FUJITSU COMPUTERS SIEMENS

SINIX/windows ONLINE Documentation

SINIX/windows User Environment V3.0
Clients Reference Manual

Edition May 1996

SINIX® Copyright © Siemens Nixdorf Informationssysteme AG 1990.
SINIX is the UNIX® System derivate of Siemens Nixdorf Informationssysteme AG.
UNIX is a registered trademark in the United States an other countries, licensed exclusively through X/Open Company Limited.
Copyright © Siemens Nixdorf Informationssysteme AG 1996.
All rights reserved.Delivery subject to availability; right of technical modifications reserved.
All Hardware and software names used are trademarks of their respective manufactors.
Siemens Nixdorf Informationssysteme AG

---

1 Preface

The SINIX®/windows User Environment is an OSF/Motif-based user interface for SINIX™ systems running SINIX V5.41 and above. It is a graphical interface with which you work interactively.
It is an integrated, consistent working environment in which you can access applications on your screen using a mouse and the keyboard. The SINIX/windows User Environment owes its consistency of appearance and behavior to the fact that it was developed on the basis of OSF/Motif™, which itself builds on the network-transparent, hardware-independent X Window System™.

1.1 Target group and manual structure

The present manual is intended for experienced users in need of a general survey of the clients that are available, the syntax for invoking them, the functionality they offer, and the associated resources.
The chapter "Clients" documents the clients in alphabetical order. It covers clients responsible for display and desktop management, the mwm window manager, the terminal emulators, and tools to help you configure your interface.
The chapter "Client resources" presents the resources available to the clients in alphabetical order.
The chapter "Assigning values to resources " explains the ordering and precedence rules a client applies in evaluating resource definitions. It tells you which files are available for resource configuration and describes the hierarchical relationship between these files. It also contains descriptions of the syntax of the configuration files and how values are assigned to resources.
The chapter "Environment variables " contains a detailed explanation of all the environment variables evaluated by the clients.
It is followed by the "Glossary".

1.2 Changes since the last version of the manual

The tables below list the functional changes made to clients between Version 2.2 and Version 3.0 of the SINIX/windows User Environment. Changes of an editorial nature and corrections of mistakes in the documentation are only listed if they are of particular significance.
There is a separate table of changes for each client which has undergone functional enhancement. The following clients have been superseded, are no longer supported in the product, or are new:

Client	Status	Brief description
Mail Manager	dropped	superseded by Communication Manager
Communication Manager	new	Manager for electronic mail, MIME files, fax

Table 1: New or dropped clients

The following changes are functional modifications.

Changes to the xdm client		Status
authName	authorization mechanism	modified
browserHasFocus	input focus	new
checkPathnames	session selection	new
debugLevel	error messages	modified
defaultHost	default host	new
defaultIcon32, defaultIcon64	default icon	modified
defaultUser	default user	new
failFont	fonts for error messages	modified
iconPath	list of files for icon images	modified
labelString	label of a menu item	new
languagePrompt	text of language selection prompt	new
onlyLocalUser	user icons	modified
promptFont	fonts for prompt	modified
resources	name of Xresources file	modifi

		ed
rootBackground	color of root window	new
rootBackgroundPixmap	pixmap for root window	new
sessionPrompt	text of session selection prompt	new
systemPath	system path settings	modified
terminateServer	X server termination	modified
userPath	user path settings	modified
xdm-config-file file	configuration file	modified
/etc/default/login file	system file parameters	modified
Changes to the xmessages client		Status
monitor	error log file	new

Table 2: Functional modifications to clients

1.3 Notational conventions

In this manual, the following conventions have been used for presenting syntax:

name	Arguments in italics can be replaced with appropriate values.
name	Arguments in normal type must be entered exactly as shown.
[name]	Arguments in square brackets are optional.
[name]...	Arguments in square brackets followed by an ellipsis can be omitted or they can be specified any number of times.
{name1} {name2}	Arguments in braces are "multiple-choice" arguments; you must select one of them.

Table 3: Notational conventions

These conventions apply to the resource syntax, the menu syntax, and the syntax used in defining key and button bindings.

2 Clients

This chapter contains descriptions of all the clients, arranged alphabetically. The clients can be assigned to six main functional groups:

Display management	
xauth	Handle user-specific authorization information
xdm	Manage displays in a network
xhost	Control access to X server
xmessages	Message output and login entry in /etc/utmp
Window management	
mwm	Control elements of man-machine interface
Desktop management	
xdesk	Manage desktools and files
xdfm	File manager
xdcm	Manage mail, fax, X.400
xdvm	Manage archives and file systems on removable media
xdtell	Access to the object managers: xdesk, xdfm, xdhm, xdnm, xdmm, xdum, and xdvm
System management	
xdhm	Manage hosts
xdnm	Manage NFS
xdsm	Manage software
xdum	Manage users
Terminal emulators	
xpct	ANSI-mode TC20-V100 terminal emulator
xterm	VT102 terminal emulator and Tektronix 4014 terminal emulator
x97801-480	97801-480 terminal emulator
Configuration tools	
xdeskbox	Create dialog boxes
xdttm	Set time and date
xlc	Set international working environment

xmbind	Control virtual keyboard and pointer maps
xmodmap	Check keyboard and pointer maps
xrdb	Load resource definitions into X server

Table 4: Functional overview of clients

Description format

The client descriptions are in a uniform pattern. Elements of this standard pattern may be omitted if they are not applicable to a particular client.
In each client description there is an introduction to the client's functionality, followed by a guide to the options you can select when calling the client. The option descriptions are arranged in alphabetical order. They include the default values the client uses if no value is specified explicitly. Note that the default values apply only if they have not been modified in resource configuration files. The configuration files and their hierarchical ordering are described in detail in the chapter "Assigning values to resources ".
The options are followed by a brief guide to the resources evaluated by the client - first in the form of an overview and then in alphabetical order of resource instance names. A number of resources support very wide ranges of values which would go beyond the scope of the brief guide in this chapter. For full descriptions of the resources, their values and the value assignment syntax, you should refer to the chapter "Client resources".
The layout of the resource descriptions is as follows:
The first line contains the name of the resource instance; the second line gives the name of the resource class to which the resource belongs. Then there are cross-references to options or resources which are functionally related to the resource instance. This is followed by a brief description of the resource. The last elements of this brief description are the value and the effect of the resource's default setting.
The resource descriptions are followed by a section describing the client's appearance and behavior. Then come the client's key and button bindings and its menus if not described in the online help screens. The next sections are alphabetical listings of the files and environment variables supported by the client.
The client description is rounded off by examples showing how to use the client.

2.1 mwm - Motif window manager

The window manager mwm provides the user, system administrator, or programmer with functions for controlling elements of the user interface such as the size and position of windows, window states (iconified/normal), input focus ownership, etc. The window manager can also be used to start and terminate clients.
With the -multiscreen option, mwm can manage one or more screens on a display.

mwm[***option*****]...**

-display [*hostname*]:*display_number*[.*screen_number*]
Identifies the X server, display and screen which the client is to contact.

hostname
symbolic name of the host or X terminal on which the X server which the client is to contact runs. *hostname* must be known in the transport protocol of the host on which the client resides.

display_number
number (unique to the X server) identifying the display for which the client is to manage I/O.

screen_number
.number (unique to the display) identifying the screen to which the client is to send output.

Default:
contents of the DISPLAY shell variable;
if DISPLAY is not set, the client terminates abnormally.

-multiscreen
See "multiScreen" resource
See -display option
This option indicates that mwm is to manage all the screens on the display.

Default:
Only one screen will be managed.
Specified by the -display option.

-name *resource_link*
resource_link is the name of the application under which the client's window resources are to be found.
The client can thus be called under different names and utilize the resources for its client instance *resource_link*, yet still utilize the same resources for its client class *clientclass*.
resource_link should not contain dots (.) or asterisks (*), as these symbols have special meanings. Please refer to the section "Value assignment syntax " for more information.

Default:
mwm

-screens $name_0$[$name_1$]...
See -multiscreen option
Specifies names for the screens managed by mwm.
If mwm is managing a single screen, the first name *name* on the list will be used.
The names *name* will be assigned in order to the screens being managed, i.e. the name $name_i$ to the screen with the screen number i. If mwm is managing more screens than names have been assigned, it uses the name $name_0$ for these screens.
You can use the screen name $name_i$ when assigning values to resources, thus setting the resource for a specific screen. In order to do this, insert the name $name_i$ between the client name and the widget path:
*clientname***name;*[**widget*]...**resource*: *value*
No default value.

-xrm "*resourceDefinition*:*value*"
Assigns a value to a client resource.
-xrm must be specified separately for each resource to be set in the client call. The "*resourceDefinition*:*value*" assignment is compared with the entries in the configuration files.

resourceDefinition
Identifies the resource to which a value is to be assigned. For *resourceDefinition* you can give the name of the resource, the name of the client and of any widgets in the appropriate resource syntax (see section "Value assignment syntax ").

value Value to be assigned to the resource.

2.1.1 Resources

You can configure the client using its resource database. To do this, you must assign the required values to the resources. You can use either configuration files or the -xrm option in the client call to access resources.
The instance name of the client is mwm and its class name Mwm. Please note that you can use the -name *instancename* option to change the instance name of the client when you call it.

2.1.1.1 Resource overview

This subsection starts with mwm's predefined component names for assigning values to resources. This is followed by a table in which the resources are grouped according to function, and then come brief descriptions of the resources listed alphabetically according to resource name.
The following component names are predefined for mwm:

client
Class: Client
affects the client window.
Example: Mwm*client*background: green
"mwm" is to use the value "green" for the resource instance "background" for all window frames of client windows.

client_instance
Class: see *client_class*
affects the client instance *client_instance*.
The name of the icon box assigned to the resource instance iconBoxName of the class IconBoxName can also be used as *client_instance*.
Example: Mwm*icon**client_instance**background: green
"mwm" is to use the value "green" for the resource instance "background" for window icons managed by the client instance "*client_instance*".

See *client_instance* as instance
Class: *client_class*
affects all client instances of the client class *client_class*.
Example: Mwm*icon**client_class**background: green

"mwm" is to use the value "green" for the resource instance "background" for window icons managed by the client instance "*client_class*".

defaults
Class: Defaults
replaces unknown resource path details.
Example: Mwm*defaults*background: green
"mwm" is to use the value "green" for the resource instance "background" for all unknown widgets in the resource path "defaults" (e.g. if a file contains "Mwm*pencil*background: green" and mwm does not know the widget "pencil").

feedback
Class: Feedback
affects feedback windows.
Example: Mwm*feedback*background: green
"mwm" is to use the value "green" for the resource instance "background" for all feedback windows having the name "feedback".

icon
Class: Icon
affects icons.
Example: Mwm*icon*background: green
"mwm" is to use the value "green" for the resource instance "background" for all its icons having the name "icon".

menu
Class: Menu
affects menus.
Example: Mwm*menu*background: green
"mwm" is to use the value "green" for the resource instance "background" for all its menus having the name "menu".

menu**menu_name*
Class: none
affects the menu *menu_name*.
Example: Mwm*menu**menu_name**background: green
"mwm" is to use the value "green" for the resource instance "background" for all menus having the name "*menu_name*".

title
Class: Title
affects the title area of a window or icon.
Example: Mwm*client*title*background: green
"mwm" is to use the value "green" for the resource instance "background" for the title area "title" of all client windows having the name "client".

In the following tables, the resources are grouped according to their function.

2.1.1.2 Component appearance resources

These resources determine the appearance of user interface components. They can be applied to window manager menus, feedback windows, window frames and icons.

Name	Value	Default
background	color	varies[1]

backgroundPixmap	pixmap_name	varies[1]
bottomShadowColor	color	varies[1]
bottomShadowPixmap	pixmap_name	varies[1]
fontList	font_name	fixed
foreground	color	varies[1]
saveUnder	True|False	False
topShadowColor	color	varies[1]
topShadowPixmap	pixmap_name	varies[1]
[1]) The default depends on the visual class of the screen		

Table 5: mwm resources for areas influenced by the window manager

Name	**Value**	**Default**
activeBackground	color	varies[1]
activeBackgroundPixmap	pixmap_name	varies[1]
activeBottomShadowColor	color	varies[1]
activeBottomShadowPixmap	pixmap_name	varies[1]
activeForeground	color	varies[1]
activeTopShadowColor	color	varies[1]
activeTopShadowPixmap	color	varies[1]
[1]) The default depends on the visual class of the screen		

Table 6: mwm resources for frames of active window and for active icons

2.1.1.3 mwm behavior resources

Resources for specifying mwm behavior apply, for example, to the policy for setting the input focus, how icons are placed on the screen, whether an icon box is created, etc.

Name	**Value**	**Default**
autoKeyFocus	True|False	True
autoRaiseDelay	msec	500
bitmapDirectory	directory	/usr/include/X11/bitmaps

buttonBindings	string	DefaultButtonBindings
cleanText	True|False	True
clientAutoPlace	True|False	True
colormapFocusPolicy	string	keyboard
configFile	pathname	
deiconifyKeyFocus	True|False	True
doubleClickTime	msec	multiClickTime
enableWarp	True|False	True
enforceKeyFocus	True|False	True
fadeNormalIcon	True|False	False
feedbackGeometry	string	*In the middle of the screen*
frameBorderWidth	pixel	*varies*
iconAutoPlace	True|False	True
iconBoxGeometry	string	6x1+0-0
iconBoxName	string	iconbox
iconBoxSBDisplayPolicy	string	all
iconBoxTitle	string	Icons
iconClick	True|False	True
iconDecoration	string	*varies*
iconImageMaximum	bxh	50x50
iconImageMinimum	bxh	16x16
iconPlacement	string	left bottom
iconPlacementMargin	pixel	*varies*
interactivePlacement	True|False	False
keyBindings	string	DefaultKeyBindings

keyboardFocusPolicy	string	explicit
limitResize	True|False	True
lowerOnIconify	True|False	True
maximumMaximumSize	bxh	2x screen
moveThreshold	pixel	4
moveOpaque	True|False	False
multiClickTime	msec	200
multiScreen	True|False	False
passButtons	True|False	False
passSelectButton	True|False	True
positionIsFrame	True|False	True
positionOnScreen	True|False	True
quitTimeout	msec	1000
raiseKeyFocus	True|False	False
resizeBorderWidth	pixel	*varies*
resizeCursors	True|False	True
screens	string	*varies*
showFeedback	string	-kill
startupKeyFocus	True|False	True
transientDecoration	string	system title
transientFunctions	string	-minimize -maximize
useIconBox	True|False	False
wMenuButtonClick	True|False	True

wMenuButtonClick2	True|False	True
xnlLanguage	string	$LANG

Table 7: mwm behavior resources

2.1.1.4 Client-specific resources

Client-specific resources include the resources used to control window frame decoration, which window manager functions are available to a client, what the matte looks like, etc.

Name	Value	Default
clientDecoration	string	all
clientFunctions	string	all
focusAutoRaise	True|False	depends on keyboardFocusPolicy
iconImage	pathname	(image)
iconImageBackground	color	icon background
iconImageBottomShadowColor	color	icon bottom shadow
iconImageBottomShadowPixmap	pixmap	icon bottom shadow pixmap
iconImageForeground	color	*varies*
iconImageTopShadowColor	color	icon top shadow color
iconImageTopShadowPixmap	pixmap	icon top shadow pixmap
matteBackground	color	background
matteBottomShadowColor	color	bottom shadow color
matteBottomShadowPixmap	pixmap	bottom shadow pixmap
matteForeground	color	foreground
matteTopShadowColor	color	top shadow color
matteTopShadowPixmap	pixmap	top shadow pixmap
matteWidth	pixel	0
maximumClientSize	string	*varies*
useClientIcon	True|False	False

usePPosition	string	nonzero
windowMenu	string	DefaultWindowMenu

Table 8: Client-specific mwm resources

2.1.1.5 Resource list

The resources available to this client are listed in alphabetical order below. You can also use them in the -xrm option.

activeBackground
Class: Background
Specifies the background color of the window when the window is active. mwm uses this color for the window frame only.

Default:
depends on the visual class of the screen.

activeBackgroundPixmap
Class: ActiveBackgroundPixmap
Specifies the pixmap to be used for creating the background of the window when the window is active.

Default:
depends on the visual class of the screen

activeBottomShadowColor
Class: Foreground
Specifies the bottom shadow color of the window when the window is active.

Default:
depends on the visual class of the screen

activeBottomShadowPixmap
Class: BottomShadowPixmap
Specifies the pixmap to be used for the active window's bottom shadow.

Default:
depends on the visual class of the screen

activeForeground
Class: Foreground
Specifies the foreground color of the window when the window is active.

Default:
depends on the visual class of the screen

activeTopShadowColor
Class: Background
Specifies the top shadow color to be used for the window when the window is active.

Default:
depends on the visual class of the screen

activeTopShadowPixmap
Class: TopShadowPixmap
Defines the pixmap for the active window's top and left 3-D shadows.

Default:
depends on the visual class of the screen

autoKeyFocus
Class: AutoKeyFocus
Determines the policy for allocating the input focus when the active window is closed or minimized.
This resource only takes effect if the keyboardFocusPolicy resource that belongs to the KeyboardFocusPolicy resource class is assigned the value "explicit".

Default:
True
If a window that has the input focus is no longer managed by mwm or is iconified, the input focus is returned to the window that had it before.

autoRaiseDelay
Class: AutoRaiseDelay
Specifies the amount of time (in milliseconds) that mwm waits before raising a window to the top of the window stack after it receives the input focus.
This resource only takes effect if the value assigned to the focusAutoRaise resource is "True" and the value "pointer" is set for the keyboardFocusPolicy resource that belongs to the KeyboardFocusPolicy resource class.

Default:
500 (milliseconds)

background
Class: Background
See "activeBackground" resource
Determines the color of the client window background.
mwm uses this color only for the window frame of inactive windows.

Default:
depends on the visual class of the screen

backgroundPixmap
Class: BackgroundPixmap
See "activeBackgroundPixmap" resource
Specifies the pixmap to be used for the background of a window when the window is inactive.

Default:
depends on the visual class of the screen

bitmapDirectory
Class: BitmapDirectory
Specifies the directory containing the pixmap and bitmap files for the window manager resources. This directory is examined whenever a bitmap is specified without an absolute pathname.

Default:
Value of the XBMLANGPATH environment variable

bottomShadowColor
Class: Foreground
See "activeBottomShadowColor" resource
Specifies the color for the bottom and right-hand 3-D window shadows.

Default:
depends on the visual class of the screen

bottomShadowPixmap
Class: BottomShadowPixmap
See "activeBottomShadowPixmap" resource
Specifies the pixmap to be used for the bottom shadow of windows.

Default:
depends on the visual class of the screen

buttonBindings
Class: ButtonBindings
Identifies the set of button bindings that bind buttons to functions.
mwm examines the file specified in the configFile resource that belongs to the ConfigFile resource class. The set of button bindings must be defined in this file.
For more information refer to the subsection headed " "Key and button bindings"".

Default:
DefaultButtonBindings
Only the predefined default button bindings apply.

cleanText
Class: CleanText
Controls the appearance of the background of fields in which mwm displays text (client title area and feedback windows).

Default:
True
The text is drawn on a clear (no stipple) background; mwm only clears the area immediately around the text, the rest of the background remains unchanged.

clientAutoPlace
Class: ClientAutoPlace
Determines the position of a window when the window has not been given a user-specified position.

Default:

True

The windows are positioned with the top left corners of the frames offset horizontally and vertically.

clientDecoration

Class: ClientDecoration

Controls the amount of window frame decoration.

clientDecoration is specified as a list of decorations which indicates their inclusion in the frame or exclusion from the frame.

Default:

all

All decorations are included (decorations mean window border, title bar with maximize button, title bar with window menu button, title bar with minimize button, window border with border resize handles, window border with title bar).

clientFunctions

Class: ClientFunctions

Specifies which mwm functions are applicable (or not applicable) to the client window. All mwm functions listed here can be made applicable or not applicable: f.maximize, f.minimize, f.resize, f.move, and f.kill. Other mwm functions that have not been listed here are not concerned.

Default:

all

All mwm functions are applicable.

colormapFocusPolicy

Class: ColormapFocusPolicy

Specifies the colormap focus policy to be used.

Default:

keyboard

The client window that has the input focus will also have the colormap focus.

configFile

Class: ConfigFile

Specifies the pathname of the mwm configuration file defining menus and key and button bindings.

Default:

Depending on the setting of the LANG environment variable, the client looks for one of the following files and evaluates the first one it finds:

$HOME/$LANG/.mwmrc

$HOME/.mwmrc

/usr/lib/X11/$LANG/system.mwmrc

/usr/lib/X11/system.mwmrc

defaultVirtualBindings

Class: DefaultVirtualBindings

Specifies the mapping of virtual buttons and keys to your keyboard.

Default:

dependent on X server

deiconifyKeyFocus

Class: DeiconifyKeyFocus

Specifies whether mwm automatically sets the input focus when a window is normalized (deiconified).
This resource only applies if the keyboardFocusPolicy (class KeyboardFocusPolicy) is assigned the value "explicit".

Default:

True

A window is assigned the input focus when it is normalized.

doubleClickTime

Class: DoubleClickTime

Specifies when two successive clicks of a button are to be considered a double-click. The value specified is the maximum time (in milliseconds) that is allowed between the clicks (button presses) that make up a double-click.

Default:

value of the multiClickTime resource

enableWarp

Class: EnableWarp

Specifies whether the pointer is positioned in the middle of the selected window when you use the cursor keys to change the size or position of windows.

Default:

True

The pointer is positioned in the middle of the selected window.

enforceKeyFocus

Class: EnforceKeyFocus

See "keyboardFocusPolicy" resource
determines whether the policy requiring a window to be explicitly assigned the input focus applies to all windows.
This applies especially to "globally active" input windows. An example of a globally active window is a scroll bar.

Default:

True

If the keyboardFocusPolicy resource has been assigned the value "explicit", this policy applies to all windows, including globally active windows, i.e. they cannot be used if another window has the input focus.

fadeNormalIcon

Class: FadeNormalIcon

This resource indicates whether icons are grayed out (de-emphasized) when the

associated window has been opened.

Default:

False

An icon that has been normalized (i.e. its window has been opened) is not grayed out.

feedbackGeometry

Class: FeedbackGeometry

Position of the window in which the client during an action outputs the current result of this action.

mwm gives you feedback of this kind when you use the controls in the window frame to change the position or size of a window.

Default:

In the middle of the screen.

focusAutoRaise

Class: FocusAutoRaise

Specifies whether or not the stacking of windows is changed when the input focus is reassigned.

Default:

If the keyboardFocusPolicy resource has been assigned the value "explicit":

True

When an obscured window is assigned the input focus, it is moved to the top of the stack and is thus completely visible.

If the keyboardFocusPolicy resource has been assigned the value "pointer":

False

When an obscured window is assigned the input focus, the stacking order of the windows does not change.

fontList

Class: FontList

Font list

mwm uses the first font in the font list in the title area of the mwm window frame. The character coding of the font should match the character encoding used for text.

Default:

fixed

foreground

Class: Foreground

See "activeForeground" resource

Window foreground color.

The foreground color is the color for text output.

Default:

depends on the visual class of the screen.

frameBorderWidth

Class: FrameBorderWidth

Specifies the width (in pixels) of a client window frame border without resize handles. The border width includes the 3-D shadows.
The value of this resource should be checked against the resizeBorderWidth resource (class ResizeBorderWidth).

Default:

depends on the size and resolution of the screen.

iconAutoPlace

Class: IconAutoPlace

Specifies icon placement on the screen.

Default:

True
mwm determines the position of the icons on the screen, in a specific area depending on the value set in the iconPlacement resource.

iconBoxGeometry

Class: IconBoxGeometry

Specifies the initial position and size of the icon box.
Icons are placed horizontally if their width exceeds their height and the iconPlacement resource (class IconPlacement) has not been assigned an explicit value.
Icons are placed vertically if their height exceeds their width and the iconPlacement resource (class IconPlacement) has not been assigned an explicit value.
The default values for width and height are supplied by the resources iconWidth and iconHeight plus an additional "padding" factor (*p*). If the coordinates *x* and *y* are not specified, mwm uses the iconPlacement resource to determine initial placement.
The units for *width* and *height* are columns and rows, those for *x_off* and *y_off* are pixels.
The actual screen size of the icon box window depends on the iconImageMaximum and iconDecoration resources.

Default:

width: 6 x iconWidth + *p*
height: iconHeight + *p*
x_off: +0
y_off: -0

iconBoxName

Class: IconBoxName

Specifies the name that is used to look up icon box resources.
Please note the difference between this resource and the iconBoxTitle resource (class IconBoxTitle).

Default:

iconbox

iconBoxSBDisplayPolicy

Class: IconBoxSBDisplayPolicy

See "iconBoxGeometry" resource and "iconPlacement" resource
Determines whether the window manager displays a horizontal or a vertical scroll bar in the icon box.

Default:
all
mwm displays both a horizontal and a vertical scroll bar.

iconBoxTitle
Class: IconBoxTitle
Specifies the name of the icon box, i.e. the text that is displayed in the title area of the icon box.

Default:
Icons

iconClick
Class: IconClick
Specifies whether the system menu is posted and left posted when an icon is clicked. This resource only takes effect if icons are not managed in an icon box.

Default:
True
The system menu is posted and left posted when an icon is clicked, even after the button is released.

iconDecoration
Class: IconDecoration
Defines general icon decoration, which is the result of combining the attributes/values listed below.

activelabel
When the icon is clicked on, the text in the icon label is displayed in full (see label below).
imageThe icon image is displayed.
label The icon label is displayed; if necessary, the label will be right-truncated to the width of the icon.

Default:
for icon box icons: label image
for stand-alone icons: activelabel label image

iconImage
Class: IconImage
Specifies the file name of a bitmap file that contains the icon image; the file name may also be in the form of a pathname.
If this resource is enabled and if the value of the resource useIconClient is "False", no values for an icon image assigned by other clients take effect. mwm evaluates the client-specific resource useClientIcon to determine whether to use a client-supplied pixmap or a user-supplied pixmap for the icon image.

Default:

mwm uses its own pixmap for the icon image.

iconImageBackground

Class: Background

See "background" resource

Specifies the background color of the icon image.

Default:

Icon background color

iconImageBottomShadowColor

Class: Foreground

See "bottomShadowColor" resource

Specifies the bottom shadow color of the icon image.

Default:

as defined for the window frame

iconImageBottomShadowPixmap

Class: BottomShadowPixmap

See "bottomShadowPixmap" resource

Specifies the pixmap to be used for the bottom shadow of the icon image.

Default:

as defined for the window frame

iconImageForeground

Class: Foreground

Specifies the foreground color of the icon image.

Default:

system-dependent

iconImageMaximum

Class: IconImageMaximum

Specifies the maximum size of the image part of an icon.

The size of an icon image is the size of the image added to the width of the icon frame. The maximum supported size is 128 x 128 (pixels). If the specified values are higher, the icon image is truncated to the right and bottom so that it has the maximum size.

Default:

50 x 50 (pixels)

iconImageMinimum

Class: IconImageMinimum

Specifies the minimum size of the image part of an icon.

The size of an icon image is the size of the image added to the width of the icon frame. The minimum supported size is 16 x 16 (pixels). If the specified values are lower, mwm ignores iconImageMinimum.

Default:

16 x 16 (pixels)

iconImageTopShadowColor
Class: Foreground

See "topShadowColor" resource

Specifies the color to be used for the top shadow of the icon image.

Default:

as defined for the window frame

iconImageTopShadowPixmap
Class: TopShadowPixmap

See "topShadowPixmap" resource

Specifies the pixmap to be used for the top shadow of the icon image.

Default:

as defined for the window frame

iconPlacement
Class: IconPlacement

Specifies the icon placement.

This resource takes effect only if the iconAutoPlace resource (class IconAutoPlace) has the value "True".

Default:

left bottom

mwm places the icons in rows from bottom to top, with the first row placed at the bottom of the screen, and new rows added above it. The first icon in each row is placed at the left of the screen.

iconPlacementMargin
Class: IconPlacementMargin

Specifies the distance between the edge of the screen and the first icon

Default:

The distance between the edge of the screen and the icon is the same as the space between the icons on the screen (this space is based on maximizing the number of icons in each row and column).

interactivePlacement
Class: InteractivePlacement

determines whether the user can specify the position of the window or whether the position is determined by mwm.

Default:

False

mwm places new windows according to the default specified for icon placement.

keyBindings
Class: KeyBindings

Specifies the set of bindings of keys to functions.
mwm searches for this set in the file specified in the configFile resource (class ConfigFile).
Please see the section "Key and button bindings" in the mwm description for more details.

Default:
DefaultKeyBindings

keyboardFocusPolicy
Class: KeyboardFocusPolicy
determines the input focus policy.

Default:
explicit
The input focus is only set to a client window when you press button 1 with the pointer on the client window or any part of the client window decoration.

limitResize
Class: LimitResize
Specifies whether or not the user is allowed to resize a window to greater than that specified in
- the resource maximumClientSize (class MaximumClientSize), or
- the general description of client window attributes, or
- the default mwm assigns to the maximum size of a client window.

Default:
True
The maximum size may not be exceeded.

lowerOnIconify
Class: LowerOnIconify
Specifies whether an icon is placed at the bottom of the window stack, or whether it retains the position that its window had before it was iconified.

Default:
True
When a window is iconified, mwm places its icon at the bottom of the window stack.

matteBackground
Class: Background
See "background" resource
Specifies the background color of the matte.
This resource is evaluated if the matteWidth resource is assigned a positive value, i.e. a matte exists.

Default:
background color

matteBottomShadowColor

Class: Foreground

See "bottomShadowColor" resource

Specifies the color of the matte's bottom shadow.

This resource is evaluated if the matteWidth resource is assigned a positive value, i.e. a matte exists.

Default:

as defined for the window frame

matteBottomShadowPixmap

Class: BottomShadowPixmap

See "bottomShadowPixmap" resource

Specifies the pixmap mwm is to use for the bottom shadow of the matte.

This resource is evaluated if the matteWidth resource is assigned a positive value, i.e. a matte exists

Default:

as defined for the window frame

matteForeground

Class: Foreground

See "foreground" resource

Specifies the color mwm is to use for the foreground of the matte.

This resource is evaluated if the matteWidth resource is assigned a positive value, i.e. a matte exists

Default:

foreground color of the client window

matteTopShadowColor

Class: Background

See "topShadowColor" resource

Specifies the color to be used for the top shadow of the matte.

This resource is evaluated if the matteWidth resource is assigned a positive value, i.e. a matte exists

Default:

as defined for the window frame

matteTopShadowPixmap

Class: TopShadowPixmap

See "topShadowPixmap" resource

Specifies the pixmap mwm is to use for the top shadow of the matte. This resource is evaluated if the matteWidth resource is assigned a positive value, i.e. a matte exists

Default:

as defined for the window frame

matteWidth

Class: MatteWidth

Specifies the width of the matte in pixels.

Default:
0 (pixels)
There is no matte

maximumClientSize
Class: MaximumClientSize
Specifies the size of a window if it is to be maximized, or the direction in which it is maximized.

Default:
the window cannot be larger than the size specified for the maximumMaximumSize resource. mwm checks the enlargement with the client to which the window belongs.

maximumMaximumSize
Class: MaximumMaximumSize
Limits the maximum size of a client window. The value is expressed in pixels.

Default:
twice the screen width and height

moveOpaque
Class: MoveOpaque
Specifies how a window is to be displayed while being moved.

Default:
False
The window is displayed as a thin rectangular frame.

moveThreshold
Class: MoveThreshold
Specifies the threshold value (in pixels) that defines when mouse movement is interpreted as a dragging operation

Default:
4 (pixels)

multiClickTime
Class: MultiClickTime
Specifies the maximum time (in milliseconds) that may elapse between 2 mouse clicks for the two clicks to be interpreted as a multiple click.

Default:
200 (milliseconds)

multiScreen
Class: MultiScreen
See option -display ("-display [hostname]: display_number [. screen_number]")
Specifies whether or not mwm manages all the screens on a display.

Default:

mwm manages only one screen.

passButtons

Class: PassButtons

specifies whether or not button press events are passed to the client in whose window the button was pressed after the window manager has responded accordingly.
Button press events notify the clients on whether a button was pressed, released, clicked, or used during a dragging operation.
This resource only takes effect if the resource keyboardFocusPolicy (KeyboardFocusPolicy) is assigned the value "explicit".

Default:

False

The window manager does not pass the button press event to the client.

passSelectButton

Class: PassSelectButton

specifies whether or not button 1 press events are passed to the client in whose window the button was pressed after the window manager has responded accordingly.
Select button press events notify the clients on whether the select button was pressed, released, clicked, or used during a dragging operation. This resource only takes effect if the keyboardFocusPolicy resource (KeyboardFocusPolicy) is assigned the value "explicit".

Default:

True

The client is notified that a button was pressed in its window, and a window management action associated with the window decoration is performed. The input focus is assigned to the window.

positionIsFrame

Class: PositionIsFrame

specifies whether window position information is to refer to the position of the mwm window frame or to the position of the client area managed by the client.

Default:

True

mwm interprets the information as the position of the mwm client window frame.

positionOnScreen

Class: PositionOnScreen

determines how a window is clipped by the edge of the screen when it becomes visible and exceeds screen limits.

Default:

True

mwm attempts to place windows so that they are not clipped by the edge of the screen. If a window is larger than the size of the screen, then at least the upper

left corner of the window will be on-screen.

quitTimeout
Class: QuitTimeout

specifies the amount of time (in milliseconds) that mwm is to wait for a client to react once it has asked the client to save itself. This resource takes effect only if the client has been programmed appropriately. See also the mwm function " f.kill".

Default:

1000 (milliseconds)

raiseKeyFocus
Class: RaiseKeyFocus

specifies whether or not the window which is normalized and raised to the top of the window stack by the mwm function f.normalize_and_raise receives the input focus. This resource is only evaluated if the keyboardFocusPolicy resource is assigned the value "explicit".

Default:

False
The input focus remains unchanged.

resizeBorderWidth
Class: ResizeBorderWidth

specifies the width of the border resize handles (in pixels) and the 3-D shadows. It makes sense to set this value in conjunction with the frameBorderWidth resource of the FrameBorderWidth class.

Default:

depends on the size and resolution of the screen.

resizeCursors
Class: ResizeCursors

determines whether or not the pointer is displayed as a resize cursor when it is moved onto the window area where the window size can be manipulated

Default:

True
As soon as the pointer is moved onto the resize borders, the shape of the pointer changes to that defined for the resize cursor

saveUnder
Class: SaveUnder

determines whether or not the X server saves objects obscured by the client's mwm window frame for as long as these objects are obscured
This resource only takes effect in a client window if

- the X server provides "save under" functionality, and
- the client in turn requests the X server to save objects when its client area obscures other windows.

Default:

False
None of the mwm components use the save-under function.

screens
Class: Screens
See "multiScreen" resource
See -screens option ("-screens name 0 [name 1]...")
specifies the names of the screens managed by mwm.

Default:
screen_number of the corresponding screen.

showFeedback
Class: ShowFeedback
This resource controls when mwm feedback information is displayed
The value for this resource is a list of contexts in which mwm is to provide feedback information.

Default:
kill
Acknowledgment request on receipt of kill signal.

startupKeyFocus
Class: StartupKeyFocus
determines whether or not a window gets the input focus when it becomes visible on the screen
This resource only takes effect if the keyboardFocusPolicy resource (class KeyboardFocusPolicy) is assigned the value "explicit".

Default:
True
The input focus is assigned to the window that becomes

topShadowColor
Class: Background
See "activeTopShadowColor" resource
Specifies the color to be used for the top shadow of the window frame

Default:
depends on the visual class of the screen

topShadowPixmap
Class: TopShadowPixmap
See "activeTopShadowPixmap" resource
Pixmap of the top and left 3D shadow for windows

Default:
depends on the visual class of the screen

transientDecoration
Class: TransientDecoration
See "clientDecoration" resource

controls the amount of decoration mwm puts on transient windows. transientDecoration is specified as a list of decorations.

Default:

menu title

Window frame with title bar, window menu button and resize border handles

transientFunctions

Class: TransientFunctions

specifies which window management functions are applicable (or not applicable) to transient windows.

All window management functions explicitly listed here can be made applicable or not applicable: -maximize -minimize, f.resize, f.move, f.kill. This does not apply to any other window management functions that are not listed.

Default:

-maximize -minimize

maximize

refers to the f.maximize function

minimize

refers to the f.minimize function

useClientIcon

Class: UseClientIcon

specifies whether a client-supplied icon image overrides a user-supplied icon image.

Default:

False

A user-supplied icon image takes precedence over a client-supplied icon image.

useIconBox

Class: UseIconBox

specifies whether or not icons are placed in an icon box.

Default:

False

No icon box is used; the icons are placed on the root window.

usePPosition

Class: UsePPosition

Specifies whether mwm adopts the default position of a window from the client when you have not specified a position.

Default:

nonzero

A position you have specified has priority over that of the client providing it is not the origin (0,0) of the screen.

windowMenu

Class: WindowMenu

specifies the name of the menu which is posted as the window menu when the

window menu button is pressed

Default:
DefaultWindowMenu

wMenuButtonClick
Class: WMenuButtonClick
specifies whether or not the system menu is posted and left posted when the window menu button is clicked

Default:
True
The system menu is posted and left posted when the window menu button is clicked, even when the button is released.

wMenuButtonClick2
Class: WMenuButtonClick2
specifies whether a double-click on the window menu button initiates f.kill.

Default:
True
A double-click on the window menu button invokes f.kill.

xnlLanguage
Class: XnlLanguage
Specifies the client's locale.

Default:
value of the LANG environment variable

2.1.2 Appearance and behavior

The following sections describe the mwm functions, virtual buttons and keys, and menu definitions. It continues the section "Window management" in the manual "[3]" "SINIX/windows User Environment; Introduction to Handling and Configuration". In that section you find a description of the default appearance and behavior of windows, window stacks, icons, and the icon box. You can alter the appearance and behavior of the window manager by changing the values of mwm resources.

2.1.2.1 mwm functions

All functions made available to you by mwm for defining menus and key and button bindings are referred to as mwm functions. The following overview tells you what you can do with which mwm functions:

Task	**Function**
Arrange icons	**f.pack_icons**
Execute shell scripts and operating system	!, f.exec

commands	
Output menus	f.menu, f.separator, f.title
Output window menus	f.post_wmenu
Output a menu name	f.title
Terminate mwm	f.quit_mwm, f.restart, f.set_behavior
Terminate a client	f.kill, f.quit_mwm, f.restart, f.set_behavior
Notify clients	f.kill, f.send_msg
Move a window	f.move
Refresh windows	f.refresh, f.refresh_win
Restart mwm	f.restart, f.set_behavior
Without functionality	f.nop
Send an acoustic signal	f.beep
Set the color table focus	f.focus_color, f.next_cmap, f.prev_cmap
Set the input focus	f.focus_key, f.next_key, f.prev_key
Separate menu items	f.separator
Reconfigure mwm	f.set_behavior
Resize windows	f.maximize, f.minimize, f.normalize, f.normalize_and_raise, f.resize, f.restore, f.restore_and_raise
Change the window stack	f.circle_down, f.circle_up, f.lower, f.normalize_and_raise, f.raise, f.raise_lower
Process key bindings	f.pass_keys
Change screen	f.screen

Table 9: Task oriented overview on mwm functions

mwm functions can be accessed with key and button bindings and/or with mwm menus. Techniques for using the mwm functions are described after "Accessed by means of:" for each function.
mwm functions can generally be used on client windows or icons, providing the relevant client window or icon is active. However, you can also access an mwm function by means of a button action when the mouse cursor is located within the root window but not on a window or icon. In this case, the mwm function applies to the root window. The object of an mwm function is referred to as its context. The possible contexts are given below for each function after "Context:". The following contexts are possible:

Context	Effect
root	Neither a client window nor an icon is selected as the object to which the function applies.
window	A client window is selected as the object to which the function is to apply. Some functions are available only if the window is displayed in its normal size (e.g. f.maximize) or its maximum size (e.g. f.normalize).
icon	An icon is selected as the object to which the function is to apply.

Table 10: Contexts for mwm functions

If the context of a function is given as "icon" or "window", and the function is accessed in an icon box, it applies to the icon box and not to the icons within the icon box.
An mwm function is treated **as if it were f.nop** (i.e. it has no effect) if:
- it is accessed by an input the function does not support
- it is used in a context in which it is not available
- mwm does not known its name.

2.1.2.2 mwm function list

The following alphabetically arranged descriptions of all mwm functions give the syntax for each mwm function followed by the objects to which it can apply, its functionality, and any arguments that can be used with it:

! *command*
Equivalent: f.exec *command*
Accessed by means of: button, key, menu
Context: root, icon, window
Causes *command* to be executed in a shell. If the MWMSHELL environment variable is set, mwm evaluates this to determine the shell. If MWMSHELL is not set, mwm evaluates the SHELL environment variable. If neither MWMSHELL nor SHELL is set, the command is executed under /bin/sh.

f.beep
Accessed by means of: button, key, menu
Context: root, icon, window
Generates a beep.

f.circle_down [*object*]
Accessed by means of: button, key, menu
Context: root, icon, window
Causes the window or icon that is on the top of the window stack to be put on the bottom of the stack so that it is no longer obscuring any other window or icon. f.circle_down affects only those windows and icons that are obscuring other windows and icons or which are obscured by other windows and icons.
Secondary windows are restacked with their associated primary window. Secondary windows always stay on top of the associated primary window and there can be no other primary windows between a secondary window and its associated primary

window.
object can be replaced by:

icon f.circle_down applies only to icons.
window
f.circle_down applies only to windows.

f.circle_up [*object*]
Accessed by means of: button, key, menu
Context: root, icon, window
Raises the window or icon at the bottom of the window stack so that it is not obscured by any other windows.
f.circle_up affects only those windows and icons that are obscuring other windows and icons or which are obscured by other windows and icons.
Secondary windows are restacked with their associated primary window.
object can be replaced by:

icon f.circle_up applies only to icons.
window
f.circle_up applies only to windows.

f.exec *command*
Equivalent: ! *command*
Accessed by means of: button, key, menu
Context: root, icon, window
Causes *command* to be executed in a shell. If the MWMSHELL environment variable is set, mwm evaluates it to determine the shell. If MWMSHELL is not set, mwm evaluates the SHELL environment variable. If neither MWMSHELL nor SHELL is set, the command is executed under /bin/sh.

f.focus_color
Accessed by means of: button, key, menu
Context: root, icon, window
Sets the colormap focus to a client window. If this function is executed in a root context, the default colormap set up by the X Window System for the screen where mwm is running is installed, and there is no specific client window colormap focus. This function is treated as f.nop if the value assigned to the colormapFocusPolicy resource is not "explicit".

f.focus_key
Accessed by means of: button, key, menu
Context: root, icon, window
Sets the keyboard input focus to a client window or icon. This function is treated as f.nop if the value assigned to the keyboardFocusPolicy resource is not "explicit", or the function is executed in a root context.

f.kill
Accessed by means of: button, key, menu
Context: icon, window
If the client has explicitly asked the X server to notify it when
one of its client windows is to be deleted, or
it needs to prepare to be terminated,

the client is sent a message indicating that f.kill has been invoked. In all other cases, the client's connection to the server is terminated.
Please also refer to the quitTimeout resource instance (class: QuitTimeout).

f.lower [-*client*]

Accessed by means of: button, key, menu
Context: icon, window
Lowers a client window or icon to the bottom of the window stack where it does not obscure any other window. Secondary windows are restacked with their associated primary window.

Default:
the context in which f.lower is invoked indicates the window or icon to lower.

-*client*
client is the name or class of the client to be lowered.

f.lower within

Accessed by means of: button, key, menu
Context: icon, window
Causes a secondary window within a group of a client's windows to be moved to the bottom of the stack, where it does not obscure any of the client's other secondary windows. The sequence of windows in the stack for clients' primary windows is not changed.
For example, you can use this function to change a sequence of dialog boxes if you have opened them all from the same window of a client. All other windows retain their positions in the window stack.

f.lower freeFamily

Accessed by means of: button, key, menu
Context: icon, window
Causes a client's secondary window to be moved to the bottom of the stack of all clients' primary windows, where it does not obscure any client's windows. It is moved "freely", i.e. without reference to the group of windows of the client from which it originates. After being moved, it is therefore under the associated primary window of the group. The sequence of windows in the stack for clients' primary windows is not changed.
For example, you can even use this function to move a dialog box to under the window from which you opened the dialog box. All other windows retain their positions in the stack.

f.maximize

Accessed by means of: button, key, menu
Context: icon and (with restriction) window
Causes a client window to be enlarged to its maximum size. The function is only supported in the "window" context if the window is in its normal size.

f.menu *menu_name*

Accessed by means of: button, key, menu
Context: root, icon, window
Posts the menu *menu_name*. When the function is accessed by means of a key, thereby opening the menu, the same key closes the menu again.

f.minimize

Accessed by means of: button, key, menu
Context: window
Causes a client window to be minimized (iconified). If no icon box is being used, the icon is placed at the bottom of the window stack where it does not obscure any other window. If an icon box is used, the client's icon changes to its iconified form inside the icon box. Secondary windows are minimized with their associated primary window. There is only one icon for a primary window and all its secondary windows.

f.move
Accessed by means of: button, key, menu
Context: icon, window
Allows a client window to be moved on the screen interactively.

f.next_cmap
Accessed by means of: button, key, menu
Context: root, icon, window
Installs the next colormap in the list of colormaps for the window with the colormap focus.

f.next_key *object*
Accessed by means of: button, key, menu
Context: root, icon, window
Sets the keyboard input focus. The function argument identifies the control which is to receive the input focus. If no argument is specified, the input focus goes to the next window or icon managed by mwm. If the next window managed by mwm has children and thus constitutes a group of windows, the input focus goes to the window which was last active in that group.
The order in which f.next_key passes on the input focus is the window and icon stacking order.
This function is treated as f.nop if the value assigned to the keyboardFocusPolicy resource is not "explicit".

