size it had before maximization.

Miniaturize will miniaturize the window.

(Un)Shade will shade the window or unshade it if is already shaded.

Hide will hide all windows of the application.

Hide Others will hide all applications except for the current one.

Move To allows you to move the window to another workspace.

Attributes... opens the *Attributes Panel* , where you can edit the Window Maker specific attributes and options for that window.

Close will close the window.

Kill will kill the application. Use this option only if the application does not provide means to close it normally or in extreme cases.

Chapter 3

The Workspace

3.1 Working with Menus

Menus provide a list of commands that you can execute. To execute a command listed in a menu, click in the corresponding item. The item will blink telling that the command is going to be executed.

Grayed commands are disabled and cannot be executed at that moment. If you click on them nothing will happen.

Some menu entries have a little triangular indicator at the right. Selecting these entries will open a submenu, with a new list of commands.

You can use the keyboard to traverse and execute commands in some of the menus. First you must hit the key used to open the menu — like `F12` for the root menu — to enable keyboard traversal of it. Then you can use the Up and Down arrow keys to change the current selected item and the Left and Right arrow keys to jump between submenus and parent menus. To execute the current selected item press `Return`. To close the menu or stop menu traversal, press `Escape`. Additionally, pressing the first letter for an menu item, will jump the current selection to that item.

You can make frequently used menus "stick" to the workspace by dragging the titlebar of the menu. This will make a close button appear in the menu titlebar. If you want to close the menu, just click in that button.

Menus are normally placed on top of other windows and cannot be obscured by them. If you want the menus to be able to be obscured by lowering them, double click the menu titlebar while holding the `Meta` key. Repeat this to make the menus not obscurable again.

3.1.1 The Root Window Menu

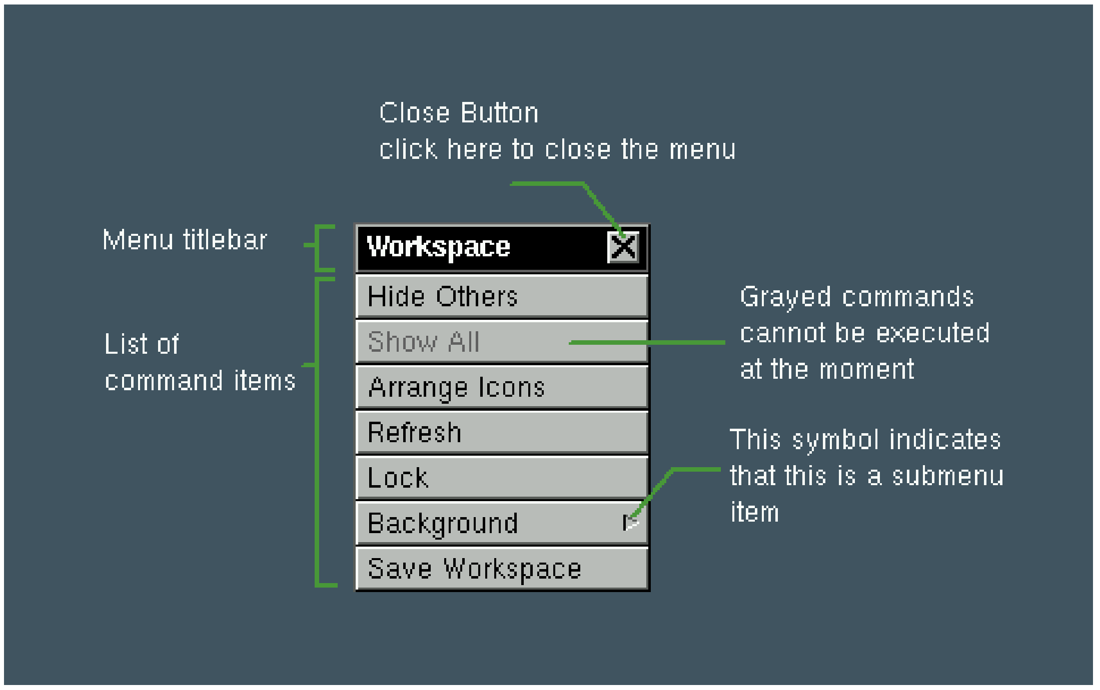
The *Root Window Menu* or *Applications Menu* has items that allow you to quickly launch applications and do some workspace management.

To open this menu, click on the workspace (root window) with the right mouse button or hit the key bound to it (`F12` by default).

The contents of the applications menu can be configured to have the applications installed in your system. To learn how to configure it, read the section on application menu configuration.

3.1.2 The Window List Menu

Clicking in the workspace with the middle mouse button will open a menu listing all windows that currently exist, with the workspace where the window is located at itś right. The current



focused window is marked by a diamond sign at the left of itś name. Clicking in an entry of that menu will focus the window, raise it and change the current workspace to the one where the window is located.

