```
# Pop up the window list in tranient mode on button 3 press & hold
Mouse 3 R A Module "FvwmWinList" FvwmWinList Transient
########################## Window-Lister ###############################
*FvwmWinListBack DarkOliveGreen
*FvwmWinListFore PaleGoldenRod
*FvwmWinListFont -*-new century schoolbook-bold-r-*-*-*-120-*-*-*-*-*-*
*FvwmWinListAction Click1 Iconify -1,Raise
*FvwmWinListAction Click2 Iconify
*FvwmWinListAction Click3 Module "FvwmIdent" FvwmIdent
*FvwmWinListUseSkipList
*FvwmWinListUseIconNames
*FvwmWinListGeometry -50-85
# I like it anchored
#*FvwmWinListNoAnchor
```

AUTHOR

Mike Finger (mfinger@mermaid.micro.umn.edu) (Mike_Finger@atk.com) (doodman on IRC, check the #linux channel)

FvwmWinList can be invoked by fvwm during initialization by inserting the line ‘Module FvwmWinList’ in the .fvwmrc file.

FvwmWinList can also be bound to a keystroke, mouse button, or menu option to be invoked later, in this case using ‘Transient’ as an argument will cause FvwmWinList to resemble the built in window list.

FvwmWinList must reside in a directory that is listed in the ModulePath option of FVWM for it to be executed by FVWM.

CONFIGURATION OPTIONS

The following options can be placed in the .fvwmrc file

***FvwmWinListGeometry {+-}<X>x{+-}<Y>**

Specifies the location and gravity of the FvwmWinList window. At the current time, size is not supported and FvwmWinList will resize itself as buttons are added. If the NoAnchor option is not specified then the windows gravity corner will be anchored, and the window will grow in the opposite direction. (i.e. If the geometry is specified -5-5, that is SoutEastGravity. This will cause the window to draw up and to the left as windows are added)

***FvwmWinListFont font**

Specifies the font to be used for labeling the buttons.

***FvwmWinListFore color**

Specifies the color to use for the button names.

***FvwmWinListBack color**

Specifies the color for the buttons.

***FvwmWinListUseSkipList**

Tells FvwmWinList to not show the windows that are listed on a WindowListSkip line if the configuration file.

***FvwmWinListNoAnchor**

By default, FvwmWinList will anchor the gravity corner so the window will grow in the opposite direction. This undoes that option.

***FvwmWinListUseIconNames**

Tells FvwmWinList to use the icon name of the window instead of the full window name. This is useful to keep the width of the window small.

***FvwmWinListAction action response[,reponse...]**

Tells FvwmWinList to do response when action is done. The currently supported actions are: Click1, Click2, Click3. The currently supported responses are any fvwm built-in commands, including modules and functions.

SAMPLE CONFIGURATION

The following are excepts from a .fvwmrc file which describe FvwmWinList initialization commands:

```
########
```

describe the horizontal and vertical window size reduction. Fvwm will search directory specified in the ModulePath configuration option to attempt to locate FvwmScroll. Although nothing keeps you from launching FvwmScroll at start-up time, you probably don't want to.

CONFIGURATION OPTIONS

FvwmScroll reads the same .fvwmrc file as fvwm reads when it starts up, and looks for lines as listed below:

***FvwmScrollFore color**

Tells the module to use color instead of grey for scroll bars themselves.

***FvwmScrollBack color**

Tells the module to use color instead of black for the window background.

BUGS

When the scroll bars are removed by clicking on the button in the lower right corner, the window does not restore its location correctly.

AUTHOR

Robert Nation

2.8.13 FvwmWinlist Manual Page

NAME

FvwmWinList - the FVWM window list module

SYNOPSIS

FvwmWinList is spawned by fvwm, so no command line invocation will work.

DESCRIPTION

The FvwmWinList module provides a widow list made up of buttons, each corresponding to a window that FVWM is managing. Clicking on the buttons with any of the three mouse buttons will either do a default action or can be user configured. Like the other modules, FvwmWinList only works when fvwm is used as the window manager.

COPYRIGHTS

The FvwmWinList module is the original work of Mike Finger.

Copyright 1994, Mike Finger. The author makes no guarantees or warranties of any kind about the use of this module. Use this modules at your own risk. You may freely use this module or any portion of it for any purpose as long as the copyright is kept intact.

INITIALIZATION

During initialization, FvwmWinList will scan the same configuration file that FVWM used during startup to find the options that pertain to it. These options are discussed in a later section.

INVOCATION

If the geometry is not specified, then a desktop reduction factor is used to calculate the pager's size. Things in the pager window are shown at 1/number of the actual size.

AUTHOR

Robert Nation

2.8.12 FvwmScroll Manual Page

NAME

FvwmScroll - the FVWM scroll-bar module

SYNOPSIS

FvwmScroll is spawned by fvwm, so no command line invocation will work.

DESCRIPTION

The FvwmScroll module prompts the user to select a target window, if the module was not launched from within a window context in Fvwm. After that, it adds scroll bars to the selected window, to reduce the total desktop space consumed by the window.

FvwmScroll reads the same .fvwmrc file as fvwm reads when it starts up, and looks for lines similar to "*FvwmScrollFore green".

FvwmScroll should not be used with windows which move or resize themselves, nor should it be used with windows which set the WM_COLORMAP_WINDOWS property. Operation is fine with windows that have a private colormap.

COPYRIGHTS

The FvwmScroll program, and the concept for interfacing this module to the Window Manager, are all original work by Robert Nation.

Copyright 1994, Robert Nation. No guarantees or warranties or anything are provided or implied in any way whatsoever.

Use this program at your own risk. Permission to use this program for any purpose is given, as long as the copyright is kept intact.

INITIALIZATION

During initialization, FvwmScroll will eventually search a configuration file which describes the colors to use. The configuration file is the same file that fvwm used during initialization.

If the FvwmScroll executable is linked to another name, ie ln -s FvwmScroll MoreScroll, then another module called MoreScroll can be started, with a completely different configuration than FvwmScroll, simply by changing the keyword FvwmScroll to MoreScroll.

INVOCATION

FvwmScroll can be invoked by binding the action 'Module FvwmScroll x y' to a menu or key-stroke in the .fvwmrc file. The parameter x and y are integers, which might want to leave out either the width or height dimension of the geometry specification

***FvwmPagerRows rows**

Tells fvwm how many rows of desks to use when laying out the pager window.

***FvwmPagerColumns columns**

Tells fvwm how many columnss of desks to use when laying out the pager window.

***FvwmPagerIconGeometry geometry**

Specifies a size (optional) and location (optional) for the pager's icon window. Since there is no easy way for FvwmPager to determine the height of the icon's label, you will have to make an allowance for the icon label height when using negative y-coordinates in the icon location specification (used to specify a location relative to the bottom instead of the top of the screen).

***FvwmPagerStartIconic**

Causes the pager to start iconified.

***FvwmPagerFont font-name**

Specified a font to use to label the desktops. If font_name is "none" then no desktop labels will be displayed.

***FvwmPagerSmallFont font-name**

Specified a font to use to label the window names in the pager. If not specified, the window labels will be omitted. Window labels seem to be fairly useless for desktop scales of 32 or greater.

***FvwmPagerFore color**

Specifies the color to use to write the desktop labels, and to draw the page-grid lines.

***FvwmPagerBack color**

Specifies the background color for the window.

***FvwmPagerHilight color**

The active page and desk label will be highlighted by using this background pattern instead of the normal background.

***FvwmPagerLabel desk label**

Assigns the text label to desk desk in the pager window. Useful for assigning symbolic names to desktops, ie

```
*FvwmPagerLabel 1 Mail
*FvwmPagerLabel 2 Maker
*FvwmPagerLabel 2 Matlab
```

DeskTopScale number

desktop, a full size image of the window will appear for you to place. There is no way to pick up a full size image of the window and move it into the pager, however. Since some mice do not have button 2, I have made provisions to drag windows in the pager by using pressing modifier-1 (usually Alt) and dragging with button 3.

Clicking mouse button 3 on a location within the current desktop will cause the viewport to move to the selected location, but will not align the viewport to a page boundary. Dragging button 3 will cause the viewport to move as you drag. Button 3 operations are limited to the current desktop only.

When iconified, the pager will work as a fully functional current page only pager. Windows and viewports can be moved within the icon of the pager. Users will want to make sure that they have no lines similar to Icon "Fvwm Pager" whatever in their .fvwmrc files.

COPYRIGHTS

The FvwmClean program, and the concept for interfacing this module to the Window Manager, are all original work by Robert Nation.

Copyright 1994, Robert Nation. No guarantees or warranties or anything are provided or implied in any way whatsoever. Use this program at your own risk. Permission to use this program for any purpose is given, as long as the copyright is kept intact.

INITIALIZATION

During initialization, FvwmPager will eventually search a configuration file which describes the time-outs and actions to take. The configuration file is the same file that fvwm used during initialization.

If the FvwmPager executable is linked to another name, ie ln -s FvwmPager OtherPager, then another module called Other- Pager can be started, with a completely different configuration than FvwmPager, simply by changing the keyword FvwmPager to OtherPager. This way multiple pager programs can be used.

KEYBOARD FOCUS CONTROL

You can direct the keyboard focus to any window on the current desktop by clicking with button 2 on its image in the pager. The window does not need to be visible, but it does need to be on the current page.

INVOCATION

The invocation method was shown in the synopsis section

CONFIGURATION OPTIONS

FvwmPager reads the same .fvwmrc file as fvwm reads when it starts up, and looks for certain configuration options:

***FvwmPagerGeometry geometry**

Completely or partially specifies the pager windows location and geometry, in standard X11 notation. In order to maintain an undistorted aspect ratio, you

Tells the module to use color instead of black for the window background.

***FvwmIdentFont fontname**

Tells the module to use fontname instead of fixed for text.

AUTHOR

Robert Nation and and Nobutaka Suzuki (nobuta-s@is.aist-nara.ac.jp).

2.8.11 FvwmPager Manual Page

NAME

FvwmPager - the FVWM Pager module

SYNOPSIS

FvwmPager is spawned by fvwm, so no command line invocation will work. From within the .fvwmrc file, FvwmPager is spawned as follows:

```
Module FvwmPager 0 3
```

or from within an fvwm pop-up menu:

```
Popup “Module-Popup”
Title “Modules”
Module “GoodStuff” GoodStuff
Module “NoClutter” FvwmClean
Module “FvwmIdentify” FvwmIdent
Module “Banner” FvwmBanner
Module “Debug” FvwmDebug
Module “SaveDesktop” FvwmSave
Module “Pager” FvwmPager 0 3
EndPopup
```

where “0” is the first desktop to show, and “3” is the last.

DESCRIPTION

The FvwmPager module shows a miniature view of the Fvwm desktops which are specified in the command line. This is a useful reminder of where your active windows are. Windows in the pager are shown in the same color as their fvwm decorations.

The pager can be used to change your viewport into the current desktop, to change desktops, or to move windows around.

Pressing mouse button 1 in the pager will cause you viewport to change to the selected page of the selected desk. If you click with button 1 in the desk-label area, you will switch desks but not pages within the desk.

Dragging mouse button 2 on a miniature view of a window will cause that window to be move to the location where you release the mouse button, but your viewport will not change. If you drag the window out of the pager and onto your

2.8.10 FvwmIdent Manual Page

NAME

FvwmIdent - the FVWM identify-window module

SYNOPSIS

FvwmIdent is spawned by fvwm, so no command line invocation will work.

DESCRIPTION

The FvwmIdent module prompts the user to select a target window, if the module was not launched from within a window context in Fvwm. After that, it pops up a window with information about the window which was selected.

FvwmIdent reads the same .fvwmrc file as fvwm reads when it starts up, and looks for lines similar to "*FvwmIdentFore green".

COPYRIGHTS

The FvwmIdent program, and the concept for interfacing this module to the Window Manager, are all original work by Robert Nation and Nobutaka Suzuki.