Default:
The keyboard input focus is set to the next window or icon.
object can be replaced by:

icon f.next_key switches the input focus to the next icon.

transient
f.next_key switches the input focus to the next window in the same window group. If the window group containing the currently active window consists of only the one currently active window, f.next_key is treated as f.nop.

window
f.next_key switches the input focus to the next window. If the next window has children and thus constitutes a window group, f.next_key switches the input focus to the window which was last active in that group.

f.nop
Accessed by means of: button, key, menu
Context: root, icon, window
This function does nothing (null operation).

f.normalize
Accessed by means of: button, key, menu

Context: icon and (with restriction) window
Causes a client window to be displayed in its normal size. Secondary windows are placed in their normal state along with their associated primary window. The function is supported in the context "window" only when the window is displayed in its maximum size.

f.normalize_and_raise
Accessed by means of: button, key, menu
Context: icon, window
Causes a client window to be opened in its normal size as uppermost window in the window stack. Secondary windows are restacked with their associated primary window.

f.pack_icons
Accessed by means of: button, key, menu
Context: root, icon, window
Is used to tidy up icons (based on the layout policy being used) on the root window or in the icon box.

f.pass_keys
Accessed by means of: button, key, menu
Context: root, icon, window
Causes the processing of key bindings by mwm to be enabled/disabled (toggled). When it disables key binding processing, all key input is passed on to the window with the keyboard input focus, and no mwm functions are executed. When processing by mwm is disabled by means of a key binding, the same key binding can be used to enable key binding processing.

f.post_wmenu
Accessed by means of: button, key
Context: root, icon, window
Posts the window menu. If f.post_menu is called using a key and a window menu button exists, the window menu is automatically placed with its top left corner at the bottom left corner of the window menu button for the client window. If there is no window menu button, the window menu is placed at the top left corner of the client window. If the function is accessed by means of a key, thereby opening the menu, the same key closes the menu again.

f.prev_cmap
Accessed by means of: button, key, menu
Context: root, icon, window
Installs the previous colormap in the list of colormaps for the window with the colormap focus.

f.prev_key[*object*]
Accessed by means of: button, key, menu
Context: root, icon, window
Sets the keyboard input focus. The function argument identifies the control which is to receive the input focus. If no argument is specified, the input focus goes to the previous window or icon managed by mxm. If the previous window managed by mwm has children and thus constitutes a group of windows, the input focus goes to the window which was last active in that group.
The order in which f.prev_key passes on the input focus is the window and icon stacking order.

This function is treated as f.nop if the value assigned to the keyboardFocusPolicy resource is not "explicit".

Default:

The keyboard input focus is set to the previous window or icon.

object can be replaced by:

icon f.prev_key switches the input focus to the previous icon.

transient

f.next_key switches the input focus to the previous window in the same window group. If the window group containing the currently active window consists of only the one currently active window, f.prev_key is treated as f.nop.

window

f.prev_key switches the input focus to the previous window. If the previous window has children and thus constitutes a window group, f.prev_key switches the input focus to the window which was last active in that group.

f.quit_mwm

Accessed by means of: button, key and (restricted) menu

Context: root, and (restricted) icon and window

Terminates mwm. The function is supported in the "icon" and "window" contexts only by means of button or key. The function can be accessed by menu in the "root" context only.

f.raise[-*client*]

Accessed by means of: button, key, menu

Context: icon, window

Raises a client window or icon to the top of the window stack where it is not obscured by any other window. Secondary windows are restacked with their associated primary window.

Default:

The context in which f.raise is invoked indicates the window or icon to raise.

-*client*

client is the name or class of the client whose window is to be raised.

f.raise within

Accessed by means of: button, key, menu

Context: icon, window

Causes a secondary window within a client's group of windows to be moved to the top of the stack, where it is not obscured by any of the client's other secondary windows. The sequence of windows in the stack for clients' primary windows is not changed. For example, you can use this function to change the sequence of dialog boxes if you have opened them from the same window of a client. All other windows retain their positions in the window stack.

f.raise freeFamily

Accessed by means of: button, key, menu

Context: icon, window

Causes a client's secondary window to be moved to the top of the stack for the primary windows of all clients, where it is not obscured by the windows of any client.

At the same time, the associated primary window is moved with all its secondary windows to the top of the stack for clients' primary windows.
For example, you can even use this function to move a dialog box over the primary windows of other clients, at the same time moving all secondary windows belonging to the same window group of the client as the dialog box.

f.raise_lower

Accessed by means of: button, key, menu
Context: icon, window
Raises a window to the top of the window stack if it is partially obscured by another window, otherwise it lowers the window to the bottom of the window stack. Secondary windows are restacked with their associated primary window.

f.raise_lower within

Accessed by means of: button, key, menu
Context: icon, window
Causes a client's secondary window that is partly obscured by another of the client's windows to be moved to the top of the group of the client's windows, where it is not obscured by any of the client's secondary windows. If, on the other hand, the secondary window is not obscured by any other of the client's windows, it is moved to the bottom of the group of the client's windows. In contrast to the effect of "f.raise_lower freeFamily", the sequence of the windows in the stack for clients' primary windows does not change.

f.raise_lower freeFamily

Accessed by means of: button, key, menu
Context: icon, window
Moves a client's secondary window that is partly obscured by another window (of the same or another client) to the top of the group of the client's windows, and at the same time moves the associated primary window with all its secondary windows to the top of the stack for clients' primary windows. It is then not obscured by any other of the client's secondary windows. If, on the other hand, the secondary window is not obscured by any other window, it is moved to the bottom of the group of the client's windows, and, at the same time, the associated primary window is moved with all its secondary windows to the bottom of the stack for clients' primary windows. In contrast to the effect of "f.raise_lower within", therefore, the sequence of the windows in the stack for clients' primary windows changes.

f.refresh

Accessed by means of: button, key, menu
Context: root, icon, window
Causes all windows to be redrawn.

f.refresh_win

Accessed by means of: button, key, menu
Context: window
Causes a specific client window to be redrawn.

f.resize

Accessed by means of: button, key, menu
Context: window
Allows a client window to be resized interactively.

f.restart

Accessed by means of: button, key and (restricted) menu

Context: root and (restricted) icon and window
Causes mwm to terminate and restart. The function is only supported in the "icon" and "window" contexts if you access it by means of a button or key. It can be accessed via menu in the "root" context only.

f.restore
Accessed by means of: button, key, menu
Context: icon and (restricted) window
Causes the client window to be restored to the size it was before being minimized.

f.restore_and_raise
Accessed by means of: button, key, menu
Context: icon and (restricted) window
Causes a client window to be restored to the size it was before it was minimized and to appear at the top of the stack.

f.screen *screen_number*
Accessed by means of: button, key, menu
Context: root, icon, window
Switches the mouse cursor from the current screen to the screen *screen_number* of the display on which you are working.

f.screen back
Accessed by means of: button, key, menu
Context: root, icon, window
Switches the mouse cursor from the current screen to the screen you were using previously of the display on which you are working.

f.screen next
Accessed by means of: button, key, menu
Context: root, icon, window
Switches the mouse cursor from the current screen to the next screen of the display on which you are working.

f.screen prev
Accessed by means of: button, key, menu
Context: root, icon, window
Switches the mouse cursor from the current screen to the previous screen of the display on which you are working.

f.send_msg *message_number*
Accessed by means of: button, key, menu
Context: icon, window
Sends a client message of the type _MOTIF_WM_MESSAGES where *message_number* indicates the message type. This function is normally only used by programmers.
The client message will only be sent if *message_number* is included in the client's _MOTIF_WM_MESSAGES property. A menu item is grayed out if it is used to do f.send_msg on a message that is not included in the _MOTIF_WM_MESSAGES property.

f.separator
Accessed by means of: menu
Context: root, icon, window
Causes a menu separator to be inserted in the menu at the specified location. The menu label is ignored.

f.set_behavior
Accessed by means of: button, key, menu
Context: root, icon, window
Causes mwm to restart with modified behavior. f.set_behavior switches between the default OSF behavior and behavior customized by the user.

f.title
Accessed by means of: menu
Context: root, icon, window
Causes a title to be displayed in a menu.

2.1.2.3 Virtual buttons and keys

Virtual buttons and keys are buttons and keys whose assignment to your mouse buttons and keyboard keys can be defined in the following manner. mwm determines the assignment by default X server-specific from the file xmbind.alias in the directory specified by the XMBINDDIR environment variable. The client obtains from this the name of the file from which it reads the virtual key and button assignment. If the environment variable is not set, the client looks for the file in the `/usr/lib/Xm/bindings` directory.
You determine the assignment in the $HOME/.motifbind file or by means of the defaultVirtualBindings resource (class: DefaultVirtualBindings), which mwm evaluates upon starting.

2.1.2.4 Syntax of a line in the file $HOME/.motifbind:

virtual_key:[*modifier*]...<Key>*key*

virtualKey	Default	virtualKey	Default
osfActivate	*unbound*	osfLeft	Left
osfAddMode	Shift F8	osfMenu	F4
osfBackSpace	Backspace	osfMenuBar	F10
osfBeginLine	Home	osfPageDown	Next
osfClear	Clear	osfPageUp	Prior
osfCopy	*unbound*	osfPaste	*unbound*
osfCut	*unbound*	osfPrimaryPaste	*unbound*
osfDelete	Delete	osfQuickPaste	*unbound*
osfDown	Down	osfRight	Right
osfEndLine	End	osfSelect	Select
osfCancel	Escape	osfUndo	Undo
osfHelp	F1	osfUp	Up
osfInsert	Insert		

Table 11: Replacement options for virtualKey in .motifbind

Modifier key	Replacement	Modifier key	Replacement
Ctrl	CTRL key	Mod1	Modifier key 1
Shift	Shift key	Mod2	Modifier key 2
Meta	Meta key	Mod3	Modifier key 3
Lock	LOCK key	Mod4	Modifier key 4
		Mod5	Modifier key 5

Table 12: Replacement options for modifier in .motifbind

modmap will supply you with information on the mapping of the values for *modifier* to keys on the keyboard.

key For *key*, you can use any character in the ISO Latin 1 character set. xmodmap will supply you with information on which keys your display supports.

2.1.3 Key and button bindings

Key and button bindings provide you with an easy way of executing frequently used functions. Only one set of key bindings and one set of button bindings can be in effect at any one time.

2.1.3.1 Button bindings

A button binding binds an action to a button sequence. A button sequence can involve either a single button or a button used in conjunction with one or more modifiers. Thus you could specify that an mwm function is to be performed when you press mouse button 1 and the ALT key simultaneously.
A set of button bindings is a number of button bindings grouped together under one name. This allows you to enable all the button bindings that have been grouped together under that name simultaneously using the buttonBindings resource instance (class: ButtonBindings).
On startup mwm reads the current set of button bindings from the file whose name is assigned to the configFile resource instance of class ConfigFile.

2.1.3.2 Button binding syntax

The general syntax for defining a button binding is as follows:

```
Buttons name_of_buttonbinding
{
button_sequence1 context1.1[|context1.2]... mwm-function1
```

button_sequence$_2$ *context*$_{2.1}$[|*context*$_{2.2}$]... *mwm-function*$_2$

button_sequence$_n$ *context*$_{n.1}$[|*context*$_{n.2}$]... *mwm-function*$_n$

}

button_sequence: The button sequence *button_sequence* is used to quickly execute the mwm function bound to this button sequence. You must use the following syntax for the button sequence:
[*modifier*...]<*button_action*>

Modifier key	Replacement	Modifier key	Replacement
Alt	Alt key, same as Meta	Mod1	Modifier key 1
Ctrl	CTRL key	Mod2	Modifier key 2
Shift	Shift key	Mod3	Modifier key 3
Meta	Meta key, same as Alt	Mod4	Modifier key 4
Lock	LOCK key	Mod5	Modifier key 5

Table 13: Replacement options for modifier for button bindings

xmodmap will supply you with information on the mapping of the values for *modifier* to keys on the keyboard.

buttonaction	Replacement
Btn1Down (analogously Btn2Down, Btn3Down, Btn4Down, Btn5Down)	Press down button 1
Btn1Click (analogously Btn2Click, Btn3Click, Btn4Click, Btn5Click)	Click (press and release) button 1
Btn1Up (analogously Btn2Up, Btn3Up, Btn4Up, Btn5Up)	Release button 1
Btn1Click2 (analogously Btn2Click2, Btn3Click2, Btn4Click2, Btn5Click2)	Press and release button 1 twice. The amount of time that may elapse between the clicks is specified in the doubleClickTime resource instance (class: DoubleClickTime).

Table 14: Replacement options for buttonaction for button binding

context: The context *context* indicates which components have the keyboard input focus.

context specifies a *component* of your user interface.

context	component
root	root window
icon	icon
window	client window including window border
app	client window excluding window border
frame	window border
title	title bar
border	window border excluding title bar

Table 15: Replacement options for context for button bindings

If no window or icon has the keyboard's input focus, mwm interprets the context as "root" when a function is activated.
mwm_function: *mwm_function* can be replaced by any window manager function. You will find a description of the mwm functions in the subsection headed " "Appearance and behavior"".

2.1.3.3 Key bindings

A key binding binds an action to a key sequence. A key sequence can involve either a single key or a combination of keys. Thus you could specify that an mwm function is to be performed when you press one or more buttons.
A set of key bindings is a number of key bindings grouped together under one name. This allows you to enable all the key bindings that have been grouped together under that name simultaneously using the keyBindings resource instance (class: KeyBindings).
On startup mwm reads the current set of key bindings from the file whose name is assigned to the configFile resource instance of class ConfigFile.

2.1.3.4 Key binding syntax

The general syntax for defining a key binding is as follows:

```
Keys name_of_keybinding
{
accelerator1 context1.1[|context1.2]... mwm_function1

accelerator2 context2.1[|context2.2]... mwm_function2

acceleratorn contextn.1[|contextn.2]... mwm_functionn

}
```

accelerator: The key sequence *accelerator* is used to quickly execute the mwm function bound to this key sequence. You must use the following syntax for *accelerator*: [*modifier*]...<Key>*key*.

Modifier key	Replacement	Modifier key	Replacement
Alt	Alt key,	Mod1	Modifier

	same as Meta		key 1
Ctrl	CTRL key	Mod2	Modifier key 2
Shift	Shift key	Mod3	Modifier key 3
Meta	Meta key, same as Alt	Mod4	Modifier key 4
Lock	LOCK key	Mod5	Modifier key 5

Table 16: Replacement options for modifier for button bindings

xmodmap will supply you with information on the mapping of the values for *modifier* to keys on the keyboard.
key can be any key on the keyboard. You can use xmodmap to obtain information on the keys supported by your display.
context: *context* indicates which components have the keyboard input focus. *context* specifies a *component* of your user interface.

context	component
root	root window
icon	icon
window	client window including window border and titlebar
app	client window including window border and titlebar
frame	client window including window border and titlebar
title	client window including window border and titlebar
border	client window including window border and titlebar

Table 17: Replacement options for context for button bindings

If no window or icon has the keyboard's input focus, mwm interprets the context as "root" when a function is activated.
mwm_function: *mwm_function* can be replaced by any window manager function. You will find a description of the mwm functions in the subsection headed " "Appearance and behavior"".

2.1.4 Menus

Menus provide you with an easy way of executing functions that you use repeatedly.
A menu is a list of functions. The window manager supplies a series of menu functions, the

mwm functions. Each element in the list has a name, i.e. the menu item. The list also has a name, the menu name.
The list of functions is displayed in a window, also referred to as menu, when a predefined key or button sequence is entered. The window determines its own size, depending on the font used and the number and length of the menu items. You can then select an item from the menu using another predefined key or button sequence.
On startup mwm reads menu definitions together with key and button bindings from the file whose name is assigned to the configFile resource instance of class ConfigFile.

2.1.4.1 Predefined menus

The window manager mwm provides you with four menus: the root menu, the window menu, the icon menu and the icon box menu. These menus are described in the section "Window management" in the manual "[3]" "SINIX/windows User Environment; Introduction to Handling and Configuration".

2.1.4.2 Defining menus

The following section provides you with the information on the definition of a menu.

Menu syntax
The general syntax for defining a menu is as follows:

```
Menu menu_name
{
menu_item1[ mnemonic1][ accelerator1] mwm_function1

menu_item2[ mnemonic2][ accelerator2] mwm_function2

menu_itemn[ mnemonicn][ acceleratorn] mwm_functionn

}
```

menu_name: The menu named *menu_name* is to be defined. The name of the menu is an arbitrary string. If the string includes blanks, the entire string must be enclosed in double quotes.
The menu name is not automatically displayed in the title area of the menu. Use the mwm function f.title to achieve this by defining a menu item called *menu_name* that invokes the mwm function f.title.
menu_item: Each line defines a menu item *menu_item* that is to be included in the menu. The menu item is either a string or a bitmap.

Menu item as a string
: The string is an arbitrary string of characters. If the string includes blanks, you must enclose it in double quotes.

Menu item as a bitmap
: Instead of *menu_item* you may specify the filename of a bitmap as follows:

@~/filename
: *filename* is the relative pathname of the file, relative to your $HOME directory

@filename
: *filename* is either the absolute pathname of the file or its relative pathname, relative to the directory specified as the value for the bitmapDirectory resource

instance (class: BitmapDirectory).

mnemonic: The mnemonic *mnemonic* is used to quickly select a menu item from an open (visible) menu by pressing a single letter on the keyboard. If a mnemonic has been defined, the corresponding letter of the menu item is underlined on the menu.
A mnemonic is a letter of the alphabet that identifies a menu item uniquely within the menu. This letter must be one of the letters making up the name of the menu item. If the mnemonic is not one of these letters, mwm will not register a mnemonic for this menu item. If a mnemonic is not unique it will be used to invoke the function of the first menu item which it matches.
If the mnemonic is an uppercase letter, the mnemonic must be entered as an uppercase letter in the menu definition; the menu item must, however, be selected using a lowercase letter. Uppercase letters are not accepted as input.

accelerator: You can use the key sequence *accelerator* assigned to a menu item to quickly select the menu item without having to display the menu. The accelerator for a menu item must be unique within the context in which the menu could be displayed.
You must use the following syntax for defining *accelerator*:
[*modifier*]...<Key>*key*.

Alt	Alt key, same as Meta	Mod1	Modifier key 1
Ctrl	CTRL key	Mod2	Modifier key 2
Shift	Shift key	Mod3	Modifier key 3
Meta	Meta key, same as Alt	Mod4	Modifier key 4
Lock	LOCK key	Mod5	Modifier key 5

Table 18: Replacement options for modifier for defining an accelerator

key can be any key on the keyboard. xmodmap will supply you with information on the mapping of the values for *modifier* to keys on the keyboard.

mwm_function: *mwm_function* can be replaced by any window manager function. You will find a description of the mwm functions in the subsection headed " "Appearance and behavior"".

2.1.5 Files

The files below are listed in alphabetical order. The resource configuration files are listed only in the form of the generalized file names $XENVIRONMENT, $XUSERFILESEARCHPATH and $XFILESEARCHPATH.

/bin/sh
: is executed by the mwm function f.exec.

/usr/include/X11/bitmaps
: is the directory for mwm resource's bitmap files.

/usr/lib/X11/$LANG/system.mwmrc
: system-wide, language-dependent mwm configuration file. The file the client actually evaluates depends on the value of the configFile resource.

/usr/lib/X11/system.mwmrc

system-wide mwm configuration file. The file the client actually evaluates depends on the value of the configFile resource.

$HOME/.motifbind
: user-specific configuration file for the virtual button and key assignments.
For more information on $HOME/.motifbind, please refer to the section "Virtual buttons and keys ".

$HOME/$LANG/.mwmrc
: system-wide, language-dependent mwm configuration file. The file the client actually evaluates depends on the value of the configFile resource.

$HOME/.mwmrc
: user-specific mwm configuration file. The file the client actually evaluates depends on the value of the configFile resource.

$HOME/.Xdefaults
: user-specific configuration file

$XAUTHORITY
: user-specific authorization file
The client uses this file as a user-specific authorization file. If XAUTHORITY is not set, the client uses the .Xauthority file in the user's HOME directory.

$XENVIRONMENT
: user-specific configuration file

$XFILESEARCHPATH
: application-specific configuration file created by the client programmer

$XUSERFILESEARCHPATH
: application-specific configuration file created by the system administrator or user

2.1.6 Environment variables

Detailed descriptions of all the environment variables, arranged in alphabetical order, are given in the chapter "Environment variables ".

DISPLAY
: The client reads the DISPLAY environment variable if the -display option is not specified in its command line. From this variable the client finds out the name of the X server and the display and screen on it.
The DISPLAY environment variable is a string in the format:
hostname:*display_number*.*screen_number*
The client terminates abnormally if the DISPLAY environment variable is not set and the -display option is not specified in the client command line.

HOMEidentifies the user's HOME directory.

LANGis evaluated in the context of NLS language support .

MAIL absolute pathname of the file for the user's incoming mail

MWMSHELL
: is evaluated by the mwm function f.exec in order to execute a command.
If MWMSHELL is not set, mwm evaluates the SHELL environment variable.

SHELLspecifies which command interpreter is to be used.
: SHELL is evaluated by the mwm function f.exec in order to execute a command if MWMSHELL is not set.

XAUTHORITY

denotes the user-specific authorization file.
If the environment variable is not set, the client searches for the $HOME/.Xauthority file.

XBMLANGPATH
: defines a list of directories the client is to search for pixmaps and bitmaps if it does not find them in the directory specified by the bitmapDirectory resource of the BitmapDirectory class.

XENVIRONMENT
: specifies the path for configuration files.

XFILESEARCHPATH
: specifies the path for configuration files.

XMBINDDIR
: specifies the path for the xmbind.alias file, from which the client obtains the name of the file from which it reads the virtual key and button assignments. If the variable is not set, the client searches for the file in the /usr/lib/Xm/bindings directory.

XUSERFILESEARCHPATH
: specifies the path for configuration files.

2.2 xauth - Handling authorization information

xauth lets you add, modify and display user-specific authorization information required for communication with the X server.
The shell script xauth_remote is provided to help you transfer authorization information to your authorization file on a remote host. xauth_remote is demonstrated in the examples at the end of this description.
The authorization mechanism is used by the xdm display manager, which generates the required authorization information and passes it to the X server. A set of authorization information consists of the name of the X server, the authorization protocol identifier, and an encoded session key in hexadecimal notation. When the authorization mechanism is active for an X server, a client must present this key to the X server to verify that it is authorized to access the X server in the course of the current session. The authorization mechanism is active if the value of the xdm client's authorize resource is True.
Once a user has successfully logged in at the xdm login window, xdm stores the session key in the $XAUTHORITY file. If the XAUTHORITY environment variable is not set, the default file is .Xauthority in the user's home directory. Any clients the user starts up within the session for the associated display retrieve the session key from the authorization file and present it to the X server. When the session is terminated and a new one is started, a new key is generated and communicated to the X server.
If clients which access the display need to be run on other hosts in the network, the session key must be made available to them. This entails extracting the session key from $XAUTHORITY (or $HOME/.Xauthority) at the start of every new session and adding it to the user-specific authorization files on the remote hosts. This is one of the functions of xauth.
Note that the host-specific access authorization granted by the xhost client takes precedence over the user-specific access authorization provided by the authorization mechanism. If a host is granted access to a server by the X server's internal host list, all clients started on that host are authorized to access that server, even if they cannot supply the valid session key. Hosts on which user-specific authorization is to apply must not appear in the X server's host-specific access list. For further information refer to the xhost client description (section "xhost - Controlling access to the X server ").
xauth itself does not communicate with the X server.

xauth[-f *authfile*[*option*]...[*command*[*argument*]...]

-b See -i option
Specifies that xauth should attempt to remove all locks from the authorization file before continuing. This option should be used only to clear obsolete file locks.

Default:
xauth does not attempt to remove locks.

-f *autfile*
Specifies which authorization file the client is to use.

Default:
xauth uses the file defined in the XAUTHORITY environment variable. If this

environment variable is not set, xauth uses the .Xauthority file in the user's home directory.

-i See -b option

Tells xauth to ignore file locks on the authorization file. If this option is not set, xauth will not attempt read or write access to an authorization file locked by another process such as xdm or another xauth process.

Default:

The client respects file locks.

command[*argument*]...

The commands described below provide access to the contents of the authorization files. The commands can be entered interactively on the xauth command line or from shell scripts.

? writes a brief list of the available xauth commands to standard output.

add *display protocol key*

generates an entry for the named display in the authorization file.

If you use xdm as your display manager, xdm generates the key, passes it to the X server and adds it to the user-specific authorization file if authorization is active and the server supports this procedure. In this case you do not need the add command.

If you do not use xdm, you can also activate authorization with xinit. In this case you must generate the key for yourself, using the add command (see example in section "Generating an authorization key").

display

identifies the display for which authorization data is to be processed, in the form:

[*hostname*]:*display_number*

hostname

symbolic name of the host the X server runs on.

display_number

number uniquely identifying a display to an X server.

protocol

is the name of the authorization protocol. The only value currently valid is MIT-MAGIC-COOKIE-1. If you enter a dot as the protocol name, it will be treated as an abbreviation for MIT-MAGIC-COOKIE-1.

key is the authorization key.

exit terminates the xauth program. If modifications have been made to the authorization file, it is saved. The EOF character is interpreted as and implicit exit command.

[n]extract *file display*...

For each display you list, the authorization entry is extracted from the authorization file and written to a file.

If an n is prefixed to the command, the entries are written in a numeric format for non-binary transfer, for example secure electronic mail. The extracted entries can be read with the merge and nmerge commands.

file is the name of the file. If *file* is simply a dash (-), [n]extract writes to standard output.

help[*string*]
writes a description of the xauth command to standard output.

string
restricts the help output to xauth commands beginning with *string*.

info writes information about the authorization file to standard output. This is typically information indicating whether changes have been made, where xauth commands are read from, and so forth.

[n]list[*display*]...
writes authorization entries for each of the displays you list, or for all displays if *display* is not specified, to standard output. nlist represents the entries in the same numeric format as is generated by nextract. list represents the entries in text form. The key data are always written in hexadecimal format.

display
name of the display.
See the description of the add command ("add display protocol key").

[n]merge[*file*]...
reads the authorization entries generated by [n]extract from a file and writes them to the authorization database, overwriting any existing entry for the specified display. nmerge uses the numeric format as generated by the nextract command.

file is the name of the file. If *file* is simply a dash, the command reads from standard input.

quit terminates xauth without saving any changes. The interrupt key <DEL> has the same effect.

remove *display*...
removes the authorization entries for the listed displays from the authorization file.

display
name of the display.
See the description of the add command ("add display protocol key").

source *file*
reads xauth commands from *file* and executes them. Blank lines and line beginning with # are ignored. If *file* is simply a dash, the command reads from standard input.

-q q - quiet
See -v option
Tells xauth to suppress status messages.

Default:

This option is set if xauth is reading commands from the command line or if its standard output is not a screen.

-v See -q option
Tells xauth to display status messages indicating the results of various operations, such as the number of records read or written.

Default:
This option is set if xauth is reading commands from standard input or if its standard output is a screen.

2.2.1 Files

The files below are listed in alphabetical order.

$XAUTHORITY
user-specific authorization file
The client uses this file as a user-specific authorization file when no file is specified by means of the -f option. If XAUTHORITY is not set, the client uses the .Xauthority file in the user's HOME directory.

2.2.2 Environment variables

Detailed descriptions of all the environment variables, arranged in alphabetical order, are given in the chapter "Environment variables ".

HOMEidentifies the user's HOME directory. xauth searches this directory for the .Xauthority file.
XAUTHORITY
denotes the user-specific authorization file.
The client uses this environment variable if the -f option is not set. If the environment variable is not set, the client searches for the $HOME/.Xauthority file.

2.2.3 Examples

A common application for xauth is to make an authorization entry usable by clients running on remote hosts.
User "custer" wants to extract the authorization entry for the display "asterix:0.0" and transfer it to his authorization file on the host "distant".

User: muster
Display:
asterix:0.0

```
xauth extract - $DISPLAY | rsh distant /usr/bin/X11/xauth merge
```

The command

```
xauth_remote distant
```

has the same effect. This shell script also allows for the fact that the entries for a local display in the authorization file look different from those for remote displays and so need to

be converted.
If a different login name is involved on the remote system, this has to be indicated appropriately.

```
xauth extract - $DISPLAY | rsh distant -l apache /usr/bin/X11/xauth merge
```

This transfers the authorization data to the authorization file for the login name "apache" on the host "distant".
Note that to be able to execute these commands the user "custer" needs access permissions for the .rhosts file of the appropriate user on the remote host and that the DISPLAY environment variable must be set.

2.2.3.1 Generating an authorization key

If you do not use the xdm display manager, you can activate the authorization mechanism with the aid of the X client xinit instead of xdm. Having logged in on the host at a local alphanumeric terminal, you can start the X server by hand and activate user-specific access protection for local displays. This entails first generating a random number and writing it to the $XAUTHORITY file.
You generate a random number using the Korn shell:

```
randomkey='ksh -c 'echo $(( $RANDOM * $RANDOM * 2 ))''
```

Using xauth you add key thus generated to the authorization file:

```
xauth add ${HOST}/unix:0 . $randomkey
xauth add :0.0 . $randomkey
xauth add ${HOST}:0 . $randomkey
```

The first two entries generate an authorization entry for local communication between X clients and the X server named unix:0. The entry you use depends on how the local display is identified on your system. The second entry generates an entry for the remote display hostname:0. Now start the server with the -auth option and pass it the authorization key:

```
$ xinit -- /usr/bin/X11/X -auth $XAUTHORITY
```

2.3 xdesk - Managing desktools and files

The xdesk client provides file manager and desktop manager functionality. The desktop manager manages desktools, the file manager manages files and directories. Desktools, files and directories are the objects which can appear on the desktop, where the user can work with them interactively.
Only one instance of xdesk can be run per display. However, it is possible to start the file manager independently of the desktop manager by calling the xdfm client. This is useful if you wish to start a file manager on a remote system.
The appearance and behavior of the client are documented in help texts. As a central component of the SINIX/windows user environment, the client is described in the product's manuals from various aspects. The appearance and behavior of the client in its standard configuration is explained in the manual "[3]" "SINIX/windows User Environment; Introduction to Handling and Configuration". How the desktop and file managers work together and what they do for the user is explained in the manual "[2]" "SINIX/windows User Environment; Guide for Experts and System Administrators".
The client's appearance and behavior can be configured dynamically using the xdtell client which, among other things, allows new client windows defined in the XDCL language to be output during runtime. The xdesk client evaluates instances of the XDCL classes `"filemanager_view", "desktop_as_window" and "desktop_as_root" for this purpose.`
The client's appearance and behavior can be configured with the XDCL language. This language allows experienced users and system administrators to change the standard configuration and customize the appearance and behavior of the client to meet their requirements. The manual "[4]" "SINIX/windows User Environment; XDCL Desktop Configuration Language" is a reference work providing a complete overview of which objects the client makes available, which characteristics can be specified, and which methods can be initiated by user actions.
On startup the client evaluates the XDESK_CFG_PATH environment variable and searches the directory named there for its .xdesk.cfg configuration file, which must conform to XDCL Desktop Configuration Language XDCL syntax. The user can edit this file so as to modify the appearance and behavior of the client.
The client automatically generates save files in XDCL whenever the user configures objects interactively and asks the client to save the current state. xdesk stores save files in the directory specified by the XDESK_SAVE_DIR environment variable. When called again, the client attempts to restore the saved state. Object definitions which are in the save files but have been modified in the configuration file are ignored; the definitions in the configuration file take precedence.

File manager

The file manager manages files and directories in directory windows. A directory window shows the contents of a directory in graphical form. It is possible to open multiple directory windows for one directory, providing the user with different views of the same directory. Views are defined in the client configuration file; a view governs the appearance of the directory window, such as the menus in the menu bar and the types of file to display. Among other things, the configuration file tells the file manager which directory views and file type are defined, which pull-down menus are on the menu bar of a directory window, which items are on the pop-up menu of a file or directory icon, and what user actions initiate which methods.

Desktop manager
The desktop manager manages desktools in dedicated windows or on the root window. Among other things, the configuration file tells the desktop manager which desktools to display, whether to place them in a separate window or on the root window, which pop-up menu is defined for a desktool, and what user actions initiate which methods.

Interaction between desktop manager and file manager
The desktop manager remembers the current positions of the objects on the desktop and provides the "drag and drop" functionality which allows the user to pick up file manager objects and position them on the desktop. Dropping a file manager object on a desktool causes the desktop manager to execute the method defined in the desktool or file type configuration.
Objects from a remote file manager cannot be placed on the desktop. If a desktool is able to process an object from a remote file manager, the desktop manager initiates a method such as "edit" or "print" which runs on the remote host.

xdesk[*option*]...

-display [*hostname*]:*display_number*[.*screen_number*]
Identifies the X server, display and screen which the client is to contact.

hostname
symbolic name of the host or X terminal on which the X server which the client is to contact runs. *hostname* must be known in the transport protocol of the host on which the client resides.
display_number
number (unique to the X server) identifying the display for which the client is to manage I/O.
.screen_number
number (unique to the display) identifying the screen to which the client is to send output.

Default:
contents of the DISPLAY shell variable;
if DISPLAY is not set, the client terminates abnormally.

-xrm "*resourceDefinition*:*value*"
Assigns a value to a client resource.
-xrm must be specified separately for each resource to be set in the client call. The "*resourceDefinition*:*value*" assignment is compared with the entries in the configuration files.

resourceDefinition
Identifies the resource to which a value is to be assigned. For *resourceDefinition* you can give the name of the resource, the name of the client and of any widgets in the appropriate resource syntax (see section "Value assignment syntax ").
value Value to be assigned to the resource.

2.3.1 Resources

You can configure the client using its resource database. To do this, you must assign the required values to the resources. You can use either configuration files or the -xrm option in the client call to access resources.
The instance name of the client is xdesk and its class name XDeskmaster.

iconPath
Class: IconPath
See resources iconSet,systemIconPath
defines the list of files the client is to search for the icon image for an object that it manages.
No default

iconSet
Class: IconSet
See resources iconPath, systemIconPath
defines the set of icon images the client uses for the objects that it manages.

Default:
depends on the number of color planes on the monitor:
mono for a monitor supporting one color plane
color for a monitor supporting more than one color plane

rootDesktop
Class: RootDesktop
Specifies the placement of desktools of the SINIX/windows desktop.

Default:
False
The desktools of the SINIX/windows desktop are placed in a window.

saveOnExit
Class: SaveOnExit
Specifies whether the client saves its status when terminating.

Default:
ask

standardIconSize
Class: StandardIconSize
Size of the icons when the client is started

Default:
Auto

systemIconPath
Class: SystemIconPath
See resources iconPath, iconSet
Specifies the system-specific part of the list of files the client searches for the icon image for an object it manages.

Default:

/usr/lib/X11/siwin/icons/desktools/%I/%N:\
/usr/lib/X11/siwin/icons/filetypes/%I/%N:\
/usr/lib/X11/siwin/icons/wm/%I/%N:\
/usr/lib/X11/siwin/icons/clients/%I/%N:\
/usr/lib/X11/siwin/icons/users/%I/%N:\
/usr/lib/X11/siwin/icons/hosts/%I/%N:\
usr/lib/X11/siwin/icons/mail/%I/%N:\
/usr/lib/X11/siwin/icons/common/%I/%N

2.3.2 Files

The files below are listed in alphabetical order. The resource configuration files are listed only in the form of the generalized file names $XENVIRONMENT, $XUSERFILESEARCHPATH and $XFILESEARCHPATH.

.interact.cfg
: save file stored in the directory defined in the XDESK_SAVE_DIR environment variable after interactive configuration and saving of the client.

.xddm.session
: desktop manager save file, stored in the directory defined in the XDESK_SAVE_DIR environment variable.

.xdesk.cfg
: XDCL configuration file of the client in a directory on the path $XDESK_CFG_PATH.

.xdfm.session
: file manager save file, stored in the directory defined in the XDESK_SAVE_DIR environment variable.

$XAUTHORITY
: user-specific authorization file
The client uses this file as a user-specific authorization file. If XAUTHORITY is not set, the client uses the .Xauthority file in the user's HOME directory.

$XENVIRONMENT
: user-specific configuration file

$XFILESEARCHPATH
: application-specific configuration file created by the client programmer

$XUSERFILESEARCHPATH
: application-specific configuration file created by the system administrator or user

2.3.3 Environment variables

Detailed descriptions of all the environment variables, arranged in alphabetical order, are given in the chapter "Environment variables ".

DISPLAY
: The client reads the DISPLAY environment variable if the -display option is not specified in its command line. From this variable the client finds out the name of the X server and the display and screen on it.

The DISPLAY environment variable is a string in the format:
hostname:*display_number*.*screen_number*
The client terminates abnormally if the DISPLAY environment variable is not set and the -display option is not specified in the client command line.

HOMEidentifies the user's HOME directory.

LANGs evaluated in the context of NLS language support .

NLSPATH
The client evaluates the NLSPATH environment variable as a search path for message files.
Thus the NLSPATH environment variable must include the search path /usr/lib/nls/msg/%l/%N.cat. "%l" is replaced by that part of the locale definition which identifies the language. The client replaces "%N" with "xdesk".

TRASHdenotes the directory in which the desktop and file managers create the directory for trash. The name of the directory for trash begins with a period (.) and is therefore only visible in $TRASH if these file and directory names are displayed. If the variable is not set, the client evaluates the HOME environment variable.

XAUTHORITY
denotes the user-specific authorization file.
If the environment variable is not set, the client searches for the $HOME/.Xauthority file.

XDESK_CFG_PATH
defines a list of directories to search for the client's XDCL configuration file. If the variable is not set, the client evaluates the HOME environment variable. If there is no XDCL configuration file of the client in $HOME, the client reads from the directory /usr/lib/X11/xdesk/cfg the XDCL configuration file xdesk.cfg.

XDESK_ICON_PATH
defines a user-specific list of files the client is to scan for the file containing the icon image for an object that it manages. If this environment variable is set, it overrides the value of the iconPath resource of class IconPath. The list may include the symbolic names described in the iconPath resource.

XDESK_SAVE_DIR
defines the directory in which the client stores its save files. If the variable is not set, the client evaluates the HOME environment variable.

XDESK_SYSTEM_ICON_PATH
defines a system-specific list of files the client is to scan for the file containing the icon image for an object that it manages. If this environment variable is set, it overrides the value of the systemIconPath resource of class SystemIconPath. The list may include the symbolic names described in the systemIconPath resource.

XENVIRONMENT
specifies the path for configuration files.

XFILESEARCHPATH
specifies the path for configuration files.

XUSERFILESEARCHPATH
specifies the path for configuration files.

2.4 xdeskbox - Creating dialog boxes

xdeskbox enables you to create various types of Motif dialog boxes. Before the client can create a dialog box, the xdesk or xdfm client must have been started for the display.
The client is capable of creating information, warning, question, working, error and prompt dialog boxes. xdeskbox returns information to its calling environment which reflects the action the user took in response to the dialog box.
The client exit status codes are as follows:

0 dialog box "OK" button pressed
1 dialog box "Cancel" button pressed
2 dialog box "Help" button pressed
9 -help option specified, or call contained invalid arguments
255 connection setup to display or xdesk client or xdfm client unsuccessful

If dialog box with a text entry box has been created, xdeskbox writes the text entered by the user to standard output.
xdeskbox[*option*]...[*string*]
The options -error, -info, -prompt, -question, -warning and -working are mutually exclusive.

-addhelp
Specifies the displayed dialog box contains a help button, which is positioned to the right of any other buttons.

Default:
The dialog box does not have a "Help" button.

-cancel *string*
Specifies the label *string* to be used for the cancel button.
Multiple words can be used as a label by enclosing them in quotes. If a cancel button is not present, this option has no effect.

-error
causes the dialog box created to be a "error" dialog box.

Default:
The client creates an information box.

-geometry *width*x*height*+*x_off*+*y_off*
Defines the size and position of the client window.

width the window width.
height
the window height.
x_off Displacement of the window along the x axis (horizontally) in pixels

+*x_off*
distance between left edge of window and screen
-*x_off*
distance between right edge of window and screen

y_off Displacement of the window along the y axis (vertically) in pixels

+*y_off*
distance between top edge of window and screen
-*y_off*
distance between bottom edge of window and screen
No default value.

-help Writes a short description of the options to standard error.
This help text is also displayed if the client cannot interpret a command-line argument.

-helplabel *string*
Allows you to specify a text string, *string*, to be used in place of the default string as the label for the dialog box help button.
Multiple words can be used as the label by enclosing them in quotes. If the dialog box does not contain a help button, this option has no effect.

-info Causes the dialog box created to be an "information" dialog box.
This is the default.

-nocancel
Causes the dialog box to be created without a cancel button.

Default:
The dialog box has a "Cancel" button.

-ok *string*
Allows you to specify a text string, *string*, to be used in place of the default string as the label for the dialog box ok button.
Multiple words can be used as the label by enclosing them in quotes.

-prompt
Creates a prompt box (a dialog box with a text entry field).

Default:
The client creates an information box.

-question
causes the dialog box created to be a "question" dialog box.

Default:
The client creates an information box.

-title *string*
Allows you to place the default dialog box title with the string *string*.
Multiple words can be used as the title by enclosing them in quotes.

Default:
The title of the dialog box is "Desktop".

-warning
Causes the dialog box created to be a "warning" dialog box.

Default:

The client generates an information box.

-working
Causes the dialog box created to be a "working" dialog box.

Default:
The client generates an information box.

string
This is a single argument specifying the text to be displayed in the dialog box.
Only the last non-option argument of xdeskbox is taken as the text to be displayed, so text consisting of multiple words must be enclosed in quotes.

Default:
The text displayed is "xdeskbox".

2.4.1 Environment variables

Detailed descriptions of all the environment variables, arranged in alphabetical order, are given in the chapter "Environment variables ".

DISPLAY
The client reads the DISPLAY environment variable if the -display option is not specified in its command line. From this variable the client finds out the name of the X server and the display and screen on it.
The DISPLAY environment variable is a string in the format:
hostname:*display_number*.*screen_number*
The client terminates abnormally if the DISPLAY environment variable is not set and the -display option is not specified in the client command line.

2.5 xdfm - File manager

The xdfm client manages file and directory objects in directory windows. A directory window shows the contents of a directory in graphical form. and directory icons. It is possible to open multiple directory windows for one directory, providing the user with different views of the same directory. Views are defined in the client configuration file; a view governs the characteristics of the directory window, such as the menus in the menu bar and the types of file to display.
The file manager is available after the xdesk client has been called. Only one instance of the file manager per system and user can be run on a display. However, it is possible to start a file manager independently of the desktop manager by calling the xdfm client. This is useful if you wish to start a file manager on a remote system.
xdfm provides "drag and drop" functionality between directory windows, allowing the user to move or copy files or directories by dragging them to a new location. In drag-and-drop operations between a local file manager and a remote file manager, objects can only be copied, not moved.
The appearance and behavior of the client are documented in help texts. As a central component of the SINIX/windows user environment, the client is described in the product's manuals from various aspects. The appearance and behavior of the client in its standard configuration is explained in the manual "[3]" "SINIX/windows User Environment; Introduction to Handling and Configuration". How the desktop and file managers work together and what they do for the user is explained in the manual "[2]" "SINIX/windows User Environment; Guide for Experts and System Administrators".
The client's appearance and behavior can be configured dynamically using the xdtell client which, among other things, allows new client windows defined in the XDCL language to be output during runtime. The xdfm client evaluates instances of the XDCL class "`filemanager_view`" for this purpose.
The client's appearance and behavior can be configured with the XDCL language. This language allows experienced users and system administrators to change the standard configuration and customize the appearance and behavior of the client to meet their requirements. The manual "[4]" "SINIX/windows User Environment; XDCL Desktop Configuration Language" is a reference work providing a complete overview of which objects the client makes available, which characteristics can be specified, and which methods can be initiated by user actions.
On startup the client evaluates the XDESK_CFG_PATH environment variable and searches the directory named there for its .xdesk.cfg configuration file, which must conform to XDCL Desktop Configuration Language XDCL syntax. The user can edit this file so as to modify the appearance and behavior of the client.
The client automatically generates save files in XDCL whenever the user configures objects interactively and asks the client to save the current state. xdesk stores save files in the directory specified by the XDESK_SAVE_DIR environment variable. When called again, the client attempts to restore the saved state. Object definitions which are in the save files but have been modified in the configuration file are ignored; the definitions in the configuration file take precedence.
Among other things, the configuration file tells the file manager which directory views and file type are defined, which pull-down menus are on the menu bar of a directory window, which items are on the pop-up menu of a file or directory icon, and what user actions initiate which methods.

xdfm[*option*]...

-display [*hostname*]:*display_number*[.*screen_number*]
Identifies the X server, display and screen which the client is to contact.

hostname
symbolic name of the host or X terminal on which the X server which the client is to contact runs. *hostname* must be known in the transport protocol of the host on which the client resides.
display_number
number (unique to the X server) identifying the display for which the client is to manage I/O.
.screen_number
number (unique to the display) identifying the screen to which the client is to send output.

Default:
contents of the DISPLAY shell variable;
if DISPLAY is not set, the client terminates abnormally.

-open *directory* [*view*]
Starts the file manager with the view *view* of the directory *directory*.
view is an instance of the XDCL object class "filemanager_view" defined in the configuration file, which is to be used as a view. If *view* is not specified, the client uses the "default" instance of the "filemanager_view" class.

Default:
If there is save information, the file manager opens all windows recorded there. If there is no save information, the file manager opens a window on the directory identified by the "initial_folder" attribute in the "default" instance of the XDCL object class "filemanager_view". If "initial_folder" is not specified, the $HOME environment variable is used.

-xrm "*resourceDefinition*:*value*"
Assigns a value to a client resource.
-xrm must be specified separately for each resource to be set in the client call. The "*resourceDefinition*:*value*" assignment is compared with the entries in the configuration files.

resourceDefinition
Identifies the resource to which a value is to be assigned. For *resourceDefinition* you can give the name of the resource, the name of the client and of any widgets in the appropriate resource syntax (see section "Value assignment syntax ").
value Value to be assigned to the resource.

2.5.1 Resources

You can configure the client using its resource database. To do this, you must assign the

required values to the resources. You can use either configuration files or the -xrm option in the client call to access resources.
The instance name of the client is xdfm and its class name XDeskmaster.

iconPath
Class: IconPath
See resources iconSet, systemIconPath
defines the list of files the client is to search for the icon image for an object that it manages.
No default
iconSet
Class: IconSet
See resources iconPath, systemIconPath
defines the set of icon images the client uses for the objects that it manages.

Default:
depends on the number of color planes on the monitor:

mono for a monitor supporting one color plane
color for a monitor supporting more than one color plane
saveOnExit
Class: SaveOnExit
Specifies whether the client saves its status when terminating.

Default:
ask
standardIconSize
Class: StandardIconSize
Size of the icons when the client is started

Default:
Auto
systemIconPath
Class: SystemIconPath
See resources iconPath, iconSet
Specifies the system-specific part of the list of files the client searches for the icon image for an object it manages.