3.2 Working with Applications

In Window Maker, the instance of a running application is represented by an application icon. Do not confuse it with the icons (miniwindows in Window Maker) displayed by other window managers when a window is iconified. Application icons and miniwindows can be differentiated by the lack of a title in the former.

> Window Maker identifies a group of windows as belonging to a single instance of an application through some standard hints that the application sets in its windows. Unfortunately not all applications that exist set these hints, preventing some application specific features from working. These hints are `WM_CLASS`, `WM_COMMAND` and `WM_CLIENT_LEADER` or the group leader in `WM_HINTS`. You can see if an application supports these hints by running the xprop utility and selecting the desired window.
>
> If the application does not set these hints, it's impossible to treat it as an application in the point of view of Window Maker.

Note: The information about applications contained in this section only applies for versions of Window Maker built without the `--enable-single-icon` compile time option. This option is unsupported and behaviour when it's enabled will not be covered in this text.

3.2.1 Hiding an Application

If you want to close an application but intend to use it later you can *hide* it. When you hide an application all windows and miniwindows that belong to that application will be removed from the screen and are hidden into its application icon.

To hide an application:

- Click in the miniaturize button of any of the windows that belong to the application while holding the `Control` key.
  OR
- Press the keyboard shortcut assigned to it, which is `Meta+H` in the default configuration.
  OR
- Use the Hide command in the window commands menu brought when the window titlebar is clicked witht the right mouse button.
  OR
- Use the (Un)Hide command in the application icon commands menu brought when the application icon is clicked witht the right mouse button.

To unhide an application:

- Double click in the application icon with the left mouse button.
  OR
- Use the (Un)Hide command in the application icon commands menu brought when the application icon is clicked with the right mouse button.

When you unhide an application, all it's windows and miniwindows will be brought back and you will be taken to the last workspace you worked with that application.

Extra Bindings

Action	**Effect**
Double-click in the application icon while holding the `Meta` key	Unhide the clicked application and hide all other applications that are present in the current workspace.
Double-click in the application icon while holding the `Shift` key	Unhide the clicked application in the current workspace. If the application is already unhidden in another workspace, the application is brought to the current workspace.
Double-click in the application icon with the middle mouse button	Unhide the clicked application and deminiaturizes all it's windows.
Double-click in the window titlebar with the right mouse button while holding the `Meta` key	Hide all applications in the current workspace except for the clicked one.

There are two other commands in the applications menu related to application hiding:

Hide Others hide all applications in the current workspace except the currently active one;

Show All unhide all applications that were hidden in the current workspace.

3.2.2 The application icon menu

A menu with commands that will apply to the application can be brought by clicking the icon with the right mouse button.

The commands available in this menu are:

Unhide Here unhides the application in the current workspace;

(Un)Hide hides the application. If the application is already hidden, unhide it and take you to the workspace where the application is located;

Set Icon. . . open the icon image selection panel for the application icon.

Kill will kill the application.

3.2.3 The application dock

The application dock is a place where you can store frequently used applications for easy and fast access. It is located, by default, in the right side of the screen.

You can click in the top icon (the one with the GNUstep logo) and drag it downward to remove most of the dock from view. You can also drag it sidewards to switch the side of the dock from right to side or vice-versa.

A menu similar to the application icon menu is brought up when you click in a docked icon with the right menu button. It contains some extra commands that are specific for docked applications.

To make the dock be uncoverable by other windows, either double-click in the top dock icon while holding the `Meta` key or select the "Keep Dock on Top" item in the dock menu. In this mode, maximized windows will not be resized over the dock.

Starting a docked application

To start an application that is docked, double-click in its icon. The icon will be briefly highlighted and the application started.

While an application is not running, an ellipsis mark is present in the bottom left corner of the icon. When the application is started and is running the ellipsis will disappear, until the application is quit.

While the application is running, the docked icon will behave just like a normal undocked application icon, except for some extra actions specific for the dock.

To start a docked application:

- Double-click in the application icon with the left mouse button.
  OR
- Use the "Launch" command in the dock menu for the icon. If the application is already running it will start another instance.
  OR
- Hold the `Control` key while double-clicking in the icon to start another instance of the application.

If a new instance of an already running application is started, it will get a new application icon.

Customizing the dock

To add new applications to the dock, you can click in an application icon and drag it into the dock. When a ghost image of the icon appear, release the mouse button and the icon will be docked.

To reorder the docked applications, drag an icon to an empty slot and move the icons around as you want.

To remove a docked application, drag it from the dock and release the mouse button when the ghost image disappears. To remove the icon of an application that is running, hold the `Meta` key while dragging it.

