ULTRIX Worksystem Software

digital

Guide to the uwm Window Manager

Order Number: AA-KU50B-TE

ULTRIX Worksystem Software Guide to the uwm Window Manager

Order No. AA-KU50B-TE

ULTRIX Worksystem Software, Version 2.0

Digital Equipment Corporation

This manual was written and produced by the ULTRIX Documentation Group in Nashua, New Hampshire.

Contents

Figures

About This Manual

The *Guide to the uwm Window Manager* describes how to use and customize the uwm window manager.

Audience

The audience for this manual is the worksystem software user who is already familiar with using the C or Bourne shell.

Organization

The *Guide to the uwm Window Manager* contains the following sections:

Starting uwm
: Discusses how to start the uwm window manager and describes the menus through which you execute uwm commands.

Using Window Operations Commands
: Discusses those basic tasks that enable you to get started with the uwm window manager and use commands in the Window Operations menu.

Using Extended Window Operations Commands
: Discusses additional uwm tasks you can perform using commands in the Extended Window Operations menu.

Customizing uwm
: Discusses how to customize the uwm window manager.

Appendixes A and B list the default built-in startup file and the XUI Toolkit startup file for uwm.

Conventions

The following conventions are used in this manual:

special	In text, each mention of a specific command, option, partition, pathname, directory, or file is presented in this type.

name(n)	References to ULTRIX commands, system calls, or subroutines include the section numbers in the *ULTRIX Reference Pages* or *ULTRIX Worksystem Software Reference Pages* where they are documented.
UPPERCASE	The ULTRIX system differentiates between lowercase and uppercase characters. In examples, enter uppercase characters only where they are specifically indicated by an example or a syntax line.
`example`	In examples, system output is printed in this type.
`example`	In examples, user input is printed in this boldface type.

Guide to the uwm Window Manager

The uwm window manager lets you manipulate and control the windows on your screen. To accomplish these tasks, you select commands from one of the default uwm menus:

- Window Operations
- Extended Window Operations

The next three sections explain how to:

- Start uwm
- Use Window Operations commands
- Use Extended Window Operations commands

The last section provides information about customizing uwm. You should read it before attempting to customize your uwm window manager.

Starting uwm

Note that, before you start uwm, you should halt any other window managers you may be running. If you run uwm with any other window manager, you will get unpredictable results.

To start uwm, use the following command:

```
% uwm &
```

This command sequence causes uwm to execute in the background. While uwm executes in background, you then can enter other commands.

To indicate that it has started, uwm beeps. When you hear a beep, you can display either default menu: Window Operations or Extended Window Operations.

To display the Window Operations menu (labeled WINDOW OPS), press and hold down both the Compose Character key and mouse button 1 (MB1), which is the left mouse button unless you have redefined it. Figure 1 illustrates the Window Operations menu.

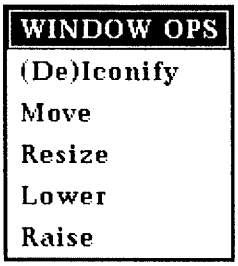


Figure 1: Window Operations Menu

To display the Extended Window Operations menu (labeled EXTENDED WINDOW OPS), press and hold down both the Compose Character key and MB2. Figure 2 illustrates the Extended Operations menu.

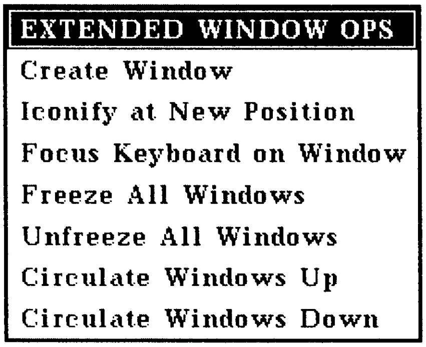


Figure 2: Extended Window Operations Menu

Using Window Operations Commands

The Window Operations menu lists these commands:

- (De) Iconify
- Move
- Resize

- Lower
- Raise

The following sections discuss these operations.

Iconifying and Deiconifying Windows

When you iconify a window, you change it to a small graphical representation that takes up little space on the screen. When you deiconify an icon, you change the icon back to a window at its original location and to its original size.

To iconify a window, follow this procedure:

1. Press and hold down both the Compose Character key and MB1.
2. Drag the mouse pointer to the (De)Iconify command.
3. Release the Compose Character key and the mouse button.
4. Move the mouse pointer to the appropriate window.
5. Press both the Compose Character key and MB1.

When you select the window, uwm immediately changes the window to its appropriate icon. Figure 3 illustrates the resulting icon.

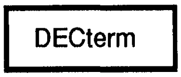


Figure 3: Iconified Window

To deiconify a window, follow this procedure:

1. Press and hold down both the Compose Character key and MB1.
2. Drag the mouse pointer to the (De)Iconify command.
3. Release the Compose Character key and the mouse button.
4. Move the mouse pointer to the appropriate window.
5. Press both the Compose Character key and MB1.

When you select the icon, uwm immediately changes it back to its appropriate window.

Moving Windows

Within your default uwm environment, you can move a window or icon to another screen location by following one of two procedures.

To use the Move command from the Window Operations menu, follow this procedure:

1. Press and hold down both the Compose Character key and MB1.
2. Drag the mouse pointer to the Move command.
3. Release the Compose Character key and the mouse button.
4. Move the mouse pointer to the appropriate window.
5. Press and hold down both the Compose Character key and MB3.
6. Move the window with the displayed grid.

To use a mouse button and key combination, follow this procedure:

1. Move the mouse pointer to the appropriate window.
2. Press and hold down both the Compose Character key and MB3.
3. Pull the mouse pointer off to the right or left of the displayed menu.
4. Move the window with the displayed grid.