Default:
/usr/lib/X11/siwin/icons/desktools/%I/%N:\
/usr/lib/X11/siwin/icons/filetypes/%I/%N:\
/usr/lib/X11/siwin/icons/wm/%I/%N:\
/usr/lib/X11/siwin/icons/clients/%I/%N:\
/usr/lib/X11/siwin/icons/users/%I/%N:\
/usr/lib/X11/siwin/icons/hosts/%I/%N:\

/usr/lib/X11/siwin/icons/mail/%I/%N:\
/usr/lib/X11/siwin/icons/common/%I/%N

2.5.2 Files

The files below are listed in alphabetical order. The resource configuration files are listed only in the form of the generalized file names $XENVIRONMENT, $XUSERFILESEARCHPATH and $XFILESEARCHPATH.

.interact.cfg
: save file stored in the directory defined in the XDESK_SAVE_DIR environment variable after interactive configuration and saving of the client.

.xdesk.cfg
: XDCL configuration file of the client in a directory on the path $XDESK_CFG_PATH.

.xdfm.session
: file manager save file, stored in the directory defined in the XDESK_SAVE_DIR environment variable.

$XAUTHORITY
: user-specific authorization file
The client uses this file as a user-specific authorization file. If XAUTHORITY is not set, the client uses the .Xauthority file in the user's HOME directory.

$XENVIRONMENT
: user-specific configuration file

$XFILESEARCHPATH
: application-specific configuration file created by the client programmer

$XUSERFILESEARCHPATH
: application-specific configuration file created by the system administrator or user

2.5.3 Environment variables

Detailed descriptions of all the environment variables, arranged in alphabetical order, are given in the chapter "Environment variables ".

DISPLAY
: The client reads the DISPLAY environment variable if the -display option is not specified in its command line. From this variable the client finds out the name of the X server and the display and screen on it.
The DISPLAY environment variable is a string in the format:
hostname:*display_number*.*screen_number*
The client terminates abnormally if the DISPLAY environment variable is not set and the -display option is not specified in the client command line.

HOMEidentifies the user's HOME directory.

LANGis evaluated in the context of NLS language support.

NLSPATH
: The client evaluates the NLSPATH environment variable as a search path for message files.
Thus the NLSPATH environment variable must include the search path /usr/lib/nls/msg/%l/%N.cat. "%l" is replaced by that part of the locale definition which

identifies the language. The client replaces "%N" with "xdesk".

TRASH
denotes the directory in which the desktop and file managers create the directory for trash. The name of the directory for trash begins with a period (.) and is therefore only visible in $TRASH if these file and directory names are displayed. If the variable is not set, the client evaluates the HOME environment variable.

XAUTHORITY
denotes the user-specific authorization file.
If the environment variable is not set, the client searches for the $HOME/.Xauthority file.

XDESK_CFG_PATH
defines a list of directories to search for the client's XDCL configuration file. If the variable is not set, the client evaluates the HOME environment variable. If there is no XDCL configuration file of the client in $HOME, the client reads from the directory /usr/lib/X11/xdesk/cfg the XDCL configuration file xdesk.cfg.

XDESK_ICON_PATH
defines a user-specific list of files the client is to scan for the file containing the icon image for an object that it manages. If this environment variable is set, it overrides the value of the iconPath resource of class IconPath. The list may include the symbolic names described in the iconPath resource.

XDESK_SAVE_DIR
defines the directory in which the client stores its save files. If the variable is not set, the client evaluates the HOME environment variable.

XDESK_SYSTEM_ICON_PATH
defines a system-specific list of files the client is to scan for the file containing the icon image for an object that it manages. If this environment variable is set, it overrides the value of the systemIconPath resource of class SystemIconPath. The list may include the symbolic names described in the systemIconPath resource.

XENVIRONMENT
specifies the path for configuration files.

XFILESEARCHPATH
specifies the path for configuration files.

XUSERFILESEARCHPATH
specifies the path for configuration files.

2.6 xdhm - Managing hosts

The xdhm client, the host manager, allows you to display the hosts and networks known to your host, as well as some of their characteristics. In addition, it enables system administrators to manage these hosts and networks as objects and to administer the NIS services. They can, for example, make another host known to their own host, change the characteristics of a known host, delete a host from the list of known hosts, and interactively assign to hosts and networks icons used, for example, by xdhm in the window for host selection. The client can run under a login name on a given host only once per display.
The appearance and behavior of the client are documented in help texts.
The client's appearance and behavior can be configured dynamically using the xdtell client which, among other things, allows new client windows defined in the XDCL language to be output during runtime. The xdhm client evaluates instances of the XDCL class "hostmanager_view" for this purpose.
The client's appearance and behavior can be configured with the XDCL language. This language allows experienced users and system administrators to change the standard configuration and customize the appearance and behavior of the client to meet their requirements. The manual "[4]" "SINIX/windows User Environment; XDCL Desktop Configuration Language" is a reference work providing a complete overview of which objects the client makes available, which characteristics can be specified, and which methods can be initiated by user actions.
On startup the client evaluates the XDESK_CFG_PATH environment variable and searches the directory named there for its .xdhm.cfg configuration file, which must conform to XDCL Desktop Configuration Language XDCL syntax. The user can edit this file so as to modify the appearance and behavior of the client.
The client automatically generates save files in XDCL whenever the user configures objects interactively and asks the client to save the current state. xdesk stores save files in the directory specified by the XDESK_SAVE_DIR environment variable. When called again, the client attempts to restore the saved state. Object definitions which are in the save files but have been modified in the configuration file are ignored; the definitions in the configuration file take precedence.

xdhm[*option*]...

-display [*hostname*]:*display_number*[.*screen_number*]
Identifies the X server, display and screen which the client is to contact.

hostname
symbolic name of the host or X terminal on which the X server which the client is to contact runs. *hostname* must be known in the transport protocol of the host on which the client resides.
display_number
number (unique to the X server) identifying the display for which the client is to manage I/O.
.screen_number
number (unique to the display) identifying the screen to which the client is to send outputs.

Default:
contents of the DISPLAY shell variable;
if DISPLAY is not set, the client terminates abnormally.

-local *hostname*
Starts the host manager with the NIS administration window for the local client. Only the NIS services can then be administered with the host manager.

hostname
Identifies the local host.

Default:
no default

-open *network* [*view*]
Starts the host manager with the view *view* on the network *network*.
view is an instance of the XDCL object class "hostmanager_view" defined in the configuration file, which is to be used as a view. If *view* is not specified, the client uses the "default" instance of the "hostmanager_view" class.

Default:
If there is save information, the host manager opens all windows recorded there. If there is no save information, the host manager opens a window with a view on the network identified by the "initial_folder" attribute in the "default" instance of the XDCL object class "hostmanager_view". If "initial_folder" is not specified, all hosts known to the local host are displayed.

-xrm "*resourceDefinition*:*value*"
Assigns a value to a client resource.
-xrm must be specified separately for each resource to be set in the client call. The "*resourceDefinition*:*value*" assignment is compared with the entries in the configuration files.

resourceDefinition
Identifies the resource to which a value is to be assigned. For *resourceDefinition* you can give the name of the resource, the name of the client and of any widgets in the appropriate resource syntax (see section "Value assignment syntax ").

value Value to be assigned to the resource.

2.6.1 Resources

You can configure the client using its resource database. To do this, you must assign the required values to the resources. You can use either configuration files or the -xrm option in the client call to access resources.
The instance name of the client is xdhm and its class name HostManager.

iconPath
Class: IconPath
See resources iconSet, systemIconPath

defines the list of files the client is to search for the icon image for an object that it manages.
No default

iconSet
Class: IconSet
See resources iconPath, systemIconPath
defines the set of icon images the client uses for the objects that it manages.

Default:
depends on the number of color planes on the monitor:

mono for a monitor supporting one color plane
color for a monitor supporting more than one color plane

saveOnExit
Class: SaveOnExit
Specifies whether the client saves its status when terminating.

Default:
ask

standardIconSize
Class: StandardIconSize
Size of the icons when the client is started

Default:
Auto

systemIconPath
Class: SystemIconPath
See resources iconPath, iconSet
Specifies the system-specific part of the list of files the client searches for the icon image for an object it manages.

Default:
/usr/lib/X11/siwin/icons/hosts/%I/%N:\
/usr/lib/X11/siwin/icons/wm/%I/%N:\
/usr/lib/X11/siwin/icons/common/%I/%N

2.6.2 Files

The files below are listed in alphabetical order. The resource configuration files are listed only in the form of the generalized file names $XENVIRONMENT, $XUSERFILESEARCHPATH and $XFILESEARCHPATH.

.xdhm.cfg
XDCL configuration file of the client in a directory on the path $XDESK_CFG_PATH.
.xdhm.ict

Save file created after interactive configuration and saving of the client in the directory defined in the XDESK_SAVE_DIR environment variable.

.xdhm.session
Save file of the client for saving views on networks. It is created in the directory defined in the XDESK_SAVE_DIR environment variable.

$XAUTHORITY
user-specific authorization file
The client uses this file as a user-specific authorization file. If XAUTHORITY is not set, the client uses the .Xauthority file in the user's HOME directory.

$XENVIRONMENT
user-specific configuration file

$XFILESEARCHPATH
application-specific configuration file created by the client programmer

$XUSERFILESEARCHPATH
application-specific configuration file created by the system administrator or user

2.6.3 Environment variables

Detailed descriptions of all the environment variables, arranged in alphabetical order, are given in the chapter "Environment variables ".

DISPLAY
The client reads the DISPLAY environment variable if the -display option is not specified in its command line. From this variable the client finds out the name of the X server and the display and screen on it.
The DISPLAY environment variable is a string in the format:
hostname:*display_number*.*screen_number*
The client terminates abnormally if the DISPLAY environment variable is not set and the -display option is not specified in the client command line.

HOMEidentifies the user's HOME directory.

LANGis evaluated in the context of NLS language support .

NLSPATH
The client evaluates the NLSPATH environment variable as a search path for message files.
Thus the NLSPATH environment variable must include the search path /usr/lib/nls/msg/%l/%N.cat. "%l" is replaced by that part of the locale definition which identifies the language. The client replaces "%N" with "xdhm".

XAUTHORITY
denotes the user-specific authorization file.
If the environment variable is not set, the client searches for the $HOME/.Xauthority file.

XDESK_CFG_PATH
defines a list of directories to search for the client's XDCL configuration file. If the variable is not set, the client evaluates the HOME environment variable. If there is no XDCL configuration file of the client in $HOME, the client reads from the directory /usr/lib/X11/xdesk/cfg the XDCL configuration file xdhm.cfg.

XDESK_ICON_PATH
defines a user-specific list of files the client is to scan for the file containing the icon

image for an object that it manages. If this environment variable is set, it overrides the value of the iconPath resource of class IconPath. The list may include the symbolic names described in the iconPath resource.

XDESK_SAVE_DIR
: defines the directory in which the client stores its save files. If the variable is not set, the client evaluates the HOME environment variable.

XDESK_SYSTEM_ICON_PATH
: defines a system-specific list of files the client is to scan for the file containing the icon image for an object that it manages. If this environment variable is set, it overrides the value of the systemIconPath resource of class SystemIconPath. The list may include the symbolic names described in the systemIconPath resource.

XENVIRONMENT
: specifies the path for configuration files.

XFILESEARCHPATH
: specifies the path for configuration files.

XUSERFILESEARCHPATH
: specifies the path for configuration files.

2.7 xdm - Display manager

The xdm client manages displays. It can manage several displays, both on its local host and on remote hosts in the network. xdm is started when the system is booted.
The display manager xdm provides services similar to those provided by init, getty and login on character-based monitors. In a login dialog, xdm prompts for the user's login name and password and checks the user's authorization to work on the host. It thus plays a major part in assuring system and network security. When configured appropriately, xdm outputs a host selection window that allows users to choose which host they want to log in at.
xdm continues managing the display until the connection is terminated by the X server or by xdm. Any number of user sessions may take place during this period. At the end of the session, the client resets the X server, and a new connection is established between the X server and xdm. xdm redisplays the login window or a host selection window and starts a new session when requested to do so.
The client is particularly useful for managing X terminals, as it is capable of managing displays on any host in a network. xdm can handle multiple displays with common characteristics as a class, thus be capable of differentiating between terminals with monochrome monitors and terminals with color monitors.
xdm is primarily a client which helps the system administrator to configure user session handling. The system-wide configuration of xdm is designed to allow users to customize their sessions to meet their own requirements. The shell script in the $HOME/.xsession file contains the user's session setup preferences. This file overrides the system-wide session management specifications.

xdm[***option*****]...**

-config *pathname*

defines the full pathname of the xdm-config file (see section "Appearance and behavior", subsection " "xdm configuration files"").
The file lists the values of resources governing the behavior of xdm on a display-specific basis.

Default:

/usr/lib/X11/xdm/xdm-config

-daemon

See -debug, -nodaemon options
See resources "daemonMode" and "debugLevel"
Starts the client in the background.
The client makes itself independent of the parent process by spawning a new process with *fork*. It also closes all file descriptors and is therefore no longer associated with the controlling terminal.

Default:

The client starts in the background.

-debug *debug_level*

See options -daemon, -nodaemon
See resources "daemonMode" and "debugLevel"

Specifies the numeric value for the debugLevel resource.
A value greater than zero causes xdm to write error messages to the screen. In addition, the -nodaemon option is automatically set. This prevents the daemonMode resource from forcing xdm to run in the background.
This option is useful for diagnostic purposes only. The recommended value for diagnostic purposes is 9.

Default:
0

-error *error_log_file*
See "errorLogFile" resource
Defines the full pathname of the xdm-errors file, which is where xdm writes error messages not assignable to a user session.
If the client cannot be started normally, or if it does not function as expected, you will find indications as to the cause in this file. The contents of the file are deleted whenever xdm is started.
Errors that xdm reports during a user session are written to the appropriate $HOME/.xerrors file.

Default:
/usr/lib/X11/xdm/xdm-errors

-nodaemon
See options -daemon, -debug
Starts the client in the foreground.
This option is useful for diagnostic purposes only.

Default:
The client starts in the background.

-resources *filename*
See ' "resources"' resource
Defines the file name of the Xresources file (see "Appearance and behavior", subsection "xdm configuration files").
This file contains resource definitions for the xdm login window displayed by the X client XLogin, for the host selection window displayed by the X client Chooser, and for the Xsetup file.

Default:
/usr/lib/X11/xdm/%l/Xresources:/usr/lib/X11/Xresources

-server *pathname*
See "servers" resource
Defines the full pathname of the Xservers file (see "Appearance and behavior", subsection "xdm configuration files") containing a list of the X servers to which xdm is to set up connections.
X servers which call up xdm via the X Display Manager Control Protocol (XDMCP) on their own initiative must not appear in this file.

Default:
/usr/lib/X11/xdm/Xservers

-session *pathname*
See "session" resource
Defines the full pathname of the Xsession file containing system-wide and user-specific session management definitions (see "Appearance and behavior", subsection "xdm configuration files").

Default:
/usr/lib/X11/xdm/Xsession

-udpPort *portNumber*
See "requestPort" resource
Specifies the port number at which xdm receives requests in accordance with XDMCP on the basis of the UDP protocol.
You should not change the port number unless it is necessary for error diagnosis.

Default:
177

-xrm "*resourceDefinition*:*value*"
Assigns a value to a client resource.
-xrm must be specified separately for each resource to be set in the client call. The "*resourceDefinition*:*value*" assignment is compared with the entries in the configuration files.

resourceDefinition
Identifies the resource to which a value is to be assigned. For *resourceDefinition* you can give the name of the resource, the name of the client and of any widgets in the appropriate resource syntax (see section "Value assignment syntax ").

value Value to be assigned to the resource.

2.7.1 Resources

You can configure the client using its resource database. To do this, you must assign the required values to the resources. You can use either the -xrm option in the client call or configuration files to access the resources of xdm. The instance name of the client is xdm and its class name DisplayManager. xdm generates the login window using xlogin, class XLogin. chooser, class Chooser, displays the host selection window and sets up the connection to the selected host.

2.7.1.1 Resource overview

This subsection starts with overviews showing which resources govern display behavior on a display-specific basis and which resources apply to the xdm login window. Then come brief descriptions of the resources listed alphabetically according to resource name.
Some display-specific resources modify the behavior of xdm for all displays, while others can act on display classes and individual displays. Where resources are to apply only to a single display class or a single display, the class or display name must be inserted into the

resource definition between the name "DisplayManager" and the resource name. At this point the name of the display must be specified without its .*screen_number* extension (see "DISPLAY ").
For example, the resource definition "DisplayManager.expo_0.startup" refers the startup resource for display "expo:0". If there is a colon (:) or a dot (.) in the display name, it must be replaced with un underscore (_) in the resource definition, as colon and dots have special meanings in resource assignment syntax.

2.7.1.2 xdm resources

The following resources govern the behavior of xdm for all displays:
Syntax:
DisplayManager.*resource*

Name	Value	Default	Option
accessFile	pathname	-[1]	
allowIcons	True|False	True	
authDir	pathname	-[1]	
autoRescan	True|False	True	
debugLevel	True|False	True	-debug
daemonMode	True|False	True	-daemon
errorLogFile	pathname	-[1]	-error
exportList	string		
keyFile	pathname	-[1]	
lockPidFile	True|False	True	
noIconUsers	pathname	-[1]	
onlyLocalUser	True|False	True	
pidFile	pathname	-[1]	
randomFile	pathname	-[1]	
removeDomainname	True|False	True	
requestPort	portnumber	177	
servers	pathname	-[1]	-servers
[1]) cf. resource description			

Table 19: xdm resources: behavior for all displays

The following resources can influence the behavior of xdm for all displays, for display classes or for single displays:
Syntax:

DisplayManager**resource*
DisplayManager.*displayclass*.*resource*
DisplayManager.*display*.*resource*

Name	Value	Default	Option
authComplain	True|False	False	
authFile	pathname	-[1]	
authName	name	-[1]	
authorize	True|False	True	
chooser	pathname	-[1]	
cpp	pathname	-[1]	
failsafeClient	pathname	-[1]	
grabServer	True|False	True	
grabTimeout	sec	3	
init	pathname	-[1]	
languageArgument	string	-[1]	
languageEntries	string	-[1]	
languageSwitch	True|False	True	
mailPath	pathname	-[1]	
openDelay	sec	15	
openRepeat	number	5	
openTimeout	sec	30	
pingInterval	min	5	
pingTimeout	min	5	
reset	pathname		
resetForAuth	True|False	False	
resetSignal	number	1	
resources	pathname	-[1]	-resources
session	pathname	-[1]	-session
sessionArgument	string	-[1]	
sessionEntries	string	-[1]	
sessionSwitch	True|False	True	

setup	pathname	-[1]	
showIcon	True|False	True	
startAttempts	number	-[1]	
startup	pathname	-[1]	
systemPath	pathname	-[2]	
systemShell	name	/bin/sh	
terminateServer	True|False	True	
termSignal	number	15	
userAuthDir	pathname	/tmp	
userPath	list	-[2]	
xrdb	pathname	-[1]	
[1]) cf. resource description [2]) depends on values of other resources			

Table 20: xdm resources: behavior for all displays, for display classes, or for single displays

2.7.1.3 Resources for the login window

The following resources govern the appearance of the xdm login window. They are defined in the Xresources file:

Syntax:

xlogin.Login.*resource*

Name	Value	Default
browserHasFocus	True|False	True
defaultIcon32	pathname	-[1]
defaultIcon64	pathname	-[1]
defaultUser	login name	-[1]
fail	string	-[1]
failColor	color	Black
failFont	fontname	-[1]
failTimeout	sec	10
foreground	color	Black
font	fontname	-[1]
greeting	string	-[1]
greetColor	color	Black
greetFont	fontname	-[1]
height	dimension	-[1]

hostname	string	-[1]
iconBackground	color	White
iconForeground	color	Black
iconLabelFont	fontname	-[1]
iconPath	string	-[1]
iconSet	string	-[1]
languageArgument	string	-[1]
languagePrompt	pathname	-[1]
loginLogo	pathname	-[1]
namePrompt	string	-[1]
onlyIconLabel	True|False	False
passwdPrompt	string	-[1]
promptFont	fontname	-[1]
promptColor	color	Black
rootBackground	color	-[1]
rootBackgroundPixmap	pixmap	-[1]
sessionArgument	string	-[1]
sessionPrompt	pathname	-[1]
showHostname	True|False	True
showIcons	True|False	True
showLogo	True|False	True
smallIconSize	True|False	False
translations	string	-[1]
width	dimension	
x	position	-[1]
y	position	-[1]
[1]) cf. resource description		

Table 21: xdm resources: login window

2.7.1.4 Resources for the host selection window

The following resources govern the appearance of the window xdm displays when users want to be able to choose for themselves the host whose xdm services they want to utilize. The resource definitions are stored in the Xresources file:

Syntax:

Chooser*_resource_

Name	Value	Default
defaultIcon32	pathname	-[1]
defaultIcon64	pathname	-[1]
defaultHost	string	
greeting	string	-[1]
greetFont	fontname	-[1]
greetColor	color	Black
height	dimension	-[1]
iconBackground	string	-[1]
iconForeground	string	-[1]
iconLabelFont	string	-[1]
iconPath	string	-[1]
iconSet	string	-[1]
loginLogo	pathname	-[1]
onlyIconLabel	True|False	False
showLogo	True|False	True
smallIconSize	True|False	False
width	dimension	
x	position	-[1]
y	position	-[1]
[1]) cf. resource description		

Table 22: xdm resources for the host selection window

2.7.1.5 Resource list

The resources available to this client are listed in alphabetical order below. You can also use them in the -xrm option.

accessFile
Class: AccessFile

Defines the full pathname of the Xaccess file which controls access for XDMCP requests.

Default:

/usr/lib/X11/xdm/Xaccess

alignment
Class: Alignment

Governs the alignment of the logo in the xdm login window or host selection window. This resource is evaluated only if the value of the showLogo resource is True.

Default:

XmALIGNMENT_BEGINNING
The logo appears on the left-hand side of the window.

allowIcons
Class: AllowIcons
See "onlyLocalUser", "noIconUsers" resources
Determines whether user login name icons are displayed in the login window on the local host.
For reasons of security it may be advisable not to display icons.

Default:
True
User login name icons are displayed.

authComplain
Class: AuthComplain
See "authorize", "unsecureGreeting" resources
Determines whether a message is displayed in the xdm login window if user-specific authorization is active for the display but is not supported by the server.
The message text can be defined using the unsecureGreeting resource.

Default:
True
A message is displayed.

authDir
Class: authDir
Defines the name of the directory in which the xdm client stores authorization information during session initialization.

Default:
/usr/lib/X11/xdm

authFile
Class: AuthFile
Defines the name of the file xdm uses to pass authorization information to the X server in conjunction with the server option -auth.
No default.

authName
Class: AuthName
Defines the mechanism used when authorization is active. The supported authorization mechanisms are MIT-MAGIC-COOKIE-1 in the international version and XDM-AUTHORIZATION-1 in the US version. If the server and xdm communicate using the XDMCP protocol, the mechanism is determined automatically and the authName resource is ignored.
Authorization is activated with the authorize resource of class Authorize.

Default:
For the international version: MIT-MAGIC-COOKIE-1
For the US version: XDM-AUTHORIZATION-1

authorize
Class: Authorize
See "authName", "authDir", "userAuthDir", "authComplain", "unsecureGreeting" resources
Defines whether the client uses a user-specific authorization mechanism based on the XDMCP protocol for local server connections.
The authName resource defines which authorization mechanism is used.

Default:
True
Authorization is active on local server connections.

autoRescan
Class: AutoRescan
Controls whether xdm automatically rescans changes to configuration files affecting the configuration, the server and the access key. If the value of the resource is True, xdm rescans updated files when setting up a connection to a server.
You can send the SIGHUP signal to the main process (i.e. the xdm process with the lowest process ID) to force the client to reread the files.

Default:
True
Auto-rescanning is performed.

browserHasFocus
Class: BrowserHasFocus
Specifies whether the input focus is on the icon browser or the login name entry box when the login window is first opened.

Default:
True

checkPathnames
Class: CheckPathnames
See "sessionArgument" resource
Specifies whether the session selection menu item defined under the name <sessionArgument> is to be displayed. The value of this resource defines a list of path names which must or must not exist in order for the session to appear in the selection list.
The path names in the list are logically combined using the Boolean operators OR (|), AND (&) and NOT (!). If the expression evaluates to True, the session is displayed on the selection menu.

Default:
No condition. Session appears on selection menu.

chooser
Class: Chooser
Defines the full pathname of the application which displays the host selection window. The host selection window offers the user a choice of icons from which to select the

host whose xdm service is to be used for display management. xdm only starts this program, known as Chooser, if the server issues an indirect XDMCP request and the entry for that server in the X access file contains the keyword CHOOSER.

Default:

/usr/lib/X11/xdm/chooser

cpp

Class: Cpp

Specifies the directory containing the C preprocessor used by the xrdb client.

Default:

/usr/bin/X11/xcpp

daemonMode

Class: DaemonMode

See "-daemon" option

determines whether the client is started as a background process.

If the value of daemonMode is False, the client runs in the foreground. This is useful for diagnostic purposes only.

Default:

True

When started, the client runs in the background.

debugLevel

Class: DebugLevel

See -debug option ("-debug debug_level")

Specifies whether or not the client runs in debug mode.

A value greater than 0 causes xdm to write error messages to the screen. It also prevents the daemonMode resource from forcing xdm to run in the background and suppress error messages.

A value greater than 9 causes xdm to stop spawning child processes for individual displays. In this mode multiuser operation is not possible.

Setting the value of this resource to a non-zero value is useful for diagnostic purposes only, and for such applications the recommended value is 9.

Default:

0

The client runs in the background and no error messages are printed.

defaultHost

Class: DefaultHost

Specifies a default host that is highlighted and selectable in the icon browser as soon as the chooser window comes up.

Default:

No default. The first icon is selected.

defaultIcon32

Class: DefaultIcon32

See "allowIcons", "smallIconSize" resources
Specifies the icon and the icon mask to be used to represent the login names in the xdm login window or the hosts in the xdm host selection window if the value of smallIconSize is True.
The resource is evaluated only if users or hosts have been defined with the aid of the user manager.

Default:

For the login window:
SWuser.s.pm if the value of iconSet is 'color'
SWuser.s.bm if the value of iconSet is 'mono'. In this case SWuser.s_m.bm is the associated mask.
For the host selection window:
SWpizzabox.s.pm if the value of iconSet is 'color'
SWpizzabox.s.bm if the value of iconSet is 'mono'. In this case SWpizzabox.s_m.bm is the associated mask.

defaultIcon64
Class: DefaultIcon64
See "allowIcons", "smallIconSize" resources
Specifies the icon and the icon mask to be used to represent the login names in the xdm login window or the hosts in the xdm host selection window if the value of smallIconSize is True.

Default:

For the login window:
SWuser.s.pm if the value of iconSet is 'color'
SWuser.s.bm if the value of iconSet is 'mono'. In this case SWuser.s_m.bm is the associated mask.
For the host selection window:
SWpizzabox.s.pm if the value of iconSet is 'color'
SWpizzabox.s.bm if the value of iconSet is 'mono'. In this case SWpizzabox.s_m.bm is the associated mask.

defaultUser
Class: DefaultUser
Specifies a user login name that is highlighted and selectable in the icon browser when the login window comes up.

Default:
No default. The first icon is selected.

errorLogFile
Class: ErrorLogFile
See -error option (" -error error_log_file")
Specifies the pathname of the file in which the client error messages are to be logged. Errors assignable to a user session are written to the appropriate $HOME/.xerrors file.

Default:
/usr/lib/X11/xdm/xdm-errors

exportList
Class: ExportList
This resource defines a list of environment variables to be exported to the xdm configuration files Xsetup, Xstartup, Xsession and Xreset in addition to the environment variables always exported to these files.
No default.

fail
Class: Fail
This resource defines a message which is displayed when user authentication fails.

Default:
Login incorrect.

failColor
Class: Foreground
Specifies the color used to display the failure message.
This is the error message displayed in the xdm login window when a user authentication fails.

Default:
black

failFont
Class: Font
Specifies the two fonts used to display the failure message.
This is the error message displayed in the xdm login window when a user authentication fails.
If the failure message does not fit into the space provided in the login window using the first font, the second font will be used instead.

Default:
-new century schoolbook-bold-r-normal--180-*,
-new century schoolbook-bold-r-normal--120-*

failTimeout
Class: FailTimeout
Specifies the time (in seconds) that the failure message is displayed.
This resource is applicable only if error messages are output in the xdm login window. That is typically the case when a chipcard terminal (ASECO) is being used.

Default:
10 (seconds)

failsafeClient
Class: FailsafeClient
Specifies the program to which the client falls back if the default session fails to execute.

The default session is the session that should be started for the user according to the specifications in the file Xsession or $HOME/.xsession.
The program is executed without arguments, but uses the same environment variables as the default session would have used.

Default:

/usr/bin/X11/xterm

font
Class: Font

Specifies the font to be used as the basic character style for text in windows and icons. Defines the font used to display the login name in the login window input field.

Default:

-new century schoolbook-medium-r-normal--180-*

foreground
Class: Foreground

Window foreground color.
The foreground color is the color for text output.

Default:

black

grabServer
Class: GrabServer

grabs the server while a user's login name and password are being read in.
In order to compensate for the security shortcomings of the X protocol, xdm grabs the server and the keyboard while reading in the user's login name and password. The server can be grabbed for the entire duration of this process (True) or only until the keyboard is locked (False).
You can use the grabTimeout resource to specify how long the client will wait for the grab to succeed. If the grab fails, xdm restarts the server and the session.

Default:

True

The server and the keyboard are disabled for the entire duration.

grabTimeout
Class: GrabTimeout

Specifies the maximum time the client will wait for a server and keyboard grab to succeed.
In order to compensate for the security shortcomings of the X protocol, xdm grabs the server and the keyboard while reading in the user's login and password. The grab may fail if another client has already grabbed the server and the network latencies are very high. Before setting the timeout to a high value, please consider the fact that a user could be fooled by a look-alike window on the screen. Failure to grab the server within the specified time results in xdm terminating the X server and the connection between xdm and the X server being rebuilt.

Default:

3 seconds

greetColor

Class: Foreground

Specifies the color used to display the login greeting.
This is the welcome message displayed in the xdm login window or host selection window.
This resource should be defined in the Xresources file identified by the value of the 'resources' resource.

Default:

black

greetFont

Class: Font

Specifies the font to be used for the login greeting.
This is the welcome message displayed in the xdm login window or host selection window.
This resource should be defined in the Xresources file identified by the value of the 'resources' resource.

Default:

-new century schoolbook-bold-i-normal--240-*

greeting

Class: Greeting

Specifies the text of the welcome message displayed once the user has logged in.
This resource should be defined in the Xresources file identified by the value of the 'resources' resource.

Default:

Welcome to SINIX/windows

height

Class: Height

Specifies the height of the xdm login window in pixels.
Specifies the height of the host selection window in pixels.
The height required for windows depends on the settings for the use of icons, logos and fonts. If you set these resources it is possible that the specified height will be not be sufficient to display all the elements of the host selection window. The window contents will then be truncated.
This resource should be defined in the Xresources file identified by the value of the 'resources' resource.

Default:

The height is calculated to match the required settings for the login window or host selection window.

hostname

Class: Hostname

Specifies the string displayed as the hostname in the login window. A string longer more than 32 characters long is truncated to 32 characters.
The value of the showHostname resource determines whether or not a hostname is displayed.
This resource should be defined in the Xresources file identified by the value of the 'resources' resource.

Default:

The hostname is established by the gethostname() function.

iconBackground
Class: IconBackground

Defines the background color for icons in the xdm login window or host selection window.
This resource is evaluated only if the icon is stored in XBM format.

Default:

white

iconForeground
Class: IconForeground

Defines the foreground color for icons in the xdm login window or host selection window.
This resource is evaluated only if the icon is stored in XBM format.

Default:

black

iconLabelFont
Class: IconLabelFont

Font for labels on icons displayed in the xdm login window or host selection window.

Default:

-new century schoolbook-medium-r-normal--120-*

iconPath
Class: IconPath

See "iconSet" resource
Defines a list of files containing icon images for displaying the login window and the host selection window. The icons are assigned to users and hosts by the user manager (see xdum client in section "xdum - Managing users ") and the host manager (see xdhm client in section "xdhm - Managing hosts") respectively.

Default:

For the login window:

```
/usr/lib/X11/siwin/icons/users/%l/%N: \
/usr/lib/X11/siwin/icons/%N:/usr/lib/X11/xdm/%N
```

For the host selection window:

/usr/lib/X11/siwin/icons/hosts/%I/%N: \
/usr/lib/X11/siwin/icons/%N:/usr/lib/X11/xdm/%N

iconSet
Class: IconSet

See "iconPath" resource

Defines the set of icon images the client uses for the objects that it manages.
xdm uses the set of icon images to display the user-specific icons in the login window and the host-specific icons in the host selection window. The icons are assigned to users and hosts by the user manager (see xdum client in section "xdum - Managing users ") and the host manager (see xdhm client in section "xdhm - Managing hosts") respectively.

Default:

depends on the number of color planes on the monitor:

mono
: for a monitor supporting one color plane

color
: for a monitor supporting more than one color plane

init
Class: Init

defines the full pathname of the Xinit file which runs under root before a session is opened (see "Appearance and behavior", subsection "xdm configuration files")

Default:

/usr/lib/X11/xdm/Xinit.

keyFile
Class: KeyFile

xdm and the terminal require a common key for XDMCP authorization using the XDM-AUTHORIZATION-1 mechanism. This resource specifies the file which contains the encryption values.
Each entry in this file consists of the name of a display and the associated common key.

Default:

For the international version:
: no default

For the US version:
: DES encryption

labelString
Class: LabelString

Specifies the label of an entry in the session option menu with the right-hand value 'sessionArgument'.
Specifies the label of an entry in the language option menu with the right-hand value 'languageArgument'. Each period '.' in languageArgument must be replaced with an underscore '_'. Also refer to section "languageArgument".

Default:

For sessionArgument:
the left-hand value of a value pair in the DisplayManager*sessionEntries resource
For languageArgument:
the left-hand value of a value pair in the DisplayManager*languageEntries resource

languageArgument
Class: LanguageArgument

See "languageEntries", "languageSwitch" resources.
Defines the variable LANG which is to be the default for a session. If the option menu 'Language' contains an entry with this variable, the entry appears as the label of the option button.
If the resource is not specified, the /etc/default/language file is evaluated. If the value in /etc/default/language is not on the option menu, it will be added to it.
If the value of languageArgument is not in the languageEntries resource or defined by /etc/default/language, the first entry in the options menu is used as the label of the option button.
If the language variable is not in the languageEntries resource, an error message to this effect is issued (see "errorLogFile" resource).

Default:
Value of the LANG variable in /etc/default/language.

languageEntries
Class: LanguageEntries

See "languageArgument", "languageSwitch" resources
Defines the language options that are offered in the client's login window by means of the option button 'Language'.
If the language variable is not in the languageEntries resource, an error message to this effect is issued (see "errorLogFile" resource).

Default:
German=De_DE.88591, English=En_US.ASCII

languagePrompt
Class: LanguagePrompt

Specifies the text of the language selection prompt.

Default:
Language

languageSwitch
Class: LanguageSwitch

See "languageArgument", "languageEntries" resources
Defines whether the option button 'Language' is displayed in the client's login window.

Default:

True

Language selection in the login window of the xdm is possible.

lockPidFile

Class: LockPidFile

uses file locking to prevent confusion between multiple xdm processes on the same host.

The process id(s) of the process(es) is (are) read from the file specified for the pidFile resource.

Default:

True

The locking mechanism is active.

loginLogo

Class: LoginLogo

See "iconSet", "showLogo" resources

Defines the logo for the xdm login window or host selection window.

Default:

For iconSet=color:
/usr/lib/X11/xdm/SNIlogo2.color
For iconSet=mono:
/usr/lib/X11/xdm/SNIlogo2

mailPath

Class: Mail

Sets the MAIL environment variable for the duration of the session. The MAIL environment variable defines the directory in which the user's mail is stored.

Default:

/var/mail

namePrompt

Class: NamePrompt

Specifies the string displayed to prompt the user for a login name.

xrdb strips trailing white space from the resource value; any blanks you want to have at the end of the string must be escaped with backslashes (\).

This resource should be defined in the Xresources file identified by the value of the 'resources' resource.

Default:

Login:

noIconUsers

Class: NoIconUsers

See "allowIcons", "onlyLocalUser" resources

Defines the name of the file that contains a list of user login names which are not to be displayed as icons in the xdm login window.

For reasons of security it may be advisable not to display all the icons.

Default:

/usr/lib/X11/NoIconUsers

onlyIconLabel

Class: onlyIconLabel

See "allowIcons" resource

Governs whether the icons in the xdm login or host selection window are to be displayed as labeled images or simply as labels.

Default:

False

The icons are displayed in the form of image plus label.

onlyLocalUser

Class: OnlyLocalUser

See "allowIcons", "noIconUsers" resources

Determines whether icons are to be displayed in the xdm login window for all users with a login name on the local host or only for local users. A login name is held to be local if:

- the user has a home directory on the local host
- the home directory belongs to the user
- the user has a valid password which has not been disabled.

Default:

True

Only the icons of local users are displayed.

openDelay

Class: OpenDelay

Specifies the length of the pause (in seconds) between a client's successive attempts to open a server.

The openDelay, openRepeat, openTimeout and startAttempts resources control the behavior of the client when a server cannot be opened.

Default:

15

openRepeat

Class: OpenRepeat

Specifies the number of attempts a client makes to open a server.

The openDelay, openRepeat, openTimeout and startAttempts resources control the behavior of the client when server a cannot be opened.

Once the client has made the number of attempts specified for openRepeat, xdm terminates, restarts the server, and again attempts to set up a connection.

Default:

5

openTimeout

Class: OpenTimeout

Specifies the duration (in seconds) of the client's attempt to open the server, i.e. the maximum period of time available for the connect() system call.
The openDelay, openRepeat, openTimeout and startAttempts resources control the behavior of the client when a server cannot be opened.
Once the period of time specified in openTimeout has elapsed, xdm terminates, restarts the server and again attempts to set up a connection.
Default:
30

passwdPrompt
Class: NamePrompt

specifies the string displayed to prompt the user for a password.
xrdb strips trailing white space from the resource value; any blanks you want to have at the end of the string must be escaped with backslashes (\).
This resource should be defined in the Xresources file identified by the value of the 'resources' resource.

Default:
Password:

pidFile
Class: PidFile

specifies the name of a file in which the process id of the main xdm process, i.e. the xdm process with the lowest process id, is stored in ASCII format.

Default:
/usr/lib/X11/xdm/xdm-pid

pingInterval
Class: PingInterval

specifies the interval (in minutes) at which the client checks whether a remote display is still active.
The client sends Xsync requests to remote displays at regular intervals to check whether the display responds.
The pingTimeout resource specifies the maximum time to wait for the display to respond.
If the display does not respond, the session is terminated.

Default:
5 minutes

pingTimeout
Class: PingTimeout

specifies the maximum time to wait for the display to respond to an Xsync request.
The client sends an Xsync request to remote displays at regular intervals to check whether the display responds. The pingInterval resource specifies the interval between checks.
If the display does not respond, the session is terminated.
If you are using X terminals which can be isolated from their associated host, it may be a good idea to increase this value. This could, however, lead to a situation where sessions continues to exist although the display is no longer running. It is also

inconvenient to have a session terminated on a workstation which happens to be for an NFS server response and is thus unable to respond to the Xsync request.

Default:

5 minutes

promptColor
Class: Foreground

specifies the color used to display prompts in the xdm login window.
This resource should be defined in the Xresources file identified by the value of the 'resources' resource.

Default:

The value of the foreground resource

promptFont
Class: Font

Specifies the two fonts used to display the prompt in the login window.
This resource should be defined in the Xresources file specified by the value of the 'resources' resource.
If the prompt string does not fit into the space provided in the login window using the first font, the second font will be used instead.

Default:

-new century schoolbook-bold-r-normal--180-*,
-new century schoolbook-bold-r-normal--120-*

randomFile
Class: RandomFile

Defines the name of the file used to generate the authorization key.

Default:

/dev/mem

removeDomainname
Class: RemoveDomainname

Specifies whether xdm strips the domain name from the names of hosts communicating with it via the XDMCP protocol.

Default:

True
The domain name is stripped from the host name.

requestPort
Class: RequestPort

See -udpPort option ("-udpPort portNumber")
Defines the number of the port at which xdm accepts XDMCP requests on the basis of the UDP protocol.

Default:

177

reset
Class: Reset
defines the full pathname of the Xreset file which runs under root after each session terminates (see section "Appearance and behavior").

Default:
/usr/lib/X11/xdm/Xreset

resetForAuth
Class: ResetForAuth
Governs whether the xdm display manager sends a SIGHUP signal to the server after changes to the authorization file, thus causing it to reboot and implement the changes.
Some servers read authorization information only when rebooted, not when setting up a connection to an xdm service.

Default:
False
xdm does not send a SIGHUP signal after changes have been made to the authorization file.

resetSignal
Class: ResetSignal
Defines the number of the signal to use to reset the server.

Default:
1 (SIGHUP)

resources
Class: Resources
See -resources option (" -resources filename")
This resource specifies the language-specific name of the Xresources file to be loaded by xrdb as the resource database onto the root window of screen 0 of the display (see section "Appearance and behavior").
The Xresources file may contain resource definitions for the setup file, the xdm login window and Chooser's host selection window.
The resource can be defined on a language-specific basis. xdm uses a POSIX-based search path to localize the resource file. The path may include the following substitution tokens:

%L	Language environment. This is either the value defined by the languageArgument resource in the /usr/lib/X11/xdm/xdm-config file or the value of the first entry in the /etc/default/language file.
%l	The part of the language environment which defines the language.
%t	The part of the language environment which defines the territory.

% c	The part of the language environment which defines the codeset.

The language environment comprises the following parts:
language[*_territory*][.*codeset*]

Default:
/usr/lib/X11/xdm/%l/Xresources:/usr/lib/X11/xdm/Xresources
Two standard resource files are supplied:
/usr/lib/X11/xdm/Xresources for the English language environment
/usr/lib/X11/xdm/De/Xresources for the German language environment

rootBackground
Class: RootBackground
Specifies the color for the root window.

Default:
For color displays: gray
Other displays: no default

rootBackgroundPixmap
Class: RootBackgroundPixmap
Specifies a pixmap for tiling the root window. The pixmap must be in XBM, GIF or XPM format.
No default

servers
Class: Servers
See -server option (" -server pathname")
Either defines the full pathname of the Xservers file (see section "Files") listing the names of the X servers managed by xdm, or specifies a single server entry (see section "Appearance and behavior").
X servers which call up the client via the X Display Manager Control Protocol (XDMCP) must not be listed in this file.
If the value of the servers resource begins with a slash (/), it is interpreted as a UNIX pathname and it is assumed that the file contains the X server entries. If the value does not begin with a slash, it is interpreted as an individual X server entry complying with the format in the Xservers file.

Default:
/usr/lib/X11/xdm/Xservers

session
Class: Session
See -session option (" -session pathname")
Defines the full pathname of the Xsession file containing system-wide and user-specific session management definitions (see section "Appearance and behavior").
The configuration data for the user's session is taken from $HOME/.xsession if this file exists.

Default:

/usr/lib/X11/xdm/Xsession

sessionArgument

Class: SessionArgument

See "sessionEntries" resource

This resource defines which menu item on the session selection option menu is selected by default when the login window comes up.

Default:

The menu item defined by the first pair of values listed for the sessionEntries resource.

sessionEntries

Class: SessionEntries

See "sessionArgument" resource

This resource defines the items on the session selection option menu in the xdm login window.

Default:

User Session=UserSession, Default Session=DefaultSession,
Light Session=LightSession, Failsafe Session=failsafe

sessionPrompt

Class: SessionPrompt

Specifies the text of the session selection prompt.

Default:

Session

sessionSwitch

Class: SessionSwitch

This resource determines whether the session selection option menu appears in the xdm login window.

Default:

True

The option menu is displayed.

setup

Class: Setup

Defines the full pathname of the Xsetup file. The Xsetup file runs under root after the login window has been displayed. The file can be used to customize the appearance of the screen and open other windows (see section "Appearance and behavior").

Default:

/usr/lib/X11/xdm/Xsetup.

showHostname

Class: SNIExtensions

specifies whether or not the hostname is to be output in the xdm login window after the welcome message.

The hostname displayed is specified by the hostname resource.
This resource should be defined in the Xresources file identified by the value of the 'resources' resource.

Default:
True
The host name is displayed.

showIcons
Class: ShowIcons
See "allowIcons" resource
Governs whether user login name icons are displayed in the xdm login window. The user can select an icon and hence a login name with the mouse instead of entering the login name at the keyboard.

Default:
True
Icons are displayed.

showLogo
Class: ShowLogo
Governs whether the logo defined by the loginLogo resource is displayed in the xdm login window or host selection window.
This resource should be defined in the Xresources file specified by the value of the 'resources' resource.

Default:
True
The logo is displayed.

smallIconSize
Class: smallIconSize
See "allowIcons", "defaultIcon32", "defaultIcon64" resources
Governs whether the size in pixels of the icons representing the user login names in the login window is 32x32 or 64x64.

Default:
False
The icon size in pixels is 64x64.

startAttempts
Class: StartAttempts
specifies the maximum number of attempts a client will make to open a server before it gives up.
The openDelay, openRepeat, openTimeout and startAttempts resources control the behavior of the client when server a cannot be opened.
Once the number of attempts specified by openRepeat have been made, or when the time specified by openTimeout for an attempt has elapsed, xdm terminates, restarts the server and again attempts to set up a connection. startAttempts specifies how often this entire process is to be repeated. If the value specified is 0, an infinite

number of attempts will be made.

Default:

0 for X terminals
5 for local servers

startup
Class: Startup

Defines the full pathname of the Xstartup file which runs after the user logs in (see section "Appearance and behavior").

Default:
/usr/lib/X11/xdm/Xstartup.

systemPath
Class: SystemPath

Sets the PATH environment variable for the Xstartup and Xreset files to the value of this resource if defined. If no path is specified, xdm evaluates the SUPATH variable in /etc/default/login.

Default:
SUPATH variable in the /etc/default/login file.

systemShell
Class: SystemShell

Sets the SHELL environment variable for the Xstartup and Xreset files to the value of this resource.

Default:
xdm evaluates the login name entry in /etc/passwd and uses the shell named there. If no shell is defined for the associated user, /bin/sh is used.

termSignal
Class: termSignal

Defines the number of the signal to use to terminate the server.

Default:
15 (SIGTERM)

terminateServer
Class: TerminateServer

Specifies whether the X server is to be terminated when a session terminates. Normally, it is simply reset. This option enables you to limit the amount of time the server is run.
This is possible on local displays only.