Configuring the docked application

To change the settings of the docked application, select the "Settings… " item in the dock menu for that icon. A settings panel for that icon will appear.

In the *Application path and arguments* field, the path for the application and its arguments can be changed. Note that you can't change the application that is represented in the icon or change anything that would cause the application name to be changed. For example, if the icon is for `xterm`, you can't change the field's value to `ghostview`; or if the icon is for `xterm -name vi`, you can't change it to `xterm -name pine`. Also note that you cannot use shell commands, such as output redirectors (>, >> etc.).

3.3 Working with Workspaces

3.3.1 The Workspaces Menu

The *Workspaces Menu* allows you to create, switch, destroy and rename workspaces.

It has the following items:

New creates a new workspace and automatically switches to it

Destroy Last destroys the last workspace unless it is occupied by some window

Workspaces Each workspace has a corresponding item in the Workspaces menu. Clicking in one of these entries will switch from the current workspace to the selected workspace.

The current active workspace is indicated by a small indicator at the left of the workspace item.

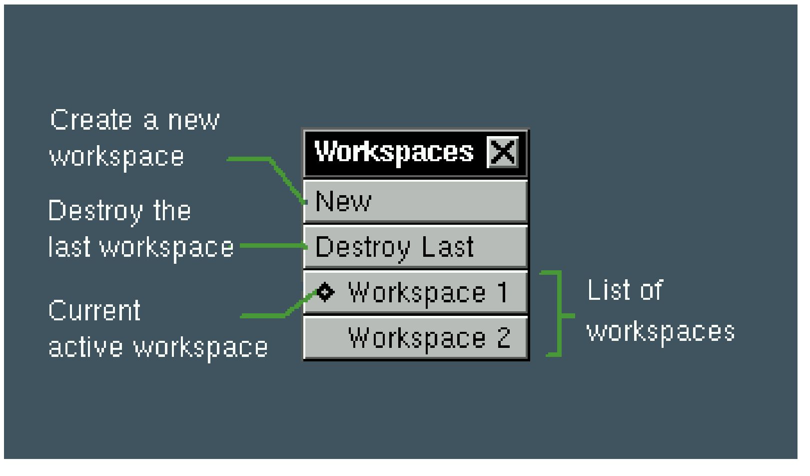


Figure 3.1: Workspaces Menu

To change the name of a workspace you must first "stick" the menu. Then `Control` click in the item corresponding to the workspace you want to rename. The item will turn into a editable text field where you can edit the workspace name. To finish editing the workspace name, press `Return`; to cancel it, press `Escape`.

There is a limit of 16 characters on the length of the workspace name.

Figure 3.2: The editable workspace menu

3.3.2 The workspace Clip

Chapter 4

Configuring Window Maker

Window Maker is fully configurable graphically, from WPrefs.app, attributes panel and other built-in configuration panels. In some cases you might want to change the configuration manually, from a text editor. This section describes the configuration files and options available.

4.1 The Defaults Database System

Window Maker uses a defaults database for storing various information, like configurations and other data that must be kept between sessions (like the list of applications of a saved session). The defaults database is stored as *property lists* in the `$(HOME)/GNUstep/Defaults` directory. Each file in the `$(HOME)/GNUstep/Defaults` directory contains data that belongs to a specific *domain*.

Any application can use the defaults database to store its configuration information. Generally an application will have one or more *domains* that belong to it.

4.1.1 Property List File Format

The syntax of a property list is simple, but if you need to change it manually, you must take care to not leave any syntax errors.

The EBNF for the property list is the following:

Description of the syntax of a property list in the Bacchus Naur Form (BNF)

```
<object>       ::=  <string> | <data> |<array> | <dictionary>
<string>       ::=  text with non-alphanumeric characters | alphanumeric text
<array>        ::=  '(' [ <object> { ',' <object> }* ] ')'
<dictionary>   ::=  '{' [ <keyval_pair> { ','<keyval_pair> }*] '}'
<keyval_pair>  ::=  <string> '=' <object> ';'
```

Example property list file

```
{
"*" = {
Icon = "defaultAppIcon.xpm";
};
"xterm.XTerm" = {
Icon = "xterm.xpm";
};
```

```
xconsole = {
Omnipresent = YES;
NoTitlebar = YES;
KeepOnTop = NO;
};
}
```

The property list above is a dictionary with 3 dictionaries inside. The first is keyed by "*", the second by "XTerm.xterm" and the last by "xconsole".

Note that all strings that have non-alphabetic or numeric characters (like a dot "." or the asterisk "*" are enclosed by double quotes. Strings with only alphanumeric characters may or may not be enclosed in double quotes, as they will not make any difference.
Here is another example:

```
{
FTitleBack = ( hgradient, gray, "#112233" );
}
```

The property list in the example above contains an array with 3 elements with a key named "FTitleBack".