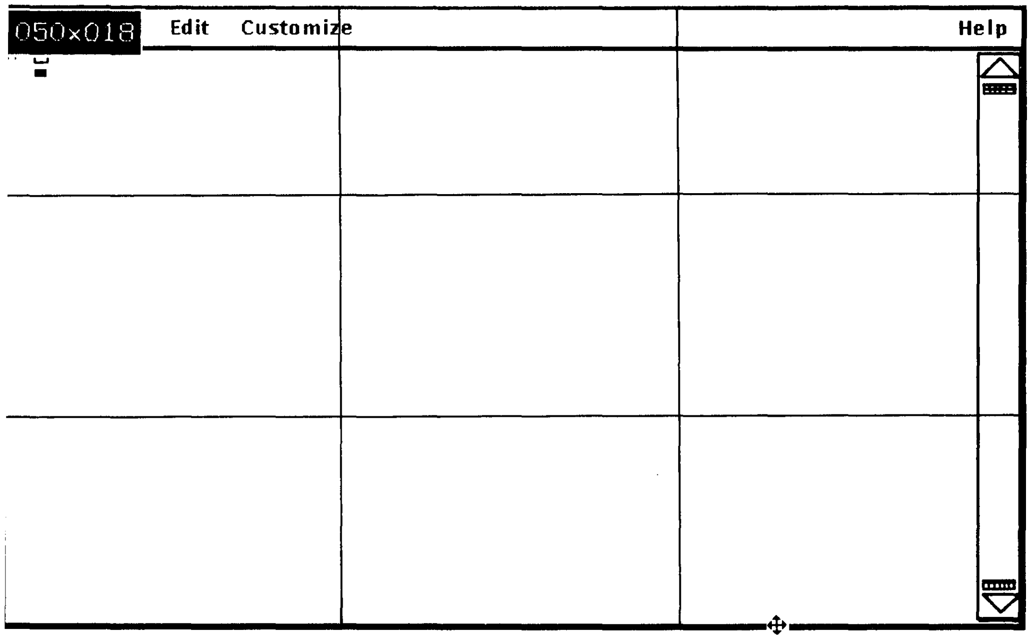
While continuing to press the mouse button, move the grid by dragging the mouse pointer to the appropriate screen location. When you have moved the grid to this screen location, release the mouse button. When you release the mouse button, uwm removes the window grid and displays the window at its new location.

Resizing Windows

To enlarge or reduce a window, follow this procedure:

1. Press and hold down both the Compose Character key and MB1.
2. Drag the mouse pointer to the Resize command.
3. Release the Compose Character key and the mouse button.
4. Move the mouse pointer to the appropriate window.
5. Press and hold down both the Compose Character key and MB1.
6. Resize the window with the displayed grid.

When you select the window, uwm overlays the window with a 9-rectangle grid and displays the window's size in columns and rows. Figure 4 illustrates a window that has been overlaid with a grid and size display.



ZK-0055U-HC

Figure 4: Resize Grid

When you select the window, you should place the mouse pointer within the appropriate grid rectangle. The following table lists each grid rectangle and the corresponding resizing action allowed.

Rectangle	Action
Center and corners	Allow you to enlarge or reduce both the width and length (horizontal and vertical dimensions) by dragging the mouse diagonally away from or towards the size display.
Right and left middle	Allow you to enlarge or reduce only the width (horizontal dimension), regardless of the direction in which you drag the mouse.
Upper and lower middle	Allow you to enlarge or reduce only the length (vertical dimension), regardless of the direction in which you drag the mouse.

Lowering and Raising Windows

When you have multiple windows on the screen, some may overlap and obscure others. You can rearrange these windows, like sheets of paper in a stack, by lowering or raising any window. Lowering a window puts it out of view by moving it to the bottom of the stack. Raising a window brings it into full view by moving it to the top of the stack.

To lower a window, follow this procedure:

1. Press and hold down both the Compose Character key and MB1.
2. Drag the mouse pointer to the Lower command.
3. Release the Compose Character key and the mouse button.
4. Move the mouse pointer to the appropriate window.
5. Press both the Compose Character key and MB1.

When you select the window, uwm immediately lowers the selected window to the bottom of the window stack.

To raise a window, follow this procedure:

1. Press and hold down both the Compose Character key and MB1.
2. Drag the mouse pointer to the Raise command.
3. Release the Compose Character key and the mouse button.
4. Move the mouse pointer to the appropriate window.
5. Press both the Compose Character key and MB1.

When you select the window, uwm immediately raises the selected window to the top of the window stack.

Using Extended Window Operations Commands

The Extended Window Operations menu lists these commands:

- Create Window
- Iconify at New Position
- Focus Keyboard on Window
- Freeze All Windows
- Unfreeze All Windows
- Circulate Windows Up
- Circulate Windows Down

The following sections discuss these operations.

Creating Windows

When you create a window from uwm, you actually create a window with the dxterm(1X) command. This dxterm window provides DECwindows DECterm terminal emulation.

To create a window, follow this procedure:

1. Press and hold down both the Compose Character key and MB2.
2. Drag the mouse pointer to the Create Window command.
3. Release the Compose Character key and the mouse button.
4. Position and size the window with the mouse.

When you select the Create Window command, uwm overlays the pointer with a temporary rubber-banded window outline and displays the window size in the upper left corner of the screen. Figure 5 illustrates the temporary rubber-banded window outline and size display.

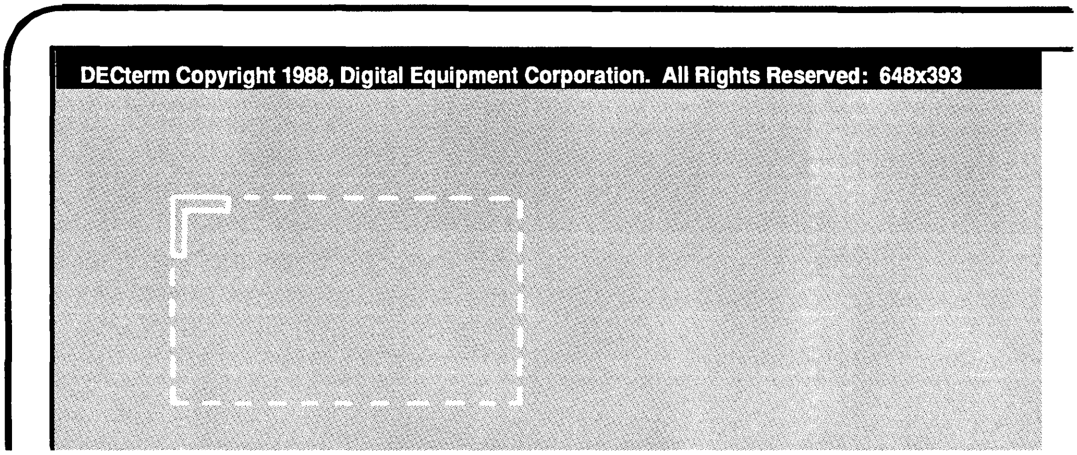

DECterm Copyright 1988, Digital Equipment Corporation. All Rights Reserved: 648x393