Default:

False for RM systems. The server is not terminated when a session terminates.
True for other systems. The server is terminated when a session terminates.

translations
Class: Translations

Specifies the key bindings for the login window.
For more information on the syntax for the translations resource refer to the section " translations ".

Default:

Ctrl<Key>C: restart-session()\n\
Ctrl<Key>\\: abort-session()\n\
Ctrl<Key>Return: finish-input()\n\
<Key>Return: finish-input()\n\

unsecureGreeting
Class: UnsecureGreeting

Defines the text displayed in the xdm login window if user-specific authorization is active for the display but is not supported by the server.
This resource should be defined in the Xresources file identified by the value of the 'resources' resource.

Default:

This is an unsecure session

userAuthDir
Class: UserAuthDir

Defines the name of the directory in which the xdm client creates the user's .Xauthority authorization file if it cannot do so in the home directory. It also defines the value of the XAUTHORITY environment variable.

Default:

/tmp

userPath
Class: UserPath

Sets the PATH environment variable for the session to the value of this resource if defined.
The value should be a colon-separated list of directories. If no path is specified, xdm evaluates the PATH variable in the /etc/default/login file.

Default:

PATH variable in /etc/default/login

width
Class: Width

Specifies the width of the xdm login window in pixels.
Specifies the width of the host selection window in pixels.
The width required for windows depends on the settings for the use of icons, logos and fonts. If you set these resources it is possible that the specified width will be not be sufficient to display all the elements of the host selection window. The window

contents will then be truncated.
This resource should be defined in the Xresources file identified by the value of the 'resources' resource.

Default:
The width is calculated to match the required settings for the login window or host selection window.

x
Class: Position
Specifies the horizontal distance, or x offset, between the window and the edge of the screen in pixels.

Default:
The window is positioned in the center of the screen. The X position is calculated according to the screen size.

xrdb
Class: Xrdb
Specifies the name of the program used to load the resources.

Default:
/usr/bin/X11/xrdb

y
Class: Position
Specifies the vertical distance, or y offset, between the window and the edge of the screen in pixels.

Default:
The window is positioned in the center of the screen. The Y position is calculated according to the screen size.

2.7.2 Appearance and behavior

The following section describes how xdm works in the following order:
- the stages in an xdm session
- connection setup between xdm and X server
- X terminal management
- authorization
- xdm configuration files
- using signals to control xdm behavior

2.7.2.1 The stages in an xdm session

1. On startup xdm reads the xdm-config file. The resource definitions in this file govern the behavior of the client for each managed display or display class.
   The subsection headed " "X terminal management"" describes how to form display classes and how to define a specific behavior for each class.
   The client writes its process ID to the xdm-pid file; it records error messages in the xdm-errors file.

2. xdm opens connections to the displays listed in the Xservers file. An entry in this file allows the client to start the X server for a host's local display. The client can also manage the displays of X servers which call it up via the X Display Manager Control Protocol (XDMCP). xdm reads the Xaccess file to find out which hosts are allowed access XDMCP.
   The subsection headed " "Connection setup between xdm and X server"" describes how the client finds out which displays to manage.
3. xdm executes the Xinit file. The function of this file is to inform the X server of settings which preferably need to take effect before a session is opened, such as the right keyboard assignment to load.
4. xdm displays the login window on the displays it is to manage. In this window the user has a choice between selecting a login name from the available icons and entering it directly. xdm authenticates the login name and password entered by the user.
   The description of the Xresources file explains how to modify the appearance and behavior of the login window.
5. Having displayed the login window, xdm executes the Xsetup file. This file can be used to influence the appearance of the screen at the time of login window output. You might want to define the color and pattern of the root window, or open other windows in addition to the login window, such as a window for displaying system messages if a suitable client is a available. As the Xsetup file is a shell script which runs under root, it must be protected against unauthorized access. The client looks in the Xresources file for resources governing the appearance and behavior of the xdm login window and the root window.
6. If login name and password authorization fails, an error message appears in a pop-up dialog box and must be acknowledged. xdm then delays the next login attempt by the time specified in LOGIN_DELAY. After MAXTRIES attempts, the login screen is restarted.
   If the user takes too long (as defined by the value of TIMEOUT) to complete the login procedure, the login process is aborted.
   xdm prompts the user to define a new password if the old one has expired or if none has been defined and PASSREQ is set in /etc/default/login.

Variable	Value
TIMEOUT	60 (seconds)
PASSREQ	NO
LOGIN_DELAY	61 (seconds)
MAXTRIES	5

Table 24: Default values of variables in /etc/default/login

   Successful and failed login attempts are logged by the syslog() call. The log messages are recorded in the /usr/adm/log/messages file, and some of them are output on the system console, depending on the configuration defined in /etc/syslog.cont.
7. Having successfully authorized the login name and the password, xdm starts a session for the user. If the authorization mechanism is active for the display, xdm generates an entry in the user's authorization file. This entry contains an authorization key valid only for the current session. Clients have to know this key in order to be able to access the server.

For further information refer to the section "Authorization".

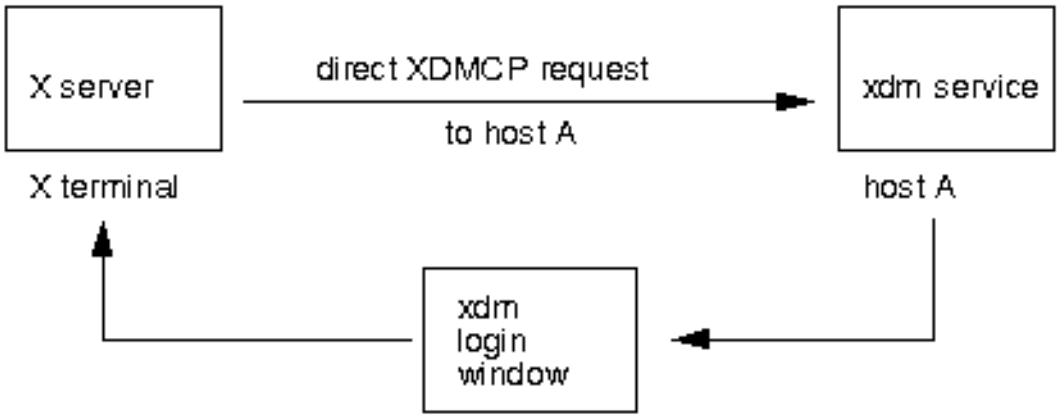
8. xdm executes the Xstartup file.
   The Xstartup file can contain definitions of measures influencing user session management. Typical examples would be mounting a user's directories via a file server, displaying a message of the day, or terminating the session in response to unauthorized logins. As the Xstartup file is a shell script which runs under root, it must be protected against unauthorized access.
9. If the Xstartup file executes successfully, xdm opens the session. The client executes the Xsession file.
   The Xsession file contains system-wide and user-specific session management definitions. These definitions apply to the user session if the user has not set up any private configuration files. If a user has set up private configuration files, a session opened for that user can be organized in accordance with the instructions they contain. Hence the Xsession file causes the session defined in the $HOME/.xsession file to be started if the latter file exists.
   The Xsession file can be passed an argument. Depending on the value of this argument, a number of different sessions can be defined for the user. The argument is defined during session selection in the login window or as a result of the user pressing a specified sequence of buttons or keys in the xdm login window after entering the appropriate login name and password.
   Users who have problems with the session started for them on the basis of their login name can start the predefined session and then do their own troubleshooting. In the passage dealing with the Xresources file in the subsection headed " "xdm configuration files"" you can find out how to pass an argument to the Xsession file.
   While the Xsession file is executing, error messages written to standard error are redirected to the user's $HOME/.xerrors file (for a SINIX Desktop session) or $HOME/.dt.errorlog (for a CDE session).
10. After the user has ended the session, xdm executes the Xreset file. The Xreset file does the opposite of the Xstartup file. It is typically used to reverse measures defined in the Xstartup file.
11. xdm then resets the server.
    Depending on the configuration, a new connection is set up and another login window or host selection window is displayed at the initiative of the server or of xdm.
    If no window is displayed, the connection between xdm and the X server is malfunctioning. The subsection headed " "Using signals to control xdm behavior"" describes how to make the client reestablish the connection to displays listed in the Xservers file.

2.7.2.2 Connection setup between xdm and X server

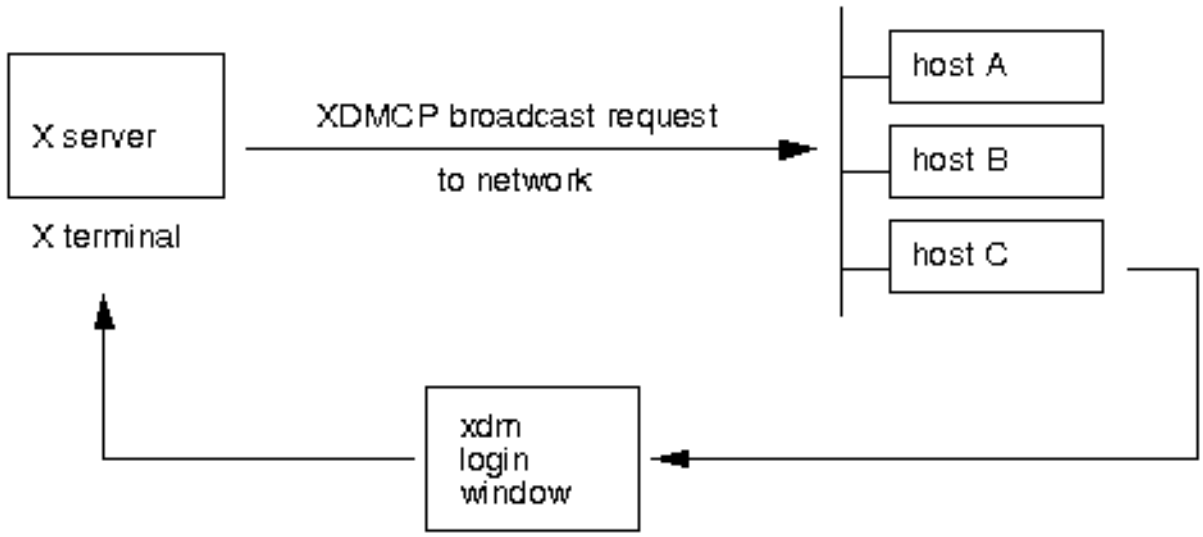
The xdm client is capable of managing multiple displays both on the local host and on remote hosts. Each display is controlled by an X server. Before xdm can manage a display, a connection must be set up between xdm and the X server. The initiative for setting up the connection can be taken either by the X server or by xdm. X servers for X terminals generally start connection set up on their own initiative. For connections initiated by the X server, the X server and the xdm service use the XDMCP protocol. XDMCP is supported by X servers written with X Window System version X11R4 and above. X servers not supporting this protocol must be managed directly by xdm. The local X server is usually started by xdm.

Display management with the X Display Manager Control Protocol (XDMCP): X servers with access to this protocol can call up the xdm client on their own initiative and request it to manage the display they control. xdm reads the Xaccess file to find out which X servers are allowed to make use of a host's xdm service.
xdm distinguishes between direct, broadcast and indirect XDMCP requests.

- Direct requests
  The server requests the xdm service of a specific host in the network. If the server is listed in the Xaccess file on that host, or is not explicitly denied access there, the display will be managed directly by the host's xdm, i.e. it will be sent a login window.

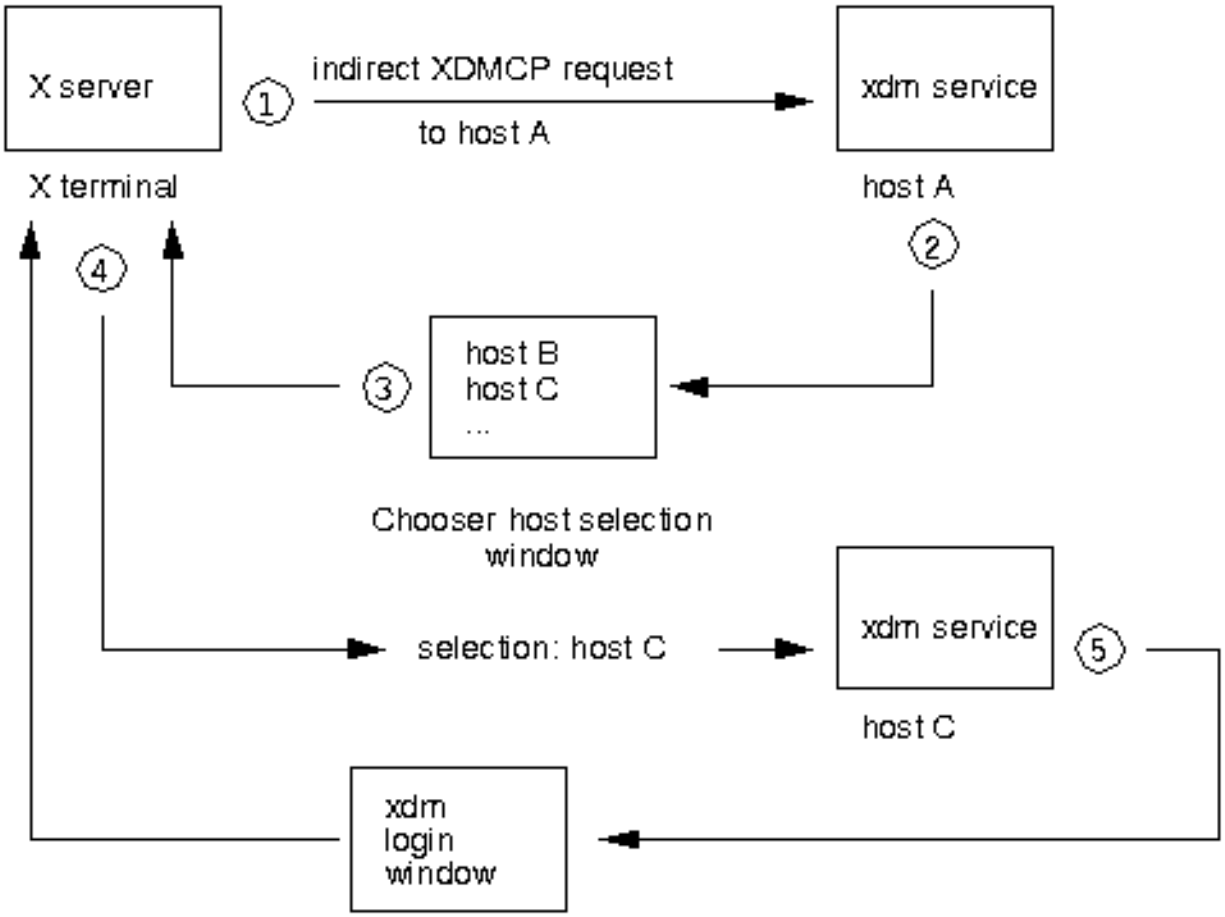


- Broadcast requests
  The server sends an XDMCP request to all the hosts on the network. All active xdm services check the server's authorization. The first xdm service to respond takes over display management.



- Indirect requests
  The server sends an indirect XDMCP request to an specific host. That host's xdm checks the entries in the host's Xaccess file and on that basis decides whether to accept the request and which hosts to forward the indirect request to. The first host to respond takes over display management.

If the entry for the display contains the keyword CHOOSER, the user at the display will be given the chance to select a host to log in on. xdm does this by starting the Chooser application, which displays a host selection window showing the available hosts on the display of the X server issuing the request. The user can select a host from the icons in this window.



If an xdm service (host A) receives an indirect XDMCP request and has a CHOOSER entry in its Xaccess file for the X server issuing the request (1), it starts the Chooser application (2). Chooser broadcasts the request to all hosts named after the keyword CHOOSER in the Xaccess file. Depending on the entries in their Xaccess files, these hosts return either a positive or a negative acknowledgment. The xdm service on host A sends the originating X server the names of the hosts which respond positively for displaying in a host selection window (3). The user selects a host, and the selection is transmitted to the chosen host's xdm (4), which sends a login window (5) at which the user can login.

The name of the application the local copy of xdm uses as Chooser is defined by the chooser resource of class Chooser.

Display management using entries in the Xservers file: This xdm configuration file contains a list of the Xservers to which the client has to take the initiative for setting up to the connection. Displays whose X servers call up xdm via XDMCP on their own initiative must not be entered in this file. X11R3 X servers must always be identified to xdm in the Xservers file because they do not support XDMCP.

xdm can start the X server of the local host in response to a command in the Xservers file. It attempts at regular intervals to set up connections to the other X servers listed in the file.

2.7.2.3 X terminal management

The xdm client simplifies X terminal management. X terminals have no hard disks, and often the only application which can run on them directly is the X server. As the X server and xdm are in this case running on different hosts, they intercommunicate via over a network. The xdm client is responsible for checking access authorization on X terminals and thus plays a major part in assuring system and network security.

Setting up display classes: The system administrator is able to form a display class from the displays of several X terminals and to define a uniform behavior for the resultant class. They way in which the display class is defined depends on the manner in which xdm is notified of the X server to which it is to set up a connection. The subsection headed " "Connection setup between xdm and X server"" describes the differences between X servers which call up xdm via XDMCP and those which are entered in the Xservers file.

X servers which call up via XDMCP can tell xdm which display class their display belongs to. In this case X terminals can only be managed under the name of the display class identified by the X terminal's X server.

In the case of displays of X terminals listed in the Xservers file, the system administrator can form a display class and define the name of the class in the Xservers file.

Defining display class behavior: The system administrator has access to a group of resources that can be used to specify xdm's behavior with respect to single displays and display classes. This principally involves using xdm configuration files to control session management for an X terminal display class. For example: the Xinit file which xdm runs before starting a user session can be set to load the correct keymap for the X terminal's display so as to enable the user to enter a login name and the associated password.

The xdm-config file holds resource definitions governing xdm's behavior in respect of displays.

Typical entries in an xdm-config file:

```
DisplayManager*init:                  /usr/lib/X11/xdm/Xinit
DisplayManager*session:               /usr/lib/X11/xdm/Xsession
DisplayManager.TX100.init:            /usr/lib/X11/xdm/Xinit.TX100
DisplayManager.TX100.session:         /usr/lib/X11/xdm/Xsession.TX100
DisplayManager.TD-9769_200_1.init:    /usr/lib/X11/xdm/Xinit.TX100
DisplayManager.TD-9769_200_1.session: /usr/lib/X11/xdm/Xsession.TX100
```

The resource definitions in this example instruct xdm to evaluate different configuration files for different displays. The same pair of configuration files applies to X terminals of display class TX100 and to X terminals of display class TD-9769_200_1. For all other displays xdm evaluates the files Xinit and Xsession.

2.7.2.4 Authorization

Instead of the host-specific access control provided by the xhost client, it is also possible to implement user-specific access control. User-specific access control is handled by an authorization mechanism which checks whether the client is authorized to access the X server in the course of a user's session. This authorization mechanism is supported by xdm. The authorization mechanism is active if the value of the xdm client's authorize resource is True. This resource can be given display-specific setting in the xdm-config file.

If the authorization mechanism is active for the local display, then when the X server is started by xdm it is supplied with a session key in a file in the /usr/lib/X11/xdm directory. On startup the X server is told the name of this file by xdm, and from then on grants access only to X clients which can present the session key.
X servers using the XDMCP protocol, such as X server for X terminals, are supplied with the session key by xdm via the XDMCP networking protocol if they request xdm services and ask for authorization to be used.
xdm performs authorization and generates the session key using a defined mechanism which can be selected with the authName resource. The international version of xdm supports the MIT-MAGIC-COOKIE-1 authorization mechanism; the US version also supports XDM-AUTHORIZATION-1, the DES encryption mechanism.
Once a user has successfully logged in at the xdm login window, xdm also stores the session key in the $XAUTHORITY file. If the XAUTHORITY environment variable is not set, the default file is .Xauthority in the user's home directory. Any clients the user starts up within the session for the associated display retrieve the session key from the authorization file and present it to the X server. The X server rejects other users' X clients because they cannot supply the current key. In this case the following error message appears on the display:

```
Client is not authorized to connect to Server
```

Permission to access the authorization file must be restricted to the owner. This ensures that it is impossible for unauthorized users to retrieve the key from this file and thus run clients on the same server.
When the session is terminated and a new one is started, a new key is generated and communicated to the X server.
To activate the authorization mechanism, add the following resource definitions to the xdm-config file:

```
DisplayManager*authorize: true
```

This entry specifies that authorization is to be activated for all connections. Authorization can be deactivated for local displays with the entry

```
DisplayManager.display.authorize: false
```

With X servers which use the XDMCP protocol, the setting (True or False) in xdm's xdm-config file is not what decides whether or not authorization is active. In this case if an X server asks for authorization to be used when it requests xdm services, xdm will always provide authorization for that X server.
With some X terminals you must define an appropriate setting in the X terminal itself to ensure that the X server provides authorization services. You should read the X terminal description to find out whether the terminal offers user-specific access control functionality.
If the message

```
This is an unsecure session
```

appears in the display's xdm login window, it means that xdm has specified that the authorization mechanism should be used, but that the X server does no support authorization or that authorization has not been activated. The text of this message can be changed with the unsecureGreeting resource.
If clients which access the display need to be run on other hosts in the network, the session key must be made available to them. This entails extracting the session key from $XAUTHORITY (or $HOME/.Xauthority) and adding it to the user-specific authorization files on the remote hosts. This is one of the functions of xauth (see section "xauth - Handling authorization information").
Note that the host-specific access authorization granted by the xhost client takes

precedence over the user-specific access authorization provided by the authorization mechanism. If a host is granted access to a server by the X server's internal host list, all clients started on that host are authorized to access that server, even if they cannot supply the valid session key. Hosts on which user-specific authorization is to apply must not appear in the X server's host-specific access list. For further information refer to the xhost client description (section "xhost - Controlling access to the X server ").

2.7.2.5 xdm configuration files

The configuration files can be identified to the xdm client in command-line options or in resource definitions. As the system administrator can change the names of the configuration files, and as there may be different configuration files for individual displays or display classes, the functions of the files are described below without regard to their actual names. The file names given here are those commonly used for these files.
If the autoRescan resource of class AutoRescan is set to True, changes to xdm configuration files take effect the next time an X server connection is set up. If it is not, you can put the changes into effect by sending a SIGHUP signal to the xdm main process. The xdm main process is the xdm process with the lowest process ID.
xdm-config is identified to the client in the -config option. If the -config option is not specified when xdm is started, the client uses the /usr/lib/X11/xdm/xdm-config file.
In this file you can define resources which govern the behavior of xdm for all displays, for specific display classes or for individual displays. In particular, in the xdm-config file you can define the names of other configuration files.
The DisplayManager*resource resource can be defined on a language-specific basis. xdm uses a POSIX-based search path to localize the resource file. The path may include the following substitution tokens:

%L	Language environment. This is either the value defined by the languageArgument resource in the /usr/lib/X11/xdm/xdm-config file or the value of the first entry in the /etc/default/language file.
%l	The part of the language environment which defines the language.
%t	The part of the language environment which defines the territory.
%c	The part of the language environment which defines the codeset.

The language environment comprises the following parts:
language[*_territory*][.*codeset*]
SINIX/windows comes with a default xdm-config configuration file as follows:

```
DisplayManager*resources: \n
/usr/lib/X11/xdm/%l/Xresources:/usr/lib/X11/xdm/Xresources
```

"DisplayManager*languageArgument" is not defined, which means that /etc/default/language is evaluated.
Typical contents of an xdm-config file:

```
DisplayManager.servers:         /usr/lib/X11/xdm/Xservers
DisplayManager.errorLogFile:    /usr/lib/X11/xdm/xdm-errors
```

```
DisplayManager.pidFile:          /usr/lib/X11/xdm/xdm-pid
DisplayManager.keyFile:          /usr/lib/X11/xdm/xdm-keys
DisplayManager.accessFile:       /usr/lib/X11/xdm/Xaccess
DisplayManager.noIconUsers:      /usr/lib/X11/xdm/NoIconUsers
DisplayManager.allowIcons:       true
DisplayManager.onlyLocalUser:    true
DisplayManager*resources: \n
/usr/lib/X11/xdm/%l/Xresources:/usr/lib/X11/xdm/Xresources
DisplayManager*init:             /usr/lib/X11/xdm/Xinit
DisplayManager*setup:            /usr/lib/X11/xdm/Xsetup
DisplayManager*startup:          /usr/lib/X11/xdm/Xstartup
DisplayManager*reset:            /usr/lib/X11/xdm/Xreset
DisplayManager*session:          /usr/lib/X11/xdm/Xsession
DisplayManager*cpp:              /usr/bin/X11/xcpp
DisplayManager*authComplain:     false
DisplayManager*authorize:        false
DisplayManager*startAttempts:    0
DisplayManager*grabServer:       true
DisplayManager._0.init:          /usr/lib/X11/xdm/Xinit.local
DisplayManager._0.startAttempts: 5
DisplayManager._0.terminateServer: true
DisplayManager._0.authorize:     true
DisplayManager._0.setup:         /usr/lib/X11/xdm/Xsetup_0
DisplayManager._0.startup:       /usr/lib/X11/xdm/GiveConsole
DisplayManager._0.reset:         /usr/lib/X11/xdm/TakeConsole
DisplayManager.*.sessionSwitch: true
DisplayManager.*.sessionEntries:        User Session=\n
UserSession,Default Session=DefaultSession,SINIX Desktop=\n
SINIXSession,Light Session=LightSession,Failsafe Session=\n
failsafe
DisplayManager.*.languageSwitch:        true
DisplayManager.*.languageEntries:       English=\n
En_US.ASCII,German=De_DE.88591
!
! Xdm evaluates /etc/default/login if userPath or systemPath
! are empty.
!
DisplayManager.*.userPath:
DisplayManager.*.systemPath:
```

Definitions for resources with tight bindings (.) to DisplayManager apply to all displays. In the examples above this is the case with the pidFile, errorLogFile and servers resources. Resources with loose bindings (*) to DisplayManager can be defined as applying to specific displays or display classes. In the examples above this is the case with the resources, init, startup, session and reset resources.

xdm-pid contains the process ID of the primary xdm process. The name of the xdm-pid file is defined by the value of the pidFile resource.

This process ID is helpful when using signals to control the behavior of xdm. For each display that it manages, xdm spawns a dedicated child process also named xdm.

xdm-errors logs xdm error messages other than those issued in the course of a user session. The name of the xdm-errors file is defined by the value of the errorLogFile resource. This file is cleared whenever xdm is restarted.

If xdm does not run as anticipated, the contents of this file may provide some indication of

where the problem lies.
The xdm-errors file does not log errors which occur after Xsession has started; errors of this type are written to the appropriate $HOME/.xerrors or $HOME/.dt/errorlog file. Error messages from the same user's previous session are also available: they are held in the corresponding $HOME/.xerrors.old or $HOME/.dt/errorlog.old file.
Xaccess contains a list of displays and servers which are thereby authorized to use the XDMCP service of the local xdm manager. In other words, xdm requests from the X servers list in the Xaccess file will be accepted by xdm.
An asterisk "*" in a line in the Xaccess file means that requests will be accepted from all X servers. This general authorization can be withdrawn from specific hosts or hosts groups. xdm evaluates entries in the Xaccess file line by line. If a line ends with a backslash "\", that line and the line which follows are treated as a single line. You can insert comments in the Xaccess file: everything that comes after a hash "#" up to the end of the line is not interpreted by xdm. xdm ignores blank lines.
There are three different types of entry:
- macro definitions;
  macros combine hosts to form host groups. Host groups can be used in entries for indirect requests.
- entries for direct and broadcast XDMCP requests;
- entries for indirect XDMCP requests.

On the distinction between direct, indirect and broadcast XDMCP requests refer to the section "Connection setup between xdm and X server" in this client description.

1. Defining a macro
   Add a line of the following form to the Xaccess file:
   %*macroname hostname*[*hostname*]...
   - %*macroname*
     identifies the start of a macro definition. %*macro-name* may be used instead of *hostname* in all subsequent entries in the Xaccess file. xdm then expands the macro name to the defined string. Multilevel expansion is supported. Recursive expansion is not.
   - *hostname*[*hostname*]...
     is a list of host names. Every host name listed here must appear together with its Internet address in the /etc/hosts file. xdm uses the Internet address when authorizing hosts to access XDMCP services. You can also give the actual Internet address rather than the host name in the Xaccess file.
     hostname may also be a previously defined macro name preceded by a percent sign "%". However, you cannot use the name of the current macro definition, as recursive expansion is not supported.
2. Granting or denying direct and broadcast access to the local host
   Add a line of the following form to the Xaccess file:
   [*mode*]*requestingHost*

   mode indicates whether or not the display of the host submitting the request is to be managed by xdm if it name matches *requestingHost*. If you do not define the mode, xdm manages the displays of hosts with matching host names. If you use an exclamation point "!" for the mode, xdm does not manage the displays of hosts with matching host names.

   requestingHost

is either a host name or a host name pattern. xdm interprets the asterisk "*" and the query "?" as wildcards.
"*" is a pattern standing for any string of 0 or more characters; "?" is a pattern standing for any one character.
Every requesting host must appear together with its Internet address in the /etc/hosts file. xdm uses the Internet address when authorizing hosts to access XDMCP services. You can also give the actual Internet address rather than the host name as *requestingHost*.

3. Granting or denying indirect access to a user-selectable host
Add a line of one of the following forms to the Xaccess file:
[*mode*]*requestingHost*[CHOOSER] *hostname*...
[*mode*]*requestingHost* CHOOSER BROADCAST

mode indicates whether or not the display of the host submitting the request is to be managed by xdm if it name matches *requestingHost*. If you do not define the mode, xdm manages the displays of hosts with matching host names. If you use an exclamation point "!" for the mode, xdm does not manage the displays of hosts with matching host names.

requestingHost
is either a host name or a host name pattern. xdm interprets the asterisk "*" and the query "?" as wildcards. "*" is a pattern standing for any string of 0 or more characters; "?" is a pattern standing for any one character.
Every requesting host must appear together with its Internet address in the /etc/hosts file. xdm uses the Internet address when authorizing hosts to access XDMCP services. You can also give the actual Internet address rather than the host name as *requestingHost*.

CHOOSER
indicates that the user on the requesting host is offered a host selection window from which to select a host to provide xdm services. If the line does not include the keyword BROADCAST, the local xdm sends XDMCP inquiries to the hosts named as *hostname*...
All the hosts that return a positive acknowledgment are offered for the user to choose from in a host selection window on the originating host. Positive acknowledgments are sent by hosts which in turn have a direct entry for the requesting X server in their Xaccess file or do not explicitly exclude xdm services for that X server.

BROADCAST
indicates that the local xdm is to broadcast the XDMCP request to all the host on the network. All the hosts that return a positive acknowledgment are offered to the user to choose from in the Chooser host selection window.

hostname...
is a list of host names. Every host name listed here must appear together with its Internet address in the /etc/hosts file. xdm uses the Internet address when authorizing hosts to access XDMCP services. You can also give the actual Internet address rather than the host name in the Xaccess file.
hostname may also be a macro name preceded by a percent sign "%". This macro name must have been defined in an earlier entry in the Xaccess file, as described above under "1. Defining a macro".

Sample Xaccess file:

```
#
# Entries for direct and broadcast requests
#
!frank.dpt1     # excludes host frank.dpt1
*.dpt1          # grants access by all hosts matching pattern *.dpt1
george.dpt2     # grants access to host/X terminal george.dpt2
#
# Entries for indirect requests
#
# Macro definitions forming host groups
%CHOOSERSELECT1      xdm.central xdm.serv1 xdm.serv2 xdm.serv3 \
xdm.serv4
%CHOOSERSELECT2      %CHOOSERSELECT1 xdm.serv5 xdm.serv6
bill.dpt4     xdm.central # Requests from hosts bill.dpt4
kay.abt6      xdm.central # and kay.dpt6 are passed to the xdm
# process of host xdm.central.
*.dpt5       CHOOSER %CHOOSERSELECT1
*.dpt6       CHOOSER %CHOOSERSELECT2
# Requests from hosts *.dpt5 are passed to the
# xdm processes of the hosts in host group
# %CHOOSERSELECT1, requests from *.dpt6 to
# %CHOOSERSELECT2. Hosts responding
# positively are displayed in the host
# selection window on the requesting hosts.
*.dpt??      CHOOSER BROADCAST
# Requests from hosts with double-digit
# department numbers are passed to the xdm
# processes all hosts on the network. Hosts
# responding positively are displayed in the
# host selection window on the requesting
# hosts.
```

Note that xdm reads the entries in the Xaccess file one after the other, stopping at the first entry it finds that matches the requesting host and handling that host in accordance with the entry it has found.

If you want to exclude a host from a host group to which you intend to grant access, the entry for the excluded host must come first. The following example clarifies this point:

```
!frank.dpt1     # excludes host frank.dpt1
*.dpt1          # grants access by all hosts matching pattern *.dpt1
```

Xservers contains a list of the xdm-managed displays to which the client is required to set up connections. X servers which call up xdm via XDMCP on their own initiative must not be entered in this file. The name of the Xservers file is defined by the value of the servers resource.

Each entry in the file is in four parts:

- display name
- display class
- display type
- command line (for local X servers)

Entries must be in the following syntax:

display-name[*display-class*] *display-type*[*command*]

display-name
designates the display xdm is to manage.
display-name must be in the format required for the DISPLAY environment variable so that it can be passed to other clients as a -display option.
display-name can be used with display-specific resources so as to define the client's behavior for the named display.

display-class
is the name of a display class to which the display belongs.
display-class can be used with display-specific resources so as to define the client's behavior for displays of the named class.

display-type
specifies whether the display is local or remote. The following types are available:

local xdm and the X server are on the same host
foreign
xdm and the X server are on different hosts

command
for a local display: program call to start the local X server. *command* is the full pathname of the program to be started as X server.

Typical contents of an Xservers file:

```
:0          local   /usr/bin/X11/X :0
chatham:0          foreign
xterm1:0 TX100    foreign
xterm2:0 TX100    foreign
xterm3:0 TX100    foreign
```

In this example the first entry refers to local display 0. The local X server is started by the command /usr/bin/X11/X.
The second entry refers to display 0 on the host named chatham.
The last three entries group three X terminals with the host names xterm1, xterm2 and xterm3 to form display class TX100, to which special configuration files can be assigned.
Xresources contains value assignments for resources the xrdb client loads into the server's resource database. The name of the Xresources file is defined by the value of the 'resources' resource.
In this file you can put resource definitions for the Xsetup file, the host selection window and the xdm login window. There are a great many resources that the system administrator can use to influence the appearance of these windows. In particular, all language-dependent strings can be defined here; to access the Xresources file xdm uses a search path in which the language variable is evaluated. Here the system administrator can also define resources which relate to the screen (color, size).
The login window is displayed by xdm together with xlogin; the host selection window is selected by the chooser client. Accordingly the Xresources file needs to contain resource value assignments for xlogin or chooser as appropriate. For an overview of all the available resources refer to the section "Resource overview" in this client description.
Assignments to the translations resource in the Xresources file allow functions to be allocated to specified key and button sequences for the login window. Some functions and their effects are explained below. The functions are described in full under the translations resource of class Translations.

abort-session()
: This terminates and then restarts the X server.

finish-input()
: If the input focus is in the login name input field, input is terminated. If the input focus is in the password input field, password input is terminated and the login data is authenticated. If the login name and the password are valid, xdm opens the session. If not, it displays an error message prompting the user to log in again.

restart-session()
: The X server is reset and the xdm login window is displayed again. This function can be used to force modified resource assignments to be evaluated.

set-session-argument()
: An argument one word long is passed to the Xsession file on startup. Depending on the value of this argument, a number of different sessions can be made available to the user.

Typical entry in an Xresources file:

```
# Entries for the login window
#  The first entries in the file are for the login window.
#  With color definition resources a distinction is drawn between
#  monochrome and color screens.
xlogin*greeting:   Hi there
xlogin*namePrompt: Please log in:\
xlogin*fail:       Not a valid login. Sorry!
#ifdef COLOR
xlogin*greetColor: Firebrick
xlogin*failColor:  Firebrick
xlogin*Foreground: Firebrick
xlogin*Background: MediumGoldenrod
xlogin*loginLogo:  /usr/lib/X11/xdm/SNIlogo.color
#else
xlogin*Foreground: black
xlogin*Background: white
xlogin*loginLogo:  /usr/lib/X11/xdm/SNIlogo
#endif
# The next entries are for the translations resource and they
# assign functions to accelerator key sequences for the login
# name and password input fields.
# If a user presses the <Return> key, input is terminated.#l
# If a user presses <F1> or <Ctrl><Return>, failsafe is
# passed to the Xsession file and input is terminated.
xlogin*passwdName.translations: #override\
<Key>Return:     set-session-argument() finish-input()
<Key>F1:         set-session-argument(failsafe) finish-input()\n\
Ctrl<Key>Return: set-session-argument(failsafe) finish-input()
xlogin*loginName.translations: #override\
<Key>Return:     set-session-argument() finish-input()
<Key>F1:         set-session-argument(failsafe) finish-input()\n\
Ctrl<Key>Return: set-session-argument(failsafe) finish-input()
# Entries for the host selection window
Chooser*greeting:      Watcher, mate
Chooser*showLogo:      true
Chooser*smallIconSize: false
```

Xinit is a shell script which runs under root before the start of each user session. The name of the Xinit file is defined by the value of the init resource
The following environment variables are set:

DISPLAY
    is the name of the associated display
PATHis the value of the systemPath resource
SHELLis the value of the systemShell resource
No arguments are passed to the script. The client waits for the script to terminate before starting the user session. xdm does not evaluate the shell script's exit status.
Xsetup is a shell script which runs under root after the login window has been displayed. The name of the Xsetup file is defined by the value of the setup resource.
In addition to the environment variables defined by the exportList resource, the following environment variables are passed to the shell script:

DISPLAY
    is the name of the associated display
PATHis the value of the systemPath resource
SHELLis the value of the systemShell resource
XAUTHORITY
    is the name of the authorization file
xdm executes the Xsetup file whenever it displays a new login window. The file can be used to customize the appearance of the screen at the time the login window is displayed.
This shell script is executed only if the grabServer resource has a value of False, as otherwise the server locked against accessing by other processes.
As this shell script runs with root privileges, it should be protected against unauthorized access.
Xstartup is a shell script which runs under root at the start of each user session once the user has successfully logged in. The name of the Xstartup file is defined by the value of the startup resource
In addition to the environment variables defined by the exportList resource, the following environment variables are passed to the shell script:

DISPLAY
    is the name of the associated display
HOMEis the user's home directory
MAIL is the name of the mail directory
PATHis the value of the systemPath resource
SHELLis the value of the systemShell resource
USERis the user's login name
XAUTHORITY
    is the name of the authorization file
No arguments are passed to the script. xdm waits for the script to terminate before continuing with the user session. If the script returns a non-zero exit status, xdm interrupts the session immediately and initiates a new authorization check.
As this shell script runs with root privileges, it should be protected against unauthorized access.
Xsession is a shell script which runs with the user's permissions at the start of each session. The name of the Xsession file is defined by the value of the session resource.

In addition to the environment variables defined by the exportList resource, the following environment variables are passed to the shell script:

DISPLAY
is the name of the associated display
HOMEis the user's home directory
LANGis the language variable in the /etc/default/language file or the language selection option menu
LOGNAME
is the user's login name
MAIL is the name of the mail directory
PATHis the value of the PATH variable in the /etc/default/login file or of the userPath resource
SHELLis the shell defined for the user in the /etc/passwd file
TZ is the name of the timezone
USERis the user's login name

This shell script can be passed an argument as a result of a user action in the xdm login window. The translations resource defines which actions are possible and which arguments can be passed. Depending on the value of the argument passed, a number of different sessions can, for example, be made available to the user (see "set-session-argument()").
Xreset is a shell script which runs under root after each user session has terminated. The name of the Xreset file is defined by the value of the reset resource.
In addition to the environment variables defined by the exportList resource, the following environment variables are passed to the shell script:

DISPLAY
is the name of the associated display
HOMEis the user's home directory
MAIL is the name of the mail directory
PATHis the value of the systemPath resource
SHELLis the value of the systemShell resource
USERis the user's login name
XAUTHORITY
is the name of the authorization file

2.7.2.6 Using signals to control xdm behavior

xdm uses UNIX signals to control X servers.

SIGHUP
resets the X server, closes down all client connections and performs various other housekeeping tasks.
SIGTERM
terminates the X server.

If these signals do not perform the expected actions, alternate signals can be defined with the resources resetSignal for SIGHUP and termSignal for SIGTERM.
xdm controls remote displays which do not support the XDMCP protocol by searching the window hierarchy of the display and using an X protocol request to clear the screen for the next session. There may be cases in which this mechanism does not terminate all clients,

since it only detects those clients which have opened windows. This problem does not arise with remote displays which support the XDMCP protocol, as the display is made to clear down all connections.
xdm itself responds to SIGHUP or SIGTERM signals. If the primary xdm process receives one of these signals, xdm responds as follows:

SIGHUP
: xdm reads the xdm-config, Xservers and Xaccess files to check whether entries have been added or deleted.
If an entry has been added to Xservers, xdm outputs its login window on the specified display. If an entry has been deleted, the client aborts any sessions running on the specified display without warning and does not start another.

SIGTERM
: xdm aborts all sessions currently running and then terminates. This signal can be used at system shutdown time.

If no login or host selection window is displayed on the root window, the connection between xdm and the X server is malfunctioning. In the case of a display listed in the Xservers file, the SIGHUP signal can be sent to xdm to cause it to set up the connection again:

1. Remove the entry for the display from the Xservers file by prefixing the comment character "#" to it.
2. Send the SIGHUP signal to the primary xdm process. You will find the process id in the xdm-pid file.
3. Restore the entry for the display in the Xservers file.
4. Again send the SIGHUP signal to the primary xdm process.

2.7.3 Files

The files below are listed in alphabetical order.

/etc/default/language
: The client evaluates the variable LANG from this system file if the language Argument resource is not specified.

/etc/default/login
: The client evaluates all parameters from this system file. The file is described in the "System Administrator's Reference Manual". Of particular importance are:

PATH	governs the value of the PATH variable if the "DisplayManager.*.userPath:" entry in the /usr/lib/X11/xdm/xdm-config file is empty.
SUPATH	governs the value of the PATH variable for the root login if the "DisplayManager.*.systemPath:" entry in the /usr/lib/X11/xdm/xdm-config file is empty.

/usr/lib/X11/app-defaults/XLogin
: contains default values for login window resources.

/usr/lib/X11/app-defaults/Chooser
: contains default values for host selection window resources.

/usr/lib/X11/xdm/chooser
: is the application which displays the host selection window. xdm uses this file if no other file has been allocated by the chooser resource of class Chooser.

/usr/lib/X11/xdm/NoIconUsers
: contains a list of login names which are not to be displayed as icons on the login screen. xdm uses this file if no other file has been allocated by the noIconUsers resource.

/usr/lib/X11/xdm/xdm-config
: is the xdm-config file read by the client if it is not passed any other file name in its -config option when called.

/usr/lib/X11/xdm/xdm-errors
: is the xdm-errors file used by the client if no other file has been assigned to the errorlogFile resource.
Once a user session has been started, the client writes errors messages to $HOME/.xerrors (SINIX Desktop session) or $HOME/.dt/errorlog (CDE session).

/usr/lib/X11/xdm/Xinit
: is the Xinit file read by the client if no other has been assigned to the 'init' resource for a display.

/usr/lib/X11/xdm/Xreset
: is the Xreset file read by the client if no other file has been assigned to the 'reset' resource for a display.

/usr/lib/X11/xdm/Xresources
: is the Xresources file read by the client if no other file has been assigned to the 'resources' resource for a display.

/usr/lib/X11/xdm/%l/Xresources
: is the Xresources file read by the client depending on the value of the LANG variable if no other file has been assigned to the 'resources' resource for a display.

/usr/lib/X11/xdm/Xservers
: is the Xservers file read by the client if no other file has been assigned to the 'servers' resource for a display.

/usr/lib/X11/xdm/Xsession
: is the Xsession file read by the client if no other file has been assigned to the 'session' resource for a display.

/usr/lib/X11/xdm/Xsetup
: is the Xsetup file read by the client if no other file has been assigned to the setup resource for a display.

/usr/lib/X11/xdm/Xstartup
: is the Xstartup file read by the client if no other file has been assigned to the startup resource for a display.

$HOME/.dt/errorlog
: holds the error messages issued in the course of a CDE session.

$HOME/.dt/errorlog.old
: holds the error messages issued in the course of the preceding CDE session.

$HOME/.Xresources
: contains the user-specific value assignments to resources which are to be loaded into the server in a SINIX Desktop session. It may also contain directives for the preprocessor used by xrdb.

$HOME/.xerrors

holds the error messages issued in the course of a SINIX Desktop session.

$HOME/.xerrors.old
holds the error messages issued in the course of the preceding SINIX Desktop session.

$HOME/.xsession
is the program that initiates a user-specific session.
In this program the user can specify the environment which is to be set up once a session has been started, the windows which are to be opened, and the configuration data which is to be used.

$XAUTHORITY
user-specific authorization file
Clients use this file as a user-specific authorization file. If the XAUTHORITY environment variable is not set, the client uses the .Xauthority file in the user's home directory.

2.7.4 Environment variables

Detailed descriptions of all the environment variables, arranged in alphabetical order, are given in the chapter "Environment variables ".

HOMEidentifies the user's HOME directory. The client searches this directory for the files .xsession and .Xresources and creates the files .xerrors, .xerrors.old and .Xauthority.

LANGdefines the values for language, territory and character set

MAIL absolute pathname of the file for the user's incoming mail

PATHis a list of the pathnames of directories which are scanned for commands.

SHELLspecifies which command interpreter is to be used.

USERidentifies the user's login name.

XAUTHORITY
denotes the user-specific authorization file.
If the environment variable is not set, the client searches for the $HOME/.Xauthority file.

2.8 xdcm - Managing mail

The xdcm client, the communication manager, allows you to manage, send and receive mail. It manages mail (electronically transmitted messages) as objects in mailboxes. The xdcm client can run under a login name on a given host only once per display.
The appearance and behavior of the client are documented in help texts.
The client's appearance and behavior can be configured dynamically using the xdtell client which, among other things, allows new client windows defined in the XDCL language to be output at runtime. The xdcm client evaluates instances of the XDCL classes "`mailmanager_view`" and "`mimemanager_view`" for this purpose.
The client's appearance and behavior can be configured with the XDCL language. This language allows experienced users and system administrators to change the standard configuration and customize the appearance and behavior of the client to meet their requirements. The manual "[4]" "SINIX/windows User Environment; XDCL Desktop Configuration Language" is a reference work providing a complete overview of which objects the client makes available, which characteristics can be specified, and which methods can be initiated by user actions.
On startup the client evaluates the XDESK_CFG_PATH environment variable and searches the directory named there for its .xdcm.cfg configuration file, which must conform to XDCL Desktop Configuration Language XDCL syntax. The user can edit this file so as to modify the appearance and behavior of the client.
The client automatically generates save files in XDCL whenever the user configures objects interactively and asks the client to save the current state. xdesk stores save files in the directory specified by the XDESK_SAVE_DIR environment variable. When called again, the client attempts to restore the saved state. Object definitions which are in the save files but have been modified in the configuration file are ignored; the definitions in the configuration file take precedence.

xdcm[*option*]...