Except for cases like file names and paths, all value strings are case insensitive, i.e.: YES = Yes = yes = yEs

4.1.2 Value Types

Here is a description of some the types of values that an option might have.

Type	**Values:**
boolean	YES or NO
integer	any integer number, usually limited by a range that will be indicated
positive integer	any integer number greater than or equal to zero (0)
speed	one of UltraFast, Fast, Medium, Slow or VerySlow
mouse button	one of Left, Middle, Right, Button1, Button2, Button3, Button4 or Button5

4.1.3 Preferences

General preference options are stored in the *Window Maker* domain, ie: in the `$(HOME)/GNUstep/Defaults/WindowMaker` file.

Changes in preference options will automatically affect the current Window Maker session, even without a restart. Some options, however, require a restart of Window Maker so that they apply. Such options are marked by a * .

Note that values marked as *Default* are values that are assumed if the option is not specified, instead of *factory default* values that are set in the preference file.

General Configuration

Option `PixmapPath`

Value list of directories separated by "`:`" (default: depends on the system)

Description A list of directories where pixmaps can be found. The pixmaps for things like icons, are searched in these paths in order of appearance.

Option `*NoDithering`

Value boolean (default: NO)

Description Disable internal dithering of images. Not recommended for displays with less than 8 bits per pixel.

Option `*ColormapSize`

Value integer number > 1 (default: 4)

Description Number of colors for each of the red, green and blue components to be used for the dithering colormap. This value must be greater than 1 and smaller than 6 for 8bpp displays. It only makes sense on PseudoColor displays. This option has not effect on TrueColor displays. Larger values result in better appearance, but leaves less colors for other applications.

Option `*ModifierKey`

Value modifier key name (default: Mod1)

Description The key to use as the modifier being referred as `Meta` in this manual, like `Meta` dragging a window to move it. Valid values are `Alt`, `Meta`, `Super`, `Hyper`, `Mod1`, `Mod2`, `Mod3`, `Mod4`, `Mod5`.

Option `UseSaveUnders`

Value boolean (default: NO)

Description Use *saveunder* in Window Maker windows. This can improve performance but increases memory usage. It also can cause problems with window refreshing in some buggy X servers.

Option `DisableClip`

Value boolean (default: NO)

Description Will remove the application Clip from the workspace.

Option `DisableDock`

Value boolean (default: NO)

Description Will remove the application Dock from the workspace.

Option `Superfluous`

Value boolean (default: NO)

Description Enable extra animations and other cosmetic things that might increase peak memory and CPU usage.

Option `SaveSessionOnExit`
Value bolean (default: NO)
Description Automatically save the state of the session when exiting Window Maker.

Option `*IconSize`
Value integer number > 4 (default: 64)
Description The size of application icons and miniwindows.

Option `OpaqueMove`
Value boolean (default: NO)
Description Whether the whole window should be moved while dragging it or only a frame representing the window should be moved. [1]

Option `*FocusMode`
Value Manual or ClickToFocus, Auto or FocusFollowsMouse, SemiAuto or Sloppy (default: Manual)
Description The mode of input focus setting. Refer to the section about focus selection (**??**) for an explanation of the focus modes.

Option `IgnoreFocusClick`
Value boolean (default: NO)
Description Whether the mouse click used to focus a window should be ignored or treated normally.

Option `AutoFocus`
Value boolean (default: NO)
Description Whether newly created windows should automatically receive input focus. Do not confuse with FocusMode=Auto.

Option `RaiseDelay`
Value integer number (default: 0)
Description How many tenths of a second to wait before raising a window in Auto or SemiAuto focus mode. 0 disables this feature.

Option `DoubleClickTime`
Value integer number (default: 250)
Description If two mouse clicks occur in this interval of time, it will be considered a double click.

Option `ColormapMode`
Value Manual or ClickToFocus, Auto or FocusFollowsMouse (default: Auto)
Description The mode of colormap setting. In *Manual* or *ClickToFocus* mode, the colormap is set to the one belonging to the current focused window. In *Auto* or *FocusFollowsMouse* mode, the colormap is set to the colormap of the window below the pointer.

[1] When OpaqueMove is turned off, Window Maker will grab the server, so that applications won't refresh their contents. This is necessary, otherwise the frame drawn over windows will not be erased correctly.

Option `CirculateRaise`

Value boolean (default: NO)

Description Whether the window should be raised when circulating (focus the next or previous window through the keyboard).

Option `OnTopTransients`

Value boolean (default: NO)

Description Whether transient windows should always be placed over their owner windows.

Option `WindowPlacement`

Value auto, cascade, manual or random (default: cascade)

Description Selects placement mode for new windows. *auto* places the window automatically in the first open space found in the workspace. *cascade* places the window in incrementing positions starting from the top-left corner of the workspace. *manual* allows you to place the window interactively with the mouse. *random* places the window randomly on the workspace.