Figure 5: Rubber-Banded Window Outline

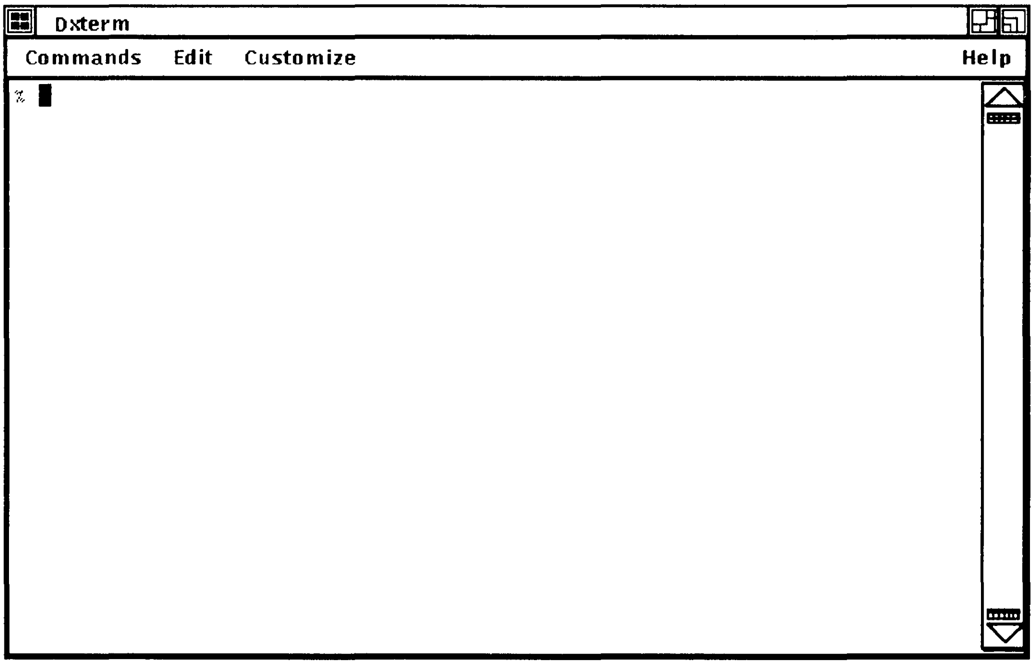
To position the rubber-banded window, move the mouse pointer to the appropriate screen location. To size the rubber-banded window, press one of the three mouse buttons:

MB1	Sizes a window with standard width (80 columns) and length (24 rows).
MB2	Sizes a window whose width and length are determined by mouse movement.
MB3	Sizes a window with a standard width (80 columns) but a length equal to the length of the screen.

When you press and hold MB2 to size the rubber-banded window, you can drag the mouse pointer in any direction on the screen. The following table lists the direction, its corresponding mouse movement, and the sizing action allowed.

Direction	Mouse Movement	Action
Diagonally	Away from stationary corner	Sizes both the width and the length (horizontal and vertical dimensions, respectively).
Horizontally	Left or right from stationary corner	Sizes only the width (horizontal dimension).
Vertically	Up or down from stationary corner	Sizes only the length (vertical dimension).

When you release the appropriate mouse button, the dxterm window is displayed. Figure 6 illustrates a sample dxterm window.



ZK-0057U-HC

Figure 6: A Sample dxterm Window

Iconifying a Window at a New Position

To iconify a window at another screen location, follow this procedure:

1. Press and hold down both the Compose Character key and MB2.
2. Drag the mouse pointer to the Iconify at New Position command.
3. Release the Compose Character key and the mouse button.
4. Move the mouse pointer to the appropriate window.
5. Press and hold down both the Compose Character key and MB2.
6. Move the grid to another screen location.

When uwm overlays the selected window with an icon-size, 9-rectangle grid, continue to press the mouse button. Next, move the icon grid to the new screen location and release the mouse button. Figure 7 illustrates a window with an icon-size grid overlaid.

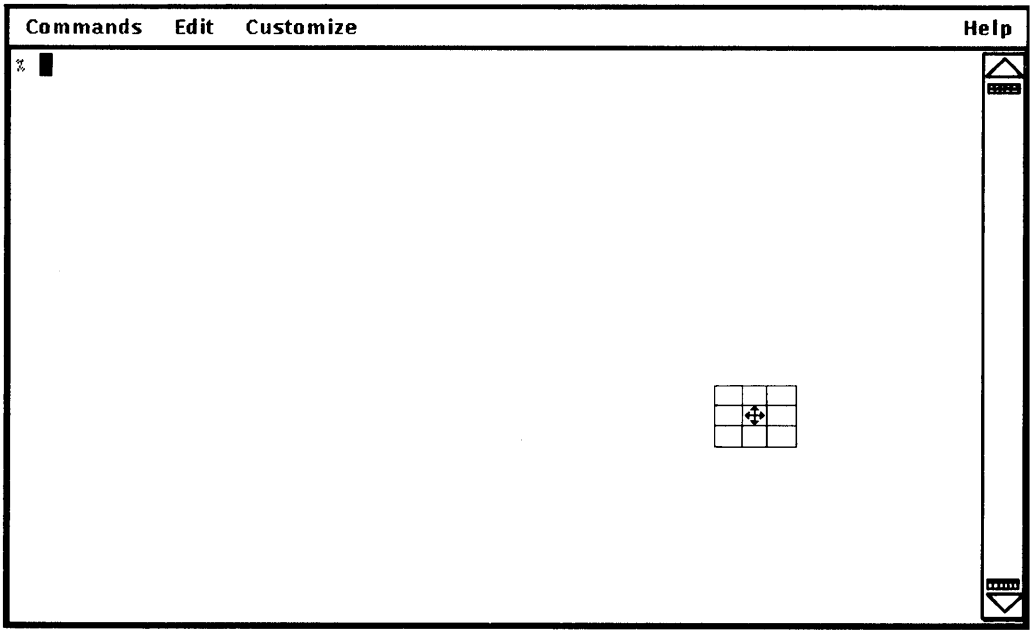


Figure 7: Icon-Sized Grid

To deiconify this icon, you can follow one of two procedures. To use the (De)Iconify command, follow this procedure:

1. Press and hold down both the Compose Character key and MB1.
2. Drag the mouse pointer to the (De)Iconify command.
3. Release the Compose Character key and the mouse button.
4. Move the mouse pointer to the appropriate icon.
5. Press the Compose Character key and MB1.