Copyright 1994, Robert Nation and Nobutaka Suzuki. No guarantees or warranties or anything are provided or implied in any way whatsoever. Use this program at your own risk. Permission to use this program for any purpose is given, as long as the copyright is kept intact.

INITIALIZATION

During initialization, FvwmIdent will eventually search a configuration file which describes the colors and font to use. The configuration file is the same file that fvwm used during initialization.

If the FvwmIdent executable is linked to another name, ie ln -s FvwmIdent MoreIdentify, then another module called MoreIdentify can be started, with a completely different configuration than FvwmIdent, simply by changing the keyword FvwmIdent to MoreIdentify. This way multiple clutter- reduction programs can be used.

INVOCATION

FvwmIdent can be invoked by binding the action 'Module FvwmIdent' to a menu or key-stroke in the .fvwmrc file. Fvwm will search directory specified in the ModulePath configuration option to attempt to locate FvwmIdent. Although nothing keeps you from launching FvwmIdent at start-up time, you probably don't want to.

CONFIGURATION OPTIONS

FvwmIdent reads the same .fvwmrc file as fvwm reads when it starts up, and looks for lines as listed below:

***FvwmIdentFore color**

Tells the module to use color instead of black for text.

***FvwmIdentBack color**

```
*FvwmIconBoxMouse 2 Click Iconify -1, Focus
*FvwmIconBoxMouse 3 Click Module “FvwmIdent” FvwmIdent
#
# Key bindings
#
*FvwmIconBoxKey r RaiseLower
*FvwmIconBoxKey space Iconify
*FvwmIconBoxKey d Close
#
# FvwmIconBox built-in functions
#
*FvwmIconBoxKey n Next
*FvwmIconBoxKey p Prev
*FvwmIconBoxKey h Left
*FvwmIconBoxKey j Down
*FvwmIconBoxKey k Up
*FvwmIconBoxKey l Right
#
# Icon file spcifications
#
*FvwmIconBox “*” unknown1.xpm
*FvwmIconBox “Fvwm*” -
*FvwmIconBox “FvwmFileMgr” folder2.xpm
*FvwmIconBox “rxvt” term.xpm
*FvwmIconBox “xterm” xterm.xpm
*FvwmIconBox “editres” editres.xpm
*FvwmIconBox “xman” xman.xpm
*FvwmIconBox “xbiff” mail1.xpm
*FvwmIconBox “xmh” mail1.xpm
*FvwmIconBox “xcalc” xcalc.xpm
*FvwmIconBox “GoodStuff” toolbox.xpm
```

AUTHOR

Nobutaka Suzuki (nobuta-s@is.aist-nara.ac.jp).

Specifies the bitmap to be displayed in iconbox for windowname. Windowname can be window name, class name, or resource name. Windowname can contain “*” and “?” like Fvwm configuration file. The bitmap-file is either the full path name to a bitmap file, or a file in the IconPath or PixmapPath. If bitmap-file is specefied to be “-”, the icon for a window corresponding to win- downame is not shown in the iconbox.

SAMPLE CONFIGURATION

The following are excepts from a .fvwmrc file which describe FvwmIconBox initialization commands:

```
##############################################################
# Make no icon to be shown in root window
# by spcifying in your .fvwmrc
# (1) “SuppressIcons” if PRUNE is defined in configure.h, or
# (2) “Style “*” NoIcon” if PRUNE is defined.
#SuppressIcons
Style “*” NoIcon
 ##############################################################
*FvwmIconBoxIconBack #cfcfcf
*FvwmIconBoxIconHiFore black
*FvwmIconBoxIconHiBack LightSkyBlue
*FvwmIconBoxBack #5f9ea0
#*FvwmIconBoxFore blue
*FvwmIconBoxGeometry 5x1+0+0
*FvwmIconBoxMaxIconSize 64x38
*FvwmIconBoxFont -adobe-helvetica-medium-r-*-*-12-*-*-*-*-*-*
*FvwmIconBoxSortIcons
*FvwmIconBoxPadding 4
*FvwmIconBoxLines 10
*FvwmIconBoxPlacement Left Top
*FvwmIconBoxPixmap fvwm.xpm
#*FvwmIconBoxHideSC Horizontal
#
# mouse bindings
#
*FvwmIconBoxMouse 1 Click RaiseLower
*FvwmIconBoxMouse 1 DoubleClick Iconify
```

Left: Icons are placed from left to right.

Right: Icons are placed from right to left.

For example, when the placement is "Left Top", icons are placed from left to right, and new rows are added from top to bottom. The default vale is "Left Bottom".

***FvwmIconBoxLines**

Specifies the number of icons placed in a line. If primary is Left or Right, this value specifies the number of columns. If primary is Top or Bottom, this value specifies the number of rows. For example, seven icons are placed in a row if *FvwmIconBoxLines is 7 and *FvwmIconBoxPlacement is "Left Top". The default value is 6.

***FvwmIconBoxHideSC direction**

Specifies the undisplayed scroll bar . Direction can be either Horizontal or Vertical.

***FvwmIconBoxGeometry <width>x<height>{+-}<X>{+-}<Y>**

Specifies the location and/or size of FvwmIconBox. width and height are measured in icons, not pixels. The default value is 6x1+0+0.

***FvwmIconBoxMaxIconSize <width>x<height>**

Specifies the maximum size of icon bitmap. A bitmap larger than this size is clipped to this size. The default value is 48x48.

***FvwmIconBoxMouse Button Action Response[, Response]**

Tells the module to do Responce when Action is done on button Button. Available Responsees are built-in commands in Fvwm (e.g. Iconify, Delete, Focus) and available Actions are Click and DoubleClick.

***FvwmIconBoxKey Key Response[, Response]**

Tells the module to do Response when Key is pressed. Available Responcees are, besides Fvwm built-in commands, the following six FvwmIconBox built-in commands: Next, Prev, Left, Right, Up, and Down.

Next: Change the hilited-icon to the next.

Prev: Change the hilited-icon to the previous.

Left: Move the slider of the horizontal scrollbar to left. Icons move to right accordingly.

Right: Move the slider of the horizontal scrollbar to right. Icons move to left accordingly.

Up: Move the slider of the vertical scrollbar to up. Icons move to down accordingly.

Down: Move the slider of the vertical scrollbar to down. Icons move to up accordingly.

***FvwmIconBox windowname bitmap-file**

Tells the module to use color instead of white for the window foreground. This option affects only the foreground color of background_bitmap specifed in *FvwmIconBoxPixmap option described below.

***FvwmIconBoxBack color**

Tells the module to use color instead of black for the window background.

***FvwmIconBoxIconFore color**

Tells the module to use color instead of black for non-selected icon text.

***FvwmIconBoxIconBack color**

Tells the module to use color instead of white for the non-selected icon background.

***FvwmIconBoxIconHiFore color**

Tells the module to use color instead of black for selected icon text.

***FvwmIconBoxIconHiBack color**

Tells the module to use color instead of white for the selected icon background.

***FvwmIconBoxPixmap pixmap**

Tells the module to use pixmap for the window background_pixmap.

***FvwmIconBoxFont fontname**

Tells the module to use fontname instead of fixed for text.

***FvwmIconBoxSortIcons**

Tells the module to sort all icons in iconbox in alphabetical order.

***FvwmIconBoxPadding number**

Specifies the number of pixels between icons. The default value is 5.

***FvwmIconBoxPlacement primary secondary**

Specifies icon placement policy. primary and secondary can be Top, Bottom, Left and Right. The following eight combinations are available:

primary secondary
Left Top
Left Bottom
Right Top
Right Bottom
Top Left
Top Right
Bottom Left
Bottom Right

Top: Icons are placed from top to bottom.

Bottom: Icons are placed from bottom to top.

The last line is a little tricky - it spawns an FvwmPager module, and captures it to display in a quadruple width button. Since "-" is used for the button label, the Pager will be resized to be as big as possible within the button's borders. "whatever", of course, is a non-existent icon file, and will be ignored by GoodStuff.

AUTHOR

Robert Nation

2.8.9 FvwmIconBox Manual Page

NAME

FvwmIconBox - the FVWM iconbox module

SYNOPSIS

FvwmIconBox is spawned by fvwm, so no command line invocation will work.

DESCRIPTION

The FvwmIconBox module provides an icon manager. The user can do operations, like iconify and deiconify, for each icon shown in the module via mouse and keyboard.

FvwmIconBox reads the same .fvwmrc file as fvwm reads when it starts up, and looks for lines similar to "*FvwmIconBoxFore green".

COPYRIGHTS

The FvwmIconBox program is original work of Nobutaka Suzuki.

Copyright 1994, Nobutaka Suzuki. No guarantees or warranties or anything are provided or implied in any way whatsoever. Use this program at your own risk. Permission to use this program for any purpose is given, as long as the copyright is kept intact.

INITIALIZATION

During initialization, FvwmIconBox will eventually search a configuration file. The configuration file is the same file that fvwm used during initialization.

If the FvwmIconBox executable is linked to another name, ie ln -s FvwmIconBox MoreIconBox, then another module called MoreIconBox can be started, with a completely different configuration than FvwmIconBox, simply by changing the keyword FvwmIconBox to MoreIconBox. This way multiple clutter-reduction programs can be used.

INVOCATION

FvwmIconBox can be invoked by binding the action 'Module FvwmIconBox' to a menu or key-stroke in the .fvwmrc file. Fvwm will search directory specified in the ModulePath configuration option to attempt to locate FvwmIconBox.

CONFIGURATION OPTIONS

FvwmIconBox reads the same .fvwmrc file as fvwm reads when it starts up, and looks for lines as listed below:

***FvwmIconBoxFore color**

Stuff to spawn an rclock process, and capture the first window whose name or resource is “Clock”, and display it in the button-bar. This is handy for applications like xclock, xbiff, and xload. Modules can be swallowed by specifying SwallowModule instead of just “swallow” NOTE: if you use xclock for this application, you will want to specify xclock -padding 0.

SAMPLE CONFIGURATION

The following are excepts from a .fvwmrc file which describe GoodStuff initialization commands:

```
#####################################################################
# Load any modules which should be started during
# fvwm initialization
ModulePath /usr/lib/X11/fvwm:/usr/bin/X11
Module GoodStuff
# Make goodstuff title-bar-less, sticky, and give it an icon
NoTitle GoodStuff
Sticky GoodStuff
Icon “GoodStuff” toolbox.xpm
#####################################################################
#Definitions used by the modules
*GoodStuffFore Black
*GoodStuffBack #908090
*GoodStuffGeometry -135-5
*GoodStuffRows 1
*GoodStuffFont -adobe-helvetica-medium-r-*-*-12-*-*-*-*-*-*
*GoodStuff Resize resize.xpm Resize
*GoodStuff Move arrows2.xpm Move
*GoodStuff Lower Down Lower
*GoodStuff Raise Up Raise
*GoodStuff Kill bomb.xpm Destroy
*GoodStuff Dopey rterm.xpm Exec “big_win” xterm -T big_win -geometry
80x50&
*GoodStuff Snoopy rterm.xpm Exec “small_win” xterm -T small_wi &
*GoodStuff Xcalc rcalc.xpm Exec “Calculator” xcalc &
*GoodStuff XMag magnifying_glass2.xpm Exec “xmag” xmag &
*GoodStuff Mail mail2.xpm Exec “xmh” xmh &
*GoodStuff(4x1) - whatever SwallowModule “Fvwm Pager” FvwmPager 0 3
```

keystroke to invoke it later. Fvwm will search directory specified in the ModulePath configuration option to attempt to locate GoodStuff.

CONFIGURATION OPTIONS

The following options can be placed in the .fvwmrc file

***GoodStuffGeometry geometry**

Specifies the GoodStuff window location and/or size. If the size is not specified, GoodStuff will auto-size itself in a reasonable manner. The geometry is a standard X11 window geometry specification.

***GoodStuffFont font**

Specifies the font to be used for labeling the buttons.

***GoodStuffRows rows**

Specifies the number of rows of buttons to be created. If unspecified, 2 rows will be used.