Option `WindowPlaceOrigin`

Value (x, y) where x and y are integer numbers. (default: (0,0))

Description Sets the offset from the top-left corner of the screen to place windows. In non-manual Window-Placement modes, windows will not be placed above or to the left of this point.

Option `AutoArrangeIcons`

Value boolean (default: NO)

Description Whether icons should be automatically arranged.

Option `ResizeDisplay`

Value center, corner, floating or line (default: corner)

Description Selects the type or position of the box that shows the window size when a window is being resized. *center* places the box in the center of the workspace, *corner* places it in the top-left corner of the workspace, *floating* places it in the center of the window being resized and *line* draws the current window size over the workspace, like in a technical drawing.

Option `MoveDisplay`

Value center, center or floating (default: corner)

Description Selects the type or position of the box that shows the window position when a window is being moved. The value meanings are the same as for the ResizeDisplay option.

Option `AlignSubmenus`

Value boolean (default: NO)

Description Whether submenus should be aligned vertically with their parent menus.

Option `WrapMenus`

Value boolean (default: NO)

Description Whether submenus should be placed at the right of itś parent menu when they dont́ fit on screen. Note that menus that are placed outside the screen can be scrolled.

Option `ScrollableMenus`

Value boolean (default: NO)

Description Whether menus that are not fully inside the screen should automatically scroll when the pointer is over them and near the border of the screen.

Option `MenuScrollSpeed`

Value speed (default: Medium)

Description The scrolling speed of menus.

Option `DontLinkWorkspaces`

Value boolean (default: NO)

Description Do not automatically switch to the next or previous workspace when a window is dragged to the edge of the screen.

Option `NoWindowOverIcons`

Value boolean (default: NO)

Description When maximizing windows, limit their sizes so that they will not cover miniwindows and application icons.

Option `StickyIcons`

Value boolean (default: NO)

Description Whether miniwindows should be present in all workspaces.

Option `CycleWorkspaces`

Value boolean (default: NO)

Description Set to YES if you want windows that are dragged past the last workspace should be moved to the first workspace, or vice-versa.

Option `AdvanceToNewWorkspace`

Value boolean (default: NO)

Description Whether windows dragged past the last workspace should create a new workspace.

Option `DisableAnimations`

Value boolean (default: NO)

Description Whether animations, like the one done during miniaturization, should be disabled.

Option `IconSlideSpeed`

Value speed (default: Medium)

Description The speed of icons when they are being slided across the workspace.

Option `ShadeSpeed`

Value speed (default: Medium)

Description The speed of the shading animation.

Option `DisableSound`

Value boolean (default: NO)

Description Whether sound support in Window Maker should be disabled. Note that to have sound support in Window Maker, you must have a sound module for Window Maker compiled for your system and a version of Window Maker compiled with the –enable-sound configure flag. The sound module is currently only available for Linux and FreeBSD systems. Read the README file for Window Maker or the Window Maker Home Page to find where to download the module.

Option `*DisableWSMouseActions`

Value boolean (default: NO)

Description Whether actions triggered by mouse clicks in the workspace should be disabled. This allows the use of file managers and desktop managers that place icons in the root window (such as KDE).

Option `SelectWindowsMouseButton`

Value mouse button (default: Left)

Description The mouse button that activates the "rubberband" selection of multiple windows in the workspace.

Option `WindowListMouseButton`

Value mouse button (default: Middle)

Description The mouse button that opens the window list menu in the workspace.

Option `ApplicationMenuMouseButton`

Value mouse button (default: Right)

Description The mouse button that opens the applications menu in the workspace.

Option `IconPosition`

Value tlh, tlv, trh, trv, blh, blv, brh, brv (default: blh)

Description The position on the screen where miniwindows and application icons should be placed. The first character can be *t* or *b* , meaning top or bottom. The second character can be *l* or *r* , meaning left or right. The third character can be *v* or *h* , meaning vertical or horizontal. So, *blh* will make the icons be placed horizontally starrting from the bottom left corner of the screen.

Option `AppIconBalloons`

Value boolean (default: NO)

Description Show a balloon with the class name (normally the name of the application, unless that application is coded in an uncommon way) of the application represented when the mouse pointer is placed over an application icon. If the application icon is docked, the command line for that application is also displayed.

Option `MiniwindowTitleBalloons`

Value boolean (default: NO)

Description Show a balloon with the title of the miniwindow when the mouse pointer is placed over a miniwindow.

Option `WindowTitleBalloons`

Value boolean (default: NO)

Description Show a balloon with the full title of the window the mouse pointer is placed over the window's titlebar and the title doesn't fully fit in the titlebar.