When you select the (De)Iconify command, uwm immediately places the window back at its original location.

To use the Iconify at New Position command, follow this procedure:

1. Press and hold down both the Compose Character key and MB2.
2. Drag the mouse pointer to the Iconify at New Position command.
3. Release the Compose Character key and the mouse button.
4. Move the mouse pointer to the appropriate icon.
5. Press both the Compose Character key and MB2.

When you select the Iconify at New Position command, uwm displays a 9-rectangle window grid with which you can move the window to a new screen location. When you move the grid and release the mouse button, uwm changes the icon back to a window.

Focusing and Unfocusing the Keyboard on Windows

If you have created several windows but expect to do your work mostly in one, you can focus all of your keyboard input to that one window. Then, regardless of where your mouse pointer is located on the screen, all keyboard input is interpreted by the selected window only.

To focus all keyboard input to a window, follow this procedure:

1. Press and hold down both the Compose Character key and MB2.
2. Drag the mouse pointer to the Focus Keyboard on Window command.
3. Release the Compose Character key and the mouse button.
4. Move the mouse pointer to the appropriate window.
5. Press both the Compose Character key and MB2.

When you release the Compose Character key and the mouse button, uwm immediately focuses all keyboard input to the selected window, regardless of the pointer's position.

To unfocus the keyboard, follow this procedure:

1. Press and hold down both the Compose Character key and MB2.
2. Drag the mouse pointer to the Focus Keyboard on Window command.
3. Release the Compose Character key and the mouse button.
4. Move the mouse pointer off all windows.
5. Press both the Compose Character key and MB2.

When you release the Compose Character key and the mouse button, uwm immediately unfocuses all keyboard input from the selected window.

Freezing and Unfreezing Windows

If you execute a large process that takes much of your system's resources, you may want to freeze all windows. When you freeze all windows, the window server does not update any windows and can dedicate its resources to other processing. Then, when you unfreeze all windows, the window server automatically updates each window with any output that was buffered while the window was frozen.

To freeze all windows, follow this procedure:

1. Press and hold down both the Compose Character key and MB2.
2. Drag the mouse pointer to the Freeze All Windows command.
3. Release the Compose Character key and the mouse button.
4. Move the mouse pointer off the menu.
5. Press both the Compose Character key and MB2.

When you release the Compose Character key and the mouse button, uwm immediately freezes all windows.

To unfreeze all windows, follow this procedure:

1. Press and hold down both the Compose Character key and MB2.
2. Drag the mouse pointer to the Unfreeze All Windows command.
3. Release the Compose Character key and the mouse button.
4. Move the mouse pointer off the menu.
5. Press and hold down both the Compose Character key and MB2.

When you release the Compose Character key and the mouse button, uwm immediately unfreezes all windows.

Circulating All Windows

When you have multiple windows on the screen, some may overlap and obscure others. You can rearrange these windows by lowering or raising any window, as if they were sheets of paper on a stack. By circulating all windows up, you raise the bottom window to the top of the stack, thereby bringing it into full view. By circulating all windows down, you lower the top window to the bottom of the stack, thereby putting it out of view.

Within your default uwm environment, you can circulate all windows up by following one of two procedures.

To use the Circulate Windows Up command from the Extended Window Operations menu, follow this procedure:

1. Press and hold down both the Compose Character key and MB2.
2. Drag the mouse pointer to the Circulate Windows Up command.
3. Release the Compose Character key and the mouse button.
4. Move the mouse pointer off all windows.
5. Press both the Compose Character key and MB2.

To use a mouse button and key combination, follow this procedure:

1. Move the mouse pointer off all windows.
2. Press both the Compose Character key and MB3.

When you release the Compose Character key and mouse button, uwm immediately circulates the last window up to the top of the stack.

To circulate all windows down, follow this procedure:

1. Press and hold down both the Compose Character key and MB2.
2. Drag the mouse pointer to the Circulate Windows Down command.
3. Release the Compose Character key and the mouse button.
4. Move the mouse pointer off all windows.
5. Press and hold down both the Compose Character key and MB2.

When you release the Compose Character key and mouse button, uwm immediately circulates the first window down to the bottom of the stack.

Customizing uwm

You can customize your uwm environment by modifying the .uwmrc file located in your home directory. If your directory does not contain this file, first copy /usr/lib/X11/uwm/default.uwmrc to .uwmrc in your home directory. Then, you can modify this uwm startup file to suit your own preferences.

When you start uwm, it establishes your window manager environment by taking the following actions, in order:

1. Using its built-in, default specifications (from /usr/lib/X11/uwm/default.uwmrc)
2. Reading the system-level specifications from /usr/lib/X11/uwm/system.uwmrc, if it exists
3. Reading specifications from .uwmrc in your home directory, if it exists
4. Reading specifications from your alternate startup file specified with the -f option, if it exists

As it reads these specifications, uwm uses the last definition it encounters for each specification when the defined values for that specification differ. Therefore, because it reads the information from your startup file last, uwm always uses the values that you define there.

The startup information used to establish your uwm environment contains three specification groupings:

Variables	Modify how uwm executes a specified action.
Functions	Assign the action that is executed when either a mouse button or a character key is pressed, or when the pointer is moved to a specified screen location.
Menus	Define each menu that you can display and from which you can select defined items.