***GoodStuffColumns columns**

Specifies the number of columns of buttons to be created. If unspecified, the number of columns will be set to the number of buttons requested, divided by the number of rows. If both the rows and columns are specified, but do not specify as many buttons as are defined, then the users columns specification will be ignored.

***GoodStuffFore color**

Specifies the color for button label text and monochrome icons.

***GoodStuffBack color**

Specifies the color for the buttons.

***GoodStuff[(w x h)] label icon command**

Specifies a window manager built-in command, as described in the fvwm man page, which should be executed when a button is pressed. The label field is the text which will be written on the button. The icon field specifies an X11 bitmap file or XPM color icon file, containing the icon to display on the button. GoodStuff will search through the path specified in the fvwm IconPath or PixmapPath configuration items to find the icon file. If the optional (w x h) is specified, then this button will be w times the normal button width and h times the normal button height. If label is "-" then the button label will not be displayed.

If command is an fvwm Exec command, then the button will remain pushed in until a window whose name or class matches the qouted portion of the command is encountered. This is intended to provide visual feed- back to the user that the action he has requested will be performed. If the qouted portion contains no characters, then the button will pop out immediately. Note that users can continue pressing the button, and re-executing the command, even when it looks "pressed in."

A single extension to the fvwm built-in command set is provided. A command of the form: *GoodStuff junk clock.xpm Swallow "Clock" rclock & will cause Good-spawned during fvwm's initialization, or can be bound to a menu or mouse button or keystroke to invoke it later. Fvwm will search directory specified in the ModulePath configuration option to attempt to locate FvwmDebug.

CONFIGURATION OPTIONS

There are no configuration options.

AUTHOR

Robert Nation

2.8.8 Fvwm Goodstuff Manual Page

NAME

GoodStuff - the FVWM button panel module

SYNOPSIS

GoodStuff is spawned by fvwm, so no command line invocation will work.

DESCRIPTION

The GoodStuff module provides a button panel which sits on the X terminal's root window. The user can press the buttons at any time, and trigger invocation of a user-specified command by the window manager. GoodStuff only works when fvwm is used as the window manager.

The button panel can be of any configuration or geometry, and can have monochrome or color icons to represent the actions which would be invoked.

COPYRIGHTS

The GoodStuff program, and the concept for interfacing this module to the Window Manager, are all original work by Robert Nation

Copyright 1993, Robert Nation. No guarantees or warranties or anything are provided or implied in any way whatsoever. Use this program at your own risk. Permission to use this program for any purpose is given, as long as the copyright is kept intact.

INITIALIZATION

During initialization, GoodStuff will search for a configuration file which describes the button panel geometry, color, icons, and actions. The format of this files will be described later. The configuration file will be the one which fvwm used during its initialization.

If the GoodStuff executable is linked to another name, ie ln -s GoodStuff TermStuff, then another module called TermStuff can be started, with a completely different configuration than GoodStuff, simply by changing the keywords Good??? to Term???. This way multiple button-bars can be used.

INVOCATION

GoodStuff can be invoked by inserting the line 'Module GoodStuff' in the .fvwmrc file. This can be placed on a line by itself, if GoodStuff is to be spawned during fvwm's initialization, or can be bound to a menu or mouse button or pletely different configuration than FvwmClean, simply by changing the keyword FvwmClean to OtherClutter. This way multiple clutter- reduction programs can be used.

INVOCATION

FvwmClean can be invoked by inserting the line 'Module FvwmClean' in the .fvwmrc file. This can be placed on a line by itself, if FvwmClean is to be spawned during fvwm's initialization, or can be bound to a menu or mouse button or keystroke to invoke it later. Fvwm will search directory specified in the ModulePath configuration option to attempt to locate FvwmClean.

CONFIGURATION OPTIONS

FvwmClean reads the same .fvwmrc file as fvwm reads when it starts up, and looks for lines similar to "*FvwmClean 3600 Iconify". The format of these lines is "*FvwmClean [time] [command]", where command is any fvwm built-in command, and time is the time in seconds between when a window looses focus and when the command is executed. At most 3 actions can be specified.

AUTHOR

Robert Nation

2.8.7 FvwmDebug Manual Page

NAME

FvwmDebug - the FVWM debugger module

SYNOPSIS

FvwmDebug is spawned by fvwm, so no command line invocation will work.

DESCRIPTION

The FvwmDebug captures the stream of module packets coming from fvwm, and displays them on stderr.

COPYRIGHTS

The FvwmDebug program, and the concept for interfacing this module to the Window Manager, are all original work by Robert Nation

Copyright 1994, Robert Nation. No guarantees or warranties or anything are provided or implied in any way whatsoever. Use this program at your own risk. Permission to use this program for any purpose is given, as long as the copyright is kept intact.

INITIALIZATION

During initialization, the FvwmDebug requests that fvwm dump a complete window list into its pipe. These packets are terminated with and "end Send_WindowList" message.

INVOCATION

FvwmDebug can be invoked by inserting the line 'Module FvwmDebug' in the .fvwmrc file. This can be placed on a line by itself, if FvwmDebug is to be

COPYRIGHTS

None.

INITIALIZATION

Nothing interesting.

INVOCATION

FvwmBanner can be invoked by binding the action ‘Module FvwmBanner’ to a menu or key-stroke in the .fvwmrc file. Fvwm will search directory specified in the ModulePath configuration option to attempt to locate FvwmBanner. Although nothing keeps you from launching FvwmBanner at start-up time, you probably don’t want to.

CONFIGURATION OPTIONS

There are no configuration options.

AUTHOR

Robert Nation

2.8.6 FvwmClean Manual Page

NAME

FvwmClean - the FVWM desktop clutter reduction module

SYNOPSIS

FvwmClean is spawned by fvwm, so no command line invocation will work.

DESCRIPTION

The FvwmClean module performs actions on windows which have not had the keyboard focus for a specific length of time. This is intended to help alleviate the problem of leaving programs running in unused portions of your desktop.

FvwmClean reads the same .fvwmrc file as fvwm reads when it starts up, and looks for lines similar to “*FvwmNoClutter 3600 Iconify”.

COPYRIGHTS

The FvwmClean program, and the concept for interfacing this module to the Window Manager, are all original work by Robert Nation

Copyright 1994, Robert Nation. No guarantees or warranties or anything are provided or implied in any way whatsoever. Use this program at your own risk. Permission to use this program for any purpose is given, as long as the copyright is kept intact.

INITIALIZATION

During initialization, FvwmClean will eventually search a configuration file which describes the time-outs and actions to take. The configuration file is the same file that fvwm used during initialization.

If the FvwmClean executable is linked to another name, ie ln -s FvwmClean OtherClutter, then another module called Other-Clutter can be started, with a com-

INVOCATION

FvwmBacker can be invoked by fvwm during initialization by inserting the line 'Module FvwmBacker' in the .fvwmrc file.

FvwmBacker must reside in a directory that is listed in the ModulePath option of FVWM for it to be executed by FVWM.

CONFIGURATION OPTIONS

The follow option is the only option supported at present

***FvwmWinListDesk DeskNumber command**

Specifies the command to execute when the specified DeskNumber becomes active.

If the command begins with -solid FvwmBacker uses the next argument as a color in the X database and sets the background to that color without generating a system call to xsetroot. Only single word color names may be used.

SAMPLE CONFIGURATION

The following are excepts from a .fvwmrc file which describe FvwmBacker initialization commands:

```
####
# Set Up Backgrounds for different desktops.
####
*FvwmBackerDesk 0 -solid steelblue
*FvwmBackerDesk 1 -solid midnightblue
*FvwmBackerDesk 2 -solid yellow
*FvwmBackerDesk 3 xpmroot /usr/include/X11/pixmaps/
background2.xpm
```

AUTHOR

Mike Finger (mfinger@mermaid.micro.umn.edu) (Mike_Finger@atk.com) (doodman on IRC, check the #linux channel)

Modified by Andrew Davison (davison@cs.monash.edu.au)

2.8.5 FvwmBanner Manual Page

NAME

FvwmBanner - the FVWM Banner

SYNOPSIS

FvwmBanner is intended to be spawned by fvwm, but it will work if run from the command line.

DESCRIPTION

The FvwmInitBanner displays an Fvwm Logo in the center of the screen for 5 seconds.

SYNOPSIS

FvwmAuto is spawned by fvwm, so no command line invocation will work. An appropriate invocation from fvwm looks like:

Module FvwmAuto 200 or, from a menu: Popup “Module-Popup”

```
Title “Modules”
Module “GoodStuff” GoodStuff
Module “NoClutter” FvwmClean
Module “FvwmIdentify” FvwmIdent
Module “Banner” FvwmBanner
 Module “Debug” FvwmDebug
Module “SaveDesktop” FvwmSave
Module “AutoRaise” FvwmAuto 200 EndPopup
```

The numeric argument is required. It specified how long a window must retain the keyboard input focus before it is raised. The delay is measured in milliseconds, and any integer 0 or greater is acceptable.

AUTHOR

FvwmAuto just appeared one day, nobody knows how.

2.8.4 FvwmBacker Manual Page

NAME

FvwmBacker - the FVWM background changer module

SYNOPSIS

FvwmBacker is spawned by fvwm, so no command line invocation will work.

DESCRIPTION

The FvwmBacker module provides functionality to change the background when changing desktops. Any command can be executed to change the backgrounds. The commands used to change the backgrounds must exit after changing the root window, (i.e. ‘xv -root zzz.gif’ does not, without the -quit option) otherwise FvwmBacker will hang.

COPYRIGHTS

The FvwmBacker module is the original work of Mike Finger.

Copyright 1994, Mike Finger. The author makes no guarantees or warranties of any kind about the use of this module. Use this modules at your own risk. You may freely use this module or any portion of it for any purpose as long as the copyright is kept intact.

INITIALIZATION

During initialization, FvwmBacker will scan the same configuration file that FVWM used during startup to find the options that pertain to it. These options are discussed in a later section.

```
*FvwmAudio add_window drip.au
*FvwmAudio raise_window swoosh.au
*FvwmAudio lower_window swoosh.au
*FvwmAudio focus_change boing.au
*FvwmAudio destroy_window explosion.au
*FvwmAudio iconify ploop.au
*FvwmAudio deiconify ploop.au
*FvwmAudio toggle_paging fwop.au
*FvwmAudio new_page beam_trek.au
*FvwmAudio new_desk beam_trek.au
*FvwmAudio configure_window huh.au
*FvwmAudio window_name beep.au
*FvwmAudio icon_name beep.au
*FvwmAudio res_class beep.au
*FvwmAudio res_name beep.au
*FvwmAudio end_windowlist twang.au
```

INVOCATION

The invocation method was shown in the synopsis section. No command line invocation is possible. FvwmAudio must be invoked by the fvwm window manager.

BUGS

It’s REALLY noisy when fvwm starts and restarts.

COPYRIGHTS

This module is heavily based on a similar Fvwm module called FvwmSound by Mark Boyns. FvwmAudio simply takes Mark’s original program and extends it to make it generic enough to work with any audio player. The concept for interfacing this module to the Window Manager, is original work by Robert Nation.

Copyright 1994, Mark Boyns and Mark Scott. No guarantees or warranties or anything are provided or implied in any way whatsoever. Use this program at your own risk. Permission to use this program for any purpose is given, as long as the copyright is kept intact.

AUTHORS

Mark Boyns (boyns@sdsu.edu)

Mark Scott (mscott@mcd.mot.com)

2.8.3 FvwmAuto Manual Page

NAME

FvwmAuto - the FVWM auto-raise module

DESCRIPTION

The FvwmAudio module communicates with the Fvwm window manager to bind audio sounds to window manager actions. Different audio queues may be assigned to any window manager action. FvwmAudio can be used with any independent audio player, and therefore requires no special audio library APIs of its own. Simple in its design, it merely binds the audio filenames to particular actions and forks off the audio player program with the appropriate filename when that action occurs.

FvwmAudio can also have builtin support for the rplay library.