-display [*hostname*]:*display_number*[.*screen_number*]
Identifies the X server, display and screen which the client is to contact.

hostname
symbolic name of the host or X terminal on which the X server which the client is to contact runs. hostname must be known in the transport protocol of the host on which the client resides.
display_number
number (unique to the X server) identifying the display for which the client is to manage I/O.
.screen_number
number (unique to the display) identifying the screen to which the client is to send output.

Default:
contents of the DISPLAY shell variable;
if DISPLAY is not set, the client terminates abnormally.

-open *mailbox* [*view*]

Starts the communication manager with the view *view* on the mailbox *mailbox*.
view is an instance of the XDCL object class `"mailmanager_view"` defined in the configuration file, which is to be used as a view. If *view* is not specified, the client uses the `"default"` instance of the `"mailmanager_view"` class.

Default:

If there is save information, the communication manager opens all windows recorded there. If there is no save information, the host manager opens a window with the view on the mailbox identified by the "`initial_folder`" attribute in the "`default`" instance of the XDCL object class "`mailmanager_view`".
If "`initial_folder`" is not specified, the /var/mail/$USER file is used.

-xrm "*resourceDefinition*:*value*"

Assigns a value to a client resource.
-xrm must be specified separately for each resource to be set in the client call. The "*resourceDefinition*:*value*" assignment is compared with the entries in the configuration files.

resourceDefinition

Identifies the resource to which a value is to be assigned. For *resourceDefinition* you can give the name of the resource, the name of the client and of any widgets in the appropriate resource syntax (see section "Value assignment syntax ").

value Value to be assigned to the resource.

2.8.1 Resources

You can configure the client using its resource database. To do this, you must assign the required values to the resources. You can use either configuration files or the -xrm option in the client call to access resources.
The instance name of the client is xdcm and its class name CommunicationManager.

fontList
Class: FontList

Font list

iconPath
Class: IconPath

See resources iconSet, systemIconPath
defines the list of files the client is to search for the icon image for an object that it manages.

No default

iconSet
Class: IconSet

See resources iconPath, systemIconPath
defines the set of icon images the client uses for the objects that it manages.

Default:

depends on the number of color planes on the monitor:

mono for a monitor supporting one color plane
color for a monitor supporting more than one color plane

saveOnExit
Class: SaveOnExit
Specifies whether the client saves its status when terminating.

Default:
ask

standardIconSize
Class: StandardIconSize
Size of the icons when the client is started

Default:
Auto

systemIconPath
Class: SystemIconPath
See resources iconPath, iconSet
Specifies the system-specific part of the list of files the client searches for the icon image for an object it manages.

Default:
/usr/lib/X11/siwin/icons/mail/%I/%N:\
/usr/lib/X11/siwin/icons/users/%I/%N:\
/usr/lib/X11/siwin/icons/wm/%I/%N:\
/usr/lib/X11/siwin/icons/common/%I/%N

2.8.2 Files

The files below are listed in alphabetical order. The resource configuration files are listed only in the form of the generalized file names $XENVIRONMENT, $XUSERFILESEARCHPATH and $XFILESEARCHPATH.

.xdcm.cfg
XDCL configuration file of the client in a directory on the path $XDESK_CFG_PATH.
.xdcm.ict
Save file created after interactive configuration and saving of the client in the directory defined in the XDESK_SAVE_DIR environment variable.
.xdcm.session
Save file of the client for saving views on mailboxes. It is created in the directory defined in the XDESK_SAVE_DIR environment variable.
$HOME/.xdcmrc
/etc/mail/xdcm.rc
$HOME/.mailrc
/etc/.mailrc

The communication manager evaluates these files in the given order, looks for an entry in the form `set sendmail = command`. The *command* in the first entry of this type that it finds is the command that will be used to send mail. The default is `/usr/bin/rmail -w`.

$XAUTHORITY
: user-specific authorization file
The client uses this file as a user-specific authorization file. If XAUTHORITY is not set, the client uses the .Xauthority file in the user's HOME directory.

$XENVIRONMENT
: user-specific configuration file

$XFILESEARCHPATH
: application-specific configuration file created by the client programmer

$XUSERFILESEARCHPATH
: application-specific configuration file created by the system administrator or user

2.8.3 Environment variables

Detailed descriptions of all the environment variables, arranged in alphabetical order, are given in the chapter "Environment variables ".

DISPLAY
: The client reads the DISPLAY environment variable if the -display option is not specified in its command line. From this variable the client finds out the name of the X server and the display and screen on it.
The DISPLAY environment variable is a string in the format:
hostname:*display_number*.*screen_number*
The client terminates abnormally if the DISPLAY environment variable is not set and the -display option is not specified in the client command line.

HOMEidentifies the user's HOME directory.

LANGis evaluated in the context of internationalization.

MAIL absolute pathname of the file for the user's incoming mail

NLSPATH
: The client evaluates the NLSPATH environment variable as a search path for message files.
Thus the NLSPATH environment variable must include the search path /usr/lib/nls/msg/%l/%N.cat. "%l" is replaced by that part of the locale definition which identifies the language. The client replaces "%N" with "xdcm".

USERidentifies the user's login name.

XAUTHORITY
: denotes the user-specific authorization file.
If the environment variable is not set, the client searches for the $HOME/.Xauthority file.

XDCM_TRASH
: specifies the directory in which xdcm creates the directory for its trash. If the variable is not set, the client uses $HOME/.xdcmtrash.

XDESK_CFG_PATH
: defines a list of directories to search for the client's XDCL configuration file. If the variable is not set, the client evaluates the HOME environment variable. If there is no

XDCL configuration file of the client in $HOME, the client reads from the directory /usr/lib/X11/xdesk/cfg the XDCL configuration file xdcm.cfg.

XDESK_ICON_PATH
: defines a user-specific list of files the client is to scan for the file containing the icon image for an object that it manages. If this environment variable is set, it overrides the value of the iconPath resource of class IconPath. The list may include the symbolic names described in the iconPath resource.

XDESK_SAVE_DIR
: defines the directory in which the client stores its save files. If the variable is not set, the client evaluates the HOME environment variable.

XDESK_SYSTEM_ICON_PATH
: defines a system-specific list of files the client is to scan for the file containing the icon image for an object that it manages. If this environment variable is set, it overrides the value of the systemIconPath resource of class SystemIconPath. The list may include the symbolic names described in the systemIconPath resource.

XENVIRONMENT
: specifies the path for configuration files.

XFILESEARCHPATH
: specifies the path for configuration files.

XUSERFILESEARCHPATH
: specifies the path for configuration files.

2.9 xdnm - Managing NFS

The xdnm client, the NFS manager, allows you to display information about the file systems and file system subtrees which are mounted over NFS (Network File System) on your host or which are on your host and are shared over NFS with other hosts. It is available to system administrators for two main purposes:

- Managing mount points as objects: this lets system administrators mount remote file systems on the local host over NFS, change the attributes of an NFS-mounted file system or remove an NFS-mounted file system from the mount table.
- Managing share points as objects: this lets system administrators perform tasks such as making file systems on the local host available to remote hosts and defining their attributes. These file systems can be mounted over NFS on approved remote hosts.

The client can run under a login name on a given host only once per display.
The appearance and behavior of the client are documented in help texts.
The client's appearance and behavior can be configured dynamically using the xdtell client which, among other things, allows new client windows defined in the XDCL language to be output during runtime. The xdnm client evaluates instances of the XDCL class "nfsmanager_view" for this purpose.
The client's appearance and behavior can be configured with the XDCL language. This language allows experienced users and system administrators to change the standard configuration and customize the appearance and behavior of the client to meet their requirements. The manual "[4]" "SINIX/windows User Environment; XDCL Desktop Configuration Language" is a reference work providing a complete overview of which objects the client makes available, which characteristics can be specified, and which methods can be initiated by user actions.
On startup the client evaluates the XDESK_CFG_PATH environment variable and searches the directory named there for its .xdnm.cfg configuration file, which must conform to XDCL Desktop Configuration Language XDCL syntax. The user can edit this file so as to modify the appearance and behavior of the client.
The client automatically generates save files in XDCL whenever the user configures objects interactively and asks the client to save the current state. xdesk stores save files in the directory specified by the XDESK_SAVE_DIR environment variable. When called again, the client attempts to restore the saved state. Object definitions which are in the save files but have been modified in the configuration file are ignored; the definitions in the configuration file take precedence.

xdnm[*option*]...

-display [*hostname*]:*display_number*[.*screen_number*]
Identifies the X server, display and screen which the client is to contact.

hostname
symbolic name of the host or X terminal on which the X server which the client is to contact runs. *hostname* must be known in the transport protocol of the host on which the client resides.

display_number
number (unique to the X server) identifying the display for which the client is to manage I/O.

.screen_number
number (unique to the display) identifying the screen to which the client is to send output.

Default:
contents of the DISPLAY shell variable;
if DISPLAY is not set, the client terminates abnormally.

-open *filesystem* [*view*]
Starts the NFS manager with the view *view* on the file system *filesystem*.
view is an instance of the XDCL object class "nfsmanager_view" defined in the configuration file, which is to be used as a view. If *view* is not specified, the client uses the "default" instance of the "nfsmanager_view" class.

Default:
If there is save information, the NFS manager opens all windows recorded there. If there is no save information, the NFS manager opens a window with a view on the file system identified by the "initial_folder" attribute in the "default" instance of the XDCL object class "nfsmanager_view". If "initial_folder" is not specified, it displays all file systems on remote hosts which are NFS-mounted on the local host and all file systems on the local host which are shared with remote hosts.

-xrm "*resourceDefinition*:*value*"
Assigns a value to a client resource.
-xrm must be specified separately for each resource to be set in the client call. The "*resourceDefinition*:*value*" assignment is compared with the entries in the configuration files.

resourceDefinition
Identifies the resource to which a value is to be assigned. For *resourceDefinition* you can give the name of the resource, the name of the client and of any widgets in the appropriate resource syntax (see section "Value assignment syntax ").

value Value to be assigned to the resource.

2.9.1 Resources

You can configure the client using its resource database. To do this, you must assign the required values to the resources. You can use either configuration files or the -xrm option in the client call to access resources.
The instance name of the client is xdnm and its class name NFSManager.

iconPath
Class: IconPath
See resources iconSet, systemIconPath
defines the list of files the client is to search for the icon image for an object that it manages.
No default

iconSet
Class: IconSet
See resources iconPath, systemIconPath
defines the set of icon images the client uses for the objects that it manages.

Default:
depends on the number of color planes on the monitor:

mono for a monitor supporting one color plane
color for a monitor supporting more than one color plane

saveOnExit
Class: SaveOnExit
Specifies whether the client saves its status when terminating.

Default:
ask

standardIconSize
Class: StandardIconSize
Size of the icons when the client is started

Default:
Auto

systemIconPath
Class: SystemIconPath
See resources iconPath, iconSet
Specifies the system-specific part of the list of files the client searches for the icon image for an object it manages.

Default:
/usr/lib/X11/siwin/icons/hosts/%l/%N:\
/usr/lib/X11/siwin/icons/wm/%l/%N:\
/usr/lib/X11/siwin/icons/common/%l/%N

2.9.2 Files

The files below are listed in alphabetical order. The resource configuration files are listed only in the form of the generalized file names $XENVIRONMENT, $XUSERFILESEARCHPATH and $XFILESEARCHPATH.

.xdnm.cfg
XDCL configuration file of the client in a directory on the path $XDESK_CFG_PATH.
.xdnm.ict
Save file created after interactive configuration and saving of the client in the directory defined in the XDESK_SAVE_DIR environment variable.
.xdnm.session

Save file of the client for saving views. It is created in the directory defined in the XDESK_SAVE_DIR environment variable.

/etc/dfs/dfstab
: File the client uses to fetch and store information about the attributes of mount points.

/etc/vfstab
: File the client uses to fetch and store information about the attributes of share points.

$XAUTHORITY
: user-specific authorization file
The client uses this file as a user-specific authorization file. If XAUTHORITY is not set, the client uses the .Xauthority file in the user's HOME directory.

$XENVIRONMENT
: user-specific configuration file

$XFILESEARCHPATH
: application-specific configuration file created by the client programmer

$XUSERFILESEARCHPATH
: application-specific configuration file created by the system administrator or user

2.9.3 Environment variables

Detailed descriptions of all the environment variables, arranged in alphabetical order, are given in the chapter "Environment variables ".

DISPLAY
: The client reads the DISPLAY environment variable if the -display option is not specified in its command line. From this variable the client finds out the name of the X server and the display and screen on it.
The DISPLAY environment variable is a string in the format:
hostname:*display_number*.*screen_number*
The client terminates abnormally if the DISPLAY environment variable is not set and the -display option is not specified in the client command line.

HOMEidentifies the user's HOME directory.

LANGis evaluated in the context of NLS language support .

NLSPATH
: The client evaluates the NLSPATH environment variable as a search path for message files.
Thus the NLSPATH environment variable must include the search path /usr/lib/nls/msg/%l/%N.cat. "%l" is replaced by that part of the locale definition which identifies the language. The client replaces "%N" with "xdnm".

XAUTHORITY
: denotes the user-specific authorization file.
If the environment variable is not set, the client searches for the $HOME/.Xauthority file.

XDESK_CFG_PATH
: defines a list of directories to search for the client's XDCL configuration file. If the variable is not set, the client evaluates the HOME environment variable. If there is no XDCL configuration file of the client in $HOME, the client reads the XDCL

configuration file xdnm.cfg from the directory /usr/lib/X11/xdesk/cfg.

XDESK_ICON_PATH
: defines a user-specific list of files the client is to scan for the file containing the icon image for an object that it manages. If this environment variable is set, it overrides the value of the iconPath resource of class IconPath. The list may include the symbolic names described in the iconPath resource.

XDESK_SAVE_DIR
: defines the directory in which the client stores its save files. If the variable is not set, the client evaluates the HOME environment variable.

XDESK_SYSTEM_ICON_PATH
: defines a system-specific list of files the client is to scan for the file containing the icon image for an object that it manages. If this environment variable is set, it overrides the value of the systemIconPath resource of class SystemIconPath. The list may include the symbolic names described in the systemIconPath resource.

XENVIRONMENT
: specifies the path for configuration files.

XFILESEARCHPATH
: specifies the path for configuration files.

XUSERFILESEARCHPATH
: specifies the path for configuration files.

2.10 xdsm - Managing software

The xdsm client, the software manager, allows you to display information on software packages. It enables system administrators to install, test and deinstall software packages. The client also offers the capability of transferring software packages from data media, such as CD-ROMs, tape cartridges or diskettes, to the system. The xdsm client is the equivalent of the software manager section of client xdvm.
The client can run under a login name on a given host only once per display.
The appearance and behavior of the client are documented in help texts.
The client's appearance and behavior can be configured dynamically using the xdtell client, which, among other things, allows new client windows defined in the XDCL language to be output during runtime. The xdsm client evaluates instances of the XDCL class "pkgmanager_view" for this purpose.
The client's appearance and behavior can be configured with the XDCL language. This language allows experienced users and system administrators to change the standard configuration and customize the appearance and behavior of the client to meet their requirements. The manual "[4]" "SINIX/windows User Environment; XDCL Desktop Configuration Language" is a reference work providing a complete overview of which objects the client makes available, which characteristics can be specified, and which methods can be initiated by user actions.
On startup the client evaluates the XDESK_CFG_PATH environment variable and searches the directory named there for its .xdsm.cfg configuration file, which must conform to Desktop Configuration Language XDCL syntax. The user can edit this file so as to modify the appearance and behavior of the client.
The client generates save files in XDCL whenever the user configures objects interactively and asks the client to save the current state. It writes the save files to the directory specified by the XDESK_SAVE_DIR environment variable. When called again, the client attempts to restore the saved state. Object definitions which are in the save files but have been modified in the configuration file are ignored.

xdsm[*option*]...

-display [*hostname*]:*display_nummer*[.*screen_number*]
identifies the X server, display and screen which the client is to contact.

hostname
symbolic name of the host or X terminal on which the X server which the client is to contact runs. *hostname* must be known in the transport protocol of the host on which the client resides.
:*display_number*
number (unique to the X server) identifying the display for which the client is to manage I/O.
.*screen_number*
number (unique to the display) identifying the screen to which the client is to send outputs.

Default:
contents of the DISPLAY variable;

if DISPLAY is not set, the client terminates abnormally.

-open *position*[*:product*] [*view*]

Starts the client with the view *view* on *position*[*:product*].

position indicates the location of a software package. The following values for *position* are possible:

system
spool
file directory with software packages
CD-ROM insertion point.

product denotes a software product. Evaluated only in the case of a product CD.

view is an instance of the XDCL object class "pkgmanager_view" defined in the configuration file, which is to be used as a view. If *view* is not specified, the client uses the "default" instance of the "pkgmanager_view" class.

Default:

If there is save information, the host manager opens all windows recorded there. If there is no save information, the host manager opens a window with a view of all installed software of the host on which it is running.

-xrm "*resourceDefinition*:*value*"

Assigns a value to a client resource.

-xrm must be specified separately for each resource to be set in the client call. The "*resourceDefinition*:*value*" assignment is compared with the entries in the configuration files.

resourceDefinition

Identifies the resource to which a value is to be assigned. For *resourceDefinition* you can give the name of the resource, the name of the client and of any widgets in the appropriate resource syntax (see section "Value assignment syntax ").

value Value to be assigned to the resource.

2.10.1 Resources

You can configure the client using its resource database. To do this, you must assign the required values to the resources. You can use either configuration files or the -xrm option in the client call to access resources.
The instance name of the client is xdsm and its class name VolumeManager.

iconPath
Class: IconPath

See resources iconSet, systemIconPath

defines the list of files the client is to search for the icon image for an object that it manages.

No default

iconSet
Class: IconSet

See resources iconPath, systemIconPath

defines the set of icon images the client uses for the objects that it manages.

Default:
depends on the number of color planes on the monitor:

mono for a monitor supporting one color plane
color for a monitor supporting more than one color plane

saveOnExit
Class: SaveOnExit
Specifies whether the client saves its status when terminating.

Default:
ask

standardIconSize
Class: StandardIconSize
Size of the icons when the client is started

Default:
Auto

systemIconPath
Class: SystemIconPath
See resources iconPath, iconSet
Specifies the system-specific part of the list of files the client searches for the icon image for an object it manages.

Default:
/usr/lib/X11/siwin/icons/volumes/%I/%N:\
/usr/lib/X11/siwin/icons/wm/%I/%N:\
/usr/lib/X11/siwin/icons/common/%I/%N

2.10.2 Files

The files below are listed in alphabetical order. The resource configuration files are listed only in the form of the generalized file names $XENVIRONMENT, $XUSERFILESEARCHPATH and $XFILESEARCHPATH.

.xdsm.cfg
XDCL configuration file of the client in a directory on the path $XDESK_CFG_PATH.

.xdsm.ict
Save file created after interactive configuration and saving of the client in the directory defined in the XDESK_SAVE_DIR environment variable.

.xdsm.session
Save file of the client for saving views on archives and file systems. It is created in the directory defined in the XDESK_SAVE_DIR environment variable.

$XAUTHORITY

user-specific authorization file
The client uses this file as a user-specific authorization file. If XAUTHORITY is not set, the client uses the .Xauthority file in the user's HOME directory.

$XENVIRONMENT
user-specific configuration file

$XFILESEARCHPATH
application-specific configuration file created by the client programmer

$XUSERFILESEARCHPATH
application-specific configuration file created by the system administrator or user

2.10.3 Environment variables

Detailed descriptions of all the environment variables, arranged in alphabetical order, are given in the chapter "Environment variables ".

DISPLAY
The client reads the DISPLAY environment variable if the -display option is not specified in its command line. From this variable the client finds out the name of the X server and the display and screen on it.
The value of the DISPLAY environment variable has the following syntax:
hostname:*display_number*.*screen_number*
The client terminates abnormally if the DISPLAY environment variable is not set and the -display option is not specified in the client command line.

HOMEidentifies the user's HOME directory.

LANGis evaluated in the context of NLS language support .

NLSPATH
The client evaluates the NLSPATH environment variable as a search path for message files.
Thus the NLSPATH environment variable must include the search path /usr/lib/nls/msg/%l/%N.cat. "%l" is replaced by that part of the locale definition which identifies the language. The client replaces "%N" with "xdvm".

XAUTHORITY
denotes the user-specific authorization file.
If the environment variable is not set, the client searches for the $HOME/.Xauthority file.

XDESK_CFG_PATH
defines a list of directories to search for the client's XDCL configuration file. If the variable is not set, the client evaluates the HOME environment variable. If there is no XDCL configuration file of the client in $HOME, the client reads from the directory /usr/lib/X11/xdesk/cfg the XDCL configuration file xdsm.cfg.

XDESK_ICON_PATH
defines a user-specific list of files the client is to scan for the file containing the icon image for an object that it manages. If this environment variable is set, it overrides the value of the iconPath resource of class IconPath. The list may include the symbolic names described in the iconPath resource.

XDESK_SAVE_DIR
defines the directory in which the client stores its save files. If the variable is not set,

the client evaluates the HOME environment variable.

XDESK_SYSTEM_ICON_PATH
: defines a system-specific list of files the client is to scan for the file containing the icon image for an object that it manages. If this environment variable is set, it overrides the value of the systemIconPath resource of class SystemIconPath. The list may include the symbolic names described in the systemIconPath resource.

XENVIRONMENT
: specifies the path for configuration files.

XFILESEARCHPATH
: specifies the path for configuration files.

XUSERFILESEARCHPATH
: specifies the path for configuration files.

2.11 xdtell - Access to xdesk, xdcm, xdfm, xdhm, xdnm, xdsm, xdum and xdvm

xdtell is an interface to the xdfm, xdhm, xdnm, xdcm, xdum and xdvm managers. Since the xdesk client also provides a file manager, xdtell can influence xdesk as well. For the xdesk client, you have to specify the type "fm" (i.e. file manager) in the -manager option.
xdtell determines which manager to access on the basis of the -manager, -host and -user options. You must specify the -manager option with all xdtell options, the only exceptions being -stop and -restart. In the case of the -stop and -restart options, you can omit the -manager option; xdtell then accesses all managers that match the options specified in the client call.
By means of an xdtell call, a manager can be accessed with all its windows, for specific windows or for individual objects that it manages.
Specifically, xdtell causes the file manager

- to be started, saved or stopped
- to open, update or close windows with a certain view on objects
- to process an object.

Only one instance of the xdesk, xdfm, xdhm, xdnm, xdcm, xdsm, xdum and xdvm managers per system and login name can be run on a display. In conjunction with the -host and -user options, xdtell allows managers on other hosts or other login names to be accessed from the local host.
The access permissions must be set such that xdtell can be called on the local host and that it in turn can access a manager on a remote host:

- The login name specified in the -user option must also exist on the remote host.
- The $HOME/.rhosts file must exist on the remote host for the login name specified in the -user option, and the name of the local host and the login name under which xdtell is called must be entered in it.
- The manager must be granted access to its display by the login name on the remote host. You can use the xauth and xhost clients to change the access permissions.

It is particularly useful to include calls to the xdtell client as external methods in the XDCL configuration file of the xdesk, xdfm, xdhm, xdnm, xdcm, xdum and xdvm managers.
The exit status codes of xdtell are as follows:
0: client invocation was successful
1: call contained invalid arguments
2: display could not be opened
3: specified manager not running
4: specified host not available
5: specified manager already running.
xdtell[-manager *manager_type*][-host *hostname*][-user *user*][*option*]...
The options -close, -exec, -open, -restart, -save, -start, -stop, -update, opentrashcan and -emptytrashcan are mutually exclusive.

-close [*container*]

See options -manager ("-manager manager_type") and -open ("-open container [view]")
Closes all views on *containers* of the manager specified in the client call.
If *container* is omitted, all views on all containers of the manager specified in the client call are closed, but the manager is not exited. If the manager is not running, this option has no effect.

-emptytrashcan

See -manager option ("-manager manager_type")
Empties the trash of the manager specified in the client call.
The objects contained in the manager's trash are therefore no longer accessible.

-exec *method file*

See -manager option ("-manager manager_type")
Instructs the specified file manager to execute the XDCL named *method* on the named *file*.
When you use this option, you must specify the value "fm" in the call for the -manager option. If you omit the -manager option, xdtell only accesses managers of the type "fm", i.e. file managers.
method, for example "edit" or "print", must be associated with the file in the configuration file.
file must be an absolute pathname.
If the file manager is not running then one will be started, specifying *method* and *file* as the initial values.

-host *hostname*

See options -manager ("-manager manager_type"), -user ("-user user")
This is the name of the host on which the manager of the type *manager_type* under the login name *user* is to be accessed.

Default:

hostname defaults to the local host.

-manager *manager_type*

See options -host ("-host hostname"), -user ("-user user")
Identifies the type of the manager to be addressed on the host *hostname* under the login name *user*.
The following are possible values for *manager_type*:

cm xdcm communication manager,
fm xdfm file manager,
nm xdnm NFS manager,
hm xdhm host manager,
sm xdsm software manager,
um xdum user manager,
vm xdvm volume manager.

It is not possible to use xdtell to address simultaneously managers of more than one type except with the default of this option. You must specify the -manager option with all xdtell options, the only exceptions being -stop and -restart.

Default:

Managers of all types are addressed.

-open *container* [*view*]

See -manager option ("-manager manager_type")
See -open option of appropriate manager
Opens a window of the manager specified in the client call with the view *view* on the container *container* for the objects that the manager of this type manages.

The window is opened irrespective of whether other windows of the manager are viewing the specified *container*.
Depending on the manager specified in the client call, *view* is an instance of the corresponding manager's XDCL object class defined in the configuration file:
- "filemanager_view" for managers of type "fm",
- "hostmanager_view" for managers of type "hm",
- "mailmanager_view" for managers of type "cm",
- "nfsmanager_view" for managers of type "nm",
- "pkgmanager_view" for managers of type "sm",
- "usermanager_view" for managers of type "um",
- "archivemanager_view" for managers of type "vm".

If *view* is not specified, the manager uses the "default" instance of the corresponding XDCL object class.
If the manager of the specified type is not running, it is started with the values *container* and *view*.

-opentrashcan
See -manager option ("-manager manager_type")
Opens a window of the manager specified in the client call with a view on those objects of the manager that are in the trash can.
The window is opened irrespective of whether other windows of the manager already offer the same view.

-restart
Terminates and restarts the manager specified in the client call.

-save See -manager option ("-manager manager_type")
Saves the current status of the manager specified in the client call.
If the manager is not running this command will fail.

-start
See -manager option ("-manager manager_type")
Starts the manager specified in the client call.

-stop Terminates the manager specified in the client call without saving the current status.
If no manager is running this command will fail.

-update [*container*]
Refreshes all views on *containers* of the manager specified in the client call.
If *container* is omitted, all views on all containers of the manager are refreshed. If the manager is not running this command will fail.

-user *user*
See options -host ("-host hostname"), -manager ("-manager manager_type")
user is the login name under which the manager of the type *manager_type* on the host *hostname* is to be addressed.

Default:
user is the login name under which xdtell is called.

2.11.1 Files

$HOME/.rhosts
contains entries for hosts and login names granted access to the local host.

$XAUTHORITY
: user-specific authorization file
The client uses this file as a user-specific authorization file. If XAUTHORITY is not set, the client uses the .Xauthority file in the user's HOME directory.

2.11.2 Environment variables

Detailed descriptions of all the environment variables, arranged in alphabetical order, are given in the chapter "Environment variables ".

DISPLAY
: The client reads the DISPLAY environment variable if the -display option is not specified in its command line. From this variable the client finds out the name of the X server and the display and screen on it.
The DISPLAY environment variable is a string in the format:
hostname:*display_number*.*screen_number*
The client terminates abnormally if the DISPLAY environment variable is not set and the -display option is not specified in the client command line.

HOMEidentifies the user's HOME directory.

XAUTHORITY
: denotes the user-specific authorization file.
If the environment variable is not set, the client searches for the $HOME/.Xauthority file.

2.12 xdum - Managing users

The xdum client, the user manager, allows you to display login names and user groups known to your host. In addition, it lets system administrators manage them as objects. System administrators can, for example, add new login names and delete existing ones, add new user groups, define audit events, and interactively assign to login names and groups icons used, for example, by xdm in the xdm login window. The client can run under a login name on a given host only once per display.
The appearance and behavior of the client are documented in help texts.
The client's appearance and behavior can be configured dynamically using the xdtell client which, among other things, allows new client windows defined in the XDCL language to be output during runtime. The xdum client evaluates instances of the XDCL class "usermanager_view" for this purpose.
The client's appearance and behavior can be configured with the XDCL language. This language allows experienced users and system administrators to change the standard configuration and customize the appearance and behavior of the client to meet their requirements. The manual "[4]" "SINIX/windows User Environment; XDCL Desktop Configuration Language" is a reference work providing a complete overview of which objects the client makes available, which characteristics can be specified, and which methods can be initiated by user actions.
On startup the client evaluates the XDESK_CFG_PATH environment variable and searches the directory named there for its .xdum.cfg configuration file, which must conform to XDCL Desktop Configuration Language XDCL syntax. The user can edit this file so as to modify the appearance and behavior of the client.
The client automatically generates save files in XDCL whenever the user configures objects interactively and asks the client to save the current state. xdesk stores save files in the directory specified by the XDESK_SAVE_DIR environment variable. When called again, the client attempts to restore the saved state. Object definitions which are in the save files but have been modified in the configuration file are ignored; the definitions in the configuration file take precedence.

xdum[*option*]...

-display [*hostname*]:*display_number*[.*screen_number*]
Identifies the X server, display and screen which the client is to contact.

hostname
symbolic name of the host or X terminal on which the X server which the client is to contact runs. *hostname* must be known in the transport protocol of the host on which the client resides.
display_number
number (unique to the X server) identifying the display for which the client is to manage I/O.
.screen_number
number (unique to the display) identifying the screen to which the client is to send output.

Default:

contents of the DISPLAY shell variable;
if DISPLAY is not set, the client terminates abnormally.

-open *group* [*view*]
Starts the user manager with the view *view* on the user group *group*.
view is an instance of the XDCL object class "usermanager_view" defined in the configuration file, which is to be used as a view. If *view* is not specified, the client uses the "default" instance of the "usermanager_view" class.

Default:
If there is save information, the file manager opens all windows recorded there. If there is no save information, the user manager opens a window with a view on the group identified by the "initial_folder" attribute in the "default" instance of the XDCL object class "usermanager_view". If "initial_folder" is not specified, all users of the local host are displayed.

-xrm "*resourceDefinition*:*value*"
Assigns a value to a client resource.
-xrm must be specified separately for each resource to be set in the client call. The "*resourceDefinition*:*value*" assignment is compared with the entries in the configuration files.

resourceDefinition
Identifies the resource to which a value is to be assigned. For *resourceDefinition* you can give the name of the resource, the name of the client and of any widgets in the appropriate resource syntax (see section "Value assignment syntax ").

value Value to be assigned to the resource.

2.12.1 Resources

You can configure the client using its resource database. To do this, you must assign the required values to the resources. You can use either configuration files or the -xrm option in the client call to access resources.
The instance name of the client is xdum and its class name UserManager.

iconPath
Class: IconPath
See resources iconSet, systemIconPath
defines the list of files the client is to search for the icon image for an object that it manages.
No default

iconSet
Class: IconSet
See resources iconPath, systemIconPath
defines the set of icon images the client uses for the objects that it manages.

Default:

depends on the number of color planes on the monitor:

mono for a monitor supporting one color plane
color for a monitor supporting more than one color plane

saveOnExit
Class: SaveOnExit
Specifies whether the client saves its status when terminating.

Default:
ask

standardIconSize
Class: StandardIconSize
Size of the icons when the client is started

Default:
Auto

systemIconPath
Class: SystemIconPath
See resources iconPath, iconSet
Specifies the system-specific part of the list of files the client searches for the icon image for an object it manages.

Default:
/usr/lib/X11/siwin/icons/users/%I/%N:\
/usr/lib/X11/siwin/icons/wm/%I/%N:\
/usr/lib/X11/siwin/icons/common/%I/%N

2.12.2 Files

The files below are listed in alphabetical order. The resource configuration files are listed only in the form of the generalized file names $XENVIRONMENT, $XUSERFILESEARCHPATH and $XFILESEARCHPATH.

.xdum.cfg
XDCL configuration file of the client in a directory on the path $XDESK_CFG_PATH.
.xdum.ict
Save file created after interactive configuration and saving of the client in the directory defined in the XDESK_SAVE_DIR environment variable.
.xdum.session
Save file of the client for saving views on groups. It is created in the directory defined in the XDESK_SAVE_DIR environment variable.
/etc/.xdum.gtrash
In this file the client stores the names of user groups which are in trash.
/etc/.xdum.utrash
In this file the client stores the names of user logins which are in trash.
$XAUTHORITY

user-specific authorization file
The client uses this file as a user-specific authorization file. If XAUTHORITY is not set, the client uses the .Xauthority file in the user's HOME directory.

$XENVIRONMENT
user-specific configuration file

$XFILESEARCHPATH
application-specific configuration file created by the client programmer

$XUSERFILESEARCHPATH
application-specific configuration file created by the system administrator or user

2.12.3 Environment variables

Detailed descriptions of all the environment variables, arranged in alphabetical order, are given in the chapter "Environment variables ".

DISPLAY
The client reads the DISPLAY environment variable if the -display option is not specified in its command line. From this variable the client finds out the name of the X server and the display and screen on it.
The DISPLAY environment variable is a string in the format:
hostname:*display_number*.*screen_number*
The client terminates abnormally if the DISPLAY environment variable is not set and the -display option is not specified in the client command line.

HOMEidentifies the user's HOME directory.

LANGis evaluated in the context of NLS language support .

NLSPATH
The client evaluates the NLSPATH environment variable as a search path for message files.
Thus the NLSPATH environment variable must include the search path /usr/lib/nls/msg/%l/%N.cat. "%l" is replaced by that part of the locale definition which identifies the language. The client replaces "%N" with "xdum".

XAUTHORITY
denotes the user-specific authorization file.
If the environment variable is not set, the client searches for the $HOME/.Xauthority file.

XDESK_CFG_PATH
defines a list of directories to search for the client's XDCL configuration file. If the variable is not set, the client evaluates the HOME environment variable. If there is no XDCL configuration file of the client in $HOME, the client reads from the directory /usr/lib/X11/xdesk/cfg the XDCL configuration file xdum.cfg.

XDESK_ICON_PATH
defines a user-specific list of files the client is to scan for the file containing the icon image for an object that it manages. If this environment variable is set, it overrides the value of the iconPath resource of class IconPath. The list may include the symbolic names described in the iconPath resource.

XDESK_SAVE_DIR
defines the directory in which the client stores its save files. If the variable is not set, the client evaluates the HOME environment variable.

XDESK_SYSTEM_ICON_PATH
: defines a system-specific list of files the client is to scan for the file containing the icon image for an object that it manages. If this environment variable is set, it overrides the value of the systemIconPath resource of class SystemIconPath. The list may include the symbolic names described in the systemIconPath resource.

XENVIRONMENT
: specifies the path for configuration files.

XFILESEARCHPATH
: specifies the path for configuration files.

XUSERFILESEARCHPATH
: specifies the path for configuration files.

2.13 xdvm - Managing archives, file systems on removable media, and software

The xdvm client, the volume manager, allows you to access archives and file systems on removable storage media, to process software installation media and to manage them as objects. This tool can also be configured as a remote service tool in order to start the volume manager on remote hosts in your network. SINIX/windows must be installed on these hosts. This typically enables you to manage storage media (CD-ROMs, floppy disks, streamer tapes) and the corresponding devices, to extract archives, and to back up archives on media on which this is possible. You can also work with file systems on media which support file systems. If the medium contains installable software, you can install this in your system.
The xdvm clients consists of three related components:

- The device manager in its window displays all the devices on the local host which are identified as removable media in the /etc/device.tab file. The device manager window opens when the volume manager starts up. It represents the devices in icon form, the top half showing archive devices, the bottom half file system devices. Entries in the icon pop-up menu let you open an archive manager window or a file manager window or a software manager window, depending on whether the selected medium supports archives, file systems or installable software.
- The archive manager in its window displays the contents of an archive in icon form. It usually does this by reading a medium in one of the devices displayed by the device manager.This is the relationship between the two components of the volume manager. However, you can also process archives which are stored in file form on a hard disk.
- The software manager in its window displays the contents of an installation medium in icon form for the individual software packages or products. It does this by reading a medium in one of the devices displayed by the device manager. This is the relationship between the software manager and the device manager. However, you can also process installation directories which are stored on a hard disk. To view the software installed in the system, you can also start the software manager directly as a separate service tool.

The client can run under a login name on a given host only once per display.
The appearance and behavior of the client are documented in help texts.
The client's appearance and behavior can be configured dynamically using the xdtell client which, among other things, allows new client windows defined in the XDCL language to be output during runtime. The xdvm client evaluates instances of the XDCL classes "`devicemanager_view`", "`pkgmanager_view`" and "`archivemanager_view`" for this purpose.
The client's appearance and behavior can be configured with the XDCL language. This language allows experienced users and system administrators to change the standard configuration and customize the appearance and behavior of the client to meet their requirements. The manual "[4]" "SINIX/windows User Environment; XDCL Desktop Configuration Language" is a reference work providing a complete overview of which objects the client makes available, which characteristics can be specified, and which methods can be initiated by user actions.
On startup the client evaluates the XDESK_CFG_PATH environment variable and searches the directory named there for its .xdvm.cfg configuration file, which must conform to XDCL Desktop Configuration Language XDCL syntax. The user can edit this file so as to modify the appearance and behavior of the client.
The client automatically generates save files in XDCL whenever the user configures objects interactively and asks the client to save the current state. xdesk stores save files in the directory specified by the XDESK_SAVE_DIR environment variable. When called again, the

client attempts to restore the saved state. Object definitions which are in the save files but have been modified in the configuration file are ignored; the definitions in the configuration file take precedence.

xdvm[*option*]...

-display [*hostname*]:*display_number*[.*screen_number*]
Identifies the X server, display and screen which the client is to contact.

hostname
symbolic name of the host or X terminal on which the X server which the client is to contact runs. *hostname* must be known in the transport protocol of the host on which the client resides.
display_number
number (unique to the X server) identifying the display for which the client is to manage I/O.
.*screen_number*
number (unique to the display) identifying the screen to which the client is to send output.

Default:
contents of the DISPLAY shell variable;
if DISPLAY is not set, the client terminates abnormally.

-open *deviceOrFile* [*view*]
Starts the client with the view *view* on *deviceOrFile* or a file *deviceOrFile*.
deviceOrFile is either a raw device identified as a removable medium by having its "removable" attribute set to "true" in the /etc/device.tab file, or it is a file.
view is an instance of the XDCL object class "archivemanager_view" defined in the configuration file, which is to be used as a view. If *view* is not specified, the client uses the "default" instance of the "archivemanager_view" class.

Default:
If there is save information, the manager opens all windows recorded there. If there is no save information, the manager opens a window with a view on all devices on the local host which are identified as removable media in the /etc/device.tab file.

-xrm "*resourceDefinition*:*value*"
Assigns a value to a client resource.
-xrm must be specified separately for each resource to be set in the client call. The "*resourceDefinition*:*value*" assignment is compared with the entries in the configuration files.

resourceDefinition
Identifies the resource to which a value is to be assigned. For *resourceDefinition* you can give the name of the resource, the name of the client and of any widgets in the appropriate resource syntax (see section "Value assignment syntax ").

value Value to be assigned to the resource.

2.13.1 Resources

You can configure the client using its resource database. To do this, you must assign the required values to the resources. You can use either configuration files or the -xrm option in the client call to access resources.
The instance name of the client is xdvm and its class name VolumeManager.

iconPath
Class: IconPath
See resources iconSet, systemIconPath
defines the list of files the client is to search for the icon image for an object that it manages.
No default
iconSet
Class: IconSet
See resources iconPath, systemIconPath
defines the set of icon images the client uses for the objects that it manages.

Default:
depends on the number of color planes on the monitor:

mono for a monitor supporting one color plane
color for a monitor supporting more than one color plane
saveOnExit
Class: SaveOnExit
Specifies whether the client saves its status when terminating.

Default:
ask
standardIconSize
Class: StandardIconSize
Size of the icons when the client is started

Default:
Auto
systemIconPath
Class: SystemIconPath
See resources iconPath, iconSet
Specifies the system-specific part of the list of files the client searches for the icon image for an object it manages.

Default:
/usr/lib/X11/siwin/icons/volumes/%I/%N:\

/usr/lib/X11/siwin/icons/wm/%I/%N:\
/usr/lib/X11/siwin/icons/common/%I/%N

2.13.2 Files

The files below are listed in alphabetical order. The resource configuration files are listed only in the form of the generalized file names $XENVIRONMENT, $XUSERFILESEARCHPATH and $XFILESEARCHPATH.

.xdvm.cfg
: XDCL configuration file of the client in a directory on the path $XDESK_CFG_PATH.

.xdvm.ict
: Save file created after interactive configuration and saving of the client in the directory defined in the XDESK_SAVE_DIR environment variable.

.xdvm.session
: Save file of the client for saving views on archives and file systems. It is created in the directory defined in the XDESK_SAVE_DIR environment variable.

/etc/device.tab
: This is the file the client reads for information on the removable-media devices which it displays.

$XAUTHORITY
: user-specific authorization file
The client uses this file as a user-specific authorization file. If XAUTHORITY is not set, the client uses the .Xauthority file in the user's HOME directory.

$XENVIRONMENT
: user-specific configuration file

$XFILESEARCHPATH
: application-specific configuration file created by the client programmer

$XUSERFILESEARCHPATH
: application-specific configuration file created by the system administrator or user

2.13.3 Environment variables

Detailed descriptions of all the environment variables, arranged in alphabetical order, are given in the chapter "Environment variables ".

DISPLAY
: The client reads the DISPLAY environment variable if the -display option is not specified in its command line. From this variable the client finds out the name of the X server and the display and screen on it.
The DISPLAY environment variable is a string in the format:
hostname:*display_number*.*screen_number*
The client terminates abnormally if the DISPLAY environment variable is not set and the -display option is not specified in the client command line.

HOMEidentifies the user's HOME directory.

LANGis evaluated in the context of NLS language support .

NLSPATH
: The client evaluates the NLSPATH environment variable as a search path for

message files.
Thus the NLSPATH environment variable must include the search path /usr/lib/nls/msg/%l/%N.cat. "%l" is replaced by that part of the locale definition which identifies the language. The client replaces "%N" with "xdvm".

XAUTHORITY
denotes the user-specific authorization file.
If the environment variable is not set, the client searches for the $HOME/.Xauthority file.

XDESK_CFG_PATH
defines a list of directories to search for the client's XDCL configuration file. If the variable is not set, the client evaluates the HOME environment variable. If there is no XDCL configuration file of the client in $HOME, the client reads from the directory /usr/lib/X11/xdesk/cfg the XDCL configuration file xdvm.cfg.

XDESK_ICON_PATH
defines a user-specific list of files the client is to scan for the file containing the icon image for an object that it manages. If this environment variable is set, it overrides the value of the iconPath resource of class IconPath. The list may include the symbolic names described in the iconPath resource.

XDESK_SAVE_DIR
defines the directory in which the client stores its save files. If the variable is not set, the client evaluates the HOME environment variable.

XDESK_SYSTEM_ICON_PATH
defines a system-specific list of files the client is to scan for the file containing the icon image for an object that it manages. If this environment variable is set, it overrides the value of the systemIconPath resource of class SystemIconPath. The list may include the symbolic names described in the systemIconPath resource.

XENVIRONMENT
specifies the path for configuration files.

XFILESEARCHPATH
specifies the path for configuration files.

XUSERFILESEARCHPATH
specifies the path for configuration files.

2.14 xhost - Controlling access to the X server

You can use the xhost client to specify the hosts from which clients can access the X server or to display a list of the machines that have access to the X server. To do this, xhost refers to the X server's internal access control list containing all hosts that can set up a connection to the X server. You can use xhost to add to and delete from this list. For more information on the X server's internal access control list, look up "xhost" or "access control" in the documentation of the X server you are using for your display. To determine the X server whose internal access control list xhost is to access, xhost evaluates the DISPLAY environment variable.
Note: Controlling access using the X server's internal access control list provides only very limited security!
There is also a user-specific authorization mechanism which provides greater security than host-specific access control: it ensures that in a user's session only clients that have an authorization key for a particular X server can set up a connection to that X server. This session-specific authorization key is generated by the xdm display manager at the start of the session and stored in a user-specific authorization file, and clients retrieve the key from this file in order to pass it to the server. The xauth client allows entries in this file to be passed to other hosts, thus making the key available to clients running on other hosts which want to set up a connection to the X server. User-specific access authorization of different hosts to a server applies irrespective of the host-specific access control lists created by the xhost client.
Note that the host-specific access authorization granted by the xhost client takes precedence over the user-specific access authorization provided by the authorization mechanism. If a host is granted access to a server by the X server's internal host list, all clients started on that host are authorized to access that server, even if they cannot supply the valid session key. Hosts on which user-specific authorization is to apply must not appear in the X server's host-specific access list.

Format 1:
xhost
Format 2:
xhost [+][*host_name*...]
Format 3:
xhost -[*host_name*...]

Format 1: Output the X server's internal access control list

xhost xhost outputs the contents of the X server's internal list on standard output. This list indicates which hosts can set up a connection to the X server and whether or not access control is currently enabled.

Format 2: Grant access to the X server

xhost [+][*hostname*...]

+ disables access control; all hosts, even those not entered in the X server's internal control list, may set up connections to the X server (enable: xhost -).

Default:

only the hosts entered in the X server's internal control list have access to the X server

[+]*hostname*

hostname is the name of the host to be entered in the X server's internal list. Only TCP/IP connections are supported.

Format 3: Revoke access rights

xhost -[*hostname*...]

\- enables access control: only hosts listed in the X server's internal list may set up connections to the X server (disable: xhost +). Only TCP/IP connections are supported.

-*hostnam*

hostname is the name of the host whose entry is to be deleted from the X server's internal list. Existing connections between the host and the X server are not broken, but no new connections can be set up.