Option `EdgeResistance`

Value integer (default: 0)

Description How many pixels to move a window against the edge of the screen to actually put it outside of the screen.

Option `IconificationStyle`

Value zoom, twist, flip or none (default: zoom)

Description The style of the animation for miniaturization of miniwindows.

Appearance Options

Fonts are specified in the X Logical Font Description format (aka Xtra Long Font Description). You can cut and paste these names from programs like `xfontsel`.

Colors are specified as color names in the standard X format. This can be any color name shown by the `showrgb` program (like black, white or gray) or a color value in the #rrggbb format, where rr, gg and bb is the intensity of the color component (like #ff0000 for pure red or #000080 for medium blue). Note that color names in the #rrggbb format must be enclosed with double quotes. There are other formats to specify colors. Look at the `XParseColor(3)` manual page (type —man XParseColor—) for more information.

Textures are specified as an array, where the first element specifies the texture type followed by a variable number of arguments.

Valid texture types are:

(solid, color) the texture is a simple solid color.

(dgradient, color1, color2) the texture is a diagonal gradient rendered from the top-left corner to the bottom-right corner. The first argument (color1) is the color for the top-left corner and the second (color2) is for the bottom-right corner.

(hgradient, color1, color2) the texture is a horizontal gradient rendered from the left edge to the right edge. The first argument (color1) is the color for the left edge and the second (color2) is for the right edge.

(vgradient, color1, color2) the texture is a vertical gradient rendered from the top edge to the bottom edge. The first argument (color1) is the color for the top edge and the second (color2) is for the bottom edge.

(mdgradient, color1, color2, . . . , color*n*) this is equivalent to dgradient, but you can specify more than two colors.

(mhgradient, color1, color2, . . . , color*n*) this is equivalent to hgradient, but you can specify more than two colors.

(mvgradient, color1, color2, . . . , color*n*) this is equivalent to vgradient, but you can specify more than two colors.

(tpixmap, pixmapFile, color) the texture will be the pixmap specified by pixmapFile and it will be tiled all over the area. color is the color Window Maker will use when it needs to know the color of the titlebar. You can specify the predominant color of the pixmap.

(spixmap, pixmapFile, color) the texture will be the pixmap specified by pixmapFile and it will be scaled to fit the area. color has the same meaning as in tpixmap.

(cpixmap, pixmapFile, color) the texture will be the pixmap specified by pixmapFile and it will be centered to fit the area. color has the same meaning as in tpixmap and will also be used to fill the back of the pixmap. Note that this has the same memory usage as spixmap.

Examples

(solid, gray)

(dgradient, gray80, gray20)

(hgradient, gray80, gray20)

(vgradient, gray80, gray20)

Option `*NewStyle`
Value boolean (default: NO)
Description Selects between N*XTSTEP style buttons in the titlebar and a newer style of buttons.

Option `WidgetColor`
Value (solid, color) where color is a color name (default: (solid, gray))
Description Chooses the color to be used in titlebar buttons if NewStyle=NO.

Option `WorkspaceBack`
Value a texture or None (default: None)
Description Default texture for the workspace background. Note: The *dgradient* and *mdgradient* textures can take a lot of time to be rendered.

Option `IconBack`
Value texture (default: (solid, gray))
Description Texture for the background of application icons and miniwindows.

Option `FTitleBack`
Value texture (default: (solid, black))
Description Texture for the focused window titlebar.

Option `PTitleBack`
Value texture (default: (solid, "#616161"))
Description Texture for the owner of the currently focused transient window.

Option `UTitleBack`
Value texture (default: (solid, gray))
Description Texture for untitled window titlebars.

Option `MenuTitleBack`
Value texture (default: (solid, black))
Description Texture for menu titlebars.

Option `MenuTextBack`
Value texture (default: (solid, gray))
Description Texture for menu items.

Option `FTitleColor`
Value color (default: white)
Description The color of the text in the focused window titlebar.

Option `PTitleColor`
Value color (default: white)
Description Color for the text in the unfocused main window titlebar or the owner of the currently focused transient window.

Option `UTitleColor`
Value color (default: black)
Description Color for the text in unfocused window titlebars.

Option `MenuTitleColor`
Value color (default: white)
Description Color for the text in menu titlebars.

Option `MenuTextColor`
Value color (default: black)
Description Color for the text in menu items.

Option `HighlightColor`
Value color (default: white)
Description Color for the highlighted item in menus.

Option `HighlightTextColor`
Value color (default: black)
Description Color for the highlighted item text in menus.

Option `MenuDisabledColor`
Value color (default: "#616161")
Description Color for the text of disabled menu items.

Option `ClipTitleColor`
Value color (default: black)
Description Color for the text in the clip.

Option `CClipTitleColor`
Value color (default: #454045)
Description Color for the text in the collapsed clip.