CONFIGURATION OPTIONS

FvwmAudio reads the same .fvwmrc file as fvwm(1) reads when it starts up, and looks for certain configuration options:

***FvwmAudioPlayCmd /usr/bin/X11/demos/auplay**

This determines the independent audio player program that will actually play the sounds. If the play command is set to builtin-rplay then the builtin rplay support will be used. For example: *FvwmAudioPlayCmd builtin-rplay

***FvwmAudioDir /usr/lib/sounds**

Specifies the directory to look for the audio files. This option is ignored when rplay is used.

***FvwmAudioDelay 5**

Specifies that sound events will only be played if they occur at least 5 seconds after the previous event. Sounds events that occur during the delay period are ignored. This option is useful if you don't want several sounds playing at the same time. The default delay is 0 which disables the audio delay.

***FvwmAudioRplayHost hostname**

Specifies what host the sounds will play on. The host- name can also be an environment variable such as $HOST-DISPLAY. This option is only valid with builtin rplay support.

***FvwmAudioRplayPriority 0**

Specifies what priority will be assigned to the sounds when they are played. This option is only valid with builtin rplay support.

***FvwmAudioRplayVolume 127**

Specifies what volume will be assigned to the sounds when they are played. This option is only valid with builtin rplay support.

***FvwmAudio window-manager_action audio_filename**

Binds particular window manager actions to sound queues.

***FvwmAudio startup TaDa.au**

***FvwmAudio shutdown Elvis_Left.au**

***FvwmAudio unknown doh.au**

Multi-screen support is only available if fvwm is compiled with -DMULTIPLE_SCREENS.

BUGS

As of fvwm 0.99 there were exactly 39.342 unidentified bugs. Identified bugs have mostly been fixed, though. Since then 9.34 bugs have been fixed. Assuming that there are at least 10 unidentified bugs for every identified one, that leaves us with 39.342 - 9.32 + 10 * 9.34 = 123.402 unidentified bugs. If we follow this to its logical conclusion we will have an infinite number of unidentified bugs before the number of bugs can start to diminish, at which point the program will be bug-free. Since this is a computer program infinity = 3.4028e+38 if you don’t insist on double- precision. At the current rate of bug discovery we should expect to achieve this point in 3.37e+27 years. I guess I better plan on passing this thing on to my children....

Binding a key to a window decoration but not to the window itself is discouraged because when the key-press event finally gets to the window it will be marked as SYNTHETIC and will be ignored by many applications.

Bug reports can be sent to fvwm@wonderland.org.

AUTHOR

Robert Nation with help from many people, based on twm code, which was written by Thomas LaStrange.

2.8.2 FvwmAudio Manual Page

NAME

FvwmAudio - the FVWM Audio module

SYNOPSIS

FvwmAudio is spawned by fvwm, so no command line invocation will work. From within the .fvwmrc file, FvwmAudio is spawned as follows:

```
Module FvwmAudio
```

or from within an fvwm pop-up menu:

```
    Popup “Module-Popup”
    Title “Modules”
    Module “GoodStuff” GoodStuff
    Module “NoClutter” FvwmClean
    Module “FvwmIdentify” FvwmIdent
    Module “Banner” FvwmBanner
    Module “Debug” FvwmDebug
    Module “Audio” FvwmAudio
    Module “Pager” FvwmPager 0 3
    EndPopup
```

Warp [name window_name]

Same as CirculateDown but de-iconifies any iconified windows as it focuses on them.

WindowsDesk new_desk

Moves the selected window the the desktop specified as new_desk.

WindowList arg1 arg2

Generates a pop-up menu (and pops it up) in which the title and geometry of each of the windows currently on the desk top are shown. The geometry of iconified windows is shown in brackets. Selecting an item from the window list pop-up menu will cause that window to be moved onto the desktop if it is currently not on it, will move the desktop viewport to the page containing the upper left hand corner of the window, will de-iconify the window if it is iconified, and will raise the window.

If arg1 is an even number then the windows will be listed using the window name (the name that shows up in the title-bar). If it is odd then the window's icon name is used.

If arg1 is less than 2 then all windows on all desktops (except those listed in WindowListSkip directives) will be shown.

If arg1 is 2 or 3 then only windows on the current desktop will be shown.

If arg1 is 4 or 5 then only windows on desktop number arg2 will be shown.

KEYBOARD SHORTCUTS

All (I think) window manager operations can be performed from the keyboard so mouseless operation should be possible. In addition to scrolling around the virtual desktop by binding the Scroll built-in to appropriate keys, pop-ups, move, resize, and most other built-ins can be bound to keys. Once a built-in function is started the pointer is moved by using the up, down, left, and right arrows, and the action is terminated by pressing return. Holding down the shift key will cause the pointer movement to go in larger steps and holding down the control key will cause the cursor movement to go in smaller steps. Standard emacs and vi cursor movement controls (^n, ^p, ^f, ^b, and ^j, ^k, ^h, ^l) can be used instead of the arrow keys.

SUPPLIED CONFIGURATION

A sample configuration file, system.fvwmrc, is supplied with the fvwm distribution. It is well commented and can be used as a source of examples for fvwm configuration.

USE ON MULTI-SCREEN DISPLAYS

If the -s command line argument is not given, fvwm will automatically start up on every screen on the specified display. After fvwm starts each screen is treated indepen dently. Restarts of fvwm need to be performed separately on each screen. The use of EdgeScroll 0 0 is strongly recommended for multi-screen displays.

You may need to quit on each screen to quit from the X session completely.

Stick

Makes a window sticky if it is not already sticky, or non-sticky if it is already sticky.

Scroll horizonal vertical

Scrolls the virtual desktop’s viewport by horizontal pages in the x-direction and vertical pages in the y- direction. Either or both entries may be negative. Both horizontal and vertical values are expressed in percent of pages, so “Scroll 100 100” means to scroll down and left by one full page. “Scroll 50 25” means to scroll left half a page and down a quarter of a page. The scroll function should not be called from pop-up menus. Normally, scrolling stops at the edge of the desktop.

If the horizontal and vertical percentages are multiplied by 1000 then scrolling will wrap around at the edge of the desktop. If “Scroll 100000 0” is executed over and over fvwm will move to the next desktop page on each execution and will wrap around at the edge of the desktop, so that every page is hit in turn.

If the letter “p” is appended to each coordinate (horizontal and/or vertical), then the scroll amount will be measured in pixels.

Title

Does nothing. This is used to insert a title line in a popup or menu.

TogglePage

Temporarily disables edge scrolling. Edge scrolling can be re-enabled by calling this again.

Wait name

This built-in is intended to be used in fvwm functions only. It causes execution of a function to pause until a new window name name appears. Fvwm remains fully functional during a wait. This is particularly useful in the InitFunction if you are trying to start windows on specific desktops:

```
Function “InitFunction”
Exec “I” exec xterm -geometry 80x64+0+0
Wait “I” xterm
Desk “I” 0 2
Exec “I” exec xmh -font fixed -geometry 507x750+0+0 &
Wait “I” xmh
Desk “I” 0 0
EndFunction
```

The above function starts an xterm on the current desk, waits for it to map itself, then switches to desk 2 and starts an xmh. After the xmh window appears control moves to desk 0.

```
Exec “Xman” exec xman &
Exec “Xmag” exec xmag &
Nop ““
Popup “Exit Fvwm” Quit-Verify
EndPopup
```

Sub-menus must be defined prior to the main menu in which they are bound. Sub-menu nesting can be arbitrarily deep.

Quit

Exits fvwm, generally causing X to exit too.

Raise

Allows the user to raise a window.

RaiseLower

Alternately raises and lowers a window.

Refresh

Causes all windows on the screen to redraw themselves.

Resize [x y]

Allows the user to resize a window.

If the optional arguments x and y are provided, then the window will be moved so that its upper left corner is at location (x,y). The units of x and y are percent-of-screen, unless a letter “p” is appended to each coordinate, in which case the location is specified in pixels.

Restart name WindowManagerName

Causes fvwm to restart itself if WindowManagerName is “fvwm”, or to switch to an alternate window manager if WindowManagerName is other than “fvwm”. If the window manager is not in your default search path, then you should use the full path name for WindowManagerName.

WindowManagerName is not quoted but name is. name is the name that appears in a menu, if that is where the function is called from. name is required even if the function is not called from a menu.

This command should not have a trailing ampersand or any command line arguments and should not make use of any environmental variables. Of the following examples, the first three are sure losers, but the third is OK:

```
Key F1 R N Restart “ “ fvwm &
Key F1 R N Restart “ “ $(HOME)/bin/fvwm
Key F1 R N Restart “ “ twm -f .mystartupfile
Key F1 R N Restart “ “ /home/nation/bin/fvwm
```

In the first example, an interactive move is indicated. In the second, the window whose title-bar is selected will be moved so that its upper left hand corner is 10 percent of the screen width in from the left of the screen, and 10 percent down from the top. The final example moves the window to coordinate (10,10) pixels.

Nop Does nothing. This is used to insert a blank line or separator in a menu. If the menu item specification is Nop “ “, then a blank line is inserted. If it looks like Nop ““, then a separator line is inserted.

Popup

This built-in has two purposes: to bind a menu to a key or mouse button, and to bind a sub-menu into a menu. The formats for the two purposes differ slightly.

To bind a previously defined pop-up menu to a key or mouse button:

The following example binds mouse buttons 2 and 3 to a pop-up called “Window Ops”, whose definition was provided as an example earlier in this man page. The menu will pop up if the buttons 2 or 3 are pressed in the window frame, side-bar, or title-bar, with no modifiers (none of shift, control, or meta).

```
Mouse 2 FST N Popup “Window Ops”
Mouse 3 FST N Popup “Window Ops”
```

Pop-ups can be bound to keys through the use of the key modifier. Pop-ups can be operated without using the mouse by binding to keys and operating via the up arrow, down arrow, and enter keys.

To bind a previously defined pop-up menu to another menu, for use as a sub-menu:

The following example defines a sub menu, “QuitVerify” and binds it into a main menu, called “Utilities”:

```
Popup “Quit-Verify”
Title “Really Quit Fvwm?”
Quit “Yes, Really Quit”
Restart “Restart Fvwm” fvwm
Nop ““
Nop “No, Don’t Quit”
EndPopup

Popup “Utilities”
Title “Utilities”
Exec “Xterm” exec xterm &
Exec “Rxvt” exec rxvt &
Exec “Top” exec rxvt -T Top -n Top -e top &
Exec “Calculator” exec xcalc &
```

allowed. It the optional argument is negative only de-iconification will be allowed.

Lower

Allows the user to lower a window.

Maximize [horizontal vertical]

Without its optional arguments Maximize causes the window to alternately switch from a full-screen size to its normal size.

With the optional arguments horizontal and vertical, which are expressed as percentage of a full screen, the user can control the new size of the window. If horizontal is greater than 0 then the horizontal dimension of the window will be set to horizontal*screen_width/100. The vertical resizing is similar. For example, the following will add a title-bar button to switch a window to the full vertical size of the screen:

```
Mouse 0 4 A Maximize 0 100
```

The following causes windows to be stretched to the full width:

```
Mouse 0 4 A Maximize 100 0
```

This makes a window that is half the screen size in each direction:

```
Mouse 0 4 A Maximize 50 50
```

Values larger than 100 can be used with caution.

If the letter “p” is appended to each coordinate (horizontal and/or vertical), then the scroll amount will be measured in pixels.

Module name ModuleName

Specifies a module which should be spawned. Modules can be short lived transient programs or can remain for the duration of the X session. Modules will be terminated by the window manager prior to restarts and quits, if possible. name is a double-qouted string which has absolutely no significance, but must exist.

Move [x y]

Allows the user to move a window. If called from some- where in a window or its border, then that window will be moved. If called from the root window then the user will be allowed to select the target window.

If the optional arguments x and y are provided, then the window will be moved so that its upper left corner is at location (x,y). The units of x and y are percent-of-screen, unless a letter “p” is appended to each coordinate, in which case the location is specified in pixels.