Warning:

You can also use -*hostname* to delete the entry for the host on which the X server is running. Subsequently no connection setup between host and X server is possible in the network, not even with an xhost call. In order to restore the connection, you will have to terminate the X server. By restarting the X server, you can initialize the X server's internal list.

2.14.1 Files

$XAUTHORITY

user-specific authorization file

The client uses this file as a user-specific authorization file. If XAUTHORITY is not set, the client uses the .Xauthority file in the user's HOME directory.

2.14.2 Environment variables

Detailed descriptions of all the environment variables, arranged in alphabetical order, are given in the chapter "Environment variables ".

DISPLAY

The client reads the DISPLAY environment variable and finds out from it the name of the X server and the display and screen on it.

The DISPLAY environment variable is a string in the format:

hostname:*display_number*.*screen_number*

The client terminates abnormally if the DISPLAY environment variable is not set.

XAUTHORITY

denotes the user-specific authorization file.

If the environment variable is not set, the client searches for the $HOME/.Xauthority file.

2.15 xmbind - Controlling virtual keyboard and pointer maps

You can use the xmbind client to set the virtual key and pointer mapping for those clients that support the defaultVirtualBindings resource of the DefaultVirtualBindings class. xmbind stores the virtual key and pointer mapping in the X server. Since mwm also stores mapping at the same location in the X server, and all clients that support the defaultVirtualBindings resource read the mapping there, you need not call xmbind unless you do not want to use mwm or you want to modify the mapping during mwm runtime.
xmbind works with the same key and pointer mapping you can display and modify using the xmodmap client. Since the default mapping is X server-specific, only the system administrator should use xmodmap and xmbind to modify key and pointer mapping.

xmbind[*option*]...[*filename*]

filename
This is the name of the file containing all the virtual key and button bindings that xmbind is to map.
Binding syntax in file: *virtualKey*:[*modifier*]...<Key>*key*

virtualKey	**Default**	**virtualKey**	**Default**
osfActivate	*unbound*	osfLeft	Left
osfAddMode	Shift F8	osfMenu	F4
osfBackSpace	Backspace	osfMenuBar	F10
osfBeginLine	Home	osfPageDown	Next
osfClear	Clear	osfPageUp	Prior
osfCopy	*unbound*	osfPaste	*unbound*
osfCut	*unbound*	osfPrimaryPaste	*unbound*
osfDelete	Delete	osfQuickPaste	*unbound*
osfDown	Down	osfRight	Right
osfEndLine	End	osfSelect	Select
osfCancel	Escape	osfUndo	Undo
osfHelp	F1	osfUp	Up
osfInsert	Insert		

Table 27: Replacement options for xmbind for virtualKey

Modifier	**Replacement**	**Modifier**	**Replacement**

Ctrl	CTRL key	Mod1	Modifier key 1
Shift	Shift key	Mod2	Modifier key 2
Meta	Meta key	Mod3	Modifier key 3
Lock	LOCK key	Mod4	Modifier key 4
		Mod5	Modifier key 5

Table 28: Replacement options for xmbind for modifier

The xmodmap client will supply you with information on the mapping of the values for *modifier* to keys on the keyboard.

key can be replaced by any character in the ISO Latin 1 character set. The xmodmap client will supply you with information on which keys your display supports.

Default:

If *filename* is not specified, xmbind first looks for the file $HOME/.motifbind and implements the mappings defined there. If $HOME/.motifbind does not exist, xmbind looks for a file named xmbind.alias, first in $HOME and then, if it is not there, in $XMBINDDIR. xmbind extracts from this file the name of the file from which it is to reads the mappings.

-display [*hostname*]:*display_number*[.*screen_number*]

Identifies the X server, display and screen which the client is to contact.

hostname

symbolic name of the host or X terminal on which the X server which the client is to contact runs. *hostname* must be known in the transport protocol of the host on which the client resides.

display_number

number (unique to the X server) identifying the display for which the client is to manage I/O.

.*screen_number*

number (unique to the display) identifying the screen to which the client is to send output.

Default:

contents of the DISPLAY shell variable;
if DISPLAY is not set, the client terminates abnormally.

2.15.1 Files

$HOME/.motifbind

user-specific configuration file for the virtual button and key assignments.

2.15.2 Environment variables

Detailed descriptions of all the environment variables, arranged in alphabetical order, are given in the chapter "Environment variables ".

DISPLAY

The client reads the DISPLAY environment variable if the -display option is not specified in its command line. From this variable the client finds out the name of the X server and the display and screen on it.

The DISPLAY environment variable is a string in the format:

hostname:*display_number*.*screen_number*

The client terminates abnormally if the DISPLAY environment variable is not set and the -display option is not specified in the client command line.

XMBINDDIR

specifies the path for the xmbind.alias file, from which the client obtains the name of the file from which it reads the virtual key and button assignments. If the variable is not set, the client searches for the file in the /usr/lib/Xm/bindings directory.

2.16 xmessages - Message output and login entry in /etc/utmp

The xmessages client displays messages sent to the system console (/dev/console), kernel messages, and messages from the system administrator.
The client reads system messages from the special files /dev/csm and /dev/osm. Messages sent to the system console are buffered in these files. Once xmessages is running, its window immediately displays any message written to one of these files.
The client does not display any system messages issued in the interval between X server startup and startup of the xmessages client. Messages arriving in this interval have to be read directly from /dev/csm and /dev/osm.
Any messages that clients write to standard output and standard error once xmessages has been started up are collected in $HOME/.xerrors.
The client allocates a pseudo terminal and records this terminal together with the user's login name in the system log file /etc/utmp. In this way the user is known to commands such as *who* or *finger* and will also receive output from the *wall* command, because the client provides a window in which to display output directed to the pseudo terminal.
If $HOME is undefined at the time of xmessages startup, the client will automatically start up in standalone mode (see "-standalone" option). Then no entries will be made in /etc/utmp.
In the configuration as shipped, the client is automatically started in a SINIX/windows session immediately after successful authentication of the user.

xmessages[*option*]...

-beep See -nobeep option
specifies that an acoustic signal is to be emitted if messages arrive while xmessages is iconized.

Default:
There is an acoustic signal

-bg *color*
bg - background
See "background" resource
defines the background color of the window.

Default:
white

-console
See -noconsole option
specifies that all messages written to the system console are to be displayed.

Default:
Messages are displayed

-daemon
See "daemon" resource
starts the client in the background.
The client makes itself independent of the parent process by spawning a new process with *fork*. It also closes all file descriptors and is therefore no longer associated with

the controlling terminal.

Default:
The client starts in the foreground.

-device *filename*
Indicates a file to be monitored instead of /dev/console.

Default:

NULL (for machines running SINIX releases prior to V5.4x. The client always monitors /dev/osm and /dev/csm)
/dev/console
(for non-SINIX machines)

-display *[hostname]:display_number[.screen_number]*
Identifies the X server, display and screen which the client is to contact.

hostnam
symbolic name of the host of the X server which the client is to contact. *hostname* must be known in the transport protocol of the host on which the client resides.
display_number
number (unique to the X server) identifying the display for which the client is to manage I/O.
screen_number
number (unique to the display) identifying the screen to which the client is to send output.
This option can be used to send system messages from the local host to a display other than the local one via the network.

Default:
Contents of the DISPLAY shell variable; if DISPLAY is not set, the client terminates abnormally.

-exitOnFail
specifies whether xmessages is to exit when unable to redirect system console message output.

Default:
False
xmessages does not exit when unable to redirect system console message output.

-fg *color*
fg - foreground
Defines the foreground color of the window. This is the color in which text characters are displayed.

Default:
black

-file *filename*
Specifies the name of a file which is to be monitored if the xerrors resource is set. Any messages which clients write to standard output and standard error after xmessages startup are redirected to this file.

Default:
$HOME/.xerrors

-geometry *width*x*height*+*x_off*+*y_off*
Specifies the position of the client window.

width is the window width.
height
is the window height.
x_off position of window on x axis (horizontally) in pixel

+x_off
distance between left edge of window and screen
-x_off
distance between right edge of window and screen
y_off position of window on y axis (vertically) in pixels

+y_off
distance between top edge of window and screen
-y_off
distance between bottom edge of window and screen

Default:
no default

-iconic
When the client is started, the mwm window manager opens it as an icon and not as a normal text window.

Default:
The mwm window manager opens the normal text window of the client.

-monitor
See -nomonitor option
Xmessages periodically attempts to open the error log file (defined in the resource file) whenever the file's status has changed. It continues attempting to do so until no more error messages can be displayed.

Default:
No monitoring.

-nobeep

See -beep option
specifies that no acoustic signal is to be emitted if messages arrive while xmessages is iconized.

Default:
There is an acoustic signal

-noconsole
See -console option
specifies that messages written to the system console are not to be displayed.

Default:
Messages are displayed

-nocsm
See "csm" resource
suppresses output of system messages which are sent to the /dev/console special file and buffered in the /dev/csm cyclic buffer.

Default:
The client outputs system messages.

-nomonitor
See -monitor option
Xmessages periodically attempts to open the error log file (defined in the resource file) whenever the file's status has changed. It continues attempting to do so until no more error messages can be displayed.

Default:
No monitoring

-noosm
See "console", "osm" resources
suppresses output of kernel messages buffered in the /dev/osm cyclic buffer.

Default:
The client outputs kernel messages from /dev/osm.

-nopopup
Suppresses opening of the text window.
This option affects the behavior of the client in icon form. If a message arrives, a window is not opened; instead the icon image changes.

Default:
The client opens its window immediately after receiving a message.

-nonotify
Specifies that the icon image should not change when messages arrive.

Default:
The icon image changes. Operates only if popup is set to False.

-notify
See "popup" resource
specifies that the icon image should change when messages arrive.

Default:
True. Operates only if popup is set to False.

-noutmp
See -utmp option
specifies that no pseudo terminal should be allocated and that no entries should be made in the /etc/utmp system log file or the /etc/wtmp file.

Default:
Entries are made in /etc/utmp and /etc/wtmp.

-noxerrors
See "xerrors" resource
specifies that the log file defined by the *file* resource is not to be monitored.

Default:
False

-popup
See "notify" resource
specifies whether xmessages is automatically to be deiconified when messages arrive.

Default:

True (for console and system messages)
False (for messages from xerrors)

-standalone
If this option is set, the menu bar is not displayed and the help manager is not supported. Instead, a control panel and a simple help text window are displayed. No entries are made in /etc/utmp. If $HOME is undefined at the time of xmessages startup, the client automatically starts in standalone mode.

Default:
The menu bar is displayed and entries are made in /etc/utmp.

-utmp See -noutmp option
specifies that a pseudo terminal is to be allocated and entries made in the /etc/utmp system log file and the /etc/wtmp file. The user will then be known to commands such as "who" and "finger" and will also receive output from the "wall" command, because the client provides a window in which to display output directed to the pseudo terminal.

Default:
Entries are made in /etc/utmp and /etc/wtmp.

-verbose
Defines whether a title line is to appear in the first line of the xmessages window.

Default:
No title line is displayed.

-xerrors
See -noxerrors, -file options
specifies that the log file identified by the -file option is to be monitored.

Default:
The log file is monitored.

-xrm "*resourceDefinition*:*value*"
Assigns a value to a client resource.
-xrm must be specified separately for each resource to be set in the client call. The "*resourceDefinition*:*value*" assignment is compared with the entries in the configuration files.

resourceDefinitio
Identifies the resource to which a value is to be assigned. For *resourceDefinition* you can give the name of the resource, the name of the client and of any widgets in the appropriate resource syntax (see section "Value assignment syntax ").

value Value to be assigned to the resource.

2.16.1 Resources

You can configure the client using its resource database. To do this, you must assign the required values to the resources. You can use either configuration files or the -xrm option in the client call to access the resources.
The instance name of the client is xmessages and its class name XMessages.

2.16.1.1 Resource overview

This subsection starts with overviews showing which resources govern overall client behavior and which resources can be defined for specific messages. Then come brief descriptions of the resources listed alphabetically according to resource name.

2.16.1.2 Resources of the xmessages client

The following resources govern the behavior of the xmessages client:
Syntax:
XMessages.*resource*

Name	Value	Default
animatedClientIcon	name	newmes.48
background	color	white

clientIcon	name	message.48
columns	number	60
console	True/False	True
csm	True/False	True
daemon	True/False	True
exitOnFail	True/False	False
file	pathname	$HOME/.xerrors
fontList	font_name	cf. resource description
foreground	color	black
geometry	size	no default
iconSet	string	cf. resource description
osm	True/False	True
monitor	True|False	False
rows	number	14
systemIconPath	string	cf. resource description
update	sec	10
utmp	True/False	True
verbose	True/False	False
xerrors	True/False	True

Table 29: xmessages resources

The following resources govern the behavior of xmessages with regard to console messages (see "console" resource), system messages (see "utmp" resource), session error messages (see "xerrors" resource) or messages in the common display window:

Syntax:

XMessages*console**_resource_*
XMessages*system**_resource_*
XMessages*xerrors**_resource_*
XMessages*comon**_resource_*

Name	**Value**	**Default**

beep	True/False	True
clearCount	number	200
clearText	True/False	True
header	string	cf. resource description
notify	True/False	True
onceOnly	True/False	True
popup	True/False	cf. resource description
savedLines	number	20
stripNonprint	True/False	True
title	string	cf. resource description
volume	number	30

Table 30: xmessages resources

2.16.1.3 Resource list

You can configure the client using its resource database. To do this, you must assign the required values to the resources. You can use either configuration files or the -xrm option in the client call to access the resources.
The instance name of the client is xmessages and its class name XMessages.

animatedClientIcon
Class: Animated ClientIcon
See "systemIconPath" resource
specifies the icon displayed when new messages are received.

Default:
newmes.48

background
Class: Background
See "-bg color" option
Background color of the window

Default:
white

beep
Class: Beep
See "-beep" option, "volume" resource
specifies whether an acoustic signal is to be emitted if messages arrive while xmessages is iconized.

Default:
True

clearCount
Class: ClearCount
See "clearText", "savedLines" resources
specifies the rhythm in which the client automatically deletes the beginning of previous output.

Default:
200

clearText
Class: ClearText
See "clearCount", "savedLines" resources
specifies whether the client is to clear the text from the window once the number of lines defined by clearCount has been reached.

Default:
True

clientIcon
Class: ClientIcon
See "systemIconPath" resource
defines the icon used by the window manager

Default:
message.48

columns
Class: Columns
Number of columns in the text field

Default:
60

console
Class: Console
See "csm", "osm" resources
specifies whether all messages written to the system console are to be displayed

Default:
True

csm
Class: Csm
See "console" resource
specifies whether messages written to /dev/console are to be redirected to /dev/csm. Interpreted only if console is set to true and on system platforms which support /dev/csm.

Default:
True

daemon
Class: Daemon
Specifies whether the client is to run in the background.

Default:
False
The client runs in the foreground.

exitOnFail
Class: ExitOnFail
specifies whether xmessages is to exit when unable to redirect system console message output.

Default:
False

file
Class: File
See "xerrors" resource
specifies the name of a file which is to be monitored if the xerrors resource is set. Any messages which clients write to standard output and standard error after xmessages startup are redirected to this file.

Default:
$HOME/.xerrors

fontList
Class: FontList
The Client uses the font list for character output.

Default:
-adobe-helvetica-bold-r-normal-14-140-75-p-82-iso8859-1

foreground
Class: Foreground
See "-fg color" option
Foreground color of the window. This is the color in which text characters are displayed.

Default:
black

geometry
Class: Geometry
See -geometry option ("-geometry width x height + x_off + y_off")
Position and size of a window. The window manager can override the value of this resource.

Default:

No default

header
Class: Header

Defines a title line for messages appearing in the message window. This title line appears whenever the message source changes. %s is replaced by the name of the host on which xmessages was started.

Default:

"-----Console Messages for %s: -----\n" for console messages
"-----System Messages: -----\n" for system messages
"-----Session Error Messages: -----\n" for xerrors messages

iconSet
Class: IconSet

Specifies the set of icon images to use as client icons.

Default:

mono (for displays with one color plane)
color (for displays with more than one color plane)

notify
Class: Notify

See "popup" resource
specifies whether the icon image should change when messages arrive. Operates only if popup is set to False.

Default:

True

onceOnly
Class: OnceOnly

Specifies that an acoustic signal is emitted only the first time a new message is received.

Default:

False

osm
Class: Osm

See "console" resource
specifies whether messages written to /dev/osm are to be displayed.

Default:

True
The client displays kernel messages from /dev/osm.

popup
Class: Popup

See "notify" resource
specifies whether xmessages automatically opens its text window when messages arrive.

Default:

True (for console and system messages)
False (for xerrors messages)

rows
Class: Rows
Number of lines in the text field

Default:
14

savedLines
Class: SavedLines
See "clearCount", "clearText" resources
specifies the number of lines of previous output that the client does not delete.

Default:
20

stripNonprint
Class: StripNonprint
Specifies whether non-printing characters are to be stripped.

Default:
True

systemIconPath
Class: SystemIconPath
Specifies the list of directories to be searched for the client icon images.

Default:
/usr/lib/X11/siwin/icons/wm/%I/N\
/usr/lib/X11/siwin/icons/clients/%I/N\
/usr/lib/X11/siwin/icons/common/%I/N

title
Class: Title
Specifies the title line for the message display area.

Default:
"Console Messages" for console messages
"System Messages" for system messages
"Session Error Messages" for xerrors messages
"All Messages" for the common window

update

Class: Update
Specifies the interval in seconds at which the log file is to be polled.

Default:
10

utmp
Class: Utmp
Specifies whether a pseudo terminal is to be allocated and entries made in the /etc/utmp system log file and the /etc/wtmp file. The user will then be known to commands such as *who* and *finger* and will also receive output from the *wall* command, because the client provides a window in which to display output directed to the pseudo terminal.

Default:
True

verbose
Class: Verbose
Defines whether a title line is to appear in the first line of the xmessages window.

Default:
False
No title line is displayed.

volume
Class: Volume
See "beep" resource
defines the volume of the acoustic signal as a percentage of the maximum.

Default:
30

xerrors
Class: Xerrors
See "file" resource
specifies whether the log file identified by the file resource is to be monitored

Default:
True

2.16.2 Files

The files below are listed in alphabetical order. The resource configuration files are listed only in the form of the generalized file names $XENVIRONMENT, $XUSERFILESEARCHPATH and $XFILESEARCHPATH.

/dev/console
console special file

/dev/csm
contains system messages sent to the /dev/console special file
/dev/osm
contains kernel messages
/etc/utmp
system log file in which the client records the allocated pseudo terminal
/etc/wtmp
log file in which the client records the allocated pseudo terminal
$HOME/.xerrors
Any messages that clients write to standard output and standard error once xmessages has been started up are collected in the $HOME/.xerrors file.
$XENVIRONMENT
user-specific configuration file
$XFILESEARCHPATH
application-specific configuration file created by the client programmer
$XUSERFILESEARCHPATH
application-specific configuration file created by the system administrator or the user

2.16.3 Environment variables

Detailed descriptions of all the environment variables, arranged in alphabetical order, are given in the chapter "Environment variables ".

DISPLAY
The client reads the DISPLAY environment variable if the -display option is not specified in its command line. From this variable the client finds out the name of the X server and the display and screen on it.
The DISPLAY environment variable is a string in the format:
hostname:*display_number*.*screen_number*
The client terminates abnormally if the DISPLAY environment variable is not set and the -display option is not specified in the client command line.
XAUTHORITY
denotes the user-specific authorization file.
If the environment variable is not set, the client searches for the $HOME/.Xauthority file.
XENVIRONMENT
specifies the path for configuration files.
XFILESEARCHPATH
specifies the path for configuration files.
XUSERFILESEARCHPATH
specifies the path for configuration files.

2.17 xmodmap - Displaying/modifying keyboard and pointer maps

You can use the xmodmap client to display and edit the keyboard modifier map, the keymap table and the pointer map. Other clients evaluate these maps in order to convert event keycodes into the appropriate keysyms (refer to the OSF/Motif and X Window System documentation listed under " "Related publications"" for further information on events).
xmodmap is generally started when the X server is booted to ensure that the keyboard is configured to meet individual requirements. The xmodmap client therefore usually serves only to provide information on the current keyboard and pointer mapping.
Clients that support the defaultVirtualBindings resource of the DefaultVirtualBindings class evaluate additional keyboard and pointer mapping. You can use the xmbind client to modify this additional mapping.
Since the default is adapted to the X server-specific mapping, only the system administrator should use xmodmap and xmbind to modify the keyboard and pointer mapping.

xmodmap[*option*]...[*filename*]

- Causes xmodmap to read from the standard input instead of from a file.

filename
this is the name of the input file. this file contains all the xmodmap expressions to be executed.

Default:
Displays the current settings.

-display [*hostname*]:*display_number*[.*screen_number*]
Identifies the X server, display and screen which the client is to contact.

hostnam
symbolic name of the host or X terminal on which the X server which the client is to contact runs. *hostname* must be known in the transport protocol of the host on which the client resides.

display_number
number (unique to the X server) identifying the display for which the client is to manage I/O.

.*screen_number*
number (unique to the display) identifying the screen to which the client is to send output.

Default:
contents of the DISPLAY shell variable;
if DISPLAY is not set, the client terminates abnormally.

-e "*xmodmap_expression*"
xmodmap_expression is executed.
You may specify as many expressions as you wish in a xmodmap call. Use the -grammar option to display information on how to specify *xmodmap_expression* or refer to the section "Appearance and behavior ".

-grammar

xmodmap outputs a help text to standard error which describes the syntax of xmodmap expressions in shell scripts and in xmodmap calls (-e option).
See the section "Appearance and behavior " for more detailed information.

-help Writes a short description of the options to standard error.
This help text is also displayed if the client cannot interpret a command-line argument.

-n xmodmap does not change keyboard layout; instead it simulates the changes and displays what the current xmodmap call would do.

Default:
xmodmap changes the keyboard layout.

-pk pk - print keymap table
See -pm option
xmodmap outputs the current keymap table to standard output.

Default:
xmodmap outputs the current modifier map.

-pm pm - print modifier map
xmodmap outputs the current modifier map to standard output.

Default:
xmodmap outputs the current modifier map to standard output.

-pp pp - print pointer map
See -pm option
xmodmap outputs the current pointer map to standard output. The pointer map contains the button codes.

Default:
xmodmap outputs the current modifier map.

-quiet
xmodmap does not print logging information as it parses its input.

Default:
-quiet is the default setting.

-verbose
See -quiet option
xmodmap prints logging information to standard output as it parses its input.

Default:
xmodmap does not print logging information.

2.17.1 Appearance and behavior

xmodmap accepts input for updating keyboard or button layout from a file (see *filename* option) or when the client is called in its command line (see options - and -e). Valid xmodmap expressions are described in the following.

2.17.1.1 xmodmap expressions

xmodmap reads a list of expressions and evaluates them before executing the first expression. This makes it possible to refer to keysyms that are redefined in the list without having to worry about name conflicts.
You can use comment lines in any files you create as input to xmodmap. xmodmap interprets all lines starting with an exclamation mark as comment lines (see section "Examples ").
All names are case sensitive with the exception of modifier names.
If you want to change the mapping of a modifier key, you must first remove it from the current modifier map.
In the following description

modifiername
: is the name of the modifier key (see "add")

keysymname
: is the keysym name

2.17.1.2 Valid expressions

add *modifiername* = *keysymname* [...]
: xmodmap adds the specified keysyms to the modifier map indicated by *modifiername*. xmodmap reads all input expressions before evaluating the keysyms (*keysymname*). This makes it easier to write expressions for swapping keys (see section " Example 5: Swapping [CTRL] and [LOCK] ").
Modifier names: Shift, Lock, Control, Mod1, Mod2, Mod3, Mod4, Mod5

clear *modifiername*
: xmodmap removes all entries in the modifier map for the given *modifiername*. Modifier names are case insensitive.
Modifier names: Shift, Lock, Control, Mod1, Mod2, Mod3, Mod4, Mod5

keycode *number* = *keysymname* [...]
: A list of keysyms (*keysymname*) is assigned to the indicated keycode. The keycode can be specified as a decimal, hexadecimal or octal number (see section "Example 6: Assigning a keysym to several keycodes ").

Default:
: each keycode is assigned only one keysym

Tip:

The server generates a MappingNotify event for each client every time a keycode expression is evaluated. Therefore, you should group all your keycode modifications together and execute them in the background. Clients which process keyboard input and ignore MappingNotify events are not informed that the keyboard mappings have been modified (refer to the documentation for OSF/Motif and the X Window System listed under " "Related publications"" for information on events).

keysym *keysymname1* = *keysymname2* [...]
: xmodmap searches for the keycodes to which *keysymname1* is assigned and replaces the input expression with the corresponding keycode expression for every keycode it finds. Then *keysymname2* is assigned to each keycode found as described above (see section " Example 2: Defining the Meta key "). If you have assigned a keysym to more than one key, xmodmap will not be able to execute this assignment correctly.

pointer = default
: xmodmap sets the pointer map back to its default settings, i.e. button 1 generates a code for button 1, button 2 generates a code for button 2, etc.

pointer = *number* [...]
: xmodmap assigns the buttons, one after the other, to one of the specified numbers. The first number is assigned to button 1, the second to button 2, the third to button 3 (see section " Example 1: Redefining the buttons for left-handed users ").

remove *modifiername* = *keysymname* [...]
: xmodmap deletes the specified keysym from the modifier map. Unlike the "add" function (see above), xmodmap evaluates the expressions as they are being read in. This allows you to remove modifier assignments without having to worry about whether or not new assignments have been made (see section " Example 5: Swapping [CTRL] and [LOCK] ").

2.17.2 Environment variables

Detailed descriptions of all the environment variables, arranged in alphabetical order, are given in the chapter "Environment variables ".

DISPLAY
: The client reads the DISPLAY environment variable if the -display option is not specified in its command line. From this variable the client finds out the name of the X server and the display and screen on it.
The DISPLAY environment variable is a string in the format:
hostname:*display_number*.*screen_number*
The client terminates abnormally if the DISPLAY environment variable is not set and the -display option is not specified in the client command line.

HOMEidentifies the user's HOME directory.
: The client searches this directory for files.

LANGis evaluated in the context of NLS language support.

2.17.3 Examples

2.17.3.1 Example 1: Redefining the buttons for left-handed users

The general functional layout of the buttons on a 3-button mouse is as follows: button 1 is used to select elements, button 2 is responsible for dragging movable elements, and button 3 opens pop-up menus.
Most pointing devices are designed so that button 1 is pressed using the index finger of the right hand. People who are left-handed should modify the button codes so that button 1 is pressed using the index finger of the left hand.
Redefining buttons on a 3-button mouse:

```
xmodmap -e "pointer = 3 2 1"
```

2.17.3.2 Example 2: Defining the Meta key

Many editor applications support the use of a Meta key. This key has a similar function to the Control key when set as a modifier key. Some servers, however, do not have a meta keysym in the default keymap table, so one needs to be added by hand.
Assign Meta to the Multi-language key (= Label Compose):

```
xmodmap -e "keysym Multi_key = Multi_key Meta_L"
```

This call takes advantage of the fact that applications that need a Meta key simply need the keycode and do not require the keysym to be in the first column of the keymap table. This also means that applications that work with the Multi-language key (including the default modifier map) will not notice any change.

2.17.3.3 Example 3: Set "rubout" key to generate an alternate keysym

A simple but convenient use of xmodmap is to set the "rubout" key to generate an alternate keysym. This frequently involves exchanging Backspace with Delete.

```
xmodmap -e keysym BackSpace = Delete
```

2.17.3.4 Example 4: Generating the characters < and >

Some keyboards do not automatically generate the characters < and > when the dot and comma keys are pressed together with the Shift modifier. Using the following file as the input file for the xmodmap call, the mappings for the keys are reset:

```
!
! Make: [SHIFT] [.] be [>]
!
keysym period = period greater
```

2.17.3.5 Example 5: Swapping [CTRL] and [LOCK]

Many keyboards have the **[CTRL]** and **[LOCK]** keys located in different places. These two keys can be swapped using xmodmap expressions:

```
!
! Swap [LOCK] (= Caps_Lock) and [CTRL] (= Control_L)
!
remove Lock = Caps_Lock
remove Control = Control_L
keysym Control_L = Caps_Lock
keysym Caps_Lock = Control_L
add Lock = Caps_Lock
add Control = Control_L
```

2.17.3.6 Example 6: Assigning a keysym to several keycodes

The keycode command allows you to assign a list of keycodes to the same keysym. Although such applications are not portable, it makes it possible to write scripts that can reset the keyboard to a known state.
The following script sets the backspace key to generate Delete (as described above), deletes all existing Caps Lock mappings, makes the CapsLock key into a CTRL key, makes F5 the Escape key and makes Break/Reset the Shift Lock key.

```
!
! Mapping keycodes to keys:
!
```

```
!   101 Backspace
!   55 Caps
!   14 Ctrl
!   15 Break/Reset
!   86 Stop
!   89 F5
!
keycode 101 = Delete
keycode 55 = Control_R
clear Lock
add Control = Control_R
keycode 89 = Escape
keycode 15 = Caps_Lock
add Lock = Caps_Lock
```

2.18 xpct - ANSI-mode TC20-V100 terminal emulator

This terminal emulator allows you to run programs that can only use alphanumeric devices of type SCO-ANSI console.
xpct[*option*]...
In cases where an option has both -*option* and +*option*, +*option* is usually used to reset the attribute specified by -*option* to its default value.

-ah ah - always highlight
See +ah option
See "alwaysHighlight" resource
This option causes the cursor to always be highlighted, even if the input focus is assigned to a different window.

Default:
Only a hollow cursor is displayed when the window loses the input focus.

+ah causes the cursor to be highlighted only if the input focus is assigned to the emulator window.

-b *number*
See "internalBorder" resource
number specifies (in pixels) the width of the internal border.
The internal border is the area between the text margin and the window border.

Default:
2 (pixels)

-bd *color*
bd - border color
See "borderColor" resource
color is the color of the window border.

Default:
black

-bg *color*
bg - background
See "background" resource
color is the color of the window background.

Default:
white

-blink
blink - blinking cursor
Specifies that the text cursor blinks.

Default:
the text cursor does not blink.

+blink

Specifies that the text cursor does not blink.

-bw *number*

bw - border width

See "borderWidth" resource

number is the width of the window border in pixels.

The window manager may ignore or adjust this value.

Default:

2 (pixels)

-cb cb - cut begin of line inhibit

See "cutToBeginningOfLine" resource

Affects line-by-line selection of text.

Text is copied to the buffer from the pointer position, not from the beginning of the line where the pointer is located.

Default:

The emulator copies text to the buffer from the beginning of the line.

+cb The terminal emulator copies text to the buffer from the beginning of the line.

-charsetmode *mode*

charsetmode - character set mode

See options -ibmcc, -ibmcharset, -isocc, -isocharset

See "charSetMode" resource

Specifies the character set mode in which the terminal emulator is to work. The terminal emulator can use either the ISO or IBM character set mode. By using the -isocharset and -ibmcharset options you can set different character sets in each character set mode.

You can switch character set mode during runtime by using an escape sequence.

The character set mode you set using the -charsetmode option determines whether the terminal emulator uses the ISO or the IBM character classes for word-by-word selection. When you switch character set mode during runtime, the character class is not affected. This can lead to strange results when text is selected word by word.

When the IBM character set mode is set, the translation table defined with the charsetMapping resource is used for keyboard input.

Default:

iso

The terminal emulator uses the ISO character set mode and the ISO character classes.

-cn cn - cut newline inhibit

See "cutNewline" resource

Affects line-by-line selection of text.

The terminating newline character of the last line selected is not copied to the cut buffer.

Default:

The last line is also terminated by a newline character.

+cn When selecting text line by line, each line, including the last, is terminated by a newline character.

-cr *color*

See "cursorColor" resource

color is the color of the text cursor when the window has the input focus.

Default:

black

-display [*hostname*]:*display_number*[.*screen_number*]

Identifies the X server, display and screen which the client is to contact.

hostname

symbolic name of the host or X terminal on which the X server which the client is to contact runs. *hostname* must be known in the transport protocol of the host on which the client resides.

display_number

number (unique to the X server) identifying the display for which the client is to manage I/O.

.*screen_number*

number (unique to the display) identifying the screen to which the client is to send output.

Default:

contents of the DISPLAY shell variable;
if DISPLAY is not set, the client terminates abnormally.

-e *command*[*argument*]...

command is called in the emulator window instead of the shell. If you do not specify the -title option in the emulator call, the emulator uses the command name as the window and the icon name.
The command is called with the specified arguments.
This option may only be specified as the last argument of the terminal emulator command line.

Default:

a shell is started in the terminal emulator window.

-fg *color*

fg - foreground

See "foreground" resource

color is the color of the window foreground.
This is the color in which text characters are displayed.

Default:

black

-geometry *width*x*height*+*x_off*+*y_off*

See "geometry" resource

Defines the size and position of the client window.
Specifications for the window size (*width*x*height*) are not evaluated: emulator windows always have a fixed size of 80 columns by 25 lines. *x_off* and *y_off* determine the window position.

width the window width. (ignored)
height
the window height. (ignored)
x_off Displacement of the window along the x axis (horizontally) in pixels

+*x_off*
distance between left edge of window and screen
-*x_off*
distance between right edge of window and screen
y_off Displacement of the window along the y axis (vertically) in pixels

+*y_off*
distance between top edge of window and screen
-*y_off*
distance between bottom edge of window and screen
Default:
80x25+0+0

-help Writes a short description of the options to standard error.
This help text is also displayed if the client cannot interpret a command-line argument.

-ibmcc *character_code*:*index*[,*character_code*:*index*]...
ibmcc - IBM character class range
See -ibmcharset, -charsetmode options
See "ibmCharClass", " ibmCharSet", "charSetMode" resources
Specifies which characters from the IBM character set are grouped together in a character class. At present IBM Code Page 437, 850, 852, 857 and 869 character sets are supported.
Character classes are important when text is selected word by word. When you double-click the mouse button BSelect, successive characters belonging to the same character class are selected for the buffer.
You create a character class by assigning a common numerical value to all its characters. You can use the decimal character code of any element of the character class as this common numerical value.

character_code
Decimal character code of the character. You can use either a single character (*character_code1*) or a sequence of characters (*character_code1-character_code2*) for *character_code*. Non-successive value assignments are separated by commas.
index Decimal value to which the characters of a class are assigned. You can use the decimal ASCII value of any character in the class for this purpose.

Default:

Value of the ibmCharClass resource

-ibmcharset *character_set*

ibmcharset - IBM character set

See -charsetmode option

See resources " ibmCharSet", "charSetMode"

Specifies the character set with which the terminal emulator works when the character set mode is set to IBM. You use the -charsetmode option to specify the character set mode, and the ibmFontList resource to specify the font for displaying the character set.

At present IBM Code Page 437, 850, 852, 857 and 869 character sets are supported.

Default:

ibm437

-iconic

Indicates that the mwm window manager is to start the client as an icon, not as a normal window.

Default:

The mwm window manager opens the normal window of the client.

-isocc *character*:*class*[,*character*:*class*]...

isocc - ISO character class range

See -isocharset, -charsetmode options

See resources "isoCharSet", "charSetMode"

Specifies which characters from the ISO character set are grouped together in a character class. Character classes are important when text is selected word by word. When you double-click the mouse button BSelect, successive characters belonging to the same character class are selected for the buffer.

You create a character class by assigning a common numerical value to all its characters. You can use the decimal character code of any element of the character class as this common numerical value.

character_code

Decimal character code of the character. You can use either a single character (*character_code1*) or a sequence of characters (*character_code1-character_code2*) for *character_code*. Non-successive value assignments are separated by commas.

index Decimal value to which the characters of a class are assigned. You can use the decimal ASCII value of any character in the class for this purpose.

Default:

value of the isoCharClass resource

-isocharset *character_set*

isocharset - ISO character set

See -charsetmode option

See resources "isoCharSet", "charSetMode"

Specifies the character set with which the terminal emulator works when the character

set mode is set to ISO. You use the -charsetmode option to specify the character set mode, and the isoFontList resource to specify the font for displaying the character set.

Default:

iso8859-1

-j j - jump scrolling
See "jumpScroll" resource
See -s option
Text within the window is scrolled in blocks, not line by line.
Jump scrolling means that several lines are moved simultaneously out of the display at the upper edge of the screen. This speeds up scrolling through large amounts of data significantly, since it reduces the load on both the terminal emulator and the X server.

Default:

text in the emulator window is scrolled line by line.

+j The text in the emulator window is scrolled line by line.

-l l - logging
See "logging" resource
See -lf option
Causes the emulator to keep a log.
The terminal emulator directs all output, including error output, both to the log file and to the screen.
You can use the -lf option to set up a log file of your own; if you do not do so, the terminal emulator uses the log file XPCTLog.*string*. The process ID of the xpct process is used for this. If you do not specify an absolute pathname, the log file is created in the directory where the emulator was started, or in the $HOME directory if you use the terminal emulator to start a login shell.

Default:

The terminal emulator does not keep a log.

+l The terminal emulator does not keep a log.

-lf *char_string*
lf - logfile
See "logFile" resource
See -l option
For *char_string* you can specify a filename *file* or set up a pipe to the command (in this case, the first character of *char_string* is the pipe symbol: |*command*).

-lf *file*

The emulator uses *file* as the log file instead of the default log file. *file* is created in the directory where the emulator was started.

-lf- |*command*

A pipe to *command* is created and *command* is executed by the shell.

Default:

XPCTLog.*string*
The process ID of the xpct process is used as the file name extension to make the file name unique.

-ls ls - login shell
See loginShell resource
The terminal emulator starts a login shell and executes the .profile or .login file. The initial working directory of the login shell is the directory in which the terminal emulator is started.

Default:
the terminal emulator starts a subshell which is not a login shell.

+ls The terminal emulator starts a subshell which is not a login shell.

-mb mb - margin bell
See "marginBell" resource
See -nb option
Enables the margin bell for the right text margin.
The margin bell sounds when the cursor reaches the text column which was specified using the -nb option or the corresponding resource.

Default:
the terminal emulator does not sound a margin bell.

+mb The terminal emulator does not sound a margin bell.

-mc *milliseconds*
mc - multi-click time
See "multiClickTime" resource
Defines the maximum time in milliseconds that may elapse between 2 mouse clicks for the two clicks to be interpreted as a multiple click.

Default:
250 (milliseconds)

-mf [*path*]*filename*
mf - mapchan file
See "mapchanFile" resource
Defines the Mapchan file for the key mapping for keyboard inputs.

path is the path on which the Mapchan file is searched at the start. If the path is not specified, the following directories are searched for the Mapchan file: 1. $HOME/mapchan 2. /usr/lib/mapchan

filename
is the name of the Mapchan file. If a Mapchan file has been found and activated at the startup of the client, the entry "key mapping" becomes sensitive in the "keyboard" pull-down menu and the button "key mapping" is actuated. If the Mapchan file has not been specified or found, the entry "key mapping" does not become sensitive.

Default:

ZERO

-ms *color*

ms - mouse pointer color
See "pointerColor" resource
See -fg option
color determines the color of the pointer.

Default:

black

-name *resource_link*

resource_link is the name of the application under which the client's window resources are to be found.
The client can thus be called under different names and utilize the resources for its client instance *resource_link*, yet still utilize the same resources for its client class *clientclass*.
resource_link should not contain dots (.) or asterisks (*), as these symbols have special meanings. Please refer to the section "Value assignment syntax " for more information.

Default:

xpct

-nb *number*

See -mb option
See resources "nMarginBell", "marginBell"
Specifies the column at which the margin bell is sounded.
number is the offset in columns from the right text margin. The margin bell is only sounded when enabled via the -mb option, the "Modes" menu or the marginBell resource.

Default:

10

-rv rv - reverse video

See "reverseVideo" resource
Inverts the foreground and background colors.

Default:

Foreground and background colors are not inverted.

-rw rw - reverse wraparound

See "reverseWrap" resource
See Mode menu (table "The "Mode" menu of xpct")
Enables the reverse wraparound function.
Reverse wraparound allows the cursor to back up from the left text margin (column 1) to the last column (right text margin) of the previous line. This is especially useful when editing long command lines.

Default:

the emulator does not support reverse wraparound.

+rw The text cursor cannot be backed up beyond the left-hand text margin.

-S*filedescriptor filename*

The terminal emulator is started in slave mode.
The terminal emulator uses a pseudo terminal for input and output. This pseudo terminal must have been allocated by the master which starts the emulator in slave mode, since the emulator does not perform any allocations in slave mode.
In slave mode, it is not possible to send signals to the process group of the terminal emulator via the menu.

filedescripto

is the decimal number of the file descriptor which the master process used to open the master side of the pseudo terminal.

filenam

is the absolute pathname of the slave side of the pseudo terminal used by the terminal emulator for input and output.

No default value.
Allows text to be scrolled asynchronously.

-s Positioning by scrolling large amounts of text is speeded up; during scrolling, the text visible in the window does not correspond to the scroll bar position.
-s is very useful when running the terminal emulator in extensive networks with long latency times.
Precondition: there must be a scroll bar.

Default:

synchronous scrolling

+s The emulator scrolls synchronously.
During scrolling, the text visible in the window corresponds to the scroll bar position.

-sb sb - scroll bar
See "saveLines" resource
See -sl option
The terminal emulator displays a vertical scroll bar.
The emulator saves the lines which are scrolled off the top of the window. The scroll bar can be used to restore scrolled lines to the window. The highlighted part of the scroll bar also shows what part of the whole text is being displayed within the window.
You can use the -sl option or the saveLines resource to specify how many of the lines which have been scrolled off the window are saved.

Default:

no scroll bar

+sb No scroll bar is displayed.

-si si - scroll to bottom on output inhibit
See -sb option
See "scrollInput" resource
When output is sent to the emulator window, this option inhibits positioning to the end of the scroll area unless it was already visible.
Precondition: There must be a scroll bar.

Default:

The emulator window is positioned to the end of the scroll area for output.

+si When output is sent to the emulator window, the window is positioned to the end of the scroll area.

-sk sk - scroll to bottom on key
See -sb option
See "scrollKey" resource
On input from the keyboard, the emulator window is positioned to the end of the scroll area (= end of text).
Precondition: There must be a scroll bar.

Default:

Positioning to the end of text is inhibited for keyboard input.

+sk On input from the keyboard, the window is not repositioned to the end of the scroll area (= end of text) if text has previously been scrolled down out of the window.

-sl *number*
sl - save lines
See "saveLines" resource
number specifies how many lines the emulator is to save.

Default:

50 (lines)

-title *char_string*
See -e option
See "title" resource
char_string is the name which the mwm window manager displays in the title bar of the emulator window.

Default:

xpct or the command name from the -e option.

-tm "*assignment*[*assignment*...]"
tm - tty modes
See ttyModes resource
This resource sets terminal driver parameters.
assignment is a value assignment for terminal attributes.

assignment

Each *assignment* consists of a keyword and the value for the terminal attribute in the form of *keyword value*; *value* is always exactly one character long.
Valid keywords: intr, quit, erase, kill, eof, eol
Control characters can be used with a leading ^ character; use ^? to specify the delete function.

Defaults:

intr Delete

quit ^\
erase ^h
kill ^x
eof ^d
eol ^@

-tn *name*
See "termName" resource
name is the identifier of the terminal type which the terminal emulator assigns to the TERM variable.
The terminal type *name* must be defined in the terminfo database.

Default:
ansi-tc20

-ut See "utmpInhibit" resource
The terminal emulator does not create a login entry in the system log file /etc/utmp.

Default:
The emulator creates a login entry.

+ut The emulator creates a login entry in the /etc/utmp file, provided that "root" is the owner of the emulator program file and that the "set-user-id" bit is set.

-vb vb - visual bell
See "visualBell" resource
On receiving CTRL-G the terminal emulator generates a visual warning instead of an acoustic one.
Window contents and background are displayed briefly in reverse video.

Default:
CTRL-G causes the terminal emulator to sound an acoustic warning.

+vb On receiving CTRL-G the terminal emulator sounds an acoustic warning.

-xrm "*resourceDefinition*:*value*"
Assigns a value to a client resource.
-xrm must be specified separately for each resource to be set in the client call. The "*resourceDefinition*:*value*" assignment is compared with the entries in the configuration files. For many resources there are corresponding command-line options. To address resources for which there are no specific options available in the client call you have to use the -xrm option.

resourceDefinition
Identifies the resource to which a value is to be assigned. For *resourceDefinition* you can give the name of the resource, the name of the client and of any widgets in the appropriate resource syntax (see section "Value assignment syntax ").

value Value to be assigned to the resource.

2.18.1 Resources

You can configure the client using its resource database. To do this, you must assign the required values to the resources. You can use either configuration files or the -xrm option in the client call to access resources. You can access the description of the resources of xpct via "Configuration" in the "Help" menu.
The instance name of the client is xpct and its class name XPct. Please note that you can use the -name *instancename* option to change the instance name of the client when you call it.