The next three sections explain the specifications that you can include in your startup files. After reading these sections, you can modify your startup file to customize uwm to your preferences. The fourth section lists guidelines for customizing uwm.

For a complete listing of the /usr/lib/X11/uwm/default.uwmrc file, see Appendix A. For a complete listing of the /usr/lib/X11/uwm/system.uwmrc file, see Appendix B.

Variable Specifications

Each of the variables you can specify modifies a certain aspect of a specific uwm function. If you want to use only the variables listed in the current startup file, specify resetvariables at the beginning of the variable list.

When modifying variables, either you can list each variable on a separate line, or you can list more than one variable on the same line by separating each with a semicolon (;).

The variables section in /usr/lib/X11/uwm/system.uwmrc lists the following specifications:

```
#
# Global variables
#
resetbindings;resetvariables;resetmenus
autoselect
delta=25
freeze
grid
hiconpad=5
hmenupad=6
iconfont=times_bold14
menufont=times_bold14
normali
resizefont=6x13
viconpad=5
vmenupad=3
volume=7
```

You can specify the following variables in the startup file:

Variable	Description
autoselect, noautoselect	The autoselect variable specifies that uwm is to place the mouse pointer on the first menu item when a menu is initially displayed. The noautoselect variable specifies that uwm is to place the mouse pointer on the menu header (title). The default is autoselect.
delta = *n*	The delta variable specifies that you must move the mouse pointer *n* screen pixels before uwm invokes those functions that subsequently are defined with delta mouse action. The default is 1.
freeze, nofreeze	The freeze variable specifies that uwm is to freeze the screen (not allow screen updates) while executing certain functions, for example, f.move or f.resize. In addition, the freeze variable prevents the window grid or window box from flickering. The nofreeze variable specifies that uwm is to allow screen updates while executing all functions. The default is freeze.
grid, nogrid	The grid variable specifies that uwm is to display a resize or move window grid when you invoke the f.resize and f.move functions. The nogrid variable specifies that uwm is to display a window box (outline) in place of the grid. The default is grid.
hiconpad = *n*	The hiconpad variable specifies that uwm is to pad the width of a text icon's sides with *n* pixels. The default is 5 pixels.
hmenupad = *n*	The hmenupad variable specifies that uwm is to pad the width of a menu's sides with *n* pixels. The default is 5 pixels.

Variable	Description
iconfont=*font*	The iconfont variable specifies that uwm is to use the indicated text font for those icons that it creates. See the /usr/lib/dwf/* directories for a list of the available fonts. The system default is "times_bold14". The built-in default is "fixed".
maxcolors=*n*	The maxcolors variable specifies that uwm is to use a maximum of *n* colors at any one time (for color displays only). If maxcolors is specified and if the indicated number of colors is exceeded, uwm uses the colors set for the root window (screen background). If maxcolors is not specified, however, uwm uses as many colors as it can. There is no default limit.
menufont=*font*	The menufont variable specifies that uwm is to use the indicated font for all menu text. The /usr/lib/dwf/* directories contains all the available fonts. The default is "times_bold14".
normali, nonormali	The normali variable specifies that uwm is to let an icon be placed only within the root window (screen background) when it is invoking the f.newiconify function. The nonormali variable specifies that uwm is to let an icon be placed partially off screen when it is invoking the f.newiconify function. The default is normali.
normalw, nonormalw	The normalw variable specifies that uwm is to let a window be placed only within the root window when it is invoking the f.newiconify function. The nonormalw variable specifies that uwm is to let a window be placed partially off screen when it is invoking the f.newiconify function. The default is normalw.
push=*n*	The push variable specifies the indicated distance that uwm is to use when invoking the f.pushup, f.pushdown, f.pushright, and f.pushleft functions. The pushabsolute and pushrelative variables determine the unit of distance. The default is 5.

Variable	Description
pushabsolute, pushrelative	The pushabsolute variable specifies that uwm is to use the pixel as the unit of measure for the push variable. The pushrelative variable specifies that uwm is to use as the unit of measure 1/*n*th of the window size, where *n* is set by the push variable. For example, if the pushrelative variable is listed and if the push variable is specified as 2, uwm moves the window one half (1/2) its width when invoking the f.pushleft function. The default is pushrelative.
resetbindings	The resetbindings variable specifies that uwm is to reset the key, mouse, and context bindings. Bindings that are subsequently specified in the startup file will supersede the reset value.
resetmenus	The resetmenus variable specifies that uwm is to reset the menu definitions. Definitions that are subsequently specified in the startup file will supersede the reset value.
resetvariables	The resetvariables variable specifies that uwm is to reset the variables. Variables that are subsequently specified in the startup file will supersede the reset value.
resizefont=*font*	The resizefont variable specifies that uwm is to use the specified font for the window size display when invoking the f.resize function. See the /usr/lib/dwf/* directories for a list of available fonts. The system default is "6x13". The built-in default is fixed.
reverse, noreverse	The reverse variable specifies that uwm interchange the foreground and background colors when default colors are used for window manager windows and icons. The noreverse variable specifies that no interchanging occurs. The default is reverse.
viconpad=*n*	The viconpad variable specifies that uwm is to pad *n* pixels above and below an icon name. The default is 5.

Variable	Description
vmenupad = *n*	The vmenupad variable specifies that uwm is to pad *n* pixels above and below a menu name and each menu item. The default is 5.
volume = *n*	The volume variable specifies that uwm is to set the keyboard bell volume to the indicated intensity (*n*) when invoking the f.beep function. The intensity can be 0 (softest) through 7 (loudest). The default is 4.
zap, nozap	The zap variable specifies that uwm is to display ghost lines when invoking the f.move or f.resize function. The nozap variable specifies that uwm is not to display ghost lines. The default is nozap.

Function Specifications

Each function specification has the following format:

function = *key* : *context* : *button* [:"*menu-name*"]

function	Specifies the name of the mapped function that is to perform the task.
key	Specifies the name of the mapped key.
context	Specifies the screen location the mouse pointer must be in before uwm invokes this function.
button	Specifies the name of the mapped mouse button and its corresponding action.
menu-name	Specifies, if applicable, the name of the menu that is to be used.