Examples:

```
Mouse 1 T A Move
Mouse 2 T A Move 10 10
Mouse 3 T A Move 10p 10p
```

If arg1 is non zero then the next desktop number will be the current desktop number plus arg1. Desktop numbers can be negative.

If arg1 is zero then the new desktop number will be arg2.

The number of active desktops is determined dynamically. Only desktops which contain windows or are currently being displayed are active. Desktop numbers must be between 2147483647 and -2147483648 (is that enough?).

Destroy

Destroys a window. Guaranteed to get rid of the window, but is a fairly violent way to terminate an application.

Exec name command

Executes command. command is not quoted but name is. name is the name that appears in a menu, if that is where the function is called from. name is required even if the function is not called from a menu.

The following example binds function key F1 in the root window, with no modifiers, to the exec function. The program rxvt will be started with an assortment of options.

```
Key F1 R N Exec “rxvt” exec rxvt -fg yellow -bg blue -e /bin/tcsh &
```

Focus

Moves the viewport or window as needed to make the selected window visible. Sets the keyboard focus to the selected window. Raises the window if needed to make it visible. Warps the pointer into the selected window in focus-follows-mouse mode. Does not de-iconify. This function is primarily for use with a module such as FvwmWinList.

Function

Used to bind a previously defined function to a key or mouse button.

The following example binds mouse button 1 to a function called “Move-or-Raise”, whose definition was provided as an example earlier in this man page. After performing this binding fvwm will execute to move-or- raise function whenever button 1 is pressed in a window title-bar.

```
Mouse 1 T A Function “Move-or-Raise”
```

GotoPage x y

Moves the desktop viewport to page (x,y). The upper left page is (0,0), the upper right is (N,0), where N is one less than the current number of horizontal pages specified in the DeskTopSize command. The lower left page is (0,M), and the lower right page is (N,M), where M is the desktop's vertical size as specified in the DeskTopSize command. The GotoPage function should not be used in a pop-up menu.

Iconify [value]

Iconifies a window if it is not already iconified or de-iconifies it if it is already iconified. If the optional argument value is positive the only iconification will be

Beep

Makes the computer beep.

CirculateDown [name window_name]

Causes the pointer to move to the next window in the list of windows for which CirculateSkip has not not been specified.

If the optional arguments are supplied then the focus will move to the first window whose name (or icon name or class) matches window_name. The optional argument name is required if window_name is supplied and is enclosed in quotes. This argument is the name which appears in menus if the function is called from a menu, but serves no purpose if the function is not called from a menu.

CirculateUp [name window_name]

Causes the pointer to move to the previous window in the list of windows for which CirculateSkip has not not been specified.

If the optional arguments are supplied then the focus will move to the first window whose name (or icon name or class) matches window_name. The optional argument name is required if window_name is supplied and is enclosed in quotes. This argument is the name which appears in menus if the function is called from a menu, but serves no purpose if the function is not called from a menu

Here's an example that move the focus to an xterm window when Alt-F1 is pressed:

```
Key F1 A M CirculateUp “whatever” xterm
```

Close

If the window accepts the delete window protocol a message is sent to the window asking it to gracefully remove itself. If the window does not understand the delete window protocol then the window is destroyed.

CursorMove horizonal vertical

Moves the mouse pointer by horizontal pages in the X direction and vertical pages in the Y direction. Either or both entries may be negative. Both horizontal and vertical values are expressed in percent of pages, so “CursorMove 100 100” means to move down and left by one full page. “CursorMove 50 25” means to move left half a page and down a quarter of a page. The CursorMove function should not be called from pop-up menus.

Delete

Sends a message to a window asking that it remove itself, frequently causing the application to exit.

Desk arg1 arg2

Changes to another desktop (workspace, room).

```
Style “xman” Icon xman.xpm
Style “matlab” Icon math4.xpm, StartsOnDesk 3
Style “xmag” Icon magnifying_glass2.xpm
Style “xgraph” Icon graphs.xpm
Style “GoodStuff” Icon toolbox.xpm
Style “Maker” StartsOnDesk 1
Style “signal” StartsOnDesk 3
```

Note that all properties for a window will be OR’ed together. In the above example “FvwmPager” gets the property StaysOnTop via an exact window name match but also gets NoHandles, Sticky, and WindowListSkip by a match to “Fvwm*”. It will get NoTitle by virtue of a match to “*”. If conflicting styles are specified for a window, then the last style specified will be used.

If the NoIcon attribute is set then the specified window will simply disappear when it is iconified. The window can be recovered through the window-list. If Icon is set without an argument then the NoIcon attrbute is cleared but no icon is specified. An example which allows only the FvwmPager module icon to exist:

```
Style “*” NoIcon
Style “Fvwm Pager” Icon
```

SuppressIcons

Prevents icon windows from being created or drawn. When used with the window-list this provides a sort of icon manager.

The Style command provides another (more general) method for specifying SuppressIcons.

WindowFont fontname

Makes fvwm use font fontname instead of “fixed” for the window title bar.

WindowListSkip windowname

Causes windows with the indicated name to be left out of the window list.

Windowname can contain the wildcards “*” and “?” which match window names in the normal Unix filename matching manner. Actual “*”, “?”, and “\” characters in a window name can be entered by preceding the character with a “\”.

The Style command provides another (more general) method for specifying WindowListSkip.

XORvalue number

Changes the value with which bits are XOR’ed when doing rubber-band window moving or resizing. Setting this value is a trial-and-error process.

BUILT IN FUNCTIONS

Fvwm supports a set of built-in functions which can be bound to keyboard or mouse buttons:

StaysPut, Sticky/Slippery, StartIconic/StartNormal, Color, ForeColor, BackColor, StartsOnDesk/StartsAnyWhere, IconTitle/NoIconTitle, and NoButton/Button.

In the above list some options are listed as style- option/opposite-style-option. The opposite-style-option for entries that have them describes the fvwm default behavior and can be used if you want to change the fvwm default behavior.

Icon takes an (optional) unquoted string argument which is the icon bitmap or pixmap to use.

StartsOnDesk takes a numeric argument which is the desktop number on which the window should be initially placed.

BorderWidth takes a numeric argument which is the width of the border to place the window if it does not have resize-handles.

HandleWidth takes a numeric argument which is the width of the border to place the window if it does have resize-handles.

Button and NoButton take a numeric argument which is the number of the title-bar button which is to be included/omitted.

Color takes two arguments. The first is the window- label text color and the second is the window decoration’s normal background color. The two colors are separated with a slash. If the use of a slash causes problems then the seperate ForeColor and BackColor options can be used.

An example:

```
# Change default fvwm behavior to no title-bars on windows!
# Also define a default icon.
Style “*” NoTitle,Icon unknown1.xpm, BorderWidth 4,HandleWidth 5
# now, window specific changes:
Style “Fvwm*” NoHandles,Sticky,WindowListSkip,BorderWidth 0
Style “Fvwm Pager” StaysOnTop, BorderWidth 0
Style “*lock” NoHandles,Sticky,StaysOnTop,WindowListSkip
Style “xbiff” Sticky, WindowListSkip
Style “GoodStuff” NoHandles,Sticky,WindowListSkip
Style “sxpm” NoHandles
Style “makerkit”
# Put title-bars back on xterms only!
Style “xterm” Title, Color black/grey
Style “rxvt” Icon term.xpm
Style “xterm” Icon rterm.xpm
Style “xcalc” Icon xcalc.xpm
Style “xbiff” Icon mail1.xpm
Style “xmh” Icon mail1.xpm, StartsOnDesk 2
```

StickyForeColor colorname

Sets the foreground color for non-selected sticky window titles to colorname. When using a monochrome screen this option is ignored and black is used. Only available if -DMORE_COLORS is used when compiling.

Sticky windowname

Sticky windows "stick to the screen's glass." That is, they don't move the the viewport into the virtual desk- top changes. windowname can be a window's name or its class.

Windowname can contain the wildcards "*" and "?" which match window names in the normal Unix filename matching manner. Actual "*", "?", and "\" characters in a window name can be entered by preceding the character with a "\".

The Style command provides another (more general) method for specifying Sticky.

StickyIcons

Causes icons to always stick to the screen's glass. That is, icons always follow you around the desktop. When a window is de-iconified it gets un-stuck. Some people find this a useful way of moving windows around.

StubbornIcons

Changes de-iconification behavior a bit. Instead of having windows always de-iconify themselves on the current page they de-iconify into their original position.

StubbornIconPlacement

When used with IconBoxes, causes icons to avoid placing themselves underneath existing windows.

StubbornPlacement

When using SmartPlacement, causes new windows to avoid placing themselves over icons.

Style windowname options

This command is intended to replace the commands NoBorder, NoTitle, StartsOnDesk, Sticky, StaysOnTop, Icon, WindowListSkip, CirculateSkip, SuppressIcons, BoundaryWidth, NoBoundaryWidth, StdForeColor, and StdBackColor with a single flexible and comprehensive command. This command is used to set attributes of a window to values other than the default or to set the window manager default styles.

windowname can be a window's name, class, or resource string. It can contain the wildcards * and/or ?, which are matched in the usual Unix filename manner.

options is a comma separated list containing some or all of the keywords BorderWidth, HandleWidth,NoIcon/Icon, NoTitle/Title, NoHandles/Handles, WindowListSkip/WindowListHit, CirculateSkip/CirculateHit, StaysOnTop/

SmartPlacement

Causes windows which would normally require user placement to be automatically placed in a smart location - a location in which they do not overlap any other windows on the screen. If no such position can be found user placement or random placement will be used as a fall-back method. For the best of all possible worlds use both RandomPlacement and SmartPlacement.

StartsOnDesk windowname desk-number

This command causes windows whose name or class is windowname to be initially placed on desktop number desk-number. windowname should be enclosed in double quotes. If the window requires interactive placement, an outline will be displayed on the current desk but the window will appear on the specified desk.

Windowname can contain the wildcards "*" and "?" which match window names in the normal Unix filename matching manner. Actual "*", "?", and "\" characters in a window name can be entered by preceding the character with a "\".

The Style command provides another (more general) method for specifying StartsOnDesk.

StaysOnTop windowname

These windows always try to stay on top of the other windows. This might be handy for clocks or mailboxes that you would always like to be visible. If the window is explicitly lowered it will not try to force its way back to the top until it is explicitly raised. windowname can be a window's name or its class.

Windowname can contain the wildcards "*" and "?" which match window names in the normal Unix filename matching manner. Actual "*", "?", and "\" characters in a window name can be entered by preceding the character with a "\".

The Style command provides another (more general) method for specifying StaysOnTop.

StdBackColor colorname

Sets the background color for menus and non-selected windows to colorname. When using a monochrome screen this option is ignored and white is used.

The Style command provides another (more general) method for specifying StdBackColor.

StdForeColor colorname

Sets the foreground color for menus and non-selected window titles to colorname. When using a monochrome screen this option is ignored and black is used.

The Style command provides another (more general) method for specifying StdForeColor.

StickyBackColor colorname

Sets the background color for non-selected sticky windows to colorname. When using a monochrome screen this option is ignored and white is used. Only available if -DMORE_COLORS is used when compiling.

```
EndMenu
```

Note that if a tab character is embedded in the caption of a menu entry then the text following the tab will be entered into a second column in the menu and the entire menu will be left-adjusted. This is intended for shortcut labeling. The tab character must really be a tab. If it is expanded into spaces it will not work!

For example:

```
Popup “Window Ops”
Title “Window Ops Alt-F1”
.
.
.
```

Is the start of a left adjusted menu. Alt-F1 will be placed toward the right side of the menu.

Shortcut keys may be specified in the menu definition by preceding the character with an ampersand. The ampersand will not be displayed but the character after it will be displayed underlined, and if the user presses the corresponding key then that item will be activated as if the user had clicked on it with the mouse. Only alphabetic and numeric characters may be used as shortcut keys. The shift state of the keyboard is ignored when testing shortcut characters. For example:

Popup “Window Ops”

```
Maximize “Ma&ximise” 100 100
EndMenu
```

When this menu is popped up the ‘x’ will be underlined and pressing the ‘x’ key will cause the current window to be maximized. Shortcut keys are not operative unless MENU_HOTKEYS was defined when building fvwm. If WINDOWLIST_HOTKEYS was also defined then hot keys are automatically added to the WindowList when it is displayed.