Option `WindowTitleFont`
Value font (default: helvetica bold 12)
Description Font for the text in window titlebars.

Option `MenuTitleFont`
Value font (default: helvetica bold 12)
Description Font for the text in menu titlebars.

Option `MenuTextFont`
Value font (default: helvetica medium 12)
Description Font for the text in menu items.

Option `IconTitleFont`
Value font (default: helvetica medium 8)
Description Font for the text in miniwindow titlebars.

Option `ClipTitleFont`
Value font (default: helvetica bold 10)
Description Font for the text in the Clip.

Option `DisplayFont`
Value font (default: helvetica medium 12)
Description Font for the text in information windows, like the size of windows during resize.

Option `TitleJustify`
Value center, left or right (default: center)
Description Justification of the text in window titlebars.

Keyboard Bindings

Keyboard shortcut specifications are in the form:

[<modifier key names> +] <key name>

Where *modifier key names* specify an optional modifier key, like `Meta` or `Shift`. Any number of modifier keys might be specified. The *key name* is the actual key that will trigger the action bound to the option.

Examples:

F10 means the `F10` key.

Meta+Tab means the `Tab` key with the `Meta` modifier key pressed at the same time.

Meta+Shift+Tab means the `Tab` key with the `Meta` and `Shift` modifier keys pressed at the same time.

Key names can be found at `/usr/X11R6/include/X11/keysymdef.h` The `XK_` prefixes must be ignored (if key name is `XK_Return` use `Return`).

Option `RootMenuKey`
Default Value None
Action Opens the root menu in the current position of the mouse pointer.

Option `WindowListKey`
Default Value None

Action Opens the window list menu in the current position of the mouse pointer.

Option `WindowMenuKey`
Default Value None
Action Opens the window commands menu of the current focused window.

Option `MiniaturizeKey`
Default Value None
Action Miniaturizes the current focused window.

Option `HideKey`
Default Value None
Action Hides the current active application.

Option `CloseKey`
Default Value None
Action Closes the current focused window.

Option `MaximizeKey`
Default Value None
Action Maximizes the current focused window.

Option `VMaximizeKey`
Default Value None
Action Maximizes the current focused window vertically.

Option `RaiseKey`
Default Value Meta+Up
Action Raises the current focused window.

Option `LowerKey`
Default Value Meta+Down
Action Lowers the current focused window.

Option `RaiseLowerKey`
Default Value None
Action Raises the window under the pointer or lower it if it is already raised.

Option `ShadeKey`
Default Value None
Action Shades the current focused window.

Option `SelectKey`
Default Value None

Action Selects the current focused window.

Option `FocusNextKey`
Default Value None
Action Switch focus to the next window.

Option `FocusPrevKey`
Default Value None
Action Switch focus to the previous window.

Option `NextWorkspaceKey`
Default Value None
Action Switches to the next workspace.

Option `PrevWorkspaceKey`
Default Value None
Action Switches to the previous workspace.

Option `NextWorkspaceLayerKey`
Default Value None
Action Switches to the next group of 10 workspaces.

Option `PrevWorkspaceLayerKey`
Default Value None
Action Switches to the previous group of 10 workspaces.

Option `Workspace1Key`
Default Value None
Action Switches to workspace 1

Option `Workspace2Key`
Default Value None
Action Switches to workspace 2, creating it if it does not exist.

Option `Workspace3Key`
Default Value None
Action Switches to workspace 3, creating it if it does not exist.

Option `Workspace4Key`
Default Value None
Action Switches to workspace 4, creating it if it does not exist.

Option `Workspace5Key`
Default Value None

Action Switches to workspace 5, creating it if it does not exist.

Option `Workspace6Key`
Default Value None
Action Switches to workspace 6, creating it if it does not exist.

Option `Workspace7Key`
Default Value None
Action Switches to workspace 7, creating it if it does not exist.

Option `Workspace8Key`
Default Value None
Action Switches to workspace 8, creating it if it does not exist.

Option `Workspace9Key`
Default Value None
Action Switches to workspace 9, creating it if it does not exist.

Option `Workspace10Key`
Default Value None
Action Switches to workspace 10, creating it if it does not exist.

Option `ClipLowerKey`
Default Value None
Action Lowers the clip.

Option `ClipRaiseKey`
Default Value None
Action Raises the clip.

Option `ClipRaiseLowerKey`
Default Value None
Action Raises the clip or lowers it if it is already raised.

4.1.4 Window Attributes

Window attributes are stored in the `$(HOME)/GNUstep/Defaults/WMWindowAttributes` file.