The default functions section in /usr/lib/X11/uwm/system.uwmrc lists the following specifications:

```
#
# Mouse button/key maps
#
# FUNCTION     KEYS   CONTEXT       BUTTON        MENU(if any)
# ========     ====   =======       ======        ===========
f.menu =       meta   :        :    MB1 down     :"WINDOW OPS"
f.menu =       meta   :        :    MB2 down     :"EXTENDED WINDOW OPS"
f.move =       meta   :wli     :    MB3 down
f.circleup =   meta   :root    :    MB3 down
```

You can map one function to several key, context, and mouse button combinations, or you can map several functions to a single key, context, and mouse button combination.

When mapping a function, take into consideration the possible conflict between the uwm mappings and mappings for other programs that may be running in a window. For example, a window created with dxterm uses the Shift key and mouse movement for cutting and pasting text. If you mapped a uwm function to the Shift key and invoked that function within a dxterm window, uwm would interpret the Shift key and dxterm would not receive the instruction.

To use only those mappings specified in your current .uwmrc startup file, include resetbindings at the beginning of the file.

Functions

You can modify existing mappings or add new mappings for any of the following functions:

Function	Description
f.beep	Emits a beep (intensity set by the volume variable) from the keyboard.
f.circledown	Lowers the top window to the bottom of the window stack.
f.circleup	Raises the bottom window to the top of the window stack.
f.continue	Unfreezes all windows, that is, allows them to be updated. This function should always be used in conjunction with f.pause.
f.exit	Exits from the current uwm session.
f.focus	Directs all keyboard input to the specified window. To reset the focus to follow the mouse cursor, invoke the f.focus function from the root window.
f.iconify	Changes a window to an icon or an icon to a window.

Function	Description
f.lower	Lowers the specified window to the bottom of the window stack.
f.menu	Displays the specified defined pop-up menu.
f.move	Moves the specified window to a new default screen location.
f.moveopaque	Moves the specified window with the mouse pointer, instead of first moving the window grid and then moving the window.
f.newiconify	Iconifies and positions a window or icon at the specified screen location.
f.pause	Freezes all windows, that is, does not allow them to be updated. This function should always be used in conjunction with f.continue.
f.pushdown	Moves a window down the amount specified by the push variable.
f.pushleft	Moves a window to the left the amount specified by the push variable.
f.pushright	Moves a window to the right the amount specified by the push variable.
f.pushup	Moves a window up the amount specified by the push variable.
f.raise	Raises the specified window to the top of the window stack.
f.refresh	Updates the entire screen.
f.resize	Resizes the specified window. Note that some clients, particularly editors, may react unpredictably if you resize the window while the client is running.
f.restart	Restarts uwm and causes it to reread all the uwm startup files and initialize variables.

Most functions require a window argument. That is, after selecting the function from a menu, you need to select the window or icon that is to be affected. To make the selection, place the mouse pointer in the window or icon and click the specified mouse button.

Keys

You can specify one or more of the following keys:

Key	Abbreviation
ctrl	c
lock	l
meta (Compose Character)	m
shift	s

To specify that two keys be pressed to invoke the named function, separate the key names with a vertical bar (|). You cannot specify more than two keys.

To specify no key (just mouse button and context used), leave this field blank.

Using the Shift key alone is not recommended, because client applications other than the window manager use the Shift key as a control key. If you bind the Shift key to a window manager function, you cannot use other client applications that require this key.

Contexts

The context is the screen location that must contain the mouse pointer when you invoke the specified command. You can specify one or more of the following contexts:

Location	Abbreviation	Meaning
icon	i	Action is invoked when the pointer is placed on an icon.
window	w	Action is invoked when the pointer is placed in a window.
root	r	Action is invoked when the pointer is placed within the root window (screen background).

To specify more than one context, separate the context names with a vertical bar (|).

To specify no context (just key and mouse button used), leave this field blank. Essentially, this specifies that all contexts are valid. This allows a function to be invoked from any screen location.

Mouse Buttons

When you map a mouse button to a named function, you must map the button name and the action of the specified button.

You can specify one of the following mouse buttons:

Button	Abbreviation
left	l or MB1
middle	m or MB2
right	r or MB3

You can assign the following actions to each mouse button:

Action	Meaning
down	Action is invoked when the button is pressed.
up	Action is invoked when the button is released.
delta	Action is invoked when the mouse button is pressed and the pointer is moved the number of pixels specified by the delta variable.

Because additional mouse movement is required to complete their operation, you can bind the following functions only to the button down action:

f.iconify
f.move
f.moveopaque
f.newiconify
f.resize

Menu Names

This entry specifies the name of a defined menu that is to be displayed when uwm invokes the corresponding f.menu function. If the menu name contains spaces, tabs, parentheses, or other special characters, place the name in quotation marks. This name is listed in the displayed menu header.

Menu Specifications

For each f.menu function you map, you must define the corresponding menu. Each menu specification has the following format:

```
menu = menu name {
item name : action item name : action
      .
      .
      .
}
```

menu name	Specifies the name of the menu.
item name	Specifies the text that is to be displayed for that item.
action	Specifies the action that is to be invoked when that item is selected.

The default menus section in /usr/lib/X11/uwm/system.uwmrc lists the following specifications:

```
#
# Menu specifications
#
menu = "WINDOW OPS" {
"(De)Iconify":   f.iconify
Move:            f.move
Resize:          f.resize
Lower:           f.lower
Raise:           f.raise
}
menu = "EXTENDED WINDOW OPS" {
Create Window:                 !"dxterm &"
Iconify at New Position:       f.newiconify
Focus Keyboard on Window:      f.focus
Freeze All Windows:            f.pause
Unfreeze All Windows:          f.continue
Circulate Windows Up:          f.circleup
Circulate Windows Down:        f.circledown
}
```

The width of the menu is determined by the longest specified menu or item name. To pad the width with extra spacing, use the hmenupad variable.

The length of the menu is determined by the number of menu items. To pad the length with extra spacing, use the vmenupad variable.

To indicate where uwm is to position the mouse pointer when the menu initially is displayed, use either the autoselect or the noautoselect variable.