RandomPlacement

Causes windows which would normally require user placement to be automatically placed in ever-so-slightly random locations. For the best of all possible worlds use both RandomPlacement and SmartPlacement.

SaveUnders

Causes the fvwm decoration frames to request save-unders. This can significantly improve the performance during opaque moves but it causes a significant increase in memory usage.

SloppyFocus

This focusing mode is like focus-follows-mouse (the default) except that the focus will not be removed from a window until your mouse enters a new window. Exiting a window to enter the root window will leave the focus unchanged.

chrome screen this option is ignored. If the NO_PAGER option is set when building fvwm this option is unavailable.

PagerBackColor colorname

Causes the pager background color to be colorname instead of white. On a monochrome screen this option is ignored. If the NO_PAGER option is set when building fvwm this option is unavailable.

PagerFont fontname

Makes fvwm use font fontname for writing window icon names in the window’s representation in the pager. If this option is omitted no names are written in the pager windows.

PagingDefault pagingdefaultvalue

Tells fvwm if it should start up with paging enabled or disabled. “PagingDefault 0” will start fvwm with paging disabled; “PagingDefault 1” will start fvwm with paging enabled by default.

PixmapPath path

Specifies a colon separated list of full path names of directories where pixmap (color) icons can be found. Each path should start with a slash.

Popup PopupName

Starts the definition of a pop-up menu which will later be bound to a mouse button or key. PopupName must be enclosed in quotes. Menu entries are included on lines following the Popup keyword. The menu definition ends with the key word EndPopup. Menu entries are specified as shown in the following example. The first word on each line is the built-in function which will be performed, followed by the caption (enclosed in quotes) which will be shown in the menu, followed by any additional arguments needed by the built-in function. Sub- menus can be specified by using the Popup built-in as long as the sub-menu was defined earlier in the configuration file.

```
Popup “Window Ops”
    Title “Window Ops”
    Move “Move”
    Resize “Resize”
    Raise “Raise”
    Lower “Lower”
    Iconify “(De)Iconify”
    Nop “ “
    Destroy “Destroy”
    Title “HARDCOPY”
    Exec “Hardcopy” exec xdpr &
    Exec “Hardcopy RV” exec xdpr -rv &
```

The Style command provides another (more general) method for specifying NoBoundaryWidth.

NoPPosition

Instructs fvwm to ignore the PPosition field when adding new windows. Adherence to the PPosition field is required for some applications, but if you don't have one of those its a real headache.

NoTitle windowname

Keeps fvwm from putting a title-bar in the decorations for windows named windowname. This is handy for clocks and similar gadgets that you don't want to take up too much space. windowname can be a window's name or its class.

Windowname can contain the wildcards "*" and "?" which match window names in the normal Unix filename matching manner. Actual "*", "?", and "\" characters in a window name can be entered by preceding the character with a "\".

The Style command provides another (more general) method for specifying NoTitle.

OpaqueMove percentage

Tells fvwm the maximum size window with which opaque window movement should be used. The percentage is percent of the total screen area. With "OpaqueMove 0" all windows will be moved using the traditional rubber-band outline. With "OpaqueMove 100" all windows will be move as solid windows. The default is "OpaqueMove 5", which allows small windows to be moved in an opaque manner but large windows are moved as rubber-bands.

OpaqueResize

Causes resize operations to be done with the window itself instead of an outline.

Pager X_Location Y_Location

Enables a paging style of moving across the desktop. A Pager window (not a popup) will appear at (X_Location, Y_Location). Miniature versions of all the non-sticky windows on the virtual desktop are shown in the pager. The color of the miniature version is the same as the color of the full-size window's border.

In the Pager window, pressing mouse button 1 will move the desktop viewport to the selected page (in click-to-focus mode; it will also move the keyboard focus to the window whose miniature you click on). Pressing button 2 on a window in the pager will begin a window move, using the miniature to quickly move the window anywhere on the desktop. Pressing button 3 will move the top-left corner of the viewport to the location of the button press, even if it does not line up with a page. Dragging button 3 will cause the selected viewport to scroll as you move the pointer. The Pager is automatically sticky but does not automatically stay on top.

PagerForeColor colorname

Causes the pager foreground color to be colorname instead of black. This is the color used to highlight the current viewport in the pager window. On a mono-

MWMFunctionHints

Causes fvwm to read the MOTIF_WM_HINTS atom from application windows and to parse and attempt to replicate the Motif behavior with regard to allowed window functions. Unlike mwm, which simply removes prohibited functions from the window's menus, fvwm simply shades out the prohibited functions. Also, because fvwm implements some functions in user defined macros that mwm implements internally, the mapping of prohibited functions is partially based on the menu item label.

MWMHintOverride

If MWMFunctionHints is used then maximization and iconfication are prohibited for transients. Also, windows can specify that the window manager should not destroy or delete them. Since these MWM rules are kind of stupid, especially with regard to the transient windows, I provide this MWMHintOverride option. When it is used menu items will be shaded out if MWM would prohibit their use, but the user can go ahead and select that item and it will operate as expected.

The override should be used cautiously because some applications will break if you override their mwm hints.

MWMMenus

Substitutes MWM look and feel menus in place of the standard fvwm versions. This option also triggers a few other mwm-style options, such as centering the size/resize window on the screen, instead of leaving it in the upper left, and switches the resize-on-initial-placement trigger action to shift-button-1 instead of the twm style press-button-2

NoBorder windowname

Keeps fvwm from putting decorative borders on windows named windowname. This command has no effect on the title-bar. This is handy for clocks and similar gadgets that you don't want to take up too much space. windowname can be a window's name or its class.

If you specify both NoBorder windowname and NoTitle windowname for the same window in your .fvwmrc file the window will be completely undecorated.

Windowname can contain the wildcards "*" and "?" which match window names in the normal Unix filename matching manner. Actual "*", "?", and "\" characters in a window name can be entered by preceding the character with a "\".

The Style command provides another (more general) method for specifying NoBorder.

NoBoundaryWidth Width

Changes the width of the decorations for windows with no titles and no borders. The default is 1. Any positive or zero value is acceptable. Decorations for these undecorated windows have the same context as the side-bars on normally decorated windows.

have their own man pages. Module can also be used as a built-in. Modules can be short lived transient programs or, like GoodStuff, can remain for the duration of the X session. Modules will be terminated by the window manager prior to restarts and quits, if possible. See the introductory section on modules.

ModulePath

Specifies a colon separated list of paths for fvwm to search when looking for a module to load. Individual directories do not need trailing slashes.

Mouse Button Context Modifiers Function

Defines a mouse binding. Button is the mouse button number. If Button is zero then any button will perform the specified function. Context describes where the binding applies. Valid contexts are R for the root window, W for an application window, T for a window title bar, S for a window side, top, or bottom bar, F for a window frame (the corners), I for an Icon window, or 0 through 9 for title-bar buttons, or any combination of these letters. A is for any context except for title- bar buttons. For instance, a context of FST will apply when the mouse is anywhere in a window’s border except the title-bar buttons.

Modifiers is any combination of N for no modifiers, C for control, S for shift, M for Meta, or A for any modifier. For example, a modifier of SM will apply when both the Meta and shift keys are down. X11 modifiers mod1 through mod5 are represented as the digits 1 through 5.

Function is one of fvwm’s built-in functions.

The title bar buttons are numbered with odd numbered buttons on the left side of the title bar and even numbers on the right. Smaller-numbered buttons are displayed toward the outside of the window while larger-numbered buttons appear toward the middle of the window (0 is short for 10). In summary, the buttons are numbered:

1 3 5 7 9 0 8 6 4 2

The highest odd numbered button which has an action bound to it determines the number of buttons drawn on the left side of the title bar. The highest even number determines the number or right side buttons which are drawn. Actions can be bound to either mouse buttons or keyboard keys.

MWMBorders

Substitutes MWM style 1 pixel wide relief lines instead of fvwm’s 2 pixel borders.

MWMButtons

Disables button press feedback for all decorations except the title bar and title-bar buttons, as in MWM.

MWMDecorHints

Causes fvwm to read the MOTIF_WM_HINTS atom from application windows and to parse and attempt to replicate the Motif behavior with regard to window decorations. Note that mwm allows function hints to affect window decorations but these effects are not replicated by this option.

IconFont fontname

Makes fvwm use font fontname for icon labels. If omitted, the menu font (specified by the Font configuration parameter) will be used instead.

IconPath path

Specifies a colon separated list of full path names of directories where bitmap (monochrome) icons can be found. Each path should start with a slash. Note: if the M4 patches are included when fvwm is built, then m4 will want to mangle the word "include" which will frequently show up in the IconPath or PixmapPath command. To fix this add undefine('include') prior to the IconPath command.

Key keyname Context Modifiers Function

Binds a keyboard key to a specified fvwm built-in function. Definition is the same as for a mouse binding except that the mouse button number is replaced with a key name. The keyname is one of the entries from /usr/include/X11/keysymdef.h, with the leading XK_ omitted. The Context and Modifiers fields are defined as in the mouse binding.

Binding a key to a title-bar button will not cause that button to appear unless a mouse binding also exists.

Lenience

The ICCCM states that if an application sets the input field of the wm_hints structure to False, then it never wants the window manager to give it the input focus. The only application that I know of which needs this is sxpm, and that is a silly bug with a trivial fix and has no overall effect on the program anyway. Rumor is that some older applications have problems too.

If this parameter is set then fvwm will ignore this ICCCM convention.

MenuBackColor colorname

Sets the menu background color. When using monochrome this option is ignored. This option is only available if fvwm is compiled with MENUCOLOR defined.

MenuForeColor colorname

Sets the menu foreground color. When using monochrome this option is ignored. This option is only available if fvwm is compiled with MENUCOLOR defined.

MenuStippleColor colorname

Sets the color for shaded out entries in menus (for functions which are not allowed on the currently selected window). When using monochrome this option is ignored and a stipple pattern is used. This option is only available if fvwm is compiled with MENUCOLOR defined.

Module ModuleName

Specifies a module which should be spawned during initialization. At the current time the available modules are FvwmAudio, FvwmBacker, FvwmBanner, FvwmClean, FvwmDebug, FvwmIconBox, FvwmIdent, FvwmPager, FvwmSave, FvwmSaveDesk, FvwmScroll, FvwmWinList, and GoodStuff. These modules

```
EndFunction
```

The clicking and double clicking concepts do not carry through to using keyboard shortcuts.

Two special functions exist: InitFunction and RestartFunction. The InitFunction will be called when fvwm is started for the first time in any X session and can be used to start modules, set background patterns, and begin programs. The restart function will be called when fvwm is restarted. It can be used to start modules and set background patterns but probably should not be used to start programs.

HiBackColor colorname

Sets the background color of the selected window to colorname. When using a monochrome screen this option is ignored and white is used.

HiForeColor colorname

Sets the color of the selected window's title to color-name. When using a monochrome screen this option is ignored and black is used.

Icon windowname bitmap-file

Specifies the bitmap to be used for a window when it is iconified. The windowname can be an application's window name or class name and must be enclosed in quotes. The bitmap-file is either the full path name to a standard X11 bitmap file or a file in the IconPath or PixmapPath. The specified bitmap/pixmap is used in preference to any icon supplied by the window itself.

If fvwm is compiled with XPM support for color icons then bitmap can be an XPM pixmap file.

windowname should be enclosed in double quotes but bitmap-file should not. Environment variables should not be used in the bitmap-file specification.

If windowname is an empty string then the specified file is the default icon, and will be used if no other icon bitmap or pixmap can be found:

```
 Icon "" my-favorite-icon
```

The Style command provides another (more general) method for specifying Icon.