2.18.1.1 Resource overview

This subsection begins with a functional overview of the resources of xpct. This is followed by brief descriptions of the resources, arranged in alphabetical order.
The resources described below are almost all resources of the vtterm widget (class: TermWidget). The resources termName, ttyModes and utmpInhibit do not belong to this widget. These are resources of the xpct client (class: XPct).
The fontList and menuAccelerator resources are resources of the client's vtmenubar widget.
Resources that affect the general behavior of the client

Resource	Value	Default	Option
allowSendEvents	True|False	False	
defaultVirtualBindings	string	-	
logFile	string	-	-lf
logging	True|False	False	-l
logInhibit	True|False	False	
loginShell	True|False	False	-ls
signalInhibit	True|False	False	
termName	string	-	-tn
translations	string	-	
ttyModes	pixmap	-	-tm
utmpInhibit	True|False	False	-ut
xnlLanguage	string	-	

Table 32: xpct resources: General behavior

Resources for specifying and assigning character sets

Resource	Value	Default	Option
charSetMapping	string	-	
charSetMode	string	iso	-charsetmode
ibmCharClass	string	-	-ibmcc

ibmCharSet	string	ibm437	-ibmcharset
ibmFontList	string	-	
isoCharClass	string	-	-isocc
isoCharSet	string	-	-isocharset
isoFontList	string	-	

Table 33: xpct resources: for character sets

Resources that affect the appearance of the client window

Resource	**Value**	**Default**	**Option**
background	pixel	-	-bg
backgroundPixmap	pixmap	-	
alwaysHighlight	True|False	False	
blinkColor	pixel	-	
blinkCursor	True|False	True	-blink
blockCursor	True|False	True	
borderColor	pixel	-	
borderWidth	pixel	-bw	
bottomShadowColor	pixel	-	
bottomShadowPixmap	pixmap	-	
cursorColor	pixel	-	-cr
cursorBlinkTime	string	-	
fontList	string	-	
foreground	pixel	-	-fg
geometry	string	-	-geometry
iconGeometry	string	-	
internalBorder	string	-	-b
monochrome	True|False	-	
noMenuBar	True|False	False	
pointerColor	string	-	-ms
pointerShape	pixmap	-	
reverseVideo	True|False	False	-rv
scrollBar	True|False	False	-sb
title	string	-	-title
topShadowColor	string	-	

topShadowPixmap	pixmap	-	
useStandardColormap	True|False	True	
visualBell	True|False	False	-vB

Table 34: xpct resources: Appearance of client window

Resources that affect the color settings for color terminals:

Resource	Value	Default	Option
blueColor	string	#0000cd	
brownColor	string	#cdcd00	
cyanColor	string	#00cdcd	
greenColor	string	#00cd00	
grayColor	string	gray	
lightBlueColor	string	#0000ff	
lightCyanColor	string	#00ffff	
lightGreenColor	string	#00ff00	
lightMagentaColor	string	#ff00ff	
lightRedColor	string	#ff0000	
lightWhiteColor	string	white	
magentaColor	string	#cd00cd	
redColor	string	#cd0000	
yellowColor	string	#ffff00	

Table 35: xpct resources: for color monitors

Resources that affect the behavior of the client on text input:

Resource	Value	Default	Option
autoWrap	True|False	True	
cutNewLine	True|False	True	-cn
cutToBeginningOf	True|False	True	-cb

Line			
jumpScroll	True|False	False	-j
mapchanFile	sting	-	-mf
marginBell	True|False	False	-mb
menuAccelerator	string	-	
multiClickTime	string	-	-mc
multiScroll	True|False	False	-s
nMarginBell	string	-	-nb
reverseWrap	True|False	False	-rw
saveLines	string	-	-sl
scrollInput	True|False	True	-si
scrollKey	True|False	False	-sk

Table 36: xpct resources: for text input

2.18.1.2 Resource list

The resources available to this client are listed in alphabetical order below. You can also use them in the -xrm option.

allowSendEvents
Class: AllowSendEvents

Defines whether or not the terminal emulator evaluates key or button events sent to it by other clients.

Default:

False
Events sent by other clients are not evaluated.

alwaysHighlight
Class: AlwaysHighlight

See "-ah" option
Defines whether or not the text cursor is always highlighted, even if the input focus is assigned to a different window.

Default:

False
The cursor is only highlighted in the active window.

autoWrap
Class: AutoWrap

Defines whether the text cursor is positioned automatically in the first column of the next line when a character is output in the last column.
In addition, the value of the resource determines whether the display is scrolled by a line when the text cursor is in the last line of the work area.

Default:
True
The text cursor is positioned automatically in the first column of the next line, and the window is scrolled by one line.

background
Class: Background
See option "-bg color"
Determines the color of the window background.

Default:
white

backgroundPixmap
Class: BackgroundPixmap
Specifies the pixmap to be used for the background of a window

Default:
depends on the visual class of the screen

blinkColor
Class: Foreground
Determines the color of "blinking" text.

Default:
red

blinkCursor
Class: Blinking
See "cursorBlinkTime" resource
Specifies whether or not the text cursor blinks.

Default:
True
The cursor blinks.

blockCursor
Class: BlockCursor
Specifies whether the text cursor appears as a rectangular block the size of a character or as an underscore the width of a character.

Default:
True
The text cursor appears as a block.

blueColor
Class: BlueColor
Color setting for color terminals

Default:
#0000cd

borderColor
Class: BorderColor
Specifies the color of the window border in the client area.
borderWidth
Class: BorderWidth
See option "-bw number"
Specifies the width of the client window border in pixels.

Default:
2 pixels
bottomShadowColor
Class: Foreground
Specifies the color for the bottom and right-hand 3-D window shadows.

Default:
depends on the visual class of the screen
bottomShadowPixmap
Class: BottomShadowPixmap
Specifies the pixmap to be used for the bottom shadow of windows.

Default:
depends on the visual class of the screen
brownColor
Class: BrownColor
Color setting for color terminals

Default:
#cdcd00
charSetMapping
Class: CharSetMapping
See "charSetMode", " ibmCharSet", "isoCharSet" resources
Defines a translation table for converting characters belonging to ISO-8859 character sets to characters belonging to the IBM Code Pages:

ISO 8859	IBM Code Page 437
ISO 8859-1	IBM Code Page 850
ISO 8859-2	IBM Code Page 852
ISO 8859-9	IBM Code Page 857
ISO 8859-5	IBM Code Page 866
ISO 8859-7	IBM Code Page 869

Table 37: Mapping of ISO 8859 character sets to IBM Code Pages

Keyboard input must be converted when working with the IBM character set, since the KeySyms supplied by the xmodmap table are always coded in the form of an ISO-8859-* character set. You can use the charSetMode resource to specify whether

the terminal emulator is to use an ISO or and IBM character set.
There is a predefined translation table for all supported ISO character sets. This normally does not have to be changed.
The resource can contain several translation tables.

Default:

The default translation table below contains values for converting all supported ISO-8859 character sets to IBM Code Pages. Characters in ISO character sets within the range from 0xa0 to 0xff are mapped to characters from 0x0 to 0xff belonging to the IBM character sets. Within the range 0x0 to 0x9f, the characters are either identical or no characters are defined.

	ISO8859-...->IBMCodePage437								ISO8859-...->IBMCodePage437						
ISO-Code	1	2	3	4	5	7	9	ISO-Code	1	2	3	4	5	7	9
a0	20	20	20	20	20	20	20	d0	44	44	5f	44	61	5f	47
a1	ad	4a	48	41	45	60	ad	d1	a5	4e	a5	4e	5f	50	a5
a2	9b	5f	5f	6b	5f	27	9b	d2	4f	4e	4f	4f	42	5f	4f
a3	9c	4c	9c	52	5f	9c	9c	d3	4f	4f	4f	4b	5f	e4	4f
a4	c2	c2	c2	c2	5f	5f	c2	d4	4f	4f	4f	4f	5f	54	4f
a5	9d	4c	5f	49	53	5f	9d	d5	4f	99	47	4f	65	5f	4f
a6	7c	53	48	4c	49	7c	7c	d6	99	99	99	99	5f	e8	99
a7	b2	b2	b2	b2	49	b2	b2	d7	2a	2a	2a	2a	5f	58	2a
a8	22	22	22	22	4a	22	22	d8	ed	52	47	ed	5f	5f	ed
a9	63	53	49	53	5f	63	63	d9	55	55	55	55	5f	ea	55
aa	a6	53	53	45	5f	5f	a6	da	55	55	55	55	6b	49	55
ab	ae	54	47	47	5f	ae	ae	db	55	9a	55	55	5f	5f	55
ac	aa	5a	4a	54	4b	aa	aa	dc	9a	9a	9a	9a	4d	e0	9a
ad	2d	2d	2d	2d	2d	2d	2d	dd	59	59	55	55	48	ee	49
ae	72	5a	5f	5a	59	5f	72	de	50	54	53	55	6f	5f	53
af	2d	5a	5a	2d	5f	2d	1d	df	e1	e1	e1	e1	5f	69	e1
b0	f8	f8	f8	f8	41	f8	f8	e0	85	72	85	61	70	75	85
b1	f1	61	68	61	5f	f1	f1	e1	a0	a0	a0	a0	63	e0	a0
b2	fd	2c	fd	2c	42	fd	fd	e2	83	83	83	83	54	e1	83
b3	c	6	c	7	5	c	c	e3	6	6	5	6	7	5	6

	0	c	0	2	f	0	1			1	1	f	1	9	f	1
b4	c 1	c 1	c 1	c 1	5 f	c 1	c 1		**e4**	8 4	8 4	8 4	8 4	5 f	e b	8 4
b5	e 6	6 c	e 6	6 9	4 5	5 f	e 6		**e5**	8 6	6 c	6 3	8 6	7 8	e e	8 6
b6	1 4	7 3	6 8	6 c	5 f	4 1	1 4		**e6**	9 1	6 3	6 3	9 1	5 f	5 f	9 1
b7	f a	5 f	f a	5 f	5 f	f a	f a		**e7**	8 7	8 7	8 7	6 9	5 f	5 f	8 7
b8	2 c	2 c	2 c	2 c	5 f	4 5	2 c		**e8**	8 a	6 3	8 a	6 3	5 f	e 9	8 a
b9	3 1	7 3	6 9	7 3	5 f	4 8	3 1		**e9**	8 2	8 2	8 2	8 2	5 f	6 9	9 2
ba	a 7	7 3	7 3	6 5	4 b	4 9	a 7		**ea**	8 8	6 5	8 8	6 5	5 f	6 b	8 8
bb	a f	7 4	6 7	6 7	5 f	a f	a f		**eb**	8 9	8 9	8 9	8 9	5 f	5 f	8 9
bc	a c	7 a	6 a	7 4	4 d	4 f	a c		**ec**	8 d	6 5	8 d	6 5	6 2	e 6	8 d
bd	a b	2 2	a b	5 f	4 8	a b	a b		**ed**	a 1	a 1	a 1	a 1	5 f	7 6	a 1
be	5 f	7 a	5 f	7 a	4 f	5 f	5 f		**ee**	8 c	8 c	8 c	8 c	5 f	5 f	8 c
bf	a 8	7 a	7 a	5 f	5 f	e a	a 8		**ef**	8 b	6 4	8 b	6 9	5 f	6 f	8 b
c0	4 1	5 2	4 1	4 1	5 0	6 9	4 1		**f0**	6 4	6 4	5 f	6 4	4 e	e 3	6 7
c1	4 1	4 1	4 1	4 1	4 3	4 1	4 1		**f1**	a 4	6 e	a 4	6 e	8 9	7 0	a 4
c2	4 1	4 1	4 1	4 1	5 4	4 2	4 1		**f2**	9 5	6 e	9 5	6 f	5 f	7 3	9 5
c3	4 1	4 1	5 f	4 1	5 9	e 2	4 1		**f3**	a 2	a 2	a 2	6 b	5 f	e 5	a 2
c4	8 e	4 1	8 e	8 e	5 f	5 f	8 e		**f4**	9 3	9 3	9 3	9 3	5 f	e 7	9 3
c5	8 f	4 c	4 3	8 f	5 8	4 5	8 f		**f5**	6 f	9 4	6 7	6 f	7 3	7 5	6 f
c6	9 2	4 3	4 3	9 2	5 f	5 a	9 2		**f6**	9 4	9 4	9 4	9 4	6 9	e d	9 4
c7	8 0	8 0	8 0	4 9	5 f	4 8	8 0		**f7**	f 6	f 6	f 6	f 6	8 b	5 8	f 6
c8	4 5	4 3	4 5	4 3	5 f	e 9	4 5		**f8**	e d	7 2	6 7	e d	6 a	5 f	e d
c9	9 0	9 0	9 0	9 0	5 f	4 9	9 0		**f9**	9 7	7 5	9 7	7 5	5 f	5 f	9 7
ca	4 5	4 5	4 5	4 5	5 f	4 b	4 5		**fa**	a 3	a 3	a 3	a 3	5 f	6 9	a 3
cb	4 5	4 5	4 5	4 5	5 f	5 f	4 5		**fb**	9 6	8 1	9 6	9 6	6 8	7 5	9 6
cc	4 9	4 5	4 9	4 5	6 2	4 d	4 9		**fc**	8 1	8 1	8 1	8 1	6 b	6 f	8 1
cd	4 9	4 9	4 9	4 9	5 f	4 e	4 9		**fd**	7 9	7 9	7 5	7 5	b 2	7 5	6 9
ce	4 9	4 9	4 9	4 9	5 f	5 f	4 9		**fe**	7 0	7 4	7 3	7 5	9 8	5 f	7 3
cf	4 9	4 4	4 9	4 9	5 f	4 f	4 9		**ff**	9 8	f 9	f 9	f 9	5 f	5 f	9 8

Table 38: xpct translation table for conversion of ISO8859-... character sets to

IBM Code Page 437 characters

charSetMode
Class: CharSetMode
See option "-charsetmode mode"
See resources " ibmCharSet", "isoCharSet"
Specifies which mode the terminal emulator uses for character input and output. The terminal emulator can use either the ISO or IBM character set mode. You can select different character sets within the character set mode by means of the isoCharSet and ibmCharSet resources.
You can switch between the two character set modes during runtime by means of an escape sequence.

Default:
iso
The terminal emulator uses the ISO character set mode and the ISO character classes.

cursorColor
Class: Foreground
See option " -cr color"
Determines the color of the text cursor.

Default:
black

cursorBlinkTime
Class: BlinkTime
See "blinkCursor" resource
Specifies the period of time (in milliseconds) during which the blinking cursor is visible.

Default:
400 milliseconds

cutNewline
Class: CutNewline
See option "-cn"
Specifies whether every line including the last line is terminated by a newline character when text is copied line by line to the cut buffer.

Default:
True
The last line is terminated by a newline character.

cutToBeginningOfLine
Class: CutToBeginningOfLine
See option "-cb"
Affects line-by-line copying of text to the cut buffer and specifies whether the selected line is copied to the buffer completely or only from the cursor position.

Default:
True
Selected lines are copied in full.

cyanColor
Class: CyanColor
Color setting for color terminals

Default:
#00cdcd

defaultVirtualBindings
Class: DefaultVirtualBindings
Specifies the mapping of virtual buttons and keys to your keyboard.

Default:
dependent on X server

fontList
Class: FontList
Font list
xpct uses the font list for character output in the menu bar.

Default:
-sni-97480i1-medium-ot-normal--13-*-*-*-c-*-iso8859-1

foreground
Class: Foreground
See option "-fg color"
Window foreground color.
The foreground color is the color for text output.

Default:
black

geometry
Class: Geometry
See option "-geometry width x height + x_off + y_off"
Defines the position and size of the window.
The value specified for this resource may be ignored by the window manager.
The client does not evaluate window height and width specifications.

Default:
80x25+0+0

greenColor
Class: GreenColor
Color setting for color terminals

Default:
#00cd00

greyColor
Class: GreyColor
Color setting for color terminals

Default:
grey

ibmCharClass
Class: IbmCharClass
See option "-ibmcc character_code : index [, character_code : index]..."
See "charSetMode" resource
Specifies which characters of the IBM Code Page character set are grouped together in a character class. The terminal emulator uses the IBM character classes when you set the character set mode to ibm by means of the charSetMode resource or the -charsetmode option. At present the IBM character sets IBM Code Page 437, 850, 852, 857, 866 and 869 are supported.
Character classes are important when text is selected word by word. If you double-click the mouse button BSelect, successive characters of the same character class are selected for the text buffer.

Default:
0:32,1-8:1,9:32,10-31:1,32:32,33:33,34:34,35:35,36:36,\
37:37,38:38,39:39,40:40,41:41,42:42,43:43,44:44,45:45,\
46:46,47:47,48-57:48,58:58,59:59,60:60,61:61,62:62,63:63,\
64:64,65-90:48,91:91,92:92,93:93,94:94,95-122:48,123:123,\
124:124,125:125,126:126,127:127,128-154:48,155:155,156:156,\
157:157,158:158,159:159,160-165:48,166:166,167:167,168:168,\
169:169,170:170,171:171,172:172,173:173,174:174,175:175,\
176-223:176,224-235:224,236:236,237:237,238:238,239:239,\
240:240,241:241,242:242,243:243,244:244,245:245,246:246,\
247:247,248:248,249:249,250:250,251:251,252:252,253:253,\
254:254,255:255

ibmCharSet
Class: IbmCharSet
See option "-ibmcharset character_set"
See "charSetMode" resource
Specifies the character set with which the terminal emulator works when the character set mode is set to IBM.
You set the character set mode using the charSetMode resource.
At present the IBM Code Pages 437, 850, 852, 857, 866 and 869 are supported.

Default:
ibm437

ibmFontList
Class: FontList
See "charSetMode", " ibmCharSet", "isoFontList" resources
Defines a list of fonts for the display of the terminal emulator when an IBM character set is selected.

The charSetMode resource determines whether an IBM or ISO character set is used. The ibmCharSet resource determines which IBM character set is used. At present the character sets IBM Code Page 437, 850, 852, 857, 866 and 869 are supported.
The font used must be the same size as the fonts in the ISO font list. Proportional fonts cannot be used. The fontList and font resources are not taken into account for the text window of the terminal emulator.
If the specified font does not exist, a replacement font is used instead (e.g. fixed).

Default:

-sni-pct-medium-ot-normal--16-*-*-*-c-*-ibm437=ibm437

iconGeometry
Class: IconGeometry

Specifies the size and position of the window's icon in pixels.
This resource is only effective if the icons are not managed in the icon box. The window manager can ignore these specifications.
No default value.

internalBorder
Class: BorderWidth

See option "-b number"
Specifies the width of the area between the text margin and the window border in pixels.

Default:

2 (pixels)

isoCharClass
Class: IsoCharClass

See option "-isocc character : class [, character : class]..."
See "charSetMode" resource
Specifies which characters from ISO character sets are grouped together in a character class. The terminal emulator uses the ISO character classes when you set the character set mode to ISO by means of the charSetMode resource or the -charsetmode option.
Character classes are important when text is selected word by word. If you double-click the mouse button BSelect, successive characters of the same character class are selected for the text buffer.

Default:

0:32,1-8:1,9:32,10-31:1,32:32,33:33,34:34,35:35,36:36,\
37:37,38:38,39:39,40:40,41:41,42:42,43:43,44:44,45:45,\
46:46,47:47,48-57:48,58:58,59:59,60:60,61:61,62:62,63:63,\
64:64,65-90:48,91:91,92:92,93:93,94:94,95-122:48,123:123,\
124:124,125:125,126:126,127-136:1,137:32,138-159:1,160:160,\
161:161,162:162,163:163,164:164,165:165,166:166,167:167,\
168:168,169:169,170:170,171:171,172:172,173:173,174:174,\
175:175,176:176,177:177,178:178,179:179,180:180,181:181,\
182:182,183:183,184:184,185:185,186:186,187:187,188:188,\
189:189,190:190,191:191,192-214:48,215:215,216:216,\

217-246:48,247:247,248:248,249-255:48
This default specification applies to all ISO character sets.

isoCharSet
Class: IsoCharSet
See option "-isocharset character_set"
See "charSetMode" resource
Specifies the character set with which the terminal emulator works when the character set mode is set to ISO.
You set the character set mode using the charSetMode resource.

Default:
iso8859-1

isoFontList
Class: FontList
See resources "ibmFontList", "charSetMode", "isoCharSet"
Defines a list of fonts for the display of the terminal emulator when an ISO character set is selected.
The charSetMode resource determines whether an IBM or ISO character set is used.
The ibmCharSet resource determines which ISO character set is used. A font can be specified for each ISO character set.
The font used must be the same size as the fonts in the IBM font list. Proportional fonts cannot be used. The fontList and font resources are not taken into account for the text window of the terminal emulator.
If the specified font does not exist, a replacement font is used instead (e.g. fixed).

Default:
-sni-97480i1-medium-ot-normal--16-*-*-*-c-*-iso8859-1=iso8859-1

jumpScroll
Class: JumpScroll
See option "-j"
Specifies whether or not the text in the terminal emulator window is to be scrolled by line or in blocks where necessary (jump scrolling).

Default:
False
Text is scrolled line by line.

lightBlueColor
Class: LightBlueColor
Color setting for color terminals

Default:
#0000ff

lightCyanColor
Class: LightCyanColor
Color setting for color terminals

Default:
#00ffff

lightGreenColor
Class: LightGreenColor
Color setting for color terminals

Default:
#00ff00

lightMagentaColor
Class: LightMagentaColor
Color setting for color terminals

Default:
#ff00ff

lightRedColor
Class: LightRedColor
Color setting for color terminals

Default:
#ff0000

lightWhiteColor
Class: LightBlueColor
Color setting for color terminals

Default:
white

logFile
Class: LogFile
See "logging", "logInhibit" resources
See option "-lf char_string"
Specifies the file to which the terminal emulator writes the log.
This resource only takes effect if the resource logging (class: Logging) is assigned the value "True".
The file is created in the directory in which the emulator was started or in the home directory if xpct was started as the login window.
If the file name begins with a pipe symbol (|), the rest of the string is interpreted as a command that is run by the shell. A pipe is created to the process associated with this command.

Default:
XPCTLog.*string*
string contains the process ID of the xpct process.

logging
Class: Logging
See "logFile", "logInhibit" resources
See option "-l"

Specifies whether or not a terminal emulator logs the session.
If the value is "True", the emulator writes the log to the file specified by the logFile resource (class: LogFile).

Default:
False
The emulator does not log the terminal session.

logInhibit
Class: LogInhibit
See "logging", "logFile" resources
Specifies whether the menu item for the logging function is locked.

Default:
False
Logging is not inhibited.

loginShell
Class: LoginShell
See option "-ls"
Specifies whether the shell started in the emulator window is a login shell.

Default:
False
No login shell is started in the emulator window.

magentaColor
Class: MagentaColor
Color setting for color terminals

Default:
#cd00cd

mapchanFile
Class: MapchanFile
See option "-mf [path] filename"
Defines the Mapchan file for the key mapping for keyboard inputs.

Default:
ZERO

marginBell
Class: MarginBell
See option "-mb"
See "nMarginBell" resource
Specifies whether the emulator triggers an acoustic signal when the text cursor reaches the text column specified in the nMarginBell resource instance (class: Column).

Default:

False
The margin bell is not enabled.

menuAccelerator
Class: Accelerators

defines a sequence of keystrokes as a menu bar accelerator.
Entering the defined keystroke sequence causes the first menu item on the menu bar to be selected.

Default:

<Key>[F10]

monochrome
Class: Monochrome

Specifies whether the client comes up in monochrome mode, in which case all escape sequences relating to color settings are ignored.

Default:

False (on color monitors)
True (on black-and-white monitors without grayscale capability)

multiClickTime
Class: MultiClickTime

See option "-mc milliseconds "
Specifies the maximum time (in milliseconds) that may elapse between 2 mouse clicks for the two clicks to be interpreted as a multiple click.

Default:

250 (milliseconds)

multiScroll
Class: MultiScroll

See option " -s"
Specifies whether text in the emulator window can be scrolled asynchronously.

Default:

False
Text is scrolled synchronously.

nMarginBell
Class: Column

See "marginBell" resource
See option "-nb number"
Specifies the column at which the margin bell is to ring. (class: MarginBell) is assigned the value "True".

Default:

10

noMenubar

Class: NoMenu

Specifies whether or not a menu bar is visible in the terminal emulator window. You can display the menu bar subsequently by using the [ALT] [ESC] key combination.

Default:

False

The menu bar is visible.

pointerColor

Class: Foreground

See option "-ms color"

See "foreground" resource

Specifies the pointer color.

Default:

black

pointerShape

Class: Cursor

Specifies the shape of the pointer.

Default:

xterm (I-beam pointer)

redColor

Class: RedColor

Color setting for color terminals

Default:

#cd0000

reverseVideo

Class: ReverseVideo

See option "-rv"

Specifies whether the colors in the client window are to be inverted.

Default:

False

The colors are not inverted.

reverseWrap

Class: ReverseWrap

See option "-rw"

Specifies whether the reverse wraparound function is enabled. Reverse wraparound allows the cursor to back up from the leftmost column of one line to the rightmost column of the previous line.

Default:

False

Moving backwards from the left to the right margin is not possible.

saveLines
Class: SaveLines
See option "-sl number"
Specifies the number of lines that are to be saved when there is a scroll bar.

Default:
50 (lines)

scrollBar
Class: ScrollBar
See "saveLines" resource
See option "-sb"
Specifies whether a vertical scroll bar is to be displayed in the emulator window.

Default:
False
The emulator window does not have a scroll bar.

scrollInput
Class: ScrollCond
See option "-si"
Specifies whether the window is positioned to the end of the scroll area (= end of text) for output to the window if the end of the text was previously not visible.

Default:
True
The window is positioned to the end of the text.

scrollKey
Class: ScrollCond
See option "-sk"
Specifies whether pressing a key causes the window with the input focus is to be scrolled forward to the end of the text if the latter was previously not visible.

Default:
False
The window is not positioned to the end of the text.

signalInhibit
Class: SignalInhibit
Governs whether the sending of signals to the shell's process group is disabled on the XPct menu and whether these menu items are disabled for the terminal emulator.

Default:
False
The menu items for sending signals to the shell's process group are selectable.

termName
Class: TermName
See option "-tn name"

This is the name of the terminal assigned to the TERM environment variable. This name must be in the /usr/lib/terminfo database.

Default:

ansi-tc20

title

Class:Title

See option "-title char_string"

Specifies the name of the emulator window, which may be displayed by the window manager in the title bar.

Default:

xpct

topShadowColor

Class: Background

Specifies the color to be used for the top shadow of the window frame

Default:

depends on the visual class of the screen

topShadowPixmap

Class: TopShadowPixmap

Pixmap of the top and left 3D shadow for windows

Default:

depends on the visual class of the screen

translations

Class: Translations

specifies the key and button bindings for menus, selection functions and programmed strings

For more information on the syntax for the translations resource refer to the section " translations ".

Default:

Alt <Key>Escape:	**programming()**	**\n**
<KeyPress>:	insert()	\n\
<KeyPress>Retur n:	insert()	\n\
~Meta Shift <Btn1Down>:	select-start()	\n\
~Meta <Btn1Down>:	select-start()	\n\
~Meta Shift	select-extend()	\n\

<Btn1Motion>:		
~Meta <Btn1Motion>:	select-extend()	\n\
~Meta ~Shift <Btn2Up>:	insert-selection(PRIMARY, CUT_BUFFER0)	\n\
~Meta Shift <Btn2Up>:	insert-selection(PRIMARY, CUT_BUFFER0)	\n\
~Ctrl ~Meta <Btn3Down>:	start-extend()	\n\
~Meta <Btn3Motion>:	select-extend()	\n\
~Meta ~Shift <BtnUp>:	select-end(PRIMARY, CUT_BUFFER0)	\n\
~Meta Shift <BtnUp>:	select-end(PRIMARY, CUT_BUFFER0)	\n\
<Btn2Down>:	ignore()	\n\
<BtnDown>:	bell(0) \	

Table 39: Default: translations

ttyModes
Class: TtyModes

See option "-tm " assignment [assignment ...]""

Allows you to set terminal functions for a terminal emulator analogously to the stty command.

You can specify one or more assignments for *value*. Specifications consist of a keyword for the terminal attribute and the desired value.

useStandardColormap
Class: UseStandardColormap

defines the behavior of the Client, if more colors are used than the display supports
If the value of useStandardColormap is set to True, an attempt is made to use colors as close as possible to the StandardColormap.
If the value of useStandardColormap is set to False, a dedicated colormap is created and the required colors are generated in it. If a dedicated colormap is allocated, the color settings for the windows of other clients will also be changed. When the input focus is moved to another window, the color settings are restored to their original values.

Default:

True

utmpInhibit
Class: UtmpInhibit

See option "-ut"

Specifies whether the login entry for the emulator process is entered in the system log file /etc/utmp.

Default:
False
The emulator generates an entry.

visualBell
Class: VisualBell
See option "-vb"
Specifies whether the emulator generates a visual warning rather than an acoustic signal when `[CTRL] [G]` is received.
The window contents are displayed briefly in reverse video.

Default:
False
An acoustic warning is sounded.

xnlLanguage
Class: XnlLanguage
Specifies the client's locale.

Default:
value of the LANG environment variable

yellowColor
Class: YellowColor
Color setting for color terminals

Default:
#ffff00

2.18.2 Appearance and behavior

xpct emulates almost all of the attributes of the ANSI emulator of the TC20-V100 terminal, with the following exceptions:

- The attribute "blinking" is implemented by using different colors.
- Of the IBM Code Pages, the character sets IBM Code Page 437, 850, 852, 857, 866 and 869 are currently supported.
- xpct does not offer scan code support.
- xpct does not offer printer support.

xpct offers the following functionality in addition to that of TC20-V100:

- A menu bar and pull-down menus for greater ease of operation
  You can use the "Xpct", "Mode", "Display", and "Keyboard" menus to influence the appearance and behavior of the terminal emulator. The "Help" menu function allows you to display help texts on how to use the terminal emulator.
- Cut and paste functionality
  This allows you to copy text to and from the clipboard either within the xpct window or between windows.
- A scroll bar for scrolling the text window
- Different font sizes
  (three sizes are shipped)
- dead keys and compose keys

This can be implemented either by the input method, in which case the variable XMODIFIERS=@im=Local must be set, or by way of the Mapchan files, in which case the required Mapchan file must be specified (-mf option or mapchanFile resource).

- The shortcut **[ALT] [ESC]** causes the menu bar to be displayed

xpct supports the following 16 colors:

Cn	Color	Cn	Color
0	Black	8	Grey
1	Blue	9	LightBlue
2	Green	10	LightGreen
3	Cyan	11	LightCyan
4	Red	12	LightRed
5	Magenta	13	LightMagenta
6	Brown	14	Yellow
7	White	15	LightWhite

Table 40: xpct - color support

The following sections describe the appearance and behavior of the default configuration of the terminal emulator.

2.18.2.1 Terminal emulator configuration

Input focus: If the input focus is assigned to the emulator window, the emulator represents this by visually highlighting the cursor (solid rectangle), if permitted by the window manager. Even if the keyboard is locked the cursor remains highlighted. You can use the alwaysHighlight resource of the AlwaysHighlight class to specify that the cursor should always be highlighted.

International implementation: The terminal emulator supports handling of NLS message files and evaluates the LANG environment variable. Currently, however, there is only one message file and this is in German. For all other languages, the terminal emulator uses the English default texts embedded in the program code.

The message file includes the messages which the emulator displays when you call it using the -help option or an invalid option.

There are two resource files for language-specific menu items: the /usr/lib/X11/app-defaults/XPct file, which contains the English menu items, and the /usr/lib/X11/De/app-defaults/XPct file with the German menu items. Depending on the contents of the LANG environment variable, either the default file or the German-language file is evaluated.

Terminfo entries: On startup the terminal emulator searches the terminfo database for the entries ansi-tc20-c, ansi-pct-c (color) or ansi-tc20, ansi-pct (monochrome), pct, xansi, ansi and dumb in that order. The TERM environment variable is set to "ansi-tc20-c" or "ansi-tc20" if the corresponding terminfo entry is in the database. If there is no such entry, the variable is set to the first entry found.

Support of the SINIX keyboard: On an X server that works with a SINIX keyboard, xpct

supports only the function keys that also exist on an MF2 keyboard. Additional keys are ignored (e.g. **[HRLP], [START], [INSERTWORD]** ...). The function keys **[F13] - [F22]** and **[SHIFT] [F13] - [SHIFT] [F22]** are exceptions. They are assigned the escape sequences **[CTRL] [F1]** to **[CTRL] [F10]** and **[CTRL] [SHIFT] [F1] - [CTRL] [SHIFT] [F10].**

2.18.2.2 Character sets

The terminal emulator supports the character sets IBM Code Page 437, 850, 852, 857, 866 and 869 and ISO 8859-1,-2,-3,-4,-5,-7 and -9. You can switch xpct to an IBM mode or an ISO mode on program startup or during program runtime. To set the character set mode you require on program startup you use the -charsetmode option or the charSetMode resource.
An explicitly assigned character set is used in each of the two character set modes. You assign a character set to a mode by means of the -ibmcharset option or ibmCharSet resource for IBM character sets and the -isocharset option or isoCharSet resource for ISO character sets.
You can use the ibmFontList and isoFontList resources to select the fonts by means of which the different character sets are to be represented. Independently of this, you can use the fontList resource to select a separate font for the menu bar.
In both character set modes, it is possible in accordance with the functionality of the emulated terminal to switch from the selected character set to one of two alternative character sets. The alternative character sets are essentially for the output of line graphics characters; an IBM Code Page is always used for this. You will find further information in the programming manual for the terminal.
Mapping IBM character codes to ISO character codes: In IBM modes, it is necessary to convert keyboard input, which is initially always coded according to ISO 8859-*, to the IBM character set. The translation tables required for this are available and can, if necessary, be changed by means of the charSetMapping resource.

Example:

The string

```
iso8859-1, ibm437 \
[ 0xcc : 0x49     \
0xa4 : 0xb0     \
0xb1 : 0xf1 ]   \
iso8859-2, ibm437 \
[ 0xc4 : 0x8e ]
```

maps the characters on the left, which belong to an ISO character set, to those of the IBM 437 character set on the right. In the example, three characters of the ISO 8859-1 set are mapped to the latter, followed by one of the ISO 8859-2 character set.

Character classes in text selection: Character classes are important when text is selected word by word to be copied to the text buffer. You select text word by word by double-clicking the mouse button BSelect. Word by word means here that successive characters of the same character class belong to a word. Since there are various requirements for how text is to be selected word by word (whether file names should be selected as a whole or whether each name component should be selected individually, for example), the default setting of the character classes can be changed. The isoCharClass resource allows you to do this for the ISO character classes and the ibmCharClass resource lets you do it for the IBM Code Pages character classes.

You define a character class by assigning one or more characters of a character set to a character class. The characters are specified by means of a decimal value of 0-255 in the character table. The position corresponds to the character code in decimal form. The value for the class can be specified by means of the position of any character belonging to the character class. In the default list, the character code of the first character in the class is usually used for this.

Example:

The string
33:48,37:48,45-47:48,64:46
specifies that the exclamation point ! (33), percent sign % (37), minus sign - (45), period . (46) and slash / (47) are to be in the same character class as normal characters and digits. The character 0 (48) is specified as the representative of the character class. In addition, the commercial at @ (64) is assigned to the character class to which the period . (46) belongs. This also belongs to the character class 48. This specification is useful when selecting UNIX file names and electronic mail addresses word by word.

Xpct determines at the start of the program which of the two character class resources is to be used. When the terminal emulator is in IBM mode, the IBM character class is used; when it is in ISO mode, the ISO character classes are used. These character classes continue to be used even if you switch to another character set mode during runtime. The word-by-word selection of strings containing characters from an IBM as well as an ISO 8859 character set can lead to unexpected behavior.

2.18.2.3 Control sequences

Additional color-related escape sequences: You initiate these control sequences with the key sequence [ESC][. You then enter one or more of the parameters listed below under n, using a semicolon (;) to separate multiple parameters. To terminate the escape sequence you enter the letter m.
On 8-bit terminals you use the [CSI] key instead of [ESC][.
The effect of the parameters is as follows:

0	reset all attributes (normal display)
1	reduced intensity (or light color)
4	underscored (where supported by hardware)
5	blinking (where supported by hardware)
7	reverse video
8	blanking (non-display)
10	primary font
11	first alternate font ASCII characters below 32 are displayed as ROM characters
12	selects the second alternate font

for foreground

30	black	34	blue
31	red	35	magenta
32	green	36	cyan
33	brown	37	white

for background

40	black	44	blue
41	red	45	magenta
42	green	46	cyan
43	brown	47	white

Additional control sequences: You initiate these control sequences with the key sequence `[ESC]` [, You then enter an equal sign followed by one of the parameters listed below. To terminate the escape sequence you enter the letter indicated in the table. On 8-bit terminals you use the `[CSI]` key instead of `[ESC]` [.

Key sequence	**Effect**
CSI=xD	Switches background color intensity on or off (x=1 on, x=0 off).
CSI=cF	sets the normal foreground color to c. (c is a decimal parameter from the colormap.)
CSI=cG	Sets the normal background color. (c is a decimal parameter from the colormap.)
CSI=cH	Sets the foreground to reverse video. (c is a decimal parameter from the colormap.)
CSI=cI	Sets the background to reverse video. (c is a decimal parameter from the colormap.)
CSI=cJ	Sets the graphical foreground. (c is a decimal parameter from the colormap.)
CSI=cK	Sets the graphical background. (c is a decimal parameter from the colormap.)
CSI=nM	Returns the currently applicable color attributes. (n=0 normal, n=1 reverse, n=2 graphical) The colors are returned in text form, with

	decimal values separated by blanks. The return values are terminated by a newline character. For example: "12 0\n" is returned if the current foreground color is lightRed on black.

Table 44: CSI key sequences for 8-bit terminals

2.18.3 Menus

The xpct terminal emulator comes with a menu bar containing four pull-down menus: X97801, Mode, Display and the pull-down Keyboard menu. You can use the menus in the menu bar if the noMenuBar resource (class: noMenu) has the value "False". If the menu bar is not visible, you can use the menu bar with the key sequence **[CTRL] [MENU] [HELP].** After providing information on the structure, contents and meaning of the menu items this section describes how to configure menus.

Menu items	**Meaning**
Logging	Enable/disable logging
Interrupt program	Send SIGINT signal
Terminate program	Send SIGTERM signal
Hangup program	Send SIGHUP signal (shell and emulator are terminated)
Kill program	Send SIGKILL signal (shell and emulator are terminated)
Unmap menu	Switch off menu bar
Quit	Close window and terminate client

Table 45: The "X97801" menu of xpct

Menu items	**Accelerator**	**Meaning**
Jump Scroll	Meta<Key>b	Scroll in blocks (instead of line by line)
Scrollbar	Meta<Key>s	Scroll bars
Scroll to bottom on key		Scroll to end when key is pressed
Scroll to bottom on input		Scroll to end on input
Visual	Meta<K	Visual signal generated shortly

Bell	ey>v	before end of line is reached
Margin Bell	Meta<Key>r	Acoustic signal generated shortly before end of line is reached
Reverse Wraparound	Meta<Key>z	Cursor moves backward from the first column in a line to the last column of the previous line
S**o**ft Reset		Reset emulator's input parser to initial state. If escape sequences have already been detected, branch to initial state of parser. The rest of the escape sequence is handled in normal mode.
F**u**ll Reset		Reset terminal modes and functions to initial state after start of xpct. Any modifications made during the session are cancelled, e.g. tabulators are reset (to every 8 columns). Wrap and smooth scroll are reset; contents of window are deleted.

Table 46: The "Mode" menu of xpct

Menu items	**Accelerator**	**Meaning**
Redraw	Meta<Key>t	Redraw window
Reverse **V**ideo		Foreground and background colors are inverted
Cursor blinking enabled		Enable blinking text cursor
B**l**ock cursor		Define cursor shape (block or underscore)

Table 47: The "Display" menu of xpct

Menu items	**Accelerator**	**Meaning**
Auto Repeat		Indicates "auto-repeat" status of X server; the effects of a change of status are not restricted to the active window
Key Click		Enables/disables key

		clicking. If the 'Key Click' option is enabled, a brief click is heard each time a key is pressed.
Key Mapping		Enables/disables key mapping. If the 'Key Mapping' option is enabled, the dead key and compose key sequences defined in the Mapchan file can be generated with the aid of the Mapchan mechanism.

Table 48: The "Keyboard" menu of xpct

Configuring menus: The configuration of all menus is affected by the resources of xpct's vtmenubar widget (class: XmRowColumn).

Example:

xpct*vtmenubar*background: green

Assigns the value "green" to the "background" resource instance of the menu bar "vtmenubar" for every client of the "Xpct" class.

Individual menus can be configured using the resources of the xpct, mode, display and keyboard widgets, which are all members of the XmRowColumn class.

Example:

Xpct*vtmenubar.display*background: green

Assigns the value "green" to the "background" resource instance of the menu bar "vtmenubar" in the menu "Display" for every client of the "Xpct" class.

2.18.4 Files

The files below are listed in alphabetical order. The resource configuration files are listed only in the form of the generalized file names $XENVIRONMENT, $XUSERFILESEARCHPATH and $XFILESEARCHPATH.

/etc/utmp

system log file in which the terminal emulator creates a login entry when the -ut option is set.

$HOME/.login

shell script executed when a login shell is started in the emulator window.

$HOME/.profile

shell script executed when a login Bourne shell is started in the emulator window. You can, for example, define Bourne shell functions here.

$XAUTHORITY

user-specific authorization file

The client uses this file as a user-specific authorization file. If XAUTHORITY is not set, the client uses the .Xauthority file in the user's HOME directory.

$XENVIRONMENT

user-specific configuration file

$XFILESEARCHPATH
application-specific configuration file created by the client programmer

$XUSERFILESEARCHPATH
application-specific configuration file created by the system administrator or user

XPCTLog.*n*
default log file for the terminal emulator xpct in the directory from which the terminal emulator is called

2.18.5 Environment variables

Detailed descriptions of all the environment variables, arranged in alphabetical order, are given in the chapter "Environment variables ".

DISPLAY
The client reads the DISPLAY environment variable if the -display option is not specified in its command line. From this variable the client finds out the name of the X server and the display and screen on it.
The DISPLAY environment variable is a string in the format:
hostname:*display_number*.*screen_number*
The client terminates abnormally if the DISPLAY environment variable is not set and the -display option is not specified in the client command line.

HOMEidentifies the user's HOME directory.

LANGis evaluated in the context of NLS language support for message texts and configuration files.

NLSPATH
The client evaluates the NLSPATH environment variable as a search path for message files.
Thus the NLSPATH environment variable must include the search path /usr/lib/nls/msg/%l/xpct.cat. "%l" is replaced by that part of the locale definition which identifies the language.

TERMdenotes the type of terminal emulated.
The terminal emulator sets the TERM variable according to the -tn option, the termName resource or the first entry in the terminfo database.

WINDOWID
The terminal emulator sets the contents of the WINDOWID variable to the number of the window, which is unique in the context of the root window.

XAUTHORITY
denotes the user-specific authorization file.
If the environment variable is not set, the client searches for the $HOME/.Xauthority file.

XFILESEARCHPATH
specifies the path for configuration files.

XUSERFILESEARCHPATH
specifies the path for configuration files.

2.19 xrdb - Loading resource definitions into the X server

xrdb loads user-specific resource value assignments into the server as a database.
Users can define their own resource assignments so as to customize the user interface and influence the behavior of their clients. You can, for example, specify

- foreground and background colors
- font type and size
- your own configuration files

Resources, the values that can be assigned to them, and resource syntax are described in the chapter "Client resources".
You can use xrdb to transfer value assignments to the server database. When a client is started, it checks whether the X server has created a resource database. If (and only if) there is no such database, it searches for appropriate entries in the $HOME/.Xdefaults file. Storing the resource database in the server is extremely useful because it is then available to all clients. Clients can access this information at any time, no matter what host they are running on.
xrdb allows you to assign values to resources dynamically in the course of your user session. The resources you define for a client come into effect the next time you start the client.
xrdb may be called by the xdm display manager at the start of a user session, typically to load the user-specific resource file $HOME/.Xresources into the X server's database.

xrdb[*option*]...[*filename*]

-backup *string*
Specifies the extension for the backup file created in conjunction with the -edit option.

string
is the string used as the extension.

Default:
.bak

-cpp *file*
Specifies the pathname of the program to be used as C preprocessor.
Although xrdb was designed to use a preprocessor, any program that acts as a filter and accepts the -D, -I and -U options can be used.

file is the absolute pathname of the file (program).

Default:
depends on the machine
/lib/cpp
/usr/att/lib/cpp

-D*name*[=*value*]
Is passed to the preprocessor as a definition of the symbol *name* for use in conditional directives such as #ifdef.
For more information on symbols, please refer to the *filename* option.

name is the name of the symbol.
value is the value assigned to the symbol.

filename

this is the name of a file passed to and processed by the c preprocessor.
the file can contain resource value assignments and preprocessor directives such as #ifdef, #if, etc..
lines beginning with an exclamation mark (!) are comments and are not evaluated.
preprocessor directives can also be passed from standard input.
the following symbols are defined by xrdb when the preprocessor is called. defaults and permissible values depend on the server being used:

bits_per_rgb=num
: is the number of significant bits in an rgb color specification. this is log base 2 of the number of different shades of the primary colors that can be displayed on the hardware.
please note that this generally does not refer to planes.

class=visualclass
: is the visual class of the root window on the default screen.
visualclass is any of: staticgray, grayscale, staticcolor, pseudocolor, truecolor, directcolor.

clienthost=hostname
: is the name of the host on which xrdb resides.

color is defined only if the value of class is staticcolor, pseudocolor, truecolor or directcolor.

height=num
: is the height of the screen in pixels.

host=hostname
: is the name of the host to which you are connected. this name is the same as that of the corresponding element in the display name (same meaning as serverhost).

planes=num
: is the number of bit planes, i.e. the depth of the main window on the default screen.

release=num
: is the server vendor's release number. the interpretation of this number depends on the contents of vendor.

revision=num
: is the oldest version of the x protocol supported by the x server (the oldest possible version being 0).

serverhost=hostname
: is the name of the host to which you are connected. this name is the same as that of the corresponding element in the display name and has the same meaning as host.

vendor=vendor
: is a string denoting the vendor of the server.

version=num
: is the latest version of the x protocol supported by the server. (this should be 11.)

default:
11

width=num
is the width of the screen in pixels.

x_resolution=num
is the x resolution of the screen in pixels per meter.

y_resolution=num
is the y resolution of the screen in pixels per meter.

-display [*hostname*]:*display_number*[.*screen_number*]
Identifies the X server, display and screen which the client is to contact.

hostname
symbolic name of the host or X terminal on which the X server which the client is to contact runs. *hostname* must be known in the transport protocol of the host on which the client resides.

display_number
number (unique to the X server) identifying the display for which the client is to manage I/O.

.*screen_number*
number (unique to the display) identifying the screen to which the client is to send output.

Default:
contents of the DISPLAY shell variable;
if DISPLAY is not set, the client terminates abnormally.

-edit *filename*
Loads the contents of the server's resource database into the specified file. Any existing values in this file are overwritten.
This allows you to write modifications you have made to your resource file in order to save comment lines or preprocessor directives.
filename is the full pathname of the file.
No default value.

-help Writes a short description of the options to standard error.
This help text is also displayed if the client cannot interpret a command-line argument.

-I*directory*
Passed to the preprocessor as the definition of the directory to be scanned for files to be included by #include directives.

directory
the name of the directory.

-load *filename*
Loads the contents of the input resource file *filename* into the server's resource database.
Any existing entries are overwritten. If you use the *-merge* option instead, the original contents are retained.

Default:

The option is set.

-merge *filename*

Loads the contents of the input resource file *filename* into the server's resource database.
The original contents are retained. Resource specifications of the same name are overwritten.
If xrdb is reading from standard input, you can use this option to modify the contents of the resource database directly from a screen or from a shell script.
Please note that this option does a lexical sort on both the input from the original database and the input from the input file. This is generally undesirable because it is not possible to undo the action.
The -load option also loads the input file into the server's database but completely overwrites the original contents.

Default:

The option is not set.

-n Specifies that modifications to the resource database made using the -load option or modifications to the resource input file made using the -edit option are first written to standard output. The modifications are not implemented immediately.

Default:

xrdb modifies the resources.

-nocpp

Causes xrdb to load the resource input file *filename* directly into the server without having the data preprocessed first.