To use only the menus defined in your current .uwmrc file, include the resetmenus variable at the beginning of the file.

Menu and Item Names

The menu name should be the same name specified in the corresponding function specification. If a menu or menu item name contains spaces, tabs, parentheses, or other special characters, enclose the name in quotation marks.

Menu Actions

A menu action can be any of the following:

- A function
- A shell command
- Text with a newline character
- Text without a newline character

A single menu can contain all of the menu action types.

Function Menu Actions

You can specify a uwm function as a menu action. If a menu contains more menu items than can be displayed on the screen, you may want to specify another menu as a menu item, nesting another menu function inside a menu.

To specify a menu as a menu action specification, include the f.menu function, a colon, and the menu name as a single entry. For example, the following defined menu contains the nested Move Commands menu:

```
menu = "Handy Commands" {
Refresh:                  f.refresh
Restart:                  f.restart
Move Windows:    f.menu : "Move Commands"
}
```

The following Move Commands menu then would contain its corresponding menu action specifications:

```
menu = "Move Commands" {
Move Down:       f.pushdown
Move Left:       f.pushleft
Move Right:      f.pushright
Move Up:         f.pushup
}
```

Shell Command Menu Actions

You can use any shell command as a menu action specification. To specify a shell command as a menu action, put an exclamation point (!) escape character in front of the command. For example, in the default Extended Window Operations Menu, the Create Window command actually executes the dxterm command from the menu. The menu action specification for this item is:

```
Create Window:               !"dxterm &"
```

You cannot include a newline character within a shell command.

Text String Menu Actions

You can use a string of text as a menu action. When a text string menu action is chosen, the text string is placed in a cut buffer. You then can paste this buffered text to a window.

You can specify a text string that ends with a newline character or you can specify a text string only.

When you use a text string with a new line, you can paste the text string to a dxterm window for immediate execution. To specify that a new line is to be appended to the text string, begin the string with a circumflex character (^). For example, to execute the date command and a newline character, list the following as a menu action specification:

```
Date:        ^date
```

When you use a text string only, you can append text to the string once it is pasted to a window. To use a text string without a new line, begin the string with a vertical bar character (|). For example, to execute a command which compiles a source file that is to be specified on the command line, list the following as a menu action specification:

```
Compile:          |  cc -O -s
```

Color Menus

If you have a color display, you can specify different colors for the menu header, each menu item, and for menu highlighting.

The color menu specification has the following format:

menu = *menu name* (*color1*:*color2*:*color3*:*color4*) {
item name: (*color5*:*color6*): *action*
item name: (*color5*:*color6*): *action*

.
.
.

}

color1	Specifies the foreground color (text) for the menu header.
color2	Specifies the background color for the menu header.
color3	Specifies the foreground color for menu highlighting.
color4	Specifies the background color for menu highlighting.
color5	Specifies the foreground color (text) for a menu item.
color6	Specifies the background color for a menu item.

For a list of available colors, see the /usr/lib/X11/rgb.txt file. To specify the color, include the color's name in the appropriate menu definition field.

The maximum number of colors that you can display is set by the maxcolors variable. However, uwm uses the colors of the root (background) window if you do one of the following:

- Run out of color map entries, either before or after invoking uwm
- Specify a foreground or background color that does not exist in the RGB database, /usr/lib/X11/rgb.txt
- Omit a foreground or background color
- Exceed the maximum number of colors specified by the maxcolors variable in your uwm startup file

- Specify no colors in your startup file
- Specify colors but have a monochrome system

When a color is displayed in a uwm menu, it cannot be changed by any other application.

Rules for Customizing the uwm Startup File

When customizing the window manager startup file, observe these rules:

- Group all like specifications together in your startup file, and list each grouping in the following order: variable specifications, then function specifications, then menu specifications.
- List resetbindings, resetmenus, and resetvariables, if you use them, as the first entries in the variables grouping.
- Use semicolons to separate variables when specifying more than one on a line.
- Enclose all character strings that contain two or more consecutive spaces or any number of tabs in double quotation marks.
- Do not embed quotation marks within quotation marks.
- Enclose all shell meta characters (; : { } # = ^) in double quotation marks.
- Enclose all character strings containing parentheses in double quotation marks.

Default Built-In Startup File A

This appendix contains a listing of the default built-in uwm startup file /usr/lib/X11/uwm/default.uwmrc. This is the first startup file read by uwm. You can copy this file to .uwmrc in your home directory and modify it to customize uwm to your own preferences.

```
# $Source $
# $Author $
# $Header $
# Copyright (c) 1987 by the Massachusetts Institute of Technology.
#
# This is a startup file for uwm that produces an xwm lookalike,
# but adds two useful menus.  It is patterned on the public
# distribution ../lib/X/uwm/jg.uwmrc file by Jim Gettys.
#
resetbindings
resetvariables
resetmenus
noautoselect
delta=5
freeze
grid
zap
pushabsolute
push=1
hiconpad=5
viconpad=5
hmenupad=3
vmenupad=0
iconfont=fixed
menufont=fixed
resizefont=fixed
volume=0
# FUNCTION       KEYS     CONTEXT          MOUSE BUTTON ACTIONS
f.newiconify=    meta     :window|icon:    delta left
f.raise=         meta     :window|icon:    delta left
f.lower=         meta     :window|icon:    left up
f.raise=         meta     :window:         middle down
f.resize=        meta     :window:         delta middle
```