IconBox left top right bottom

Defines regions of the screen in which to place icons. Up to four icon boxes can be defined. If an IconBox line is provided then icons will automatically be placed in them, if possible. Each time a window is iconified a new place is found for it. Icon boxes are searched for space going left to right, then top to bottom. Icons will not be auto-placed on top of other icons but they may be placed underneath application windows. If left or right is negative, then fvwm will add the screen width to it. If top or bottom is negative, then fvwm will add the screen height to it. NOTE: -0 is not parsed as the right or bottom pixel on the screen. You have to use -1 instead.

If no IconBox line is provided or all icon boxes are full, then fvwm will place icons near the current pointer location.

EdgeScroll horizontal vertical

Specifies the percentage of a page to scroll when the cursor hits the edge of a page. If you don't want any paging or scrolling when you hit the edge of a page include "EdgeScroll 0 0" in your .fvwmrc file. If you want whole pages, use "EdgeScroll 100 100". Both horizontal and vertical should be positive numbers.

If the horizontal and vertical percentages are multiplied by 1000 then scrolling will wrap around at the edge of the desktop. If "EdgeScroll 100000 100000" is used fvwm will scroll by whole pages, wrapping around at the edge of the desktop.

Font fontname

Makes fvwm use font fontname instead of "fixed" for menus, the resize indicators, and icon labels (if Icon- Font is not specified).

Function FunctionName

Starts the definition of a complex function, composed of the fvwm built-in functions, which will later be bound to a mouse button or key. FunctionName must be enclosed in quotes. Function entries are included on lines following the Function keyword. The definition ends with the key word EndFunction. Function entries are specified as shown in the following example. The first word on each line is the built-in function which will be performed, followed the type of event which should trigger the action (enclosed in quotes), followed by any additional arguments needed by the built- in function. Menus can be specified by using the Popup built-in as long as the menu was defined earlier in the configuration file.

The trigger actions which are recognized are Immediate, Motion, Click, and DoubleClick. Immediate actions are executed as soon as the function is activated, even if a window has not been selected. If there are actions other than immediate ones, fvwm will wait to see if the user is clicking, double-clicking, or dragging the mouse. After the decision is made, fvwm will execute only the built-ins from the function definition whose trigger action matches the action performed by the user.

If the following example were bound to button 1 in a window title-bar, then, when button 1 is pressed, fvwm would wait 150 msec to see if the button is released. If the button is not released fvwm will start a move operation. When the move operation is complete a raise operation will be performed. If a button release is detected then fvwm will wait another 150 msec for a second click. If only one click is detected then the window will be raised. If two clicks are detected the window will be alternately raised and lowered. The 150 msec wait duration can be altered using the ClickTime option.

Function "Move-or-Raise"

```
    Move "Motion"
    Raise "Motion"
    Raise "Click"
    RaiseLower "DoubleClick"
```

128 target 130 tcross
132 top_left_arrow 134 top_left_corner
136 top_right_corner 138 top_side
140 top_tee 142 trek
144 ul_angle 146 umbrella
148 ur_angle 150 watch
152 xterm

DecorateTransients

Causes transient windows, which are normally left undecorated, to be given the usual fvwm decorations. Note that some pop-up windows, such as the xterm menus, are not managed by the window manager and still do not receive decorations.

DeskTopScale Scale

Defines the virtual desktop scale with respect to the screen.

DeskTopSize HorizontalxVertical

Defines the virtual desktop size in units of the physical screen size.

DontMoveOff

Prevents windows from being moved off or initially placed off of the desktop. A few programs will not work correctly if you use this option. This only keeps windows from being completely lost off the edge of the desktop. It insists on keeping 16 pixels on the desktop but doesn't care a bit about keeping the whole window on the desk. See EdgeResistance if you don't like having windows partially off the screen.

EdgeResistance scrolling moving

Tells how hard it should be to change the desktop viewport by moving the mouse over the edge of the screen and how hard it should be to move a window over the edge of the screen.

The first parameter tells how milliseconds the pointer must spend on the screen edge before fvwm will move the viewport. This is intended for people who use "EdgeScroll 100 100" but find themselves accidentally flipping pages when they don't want to.

The second parameter tells how many pixels over the edge of the screen a window's edge must move before it actually moves partially off the screen.

Note that, with "EdgeScroll 0 0", it is still possible to move or resize windows across the edge of the current screen. By making the first parameter to EdgeResistance 10000 this type of motion is impossible. With EdgeResistance less than 10000 but greater than 0 moving over pages becomes difficult but not impossible.

The cursor_type argument is a number which tells the cursor shape to use. The available numbers can be found in /usr/include/X11/cursorfont.h and are currently even numbers between 0 and 152. At the current time, the following cursor types are available:

0 X_cursor 2 arrow
4 based_arrow_down 6 based_arrow_up
8 boat 10 bogosity
12 bottom_left_corner 14 bottom_right_corner
16 bottom_side 18 bottom_tee
20 box_spiral 22 center_ptr
24 circle 26 clock
28 coffee_mug 30 cross
32 cross_reverse 34 crosshair
36 diamond_cross 38 dot
40 dotbox 42 double_arrow
44 draft_large 46 draft_small
48 draped_box 50 exchange
52 fleur 54 gobbler
56 gumby 58 hand1
60 hand2 62 heart
64 icon 66 iron_cross
68 left_ptr 70 left_side
72 left_tee 74 leftbutton
76 ll_angle 78 lr_angle
80 man 82 middlebutton
84 mouse 86 pencil
88 pirate 90 plus
92 question_arrow 94 right_ptr
96 right_side 98 right_tee
100 rightbutton 102 rtl_logo
104 sailboat 106 sb_down_arrow
108 sb_h_double_arrow 110 sb_left_arrow
112 sb_right_arrow 114 sb_up_arrow
116 sb_v_double_arrow 118 shuttle
120 sizing 122 spider
124 spraycan 126 star

CirculateSkip windowname

Causes windows with the indicated name to be skipped over when the circulate-up or circulate-down functions are invoked. windowname can be a window's name or its class.

The Style command provides another (more general) method for specifying CirculateSkip.

CirculateSkipIcons

Causes circulate and warp operations to skip over iconified windows.

ClickTime delay

Specifies the maximum delay (in milliseconds) between a button press and a button release for the Function built-in to consider the action a mouse click. The default delay is 150 milliseconds.

ClickToFocus

Normally keyboard input goes to the window the mouse pointer is in. If this option is set the keyboard input stays with one window until the mouse is clicked with the pointer positioned in a new window.

Cursor cursor_num cursor_type

This provides a very awkward way of changing cursor styles. Cursor_num tells which cursor you are changing, and is a number between 0 and 12, as follows:

0 POSITION - used when initially placing windows.

1 TITLE - used in a window title-bar.

2 DEFAULT - used in windows that don't set their cursor.

3 SYS - used in one of the title-bar buttons.

4 MOVE - used when moving or resizing windows.

5 WAIT - used during an EXEC builtin command.

6 MENU - used in menus.

7 SELECT - used for various builtin commands such as iconify.

8 DESTROY - used for DESTROY and DELETE built-ins.

9 TOP - used in the top side-bar of a window.

10 RIGHT - used in the right side-bar of a window.

11 BOTTOM - used in the bottom side-bar of a window.

12 LEFT - used in the left side-bar of a window.

13 TOP_LEFT - used in the top left corner of a window.

14 TOP_RIGHT - used in the top right corner of a window.

15 BOTTOM_LEFT - used in the bottom left corner of a window.

16 BOTTOM_RIGHT - used in the bottom right corner of a window.

options. Lines beginning with ‘#’ will be ignored by fvwm. Lines starting with ‘*’ are expected to contain module configuration commands (rather than configuration commands for fvwm itself).

AppsBackingStore

Causes application windows to request backing store. This option compromises the ICCCM compliance of the window manager. While this option can speed things up in an X-terminal, where redraws of windows are expensive, it may not help much on regular workstations.

AutoRaise delay

Enables auto-raising of windows and specifies the time delay (in milliseconds) between when a window acquires the input focus and when it is automatically raised. This option works in focus-follows-mouse mode, and in click-to-focus mode if the focus is changed by clicking in the application window instead of a decoration window. In click-to-focus mode, you can suppress the raise-on-focus behavior by specifying a negative delay value.

BackingStore

Causes fvwm decorations to request backing store. See the discussion on AppsBackingStore.

BoundaryWidth Width

Changes the boundary width on decorated windows to the specified value. The default is 6 pixels.

The Style command provides another (more general) method for specifying BoundaryWidth.

ButtonStyle button# WidthxHeight

Defines the rectangular decoration shape to be used in a title-bar button. button# is the title-bar button number, and is between 0 and 9. A description of title-bar button numbers is given in the Mouse section below. Width is the percentage of the full button width which is to be used. Height is the percentage of the full height to be used. Negative numbers cause the shading to be inverted.

And that’s not all! If you use a line like:

```
 ButtonStyle : 2 4 50x30@1 70x70@0 30x70@0 50x30@1
```

then the button 2 decoration will use a 4-point pattern consisting of a line from (x=50,y=30) to (70,70) in the shadow color (@0), and then to (30,70) in the shadow color, and finally to (50,30) in the highlight color (@1). Is that too confusing? See the sample system.fvwmrc.

CenterOnCirculate

When circulating, the desktop page containing the window which the pointer is moving to is automatically selected. If CenterOnCirculate is selected then fvwm will do its best to center the target window in the desktop viewport, rather than just lining up to the closest page.

-f config_file

Causes fvwm to use config_file in the user's home directory instead of .fvwmrc as the window manager configuration file.

-debug

Puts X transactions in synchronous mode, which dramatically slows things down, but guarantees that fvwm's internal error messages are correct.

-d displayname

Manage the display called "displayname" instead of the name obtained from the environment variable $DISPLAY.

-s

On a multi-screen display, run fvwm only on the screen named in the $DISPLAY environment variable or provided through the -d option. Normally, fvwm will attempt to start up on all screens of a multi-screen display.

-version

Print the version of fvwm to stderr.

The following options are available only if fvwm is compiled with the M4 option.

-no-m4

Do not use m4 to preprocess the .fvwmrc. The default is to preprocess the startup file using m4(1).

-m4-prefix

If GNU m4 is available, cause m4 to prefix all builtin commands with m4_.

-m4opt option

Pass this option to m4. The option can be any string of characters without spaces. This option can occur multiple times. If GNU m4 is available, DO NOT pass the -P option here. Use -m4-prefix instead.

-m4-squote string

Use this given string as the starting quote characters. You must also specify -m4-equote.

-m4-equote string

Use this given string as the ending quote characters. You must also specify -m4-squote.

-m4prog path

Use path as the location of the desired m4 processor. By default, m4prog is set to "m4" which must exist somewhere on the user's path. This option allows the user to explicitly choose the version of m4 to use.

CONFIGURATION FILES

The configuration file is used to describe mouse and button bindings, colors, the virtual display size, and related items. This section describes the configuration

OPTIONS This is a string of compile time options used. Each option is separated from the other by a space.

PLANES The number of bit planes your display supports in the default root window.

RELEASE The release number of your X server. For MIT X11R5 this is 5.

REVISION The X minor protocol revision. As seen by ProtocolRevision(3).

SERVERHOST This variable is set to the name of the machine that is running the X server.

TWM_TYPE Tells which twm offshoot is running. It will always be set to the string "fvwm" in this program. This is useful for protecting parts of your .twmrc file that fvwm proper won't understand (like WorkSpaces) so that it is still usable with other twm programs.

USER The name of the user running the program. Obtained from the environment.

VENDOR The vendor of your X server. For example: MIT X Consortium.

VERSION The X major protocol version. As seen by ProtocolVersion(3).

WIDTH The width of your display in pixels.

X_RESOLUTION The X resolution of your display in pixels per meter.

Y_RESOLUTION The Y resolution of your display in pixels per meter.