Syntax

The contents of this file is a dictionary of attribute dictionaries keyed by window names. Like this:

```
{
"*" = {
Icon = "defaultAppIcon.xpm";
};
"xterm.XTerm" = {
```

```
Icon = "xterm.xpm";
};
xconsole = {
Omnipresent = YES;
NoTitlebar = YES;
KeepOnTop = NO;
};
}
```

Window names are in the form: [2]

<window instance name> . <window class name>
OR
<window instance name>
OR
<window class name>

Placing an asterisk as the window name means that the values set for that key are to be used as default values for all windows. So, since xconsole does not specify an `Icon` attribute, it will use the default value, which in the above example is `defaultAppIcon.xpm`.

Options

The default value is NO for all options.

Attribute `Icon`
Value pixmap file name
Description Assigns a pixmap image to be used as the icon for that window.

Attribute `NoTitlebar`
Value boolean
Description Do not put a titlebar in the window.

Attribute `NoResizebar`
Value boolean
Description Do not put a resizebar in the window.

Attribute `NoMiniaturizeButton`
Value boolean
Description Remove the miniaturize button from the titlebar.

Attribute `NoCloseButton`
Value boolean
Description Remove the close button from the titlebar.

Attribute `NoHideOthers`
Value boolean

[2]You can get the values for these information by running the `xprop` utility on the desired window. When you do that, it will show the following line, among other things:
`WM_CLASS(STRING) = "xterm", "XTerm"`
The first string (`xterm`) is the window instance name and the second (`XTerm`) the window class name.

Description Do not hide the window or the application to which the window belongs when a —Hide Others— command is issued.

Attribute `NoMouseBindings`
Value boolean
Description Do not grab mouse buttons in that window. This means that actions like `Meta` Click on the window will be caught by the application instead of being intercepted by Window Maker.

Attribute `NoKeyBindings`
Value boolean
Description Do not grab keys in that window. This means that keystrokes that would be normally intercepted by Window Maker (because they are bound to some action), like `Meta` + `Up`, will be passed to the application.

Attribute `NoAppIcon`
Value boolean
Description Do not create an application icon for the window. This is usefull for some applications that incorrectly get more than one application icon.

Attribute `KeepOnTop`
Value boolean
Description Always keep the window over other normal windows.

Attribute `Omnipresent`
Value boolean
Description Make the window be present in all workspaces. AKA sticky window.

Attribute `SkipWindowList`
Value boolean
Description Do not list the window in the window list menu.

Attribute `KeepInsideScreen`
Value boolean
Description Always keep window inside the visible area of the screen.

Attribute `Unfocusable`
Value boolean
Description Do not let the window be focused.

Attribute `StartWorkspace`
Value workspace number or name
Description Make the window always be initially shown in the indicated workspace.

Attribute `EmulateAppIcon`
Value boolean
Description Make the window emulate an application that provides enough information for Window Maker to create an application icon for it. If you want to use this attribute in a application that opens other windows, set it only in the main window of that application (ie: the first window of the application that is shown and the one that quits the whole application when closed).

4.1.5 Applications Menu

The applications menu (aka: root menu) can be defined in two distinct ways:

- in the form of an array in property list format, in `$(HOME)/GNUstep/Defaults/WMRootMenu`
- in the form of a text file, which location is present in `$(HOME)/GNUstep/Defaults/WMRootMenu`

In any of these ways, the contents of the menu can be defined statically in the configuration file or dynamically, by using the contents of a directory or by the output of some script.

Tips

- If the size of a window is so large that it doesn't fit on the screen and you can't manipulate it, you can simply hold the `Meta` key while dragging the window in the client area. This way you can move the window up or down and resize it, if you want.
- If you want windows to be able to cover the dock, you can make the dock lowerable by double clicking the first dock icon while holding the `Meta` key. Then, you can raise and lower the dock through the first icon, just like you do with windows.
- If you want windows to be able to cover menus, you can make them lowerable just like the dock by double clicking the titlebar of the menu with the `Meta` key pressed.
- You can make a "taskbar" by opening the window list menu, attaching it to the workspace and moving it to the bottom of the screen, leaving only it's title visible. Then, when you move the mouse over it, the menu will become visible.
- If you removed the titlebar of a window, you can still access the attributes panel for it by focusing the window and hitting the shortcut for opening the window commands menu (default is `Control+Escape`).
- If you need to execute a command every time Window Maker is started, you can place the command in the `$(HOME)/GNUstep/Library/WindowMaker/autostart` script. Similarly, the `$(HOME)/GNUstep/Library/WindowMaker/exitscript` script is always executed when Window Maker is exited.

Glossary

drag to click in an object with the mouse and move the mouse while holding the mouse button, wich cause that object to be "dragged" by the mouse.

miniaturize (iconify, minimize) to temporarily put a window aside, replacing the window with a miniature representation of it.

Meta key depending on the system, keyboard type and configuration, this can mean different keys. On PCs, it is usually the `Alt` or `Alternate` key.