(continued on next page)

```
f.iconify=      meta    :icon:          middle up
f.raise=        meta    :window|icon:    right down
f.move=         meta    :window|icon:    delta right
f.circledown=   meta    :root:          left down
f.circleup=     meta    :root:          right down
f.circledown=   m|s     ::               left down
f.menu=                 :root:           middle down    : "WindowOps"
f.menu=         m|s     ::               middle down    : "WindowOps"
f.menu=         m|s     ::               middle down    : "Preference
f.circleup=     m|s     ::               right down
f.iconify=      m|c     :window|icon:    left down
f.newiconify=   m|l     :window|icon:    left down
f.raise=        m|l     :window|icon:    left up
f.pushright=    m|l     :window|icon:    right down
f.pushleft=     m|c     :window|icon:    right down
f.pushup=       m|l     :window|icon:    middle down
f.pushdown=     m|c     :window|icon:    middle down
menu = "WindowOps" {
New Window:   !"xterm&"
RefreshScreen:    f.refresh
Redraw:       f.redraw
Move:         f.move
Resize:       f.resize
Lower:        f.lower
Raise:        f.raise
CircUp:       f.circleup
CircDown:     f.circledown
AutoIconify:  f.iconify
LowerIconify: f.newiconify
NewIconify:   f.newiconify
Focus:        f.focus
Freeze:       f.pause
UnFreeze:     f.continue
Restart: f.restart
}
menu = "Preferences" {
Bell Loud:    !"xset b 7&"
Bell Normal:  !"xset b 3&"
Bell Off:     !"xset b off&"
Click Loud:   !"xset c 8&"
Click Soft:   !"xset c on&"
Click Off:    !"xset c off&"
Lock On: !"xset l on&"
Lock Off:     !"xset l off&"
Mouse Fast:   !"xset m 4 2&"
Mouse Normal: !"xset m 2 5&"
Mouse Slow:   !"xset m 1 1&"
}
```

XUI Toolkit Startup File B

This appendix contains a listing of the default startup file /usr/lib/X11/uwm/system.uwmrc. This file overrides the /usr/lib/X11/uwm/default.uwmrc file and is provided to give the XUI Toolkit "look and feel." You can copy this file to .uwmrc in your home directory and modify it to customize uwm to your own preferences.

```
#
# Global variables
#
resetbindings;resetvariables;resetmenus
autoselect
delta=25
freeze
grid
hiconpad=5
hmenupad=6
iconfont=times_bold14
menufont=times_bold14
normali
resizefont=6x13
viconpad=5
vmenupad=3
volume=7
#
# Mouse button/key maps
#
# FUNCTION     KEYS   CONTEXT       BUTTON       MENU(if any)
# ========     ====   =======       ======       ============
f.menu =       meta   :         :   left down    :"WINDOW OPS"
f.menu =       meta   :         :  middle down   :"EXTENDED WINDOW OPS"
f.move =       meta   :wli      :   right down
f.circleup =   meta   :root     :   right down
#
# Menu specifications
#
menu = "WINDOW OPS" {
"(De)Iconify":     f.iconify
Move:           f.move
Resize:         f.resize
Lower:          f.lower
Raise:          f.raise
```

(continued on next page)

```
}
menu = "EXTENDED WINDOW OPS" {
Create Window:                  !"xterm &"
Iconify at New Position:        f.newiconify
Focus Keyboard on Window:       f.focus
Freeze All Windows:             f.pause
Unfreeze All Windows:           f.continue
Circulate Windows Up:           f.circleup
Circulate Windows Down:         f.circledown
}
```

Index

HOW TO ORDER ADDITIONAL DOCUMENTATION

DIRECT TELEPHONE ORDERS

In Continental USA
and New Hampshire,
Alaska or Hawaii
call **800-DIGITAL**

In Canada
call **800-267-6215**

DIRECT MAIL ORDERS (U.S. and Puerto Rico*)

DIGITAL EQUIPMENT CORPORATION
P.O. Box CS2008
Nashua, New Hampshire 03061

DIRECT MAIL ORDERS (Canada)

DIGITAL EQUIPMENT OF CANADA LTD.
100 Herzberg Road
Kanata, Ontario K2K 2A6
Attn: Direct Order Desk

INTERNATIONAL

DIGITAL EQUIPMENT CORPORATION
PSG Business Manager
c/o Digital's local subsidiary
or approved distributor

Internal orders should be placed through the Software Distribution Center (SDC), Digital Equipment Corporation, Westminster, Massachusetts 01473

*Any prepaid order from Puerto Rico must be placed
with the Local Digital Subsidiary:
809–754–7575

ULTRIX
Worksystem Software
Guide to uwm
AA–KU50B–TE

Reader's Comments

Note: This form is for document comments only. DIGITAL will use comments submitted on this form at the company's discretion. If you require a written reply and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Did you find this manual understandable, usable, and well-organized? Please make suggestions for improvement. ____________________

____

____

____

Did you find errors in this manual? If so, specify the error and the page number.

____

____

____

Please indicate the type of user/reader that you most nearly represent.

- ☐ Assembly language programmer
- ☐ Higher-level language programmer
- ☐ Occasional programmer (experienced)
- ☐ User with little programming experience
- ☐ Student programmer
- ☐ Other (please specify) ____________________

Name ____________________ Date ____________________

Organization ____________________

Street ____________________

City ____________________ State ______ Zip Code or Country ____________________

------Do Not Tear - Fold Here and Tape --

digital

No Postage
Necessary
if Mailed in the
United States

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO.33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

Digital Equipment Corporation
Documentation Manager
ULTRIX Documentation Group
ZKO3-3/X18
Spit Brook Road
Nashua, N.H.
03063

------Do Not Tear - Fold Here and Tape --

ULTRIX
Worksystem Software
Guide to uwm
AA–KU50B–TE

Reader's Comments

Note: This form is for document comments only. DIGITAL will use comments submitted on this form at the company's discretion. If you require a written reply and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Did you find this manual understandable, usable, and well-organized? Please make suggestions for improvement. ____________________

____

____

____

Did you find errors in this manual? If so, specify the error and the page number.

____

____

____

Please indicate the type of user/reader that you most nearly represent.

- ☐ Assembly language programmer
- ☐ Higher-level language programmer
- ☐ Occasional programmer (experienced)
- ☐ User with little programming experience
- ☐ Student programmer
- ☐ Other (please specify) ____________________

Name ____________________ Date ____________________

Organization ____________________

Street ____________________

City ____________________ State ________ Zip Code or Country ____________

------Do Not Tear - Fold Here and Tape --

digital

No Postage
Necessary
if Mailed in the
United States

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO.33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

Digital Equipment Corporation
Documentation Manager
ULTRIX Documentation Group
ZKO3-3/X18
Spit Brook Road
Nashua, N.H.
03063

------Do Not Tear - Fold Here and Tape --