You may well find that if you research the m4(1) manual well and understand the power of m4, this will be a very useful and powerful tool. But if you use any of the symbols which are predefined by m4, you are in severe danger! For example, Sun's m4 predefines include, so if you use that name in your .fvwmrc, you are out of luck. The correct solution to this problem is to put a set of quotes around the troublesome word: 'include'.

To help alleviate this problem, the following options may be useful. To change the quoting characters used by m4, use the options -m4-squote and -m4-equote. Be sure to specify both options otherwise m4 will be confused. When these are given, a changequote macro is given before the users fvwmrc file is processed.

NOTE: Some versions of m4 are broken with respect to changing quoting characters and included files. When the quoting strings are longer than one character, the macro "include(<<file>>)", where "<<" and ">>" are the quoting characters, contains extra characters around the contents of the included file. This will confuse fvwm. SunOS 4.1.3 is known to have this problem.

If you are using GNU m4 an additional option is available.

By specifying -m4-prefix when starting fvwm, m4 is instructed to prefix all builtin macros with m4_. Thus, include becomes m4_include.

The availability of the m4 preprocessing is subject to the compilation define M4.

OPTIONS

These are the command line options that are recoginzed by fvwm:

mood use -m4opt -DSunny; for a dark mood use -m4opt -DDark. Your .fvwmrc file might then contain:

```
 ifdef(‘Sunny’,‘
 StdForeColor Black
 StdBackColor LightSkyBlue
 HiForeColor yellow
 HiBackColor PeachPuff1
 PagerBackColor BlanchedAlmond ’)

 ifdef(‘Dark’,‘
 StdForeColor Black
 StdBackColor #60a0c0
 HiForeColor black
 HiBackColor #c06077
 PagerBackColor #5c54c0
 PagerForeColor orchid
 StickyForeColor Black
 StickyBackColor #60c0a0 ’)
```

The following m4 symbols are predefined by fvwm:

BITS_PER_RGB The number of significant bits in an RGB color. (log base 2 of the number of distinct colors that can be created. This is often different from the number of colors that can be displayed at once.)

CLASS Your visual class. Will return one of StaticGray, GrayScale, StaticColor, PseudoColor, TrueColor, DirectColor, or, if it cannot determine what you have, NonStandard.

CLIENTHOST The machine that is running the clients.

COLOR This will be either ‘Yes’ or ‘No’. This is just a wrapper around the CLASS definition. Returns ‘Yes’ on *Color and ‘No’ on StaticGray and GrayScale.

FVWMDIR This is set to the path where the modules were configured to be installed.

FVWM_VERSION This is a string containing the version of fvwm.

HEIGHT The height of your display in pixels.

HOME The user’s home directory. Obtained from the environment.

HOSTNAME The canonical hostname running the clients (ie. a fully-qualified version of CLIENTHOST).

ICONS

The basic Fvwm configuration uses monochrome bitmap icons, similar to twm. If XPM extensions are compiled in, then color icons similar to ctwm, MS-Windows, or the Macintosh icons can be used. In order to use these options you will need the XPM package, as described in the Makefile.noImake and the Imakefile.

If both the SHAPE and XPM options are compiled in you will get shaped color icons, which are very spiffy.

MODULES

A module is a separate program which runs as a separate Unix process but transmits commands to fvwm to execute. Future releases are expected to provide a means for these modules to extract window information from fvwm. Users can write their own modules to do any weird or bizarre manipulations without affecting the integrity of fvwm itself.

Modules MUST be spawned by fvwm so that it can set up two pipes for fvwm and the module to communicate with. The pipes will already be open for the module when it starts and the file descriptors for the pipes are provided as command line arguments.

Modules can be spawned during fvwm initialization via the Module option, or at any time during the X session by use of the Module built-in. Modules can exist for the duration of the X session, or can perform a single task and exit. If the module is still active when fvwm is told to quit, then fvwm will close the communication pipes and wait to receive a SIGCHLD from the module, indicating that it has detected the pipe closure and has exited. If modules fail to detect the pipe closure fvwm will exit after approximately 30 seconds anyway. The number of simultaneously executing modules is limited by the operating system’s maximum number of simultaneously open files, usually between 60 and 256.

Modules simply transmit text commands to the fvwm built-in command engine. Text commands are formatted just as in the case of a mouse binding in the .fvwmrc setup file. Certain auxiliary information is also transmitted, as in the sample module GoodStuff. The GoodStuff module is documented in its own man page.

ICCCM COMPLIANCE

Fvwm attempts to be ICCCM 1.1 compliant. As of this (1.20l) colormap handling is not completely ICCCM compliant. In addition, ICCCM states that it should be possible for applications to receive ANY keystroke, which is not consistent with the keyboard shortcut approach used in fvwm and most other window managers.

M4 PREPROCESSING

If fvwm is compiled with the M4 option, fvwm uses m4(1) to preprocess its setup files before parsing. This way you can use m4 macros to perform operations at runtime. This makes it very easy to work with different displays with different characteristics.

For example, depending on your mood, you might want different color schemes. One way of doing this is by using the -m4opt to specify your mood. For a sunny desktop, but off the screen. For example, if the visible screen is 1000 by 1000 pixels, and the desktop size is 3x3, and the current viewport is at the upper left hand corner of the desktop, then invoking:

```
xterm -geometry +1000+1000
```

will place the window just off of the lower right hand corner of the screen. It can be found by moving the mouse to the lower right hand corner of the screen and waiting for it to scroll into view. There is currently no way to cause a window to map onto a desktop other than the currently active desk.

A geometry specified as something like:

```
xterm -geometry -5-5
```

will generally place the window's lower right hand corner 5 pixels from the lower right corner of the visible portion of the screen. Not all applications support window geometries with negative offsets.

Some applications, like xterm and xfontsel, allow the user to specify the start-up desk on the command line:

```
xterm -xrm “*Desk:1”
```

will start an xterm on desk number 1. Not all applications understand this option, however.

INITIALIZATION

During initialization, fvwm will search for a configuration file which describes key and button bindings, and a few other things. The format of these files will be described later. First, fvwm will search for a file named .fvwmrc in the users home directory. Failing that, it will look for /usr/lib/X11/fvwm/system.fvwmrc for system-wide defaults. If that file is not found, fvwm will exit.

Fvwm will set two environment variables which will be inherited by its children. These are $DISPLAY which describes the display on which fvwm is running. $DISPLAY may be unix:0.0 or :0.0, which doesn't work too well when passed through rsh to another machine, so $HOSTDISPLAY will also be set and will use a network-ready description of the display. $HOST- DISPLAY will always use the TCP/IP transport protocol (even for a local connection) so $DISPLAY should be used for local connections, as it may use Unix-domain sockets, which are faster.

SHAPED WINDOWS

If you typically use shaped windows such as xeyes or oclock, you have several options. You can make them all undecorated (NoBorder oclock and NoTitle oclock, for example) or you can use the default configuration and leave them decorated, in which case a decorative border and a solid-color backdrop are shown. Alternately, you can compile in the SHAPE extensions by changing a flag in the Makefile, in which case you get the shaped window with no backdrop, and a title bar floats above the window. The shaped window extensions increase the window manager's memory consumption by about 60 Kbytes when no shaped windows are present but have little effect when shaped windows are present.

CIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTUOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

ANATOMY OF A WINDOW

Fvwm puts a decorative border around most windows. This border consists of a bar on each side and a small "L" shaped section on each corner. There is an additional top bar called the title bar which is used to display the name of the window. In addition, there are up to 10 title-bar buttons. The top, side, and bottom bars are collectively known as the side-bars. The corner pieces are called the frame.

Unless the standard defaults files are modified, pressing mouse button 1 in the title or side-bars will begin a move operation on the window. Pressing button 1 in the corner frame pieces will begin a resize operation. Pressing button 2 anywhere in the border brings up an extensive list of window operations.

Up to ten title-bar buttons may exist. Their use is completely user definable. The default configuration has a title-bar button on each side of the title-bar. The one on the left is used to bring up a list of window options, regardless of which mouse button is used. The one on the right is used to iconify the window. The number of title-bar buttons used depends on which ones have mouse actions bound to them. See the section on the "Mouse" configuration parameter below.

THE VIRTUAL DESKTOP

Fvwm provides multiple virtual desktops for users who wish to use them. The screen is a viewport onto a desktop which may be larger than the screen. Several distinct desktops can be accessed (concept: one desktop for each project, or one desktop for each application, when view applications are distinct). Since each desktop can be larger than the physical screen, windows which are larger than the screen or large groups of related windows can easily be viewed.

The size of the virtual desktops can be specified at startup. All virtual desktops must be the same size. The total number of distinct desktops need not be specified, but is limited to approximately 4 billion total. All windows on the current desktop can be displayed in a Pager, a miniature view of the current desktop. Windows which are not on the current desktop can be listed, along with their geometries, in a window list, accessible as a pop-up menu.

"Sticky" windows are windows which transcend the virtual desktop by "Sticking to the screen's glass." They always stay put on the screen. This is convenient for things like clocks and xbiff's, so you only need to run one such gadget and it always stays with you.

Window geometries are specified relative to the current viewport. That is:

```
 xterm -geometry +0+0
```

will always show up in the upper-left hand corner of the visible portion of the screen. It is permissible to specify geometries which place windows on the virtual

SYNOPSIS

fvwm [options]

DESCRIPTION

Fvwm is a window manager for X11. It is a derivative of twm, redesigned to minimize memory consumption, provide a 3-D look to window frames, and provide a simple virtual desktop. Memory consumption is estimated at about one-half to one- third the memory consumption of twm, due primarily to a redesign of twm's method of storing mouse bindings. In addition, many of the configurable options of twm have been removed.

The name "FVWM" used to stand for something, but I forgot what. (Feeble, famous, foobar? It doesn't really matter, this is an acronym based society anyway.)

SPECIAL NOTE FOR XFREE86 USERS

XFree86 provides a virtual screen whose operation can be confusing when used in conjunction with fvwm. With XFree86 all windows which appear on the virtual screen actually get drawn into video memory (whether or not they appear on the physical screen), so the virtual screen size is limited by available video memory.

With fvwm's virtual desktop, windows which do not appear on the screen do not actually get drawn into video RAM. The size of the virtual desktop is limited to about 32,000 by 32,000 pixels, but it is probably impractical to use a virtual desktop more than about 5 times the visible screen in each direction. Note that memory usage is a function of the number of windows which exist - the size of the desktop makes no difference.

When becoming familiar with fvwm it is recommended that you disable XFree86's virtual screen by setting the virtual screen size to the physical screen size. After you become familiar with fvwm you may want to re-enable XFree86's virtual screen.

COPYRIGHTS

Since fvwm is derived from twm code it shares twm's copy- rights.

fvwm is copyright 1988 by Evans and Sutherland Computer Corporation, Salt Lake City, Utah, and 1989 by the Massachusetts Institute of Technology, Cambridge, Massachusetts, All rights reserved. It is also copyright 1993 and 1994 by Robert Nation.

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the names of Evans & Sutherland and M.I.T. not be used in advertising in publicity pertaining to distribution of the software without specific, written prior permission.

ROBERT NATION, EVANS & SUTHERLAND, AND M.I.T. DISCLAIM ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL EVANS & SUTHERLAND OR M.I.T. BE LIABLE FOR ANY SPE-

FVWM - Virtual Window Manager

This chapter describes the FVWM Virtual Window Manager and its associated utility modules. These all run as ViewStation local clients.

Section 2.0 - HDS ViewStation Local Clients

2.8 FVWM Virtual Window Manager

This chapter contains the manual pages for the FVWM Virtual Window Manager version 1.2 and its associated utility modules:

- FvwmAudio
- FvwmAuto
- FvwmBacker
- FvwmBanner
- FvwmClean
- FvwmDebug
- Fvwm Goodstuff
- FvwmIconBox
- FvwmIdent
- FvwmPager
- FvwmScroll
- FvwmWinlist
- xpmroot (often used by FVWM) is described in Section 2.6.19.

2.8.1 FVWM Manual Page

NAME

fvwm - F(?) Virtual Window Manager for X11