Default:

The data is preprocessed.

-query

Causes the current contents of the server's resource database to be written to standard output.
Please note that preprocessor directives in the resource input file are part of the input file, not part of the server's database, and thus do not appear in the output.
You can use the -edit option to write the contents of the database back to the resource input file without affecting preprocessor directives.

-quiet

Suppresses warnings about identical entries.

Default:

Warnings are issued.

-remove

Removes the server's resource database from the root window of screen 0.

-retain

Specifies that the server is not to be reset if xrdb is the first client.
This means that the server is not restarted. If xrdb is the first client, it may when terminated also be the only client. This means that when it is terminated, the last

client is terminated, causing the server to be restarted and all data to be lost.

Default:
The option is not set.

-symbols
Causes the symbols defined for the preprocessor to be written to standard output. The symbols which are available are included in the description of the *filename* option. This option can be combined with the -query option, but not with the options that modify the server's resource database (-load, -merge).

Default:
The symbols are not written to standard output.

-U*name*
Passed to the preprocessor, which uses it to remove all definitions of the symbol *name*.
Symbols set by xrdb are listed in the description of the *filename* option.

name is the name of the symbol.

2.19.1 Files

The files below are listed in alphabetical order.

$HOME/.Xdefaults
user-specific configuration file

$HOME/.Xresources
contains user-specific value assignments which xdm is to load into the server for the current session.
It may also contain directives for the preprocessor used by xrdb.

$XAUTHORITY
user-specific authorization file
The client uses this file as a user-specific authorization file. If XAUTHORITY is not set, the client uses the .Xauthority file in the user's HOME directory.

2.19.2 Environment variables

Detailed descriptions of all the environment variables, arranged in alphabetical order, are given in the chapter "Environment variables ".

DISPLAY
The client reads the DISPLAY environment variable if the -display option is not specified in its command line. From this variable the client finds out the name of the X server and the display and screen on it.
The DISPLAY environment variable is a string in the format:
hostname:*display_number*.*screen_number*
The client terminates abnormally if the DISPLAY environment variable is not set and

the -display option is not specified in the client command line.

HOMEidentifies the user's HOME directory.

XAUTHORITY

denotes the user-specific authorization file.
If the environment variable is not set, the client searches for the $HOME/.Xauthority file.

XFILESEARCHPATH

specifies the path for configuration files.

XUSERFILESEARCHPATH

specifies the path for configuration files.

2.20 xterm - VT102 and Tektronix 4014 terminal emulators

This terminal emulator allows you to run programs that can only use alphanumeric devices of types DEC VT102 and Tektronix 4014.
Not all the functions provided by VT102 and Tektronix 4014 terminals are supported, e.g. no termcap entries are supported, i.e. xterm cannot set or evaluate termcap entries.
xterm provides both a VT102 and Tektronix 4014 emulator, each with its own window thus enabling you to edit text in the VT102 window and work with graphics in the Tektronix 4014 window. The output area for the Tektronix graphics is reduced to the smallest possible size that will still fit in the window while retaining the correct 4014 aspect ratio (height/width). This graphics output area is located in the upper left part of the 4014 window.
You can start xterm in one of the two emulators and then switch to the other. xterm can display both windows at the same time but only one window is ever active at any one time, usually the window from which xterm was started. Only the active window receives keyboard input and terminal output. This is the window that contains the text cursor and whose border highlights whenever the pointer is in either window. You select the active window by means of escape sequences, the "Modes" menu (in the VT102 window) or the "Tektronix" menu (in the 4014 window).
xterm[*option*]...
In cases where an option has both -*option* and +*option*, +*option* is usually used to reset the attribute specified by -*option* to its default value.

-132 See "c132" resource (class C132)
Allows the VT102 emulator window to be 132 columns wide (instead of 80). This option enables you to use the DECCOLM escape sequence **[ESC]**[?3h to switch from 80-column mode to 132-column mode and the DECCOLM escape sequence **[ESC]**[?3l to switch from 132-column mode to 80-column mode; the emulator window is resized accordingly.

Default:
the VT102 emulator window is 80 columns wide; xterm ignores the DECCOLM escape sequence.

-ah ah - always highlight
See option "+ah"
See "alwaysHighlight" resource
This option causes the cursor to always be highlighted, even if the input focus is assigned to a different window.

Default:
Only a hollow cursor is displayed when the window loses the input focus.

+ah causes the cursor to be highlighted only if the input focus is assigned to the emulator window.

-b *number*
See "internalBorder" resource
number specifies (in pixels) the width of the internal border.
The internal border is the area between the text margin and the window border.

Default:
2 (pixels)

-bd *color*
bd - border color
See "borderColor" resource
color is the color of the window border.

Default:
black

-bg *color*
bg - background
See "background" resource
color is the color of the window background.

Default:
white

-bw *number*
bw - border width
See "borderWidth" resource
number is the width of the window border in pixels.
The window manager may ignore or adjust this value.

Default:
2 (pixels)

-cc *position*:*index*[,...]
See "charClass" resource
Specifies how the terminal emulator handles characters when text is selected word by word.

position
identifies the characters which are to be grouped to form a character class in a terminal emulator. *position* can be a number or a value range in the form $number_1$-$number_2$.

index specifies the character class and thus how the characters selected by *position* are handled.

Default:
Default character class

-cr *color*
See "cursorColor" resource
color is the color of the text cursor when the window has the input focus.

Default:
black

-cu See " curses" resource
Causes leading tabs at the beginning of continuation lines to be suppressed.

This option applies to the cursor motion package *curses* (see the CES manual).

Default:
leading tabs are output.

+cu Causes leading tabs to be output.

-display [*hostname*]:*display_number*[.*screen_number*]
Identifies the X server, display and screen which the client is to contact.

hostname
symbolic name of the host or X terminal on which the X server which the client is to contact runs. *hostname* must be known in the transport protocol of the host on which the client resides.

display_number
number (unique to the X server) identifying the display for which the client is to manage I/O.

.*screen_number*
number (unique to the display) identifying the screen to which the client is to send output.

Default:
contents of the DISPLAY shell variable;
if DISPLAY is not set, the client terminates abnormally.

-e *command*
command is called in the emulator window instead of the shell. If you do not specify either the -T or the -n option in the emulator call, the emulator uses the command name as the window and the icon name.
This option may only be specified as the last argument of the terminal emulator command line.

Default:
a shell is started in the terminal emulator window.

-fb *font_name*
fb - font bold
See "boldFont" resource
font_name is the name of the font to be used for bold text.
The options -fb and -fn allow you to specify the font to be used for normal text (-fn) and the font for bold text (-fb). These fonts must have the same height and width. If you specify only one font (either -fb or -fn), xterm will use it as the normal font and create the bold font by widening the normal font.
Use only fonts whose characters have a fixed width (pitch) as xterm cannot handle fonts with variable character widths (proportional fonts) correctly.

Default:
vtbold

-fg *color*
fg - foreground

See "foreground" resource
color is the color of the window foreground.
This is the color in which text characters are displayed.

Default:
black

-fn *font_name*
fn - font normal
See "font" resource
font is the name of the font for the basic character style.

Default:
vtsingle

%geomDefines the preferred size and position of the Tektronix window.
%geom applies to the tekGeometry resource.
This option is only provided for compatibility with older versions as the -geometry option has superseded %geom. It can be assumed that %geom will cease to be supported in the next release.

Default:

width 750 + 2 x border (pixels)
height
565 + 2 x border (pixels)

#geom *width*x*height*+*x_off*+*y_off*
See "iconGeometry" resource
#geom specifies the preferred size and position of the icon of the emulator window.
This option is only provided for compatibility with older versions. It can be assumed that it will cease to be supported in the next release.
No default value.

-geometry *width*x*height*+*x_off*+*y_off*
See "geometry" resource
Defines the size and position of the VT102 terminal emulator window.

width the window width.
height
the window height.
x_off Displacement of the window along the x axis (horizontally) in pixels

+*x_off*
distance between left edge of window and screen
-*x_off*
distance between right edge of window and screen

y_off Displacement of the window along the y axis (vertically) in pixels

+y_off
distance between top edge of window and screen
-y_off
distance between bottom edge of window and screen

Default:
80x25+0+0

-help Writes a short description of the options to standard error.
This help text is also displayed if the client cannot interpret a command-line argument.

-iconic
Indicates that the mwm window manager is to start the client as an icon, not as a normal window.

Default:
The mwm window manager opens the normal window of the client.

-j j - jump scrolling
See "jumpScroll" resource
See option " -s"
See Modes menu (table "The "Modes" menu of xterm")
Text within the window is scrolled in blocks, not line by line.
Jump scrolling means that several lines are moved simultaneously out of the display at the upper edge of the screen. This speeds up scrolling through large amounts of data significantly, since it reduces the load on both the terminal emulator and the X server.

Default:
text in the emulator window is scrolled line by line.

+j The text in the emulator window is scrolled line by line.

-L
Specifying -L in the terminal emulator call prevents the terminal emulator from allocating a pseudo terminal
The terminal emulator assumes that this has already been done by /etc/init. The name of the pseudo terminal must be the last argument of the terminal emulator call.
-L also causes a getty process to be started instead of the user shell. Therefore, the -L option should only be used when the emulator is started by /etc/init. Please refer to the system administration documentation for your operating system for more details on /etc/init and getty.
-L is used when the emulator is started by the /etc/init process.
As of SINIX V5.4 -L is not supported.

Default:
the terminal emulator allocates a pseudo terminal and starts a shell.

-l l - logging
See "logging" resource
See option "-lf char_string"
Causes the emulator to keep a log.

The terminal emulator directs all output, including error output, both to the log file and to the screen.
You can use the -lf option to set up a log file of your own; if you do not do so, the terminal emulator uses the log file XtermLog.*string*. *string* is an extension to the file name which makes it unique within the directory in which the file is stored. If you do not specify an absolute pathname, the log file is created in the directory where the emulator was started.

Default:
The terminal emulator does not keep a log.

+l The terminal emulator does not keep a log.

-lf *char_string*
lf - logfile
See "logFile" resource
See option "-l"
For *char_string* you can specify a filename *file* or set up a pipe to the command (in this case, the first character of *char_string* is the pipe symbol: |*command*).

-lf␣*file*
The emulator uses *file* as the log file. *file* is created in the directory where the emulator was started.
-lf-␣|*command*
A pipe to *command* is created and *command* is executed by the shell.

Default:
XtermLog.*string*
string is an extension to the file name which makes it unique within the directory in which the file is stored.

-ls ls - login shell
See "loginShell" resource
The terminal emulator starts a login shell and executes the .profile or .login file. The initial working directory of the login shell is the directory in which the terminal emulator is started.

Default:
the terminal emulator starts a subshell which is not a login shell.

+ls The terminal emulator starts a subshell which is not a login shell.

-mb mb - margin bell
See "marginBell" resource
See option "-nb number"
Enables the margin bell for the right text margin.
The margin bell sounds when the cursor reaches the text column which was specified using the -nb option or the corresponding resource.

Default:
the terminal emulator does not sound a margin bell.

+mb The terminal emulator does not sound a margin bell.

-mc *milliseconds*

mc - multi-click time
See "multiClickTime" resource
Defines the maximum time in milliseconds that may elapse between 2 mouse clicks for the two clicks to be interpreted as a multiple click.

Default:
250 (milliseconds)

-ms *color*

ms - mouse pointer color
See "pointerColor" resource
See option "-fg color"
color determines the color of the pointer.

Default:
black

-n *char_string*

char_string determines the name given to the icon of the emulator window.
It can be assumed that the -n option will cease to be supported in a future release.
Please note the difference between this option and the -name option.

Default:
xterm

-name *resource_link*

resource_link is the name of the application under which the client's window resources are to be found.
The client can thus be called under different names and utilize the resources for its client instance *resource_link*, yet still utilize the same resources for its client class *clientclass*.
resource_link should not contain dots (.) or asterisks (*), as these symbols have special meanings. Please refer to the section "Value assignment syntax " for more information.

Default:
xterm

-nb *number*

See option "-mb"
See resources "nMarginBell", "marginBell"
Specifies the column at which the margin bell is sounded.
number is the offset in columns from the right text margin. The margin bell is only sounded when enabled via the -mb option, the "Modes" menu or the marginBell resource.

Default:
10

-r See option "-rv"
Causes xterm to simulate reverse video by inverting the foreground and background colors.
This option is only provided for compatibility with older versions. As the -rv option has superseded the -r option, the -r option may cease to be supported in a future release.

Default:
foreground and background colors are not inverted.

-rv rv - reverse video
See "reverseVideo" resource
Inverts the foreground and background colors.

Default:
Foreground and background colors are not inverted.

-rw rw - reverse wraparound
See "reverseWrap" resource
See Modes menu (table "The "Modes" menu of xterm")
Enables the reverse wraparound function.
Reverse wraparound allows the cursor to back up from the left text margin (column 1) to the last column (right text margin) of the previous line. This is especially useful when editing long command lines.

Default:
the emulator does not support reverse wraparound.

+rw The text cursor cannot be backed up beyond the left-hand text margin.

-S*cn*
The terminal emulator is started in slave mode.
The terminal emulator uses a pseudo terminal for input and output. This pseudo terminal must have been allocated by the master which starts the emulator in slave mode, since the emulator does not perform any allocations in slave mode.
In slave mode, it is not possible to send signals to the process group of the terminal emulator via the menu.

c is the variable part of the name of the pseudo terminal on which the emulator is started. In SINIX V5.2x this variable part is two characters long. As of SINIX V5.4, it has to be given as a string of exactly four characters. If the name has fewer than four characters, it has to be padded to four with leading dashes ('-' = 0x2d).

n is the one or two-digit decimal number of the file descriptor, i.e. the channel number which the master process used to open the master side of the pseudo terminal.

No default value.

-s Allows text to be scrolled asynchronously.
Positioning by scrolling large amounts of text is speeded up; during scrolling, the text visible in the window does not correspond to the scroll bar position.
-s is very useful when running the terminal emulator in extensive networks with long

latency times.
Precondition: there must be a scroll bar.

Default:
synchronous scrolling

+s The emulator scrolls synchronously.
During scrolling, the text visible in the window corresponds to the scroll bar position.

-sb sb - scroll bar
See "saveLines" resource
See option "-sl number"
The terminal emulator displays a vertical scroll bar.
The emulator saves the lines which are scrolled off the top of the window. The scroll bar can be used to restore scrolled lines to the window. The highlighted part of the scroll bar also shows what part of the whole text is being displayed within the window. You can use the -sl option or the saveLines resource to specify how many of the lines which have been scrolled off the window are saved.

Default:
no scroll bar

+sb No scroll bar is displayed.

-sf sf - Sun functions
Causes Sun function key escape codes to be assigned to function keys.

Default:
standard escape codes

+sf The function keys are assigned standard escape codes.

-si si - scroll to bottom on output inhibit
See option "-sb"
See "scrollInput" resource
When output is sent to the emulator window, this option inhibits positioning to the end of the scroll area unless it was already visible.
Precondition: There must be a scroll bar.

Default:
The emulator window is positioned to the end of the scroll area for output.

+si When output is sent to the emulator window, the window is positioned to the end of the scroll area.

-sk sk - scroll to bottom on key
See option "-sb"
See "scrollKey" resource
On input from the keyboard, the emulator window is positioned to the end of the scroll area (= end of text).
Precondition: There must be a scroll bar.

Default:
Positioning to the end of text is inhibited for keyboard input.

+sk On input from the keyboard, the window is not repositioned to the end of the scroll area (= end of text) if text has previously been scrolled down out of the window.

-sl *number*

sl - save lines

See "saveLines" resource

number specifies how many lines the emulator is to save.

Default:

64 (lines)

-T *char_string*

See option "-title char_string"

char_string is the name of the emulator window which is displayed in the title bar. The -T option is provided only for reasons of compatibility with previous versions; instead of -T you should use the -title option, which is equivalent to -T. It may be assumed that the -T option will cease to be supported in a future release.

Default:

xterm

-t t - Tektronix

xterm starts the Tektronix 4014 emulator as the active window.

You can use the "Modes" menu to switch back and forth between the Tektronix 4014 and VT102 emulators.

Default:

xterm starts the VT102 emulator as the active window.

+t xterm starts the VT102 emulator as the active window.

You can use the "Modes" menu to switch back and forth between the VT102 and Tektronix 4014 emulators.

-title *char_string*

See option "-e command"

See "title" resource

char_string is the name which the mwm window manager displays in the title bar of the emulator window.

Default:

xterm or the command name from the -e option.

-tm "*assignment*[*assignment*...]"

tm - tty modes

See "ttyModes" resource

This resource sets terminal driver parameters.

assignment is a value assignment for terminal attributes.

If the emulator command line contains the -L option, specifications for -tm are ignored since the getty process started by the emulator resets the terminal attributes. For further information on terminal attributes please refer to the stty command description in the operating system commands manual.

assignment

Each *assignment* consists of a keyword and the value for the terminal attribute in the form of *keyword value*; *value* is always exactly one character long.
Valid keywords: intr, quit, erase, kill, eof, eol
Control characters can be used with a leading ^ character; use ^? to specify the delete function.

Defaults:

intr	Delete
quit	^\
erase	^h
kill	^x
eof	^d
eol	^@

-tn *name*

See "termName" resource
name is the identifier of the terminal type which the terminal emulator assigns to the TERM variable.
The terminal type *name* must be defined in the terminfo database.

Default:

VT102

-ut See "utmpInhibit" resource
The terminal emulator does not create a login entry in the system log file /etc/utmp.

Default:

The emulator creates a login entry.

+ut The emulator creates a login entry in the /etc/utmp file, provided that "root" is the owner of the emulator program file and that the "set-user-id" bit is set.

-vb vb - visual bell
See "visualBell" resource
On receiving **[CTRL] [G]** the terminal emulator generates a visual warning instead of an acoustic one.
Window contents and background are displayed briefly in reverse video.

Default:

[CTRL] [G] causes the terminal emulator to sound an acoustic warning.

+vb On receiving **[CTRL] [G]** the terminal emulator sounds an acoustic warning.

-w *number*

number is the width of the window border in pixels.
This option is only provided for compatibility with older versions, as the -bw option now perform the same function. -w may not be supported in the next release.

Default:

2 (pixels)

-xrm "*resourceDefinition*:*value*"

Assigns a value to a client resource.

-xrm must be specified separately for each resource to be set in the client call. The "*resourceDefinition*:*value*" assignment is compared with the entries in the configuration files. For many resources there are corresponding command-line options. To address resources for which there are no specific options available in the client call you have to use the -xrm option.

resourceDefinition

Identifies the resource to which a value is to be assigned. For *resourceDefinition* you can give the name of the resource, the name of the client and of any widgets in the appropriate resource syntax (see section "Value assignment syntax ").

value Value to be assigned to the resource.

2.20.1 Resources

You can configure the client using its resource database. To do this, you must assign the required values to the resources. You can use either configuration files or the -xrm option in the client call to access resources.
The instance name of the client is xterm and its class name XTerm. Please note that you can use the -name *instancename* option to change the instance name of the client when you call it.

2.20.1.1 Resource overview

This subsection starts with overviews showing which resources can be used to configure which widgets, followed by brief descriptions of the resources listed alphabetically according to resource name.

Resource	Value	Default	Option
iconGeometry	size	-See resource description	
sunFunctionKeys	True|False	False	-sf
termName	terminaltyp	-Depending on the setting of other resources	-tn
title	string	xterm	-title
ttyModes	string	-[2]	-tm
utmpInhibit	True|False	False	-ut

Table 50: xterm resources: widget "xterm" (widget class: Xterm)

Resource	Value	Default	Option
alwaysHighlight	True|False	False	-ah
background	color	white	-bg
boldFont	string	vtbold	-fb
c132	True|False	False	-132
charClass	pos:index	-Dependent on the setting of other resources	-cc
curses	True|False	False	-cu
cursorColor	color	-[1]	-cr
font	string	vtsingle	-fn
foreground	color	black	-fg
geometry	size	80x25+0+0	
tekGeometry	size	-[1]	
internalBorder	number	2	-b
jumpScroll	True|False	False	-j
loginShell	True|False	False	-ls
marginBell	True|False	False	-mb
multiClickTime	number	250	-mc
multiScroll	True|False	False	-s
nMarginBell	number	10	-nb
pointerColor	color	-[1]	-ms
pointerShape	string	xterm	
reverseVideo	True|False	False	-rv
reverseWrap	True|False	False	-rw
saveLines	number	64	-sl
scrollBar	True|False	False	-sb
scrollInput	True|False	True	-si
scrollKey	True|False	False	-sk
signalInhibit	True|False	False	
tekGeometry	size	-[1]	
tekInhibit	True|False	False	
tekStartup	True|False	False	-t

translations	string	-[1]	
visualBell	True|False	False	-vb

Table 51: xterm resources: widget "vt100" (widget class: VT100)

Resource	Value	Default	Option
menuBorder	number	2	
menuFont	string	Dependent on the setting of other resources)	
menuPad	number	3	

Table 52: xterm resources: Widget "tek4014" (Widget class: Tek4014)

Resource	Value	Default	Option
background	color	black	
foreground	color	white	
thickness	number	24	

Table 53: xterm resources: Widget "scrollbar" (Widget class: Scrollbar - Athena Widget Set)

2.20.1.2 Resource list

The resources available to this client are listed in alphabetical order below. You can also use them in the -xrm option.

alwaysHighlight
Class: AlwaysHighlight
See option "-ah"
Defines whether or not the text cursor is always highlighted, even if the input focus is assigned to a different window.

Default:
False
The cursor is only highlighted in the active window.

background
Class: Background
See option "-bg color"
Determines the color of the window background.

Default:
white

You can customize this resource specifically for the *scrollbar* widget by specifying xterm*_scrollbar_*background. This defines the color of the scroll bar. In the scrollbar widget:
specifies the background color of the scroll bar. The highlighted area, which indicates the relative amount and position of the text displayed in the window, is created using a checkerboard pattern for which pixels are alternated in the foreground and background colors.

Default:
black

boldFont
Class: Font
See option "-fb font_name"
Specifies which font is to be used for bold text in the emulator window.

Default:
vtbold

borderColor
Class: BorderColor
Specifies the color of the window border in the client area.

borderWidth
Class: BorderWidth
See option "-bw number"
Specifies the width of the client window border in pixels.

Default:
2 pixels

c132
Class: C132
See option "-132"
Specifies whether or not the DECCOLM escape sequence is accepted.
The DECCOLM escape sequence `[ESC]`[?3h enables you to switch from 80-column mode to 132-column mode; the DECCOLM escape sequence `[ESC]`[?3l enables you to switch from 132-column mode to 80-column mode.

Default:
False

charClass
Class: CharClass
See option "-cc position : index [,...]"
Determines which characters are grouped together as a character class.
You form a character class by assigning all characters of the desired class a common numeric value. As common numeric value for all elements of the character class, you can choose the decimal value of the position of any element of the class.

Default:

default character classes

Position	0	1	2	3	4	5	6	7
Character	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL
Character Class	32	1	1	1	1	1	1	1
Position	8	9	10	11	12	13	14	15
Character	BS	HT	NL	VT	NP	CR	SO	SI
Character Class	1	32	1	1	1	1	1	1
Position	16	17	18	19	20	21	22	23
Character	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB
Character Class	1	1	1	1	1	1	1	1
Position	24	25	26	27	28	29	30	31
Character	CAN	EM	SUB	[ESC]	FS	GS	RS	US
Character Class	1	1	1	1	1	1	1	1
Position	32	33	34	35	36	37	38	39
Character	SP	!	"	#	$	%	&	'
Character Class	32	33	34	35	36	37	38	39
Position	40	41	42	43	44	45	46	47
Character	(	)	*	+	,	-	.	/
Character Class	40	41	42	43	44	45	46	47
Position	48	49	50	51	52	53	54	55
Character	0	1	2	3	4	5	6	7
Character Class	48	48	48	48	48	48	48	48
Position	56	57	58	59	60	61	62	63
Character	8	9	:	;	<	=	>	?
Character Class	48	48	58	59	60	61	62	63
Position	64	65	66	67	68	69	70	71
Character	@	A	B	C	D	E	F	G
Character Class	64	48	48	48	48	48	48	48

Position	72	73	74	75	76	77	78	79
Character	H	I	J	K	L	M	N	O
Character Class	48	48	48	48	48	48	48	48
Position	80	81	82	83	84	85	86	87
Character	P	Q	R	S	T	U	V	W
Character Class	48	48	48	48	48	48	48	48
Position	88	89	90	91	92	93	94	95
Character	X	Y	Z	[	\	]	^	_
Character Class	48	48	48	91	92	93	94	48
Position	96	97	98	99	100	101	102	103
Character	`	a	b	c	d	e	f	g
Character Class	96	48	48	48	48	48	48	48
Position	104	105	106	107	108	109	110	111
Character	h	i	j	k	l	m	n	o
Character Class	48	48	48	48	48	48	48	48
Position	112	113	114	115	116	117	118	119
Character	p	q	r	s	t	u	v	w
Character Class	48	48	48	48	48	48	48	48
Position	120	121	122	123	124	125	126	127
Character	x	y	z	{	³	}	~	DEL
Character Class	48	48	48	123	124	125	126	1

Table 54: charClass resource, default character classes

curses
Class: Curses

See option "-cu"

Causes leading tabs at the beginning of continuation lines, which occur when line breaks are inserted into lines which are too long, to be suppressed. (See the CES manual for further information on the cursor motion package *curses*)

Default:

False
leading tabs are output.

cursorColor
Class: Foreground
See option " -cr color"
Determines the color of the text cursor.

Default:
black

font
Class: Font
See option "-fn font_name"
Specifies the font to be used as the basic character style for text in windows and icons.

Default:
vtsingle

foreground
Class: Foreground
See option "-fg color"
Window foreground color.
The foreground color is the color for text output.

Default:
black
You can customize this resource specifically for the *scrollbar* widget by specifying "xterm**scrollbar**foreground". This determines the foreground color of the scroll bar.
In the scrollbar widget:
specifies the foreground color of the scroll bar. The highlighted area, which indicates the relative amount and position of the text displayed in the window, is created using a checkerboard pattern for which pixels are alternated in the foreground and background colors.

Default:
white

geometry
Class: Geometry
See option "-geometry width x height + x_off + y_off"
Defines the position and size of the window.
The value specified for this resource may be ignored by the window manager.

Default:
80x25+0+0

height
Class: Height
specifies the height of the Tektronix window in pixels.

Default:

565 + 2 x border (pixels)

iconGeometry

Class: IconGeometry

Specifies the size and position of the window's icon in pixels.
This resource is only effective if the icons are not managed in the icon box. The window manager can ignore these specifications.
No default value.

internalBorder

Class: BorderWidth

See option "-b number"
Specifies the width of the area between the text margin and the window border in pixels.

Default:

2 (pixels)

jumpScroll

Class: JumpScroll

See option "-j"
Specifies whether or not the text in the terminal emulator window is to be scrolled by line or in blocks where necessary (jump scrolling).

Default:

False
Text is scrolled line by line.

logFile

Class: LogFile

See "logging", "logInhibit" resources
See option "-lf char_string"
Specifies the file to which a terminal emulator writes a session log.
This resource only takes effect if the logging resource (class: Logging) is assigned the value "True".
The file is created in the directory from which the emulator was started.

Default:

XtermLog.*string*
string is an extension to the file, making it unique in the directory in which the file is stored.

logging

Class: Logging

See "logFile", "logInhibit" resources
See option "-l"
Specifies whether or not a terminal emulator logs the session.
If the value is "True", the emulator writes the log to the file specified by the logFile resource instance (class: LogFile).

Default:
False
The emulator does not log the terminal session.

logInhibit
Class: LogInhibit
See "logging", "logFile" resources
Specifies whether the menu item for the logging function is locked.

Default:
False
Logging is not inhibited.

loginShell
Class: LoginShell
See option "-ls"
Specifies whether the shell started in the emulator window is a login shell.

Default:
False
No login shell is started in the emulator window.

marginBell
Class: MarginBell
See option "-mb"
See "nMarginBell" resource
Specifies whether the emulator triggers an acoustic signal when the text cursor reaches the text column specified in the nMarginBell resource instance (class: Column).

Default:
False
The margin bell is not enabled.

menuBorder
Class: MenuBorder
Specifies the width of the border surrounding menus in pixels.

Default:
2 (pixels)

menuFont
Class: Font
See "font" resource
Specifies the name of the font to be used for menu items.

Default:
font for the basic character style

menuPad
Class: MenuPad
Specifies the distance between menu items and the menu border in pixels.

Default:
3 (pixels)

multiClickTime
Class: MultiClickTime
See option "-mc milliseconds "
Specifies the maximum time (in milliseconds) that may elapse between 2 mouse clicks for the two clicks to be interpreted as a multiple click.

Default:
250 (milliseconds)

multiScroll
Class: MultiScroll
See option " -s"
Specifies whether text in the emulator window can be scrolled asynchronously.

Default:
False
Text is scrolled synchronously.

nMarginBell
Class: Column
See "marginBell" resource
See option "-nb number"
Specifies the column at which the margin bell is to ring.
This resource is only available if the marginBell resource (class: MarginBell) is assigned the value "True".

Default:
10

pointerColor
Class: Foreground
See option "-ms color"
See "foreground" resource
Specifies the pointer color.

Default:
black

pointerShape
Class: Cursor
Specifies the shape of the pointer.

Default:
xterm (I-beam pointer)

reverseVideo
Class: ReverseVideo

See option "-rv"
Specifies whether the colors in the client window are to be inverted.

Default:
False
The colors are not inverted.

reverseWrap
Class: ReverseWrap
See option "-rw"
Specifies whether the reverse wraparound function is enabled. Reverse wraparound allows the cursor to back up from the leftmost column of one line to the rightmost column of the previous line.

Default:
False
Moving backwards from the left to the right margin is not possible.

saveLines
Class: SaveLines
See option "-sl number"
Specifies the number of lines that are to be saved when there is a scroll bar.

Default:
64 (lines)

scrollBar
Class: ScrollBar
See "saveLines" resource
See option "-sb"
Specifies whether a vertical scroll bar is to be displayed in the emulator window.

Default:
False
The emulator window does not have a scroll bar.

scrollInput
Class: ScrollCond
See option "-si"
Specifies whether the window is positioned to the end of the scroll area (= end of text) for output to the window if the end of the text was previously not visible.

Default:
True
The window is positioned to the end of the text.

scrollKey
Class: ScrollCond
See option "-sk"
Specifies whether pressing a key causes the window with the input focus is to be scrolled forward to the end of the text if the latter was previously not visible.

Default:
False
The window is not positioned to the end of the text for keyboard input.

signalInhibit
Class: SignalInhibit
Specifies whether the xterm X11 menu inhibits the sending of signals to the process group of the shell and whether the associated terminal emulator menu items are locked.

Default:
False
The menu items for sending signals to the process group of the shell are selectable.

sunFunctionKeys
Class: SunFunctionKeys
See option "-sf"
specifies whether Sun escape codes are generated for the function keys.

Default:
False
Sun escape codes are not generated.

tekGeometry
Class: Geometry
specifies the preferred size (*width*x*height*) and position (*x_off*+*y_off*) of the Tektronix 4014 window.
Default:

The value specified for this resource may be ignored by the window manager.

width 750 + 2 x border (pixels)
height
565 + 2 x border (pixels)

tekInhibit
Class: TekInhibit
specifies whether switching to the Tektronix emulator window can be inhibited, i.e. whether the corresponding menu item is locked.

Default:
False
switching between VT102 and Tektronix emulator is not inhibited

tekStartup
Class: TekStartup
See option " -t"
specifies whether the Tektronix emulator is started by the xterm call

Default:

False

the VT102 emulator is started, not the Tektronix emulator.

termName

Class: TermName

See option "-tn name"

This is the name of the terminal assigned to the TERM environment variable. This name must be in the /usr/lib/terminfo database.

Default:

See section "Appearance and behavior ", subsection "VT102 terminal emulator"

thickness

Class: Thickness

specifies the width of the scroll bar in pixels.

Default:

24 (pixels)

title

Class:Title

See option "-title char_string"

Specifies the name of the emulator window, which may be displayed by the window manager in the title bar.

Default:

xterm

translations

Class: Translations

specifies the key and button bindings for menus, selection functions and programmed strings

For more information on the syntax for the translations resource refer to the section " translations ".

Default:

<KeyPress>:	insert()	\ n \
~Meta <BtnUp>:	select-end(PRIMARY, CUT_BUFFER0)	\ n \
<BtnDown>:	bell(0)	
Ctrl ~Meta <Btn1Down>:	mode-menu()	\ n \

~Meta <Btn1Down>:	select-start()	\ n \
~Meta <Btn1Motion>:	select-extend()	\ n \
Ctrl ~Meta <Btn2Down>:	mode-menu()	\ n \
~Ctrl ~Meta <Btn2Down>:	ignore()	\ n \
~Meta <Btn2Up>:	insert-selection(PRIMARY, CUT_BUFFER0)	\ n \
~Ctrl ~Meta <Btn3Down>:	start-extend()	\ n \
~Meta <Btn3Motion>:	select-extend()	\ n \

Table 55: Default: translations

ttyModes
Class: TtyModes
See option "-tm " assignment [assignment ...]""
Allows you to set terminal functions for a terminal emulator analogously to the stty command. If the -L option is specified in the xterm call, the program ignores any ttyModes specifications.
You can specify one or more assignments for *value*. Specifications consist of a keyword for the terminal attribute and the desired value.

utmpInhibit
Class: UtmpInhibit
See option "-ut"
Specifies whether the login entry for the emulator process is entered in the system log file /etc/utmp.

Default:
False
The emulator generates an entry.

visualBell
Class: VisualBell
See option "-vb"
Specifies whether the emulator generates a visual warning rather than an acoustic signal when CTRL-G is received.
The window contents are displayed briefly in reverse video.

Default:
False
An acoustic warning is sounded.

width
Class: Width
specifies the width of the Tektronix window (in pixels).

Default:
750 + 2 x border (pixels)

xnlLanguage
Class: XnlLanguage
Specifies the client's locale.

Default:
value of the LANG environment variable

2.20.2 Appearance and behavior

VT102 terminal emulator: The client xterm emulates almost all the characteristics of the terminal VT102, with the following exceptions:
- the blinking function
- double-width and double-size character sets
- the termcap database (the /etc/terminfo database instead)
- the TERMCAP environment variable

Tektronix 4014 terminal emulator: The client xterm supports four different font sizes and five different line types. Tektronix text and graphics commands are recorded internally by xterm and may be written to a file by sending the COPY escape sequence or by means of the "Tektronix" menu; the name of the file indicates when the COPY escape sequence was executed:
`COPY`*`year-month-day.hours:minutes::seconds`*
This file is created in the directory from which xterm was started or, if a login shell is started in the xterm window, in the home directory.
The following sections describe the appearance and behavior of the default configuration of the terminal emulator with regard to escape sequences, the Tektronix-GIN mode, and the supported terminfo entries. The terminal emulator supports cut-and-paste functionality, as described in the section "Copying text ". You will find information on how to configure the optional scroll bars in the section "Scroll bars ". For more information on the terminal emulator menus, please refer to the section "Menus".

2.20.2.1 Terminal emulator configuration

Tektronix GIN mode: In this mode the shape of the pointer changes from an arrow into a cross. Pressing any key transmits the keycode and the coordinates of the hot spot, i.e. the reference point of the pointer. Pressing button 1, 2 or 3 will return the letters "l", "m" and "r", respectively. If the Shift key is pressed at the same time as one of the buttons, the corresponding uppercase letter is transmitted.
The highest order bit is used to distinguish between key and button codes. This bit, however, is normally truncated unless the terminal mode is set to RAW. For more

information on RAW mode please refer to the stty command description in the operating system commands manual.
Terminfo entries: The terminal emulator searches the terminfo database for the entries "xterm", "vt102", "vt100", "ansi" and "dumb" and sets the TERM environment variable to the first entry found.

2.20.2.2 Copying text

Using the mouse or puck buttons, text in an emulator window can be selected and copied either within the same window or into another emulator window. You use button 1 to select and copy text to a text buffer and button 2 to copy this text from the buffer to the cursor position in the active window. Please note that it is possible to change the binding of buttons to actions.
Selecting text: Position the pointer at the beginning of the text, then press button 1. Hold down button 1 and move the pointer to the end of the text to be selected. The selected text is highlighted. As soon as you release button 1, the terminal emulator copies the selected text to a global text buffer (referred to as the cut buffer).
Loading excessively large amounts of data into the buffer can cause a deadlock. If necessary, the terminal emulator displays an error message several times before you can continue. Note that the data in the text buffer may not be complete!
Copying text to a cut buffer makes it the primary selection. The cut buffer can be referenced via the name CUT_BUFFER0 or BPRIMARY.
Pasting text: Position the text cursor at the location where the selected text is to be pasted. Click button 2 to start copying the text from the global text buffer to the cursor position. The terminal emulator treats this text like entries made using the keyboard.
Speeding up text selection: With a double click of button 1 you enable word-by-word text selection, with a triple click line-by-line selection. Clicking four times causes a reset to character-by-character selection. You set the maximum time between clicks with the multiClickTime resource (class: MultiClickTime).
Line-by-line selection, i.e. selection following a triple click, can be affected by the -cb option or cutToBeginningOfLine resource (see above).
Word-by-word selection, i.e. selection following a double click, takes into account the character classes defined using the charClass resource. It is especially useful to define your own character class if names made up of characters from a certain character set are to be selected as a whole.
For example, following the value assignment
*charClass: "33:48,37:48,45-47:48,64:48"
the characters !, %, -, ., / and & will be treated like alphabetic characters and figures.
Appending text: If you want to append text, you must not copy the last newline character into the text buffer when selecting the text. In line-by-line selection mode you can use the -cn option or the cutNewline resource to truncate the last newline character.
Changing the size of the selected area: You can use button 3 to change the size of the current selection. What effect the button has depends on the position of the cursor in relation to the edge of the selected text: the edge closest to the cursor is changed. If, for example, the cursor is closest to the right edge of the selection, the right edge of the selection is extended or reduced. If the cursor is outside of the selected area, the client assumes that you really meant the right or left edge, as appropriate, and restores the original selection, and then extends or reduces the selection by the selection unit last used. In other words, if selection was previously done word by word using button 1, the selection

is extended/ reduced by words. Multiple-clicking using button 3 changes the selection mode as described for button 1.

2.20.2.3 Scroll bars

You can use the -sb option or the scrollBar resource (class: ScrollBar) to specify whether or not the terminal emulator displays a scroll bar on the window border. It is assumed in the following section that a scroll bar is displayed.
The highlighted part of the scroll bar indicates which portion of the text is displayed in the window. It corresponds to the position in the text as a whole. Using the scroll bar you can redisplay text which has been scrolled off the top of the window:
Scrolling text: Position the pointer in the highlighted part of the scroll bar, press and hold down button 2 while moving the pointer in the scroll bar.
Positioning the text: Position the pointer above or below the highlighted part of the scroll bar at approximately the region that you want to see. Click Button 2.
Configuring scroll bars: You use the resources of the scrollbar widget (class: Scrollbar) to customize your scroll bar.

2.20.3 Key and button bindings

This section provides an overview of the key and button bindings that have been predefined for this client.
For more detailed information on how to use key and button bindings when utilizing events please refer to the chapter "Client resources", especially the description of the translations resource (class: Translations) in section " translations ".

2.20.3.1 Key bindings

It is possible to define how the vt100 and tek4014 widgets are to evaluate key bindings as strings.
Binding keys or key sequences to strings is known as defining a key binding.

2.20.3.2 Button bindings

In the VT100 widget, the key and button bindings for selecting text, pasting text and activating the menus are controlled by the button bindings.
Binding buttons or button sequences to strings is known as defining a button binding. For more detailed information please refer to the chapter "Client resources", especially the description of the translations resource (class: Translations) in section " translations ".

2.20.3.3 Example

You want to change the binding for keys F14, F17, F18, F19 and F20 when calling the command *command* in the VT100 widget. The appropriate value assignment for the translations resource is

*VT100.translations:		
	#override <Key>F13:	keymap(*kommando*)
*VT100.*kommando*Keymap.translations:		
	<Key>F14:	keymap(None)\n\

	<Key>F17:	string("next") string(0x0d)\n\
	<Key>F18:	string("step") string(0x0d)\n\
	<Key>F19:	string("continue") string(0x0d)\n\
	<Key>F20:	string("print ") insert-selection (PRIMARY, CUT_BUFFER0)

Table 56: Example of button bindings

2.20.3.4 Default bindings

<KeyPress>:	insert() \n\
~Meta <BtnUp>:	select-end(PRIMARY, CUT_BUFFER0) \n\
<BtnDown>:	bell(0)
Ctrl ~Meta <Btn1Down>:	mode-menu() \n\
~Meta <Btn1Down>:	select-start() \n\
~Meta <Btn1Motion>:	select-extend() \n\
Ctrl ~Meta <Btn2Down>:	mode-menu() \n\
~Ctrl ~Meta <Btn2Down>:	ignore() \n\
~Meta <Btn2Up>:	insert-selection(PRIMARY, CUT_BUFFER0) \n\
~Ctrl ~Meta <Btn3Down>:	start-extend() \n\
~Meta <Btn3Motion>:	select-extend() \n\

Table 57: Default bindings

2.20.4 Menus

The xterm terminal emulator comes with three menus: the "xterm X11" menu, the "Modes" menu and the "Tektronix" menu.
The menus contain options which you can use to control the terminal attributes (upper menu section), and menu items which you can use to start command functions (lower menu section). If you select an active option it is disabled; selecting a disabled option causes the attribute to be enabled.
Selecting a menu item in a popup menu: There are several ways to select a menu item:

- Position the pointer on the desired menu item and click button 3. The next section explains how to post a particular pop-up menu.
- Enter the character which is underlined in the menu item text (always use lowercase even if the underlined letter is uppercase). You must, however, first post the desired pop-up menu. This is explained in the next section.
- If the menu has been posted via an accelerator string, use the cursor keys to select the desired item. Please note that some menu items are not applicable in some situations and therefore cannot be selected. An activated menu item is selected by means of the **[Return]** key. Selecting an item closes the menu. Should you wish to close the menu without making a selection, press the **[Esc]** key.
- If the desired menu option has been assigned an accelerator string, enter this accelerator; the menu will not be displayed. If an accelerator has been defined for a menu item, you will find it to the right of that menu item.

The following section explains how you post a particular menu and gives you information on the structure, contents and meaning of the menu items.
Selecting the xterm X11 menu: Position the pointer in the xterm window; press the **[CTRL]** key and button 1 simultaneously; the xterm X11 menu is displayed. The menu includes entries for terminal attributes and items in the command part which you can select to send signals to the process group which runs under xterm. The menu items apply to the VT102 window as well as the Tektronix window.

Menu items	Meaning
Secure Keyboard	Secure keyboard
Allow SendEvents	Allow the sending of events
Log to File	Activate logging
Redraw Window	Redraw the window
Send STOP Signal	Send signal SIGSTP
Send CONT Signal	Send signal SIGCONT (useful if **[CTRL]-[Z]** pressed by accident, i.e. process is hung up)
Send INT Signal	Send signal SIGINT
Send HUP Signal	Send signal SIGHUP (shell and emulator are terminated)
Send TERM Signal	Send signal SIGTERM
Send KILL Signal	Send signal SIGKILL (shell and emulator are terminated)
Quit	Close window and terminate client

Table 58: The "xterm X11" menu of xterm

Selecting the Modes menu: Position the pointer in the VT102 window; press the CTRL key and button 2 simultaneously. The menu items apply only to the VT102 emulator

Menu items	Meaning
Enable Scrollbar	Make scroll bar visible
Enable Jump Scroll	Scroll in blocks (instead of line by line)
Enable Reverse Video	Foreground and background colors are inverted

Enable Auto Wraparound	Cursor moves automatically from the last column of a line to the first column of the same line
Enable Reverse Wraparound	Cursor moves backwards from the first column in a line to the last column of the previous line
Enable Auto Linefeed	Automatic linefeed
Enable Application Cursor Keys	Enable application cursor keys
Enable Application Keypad	Enable application keypad mode
Scroll to bottom on keypress	Scroll to end when key is pressed
Scroll to bottom on tty output	Scroll to end on output to screen
Allow 80/132 switching	Switching from 80-column mode to 132-column mode permitted
Enable Curses Emulation	Enable curses emulation
Enable Visual Bell	Visual signal generated shortly before end of line is reached
Enable Margin Bell	Acoustic signal generated shortly before end of line is reached
Show Alternate Screen	Alternate screen (disabled)
Do Soft Reset	Reset emulator's input parser to initial state. If escape sequences have already been detected, branch to initial state of parser. The rest of the escape sequence is handled in normal mode.
Do Full Reset	Reset terminal modes and functions to initial state after start of xterm. Any modifications made during the session are cancelled; e.g. tabulators are reset (to every 8 columns). Wrap and smooth scroll functions are reset; contents of window are deleted.

Show Tek Window	Open Tek window
Switch to Tek Mode	Switch to Tektronix mode
Hide VT Window	Hide VT window

Table 59: The "Modes" menu of xterm

Selecting the Tektronix menu: Position the pointer in the Tektronix emulator window; press the CTRL key and button 2 simultaneously. The menu items apply only to Tektronix 4014 emulator:

Menu items	**Meaning**
Large Characters	Large font
#2 Size Characters	Font size #2
#3 Size Characters	Font size #3
Small Characters	Small font
PAGE	Page
RESET	Reset
COPY	Copy
Show VT Window	Open VT window
Switch to VT Mode	Switch to VT mode
Hide Tek Window	Hide Tektronix window

Table 60: The "Tektronix" menu of xterm

Configuring menus: The configuration of menus is affected by the resources of the menu widget (widget class: Menu).

Example:

XTerm*menu*background: green
Means that every client that belongs to the Xterm client class is to use the value "green" for the background resource in the menu "menu".

2.20.5 Files

The files below are listed in alphabetical order. The resource configuration files are listed only in the form of the generalized file names $XENVIRONMENT, $XUSERFILESEARCHPATH and $XFILESEARCHPATH.

/etc/utmp
: system log file in which the terminal emulator creates a login entry when the -ut option is set.

$HOME/.login
: shell script executed when a login shell is started in the emulator window.

$HOME/.profile
: shell script executed when a login Bourne shell is started in the emulator window. You can, for example, define Bourne shell functions here.

$HOME/.Xdefaults
: user-specific configuration file

$XAUTHORITY
: user-specific authorization file
The client uses this file as a user-specific authorization file. If XAUTHORITY is not set, the client uses the .Xauthority file in the user's HOME directory.

$XENVIRONMENT
: user-specific configuration file

$XFILESEARCHPATH
: application-specific configuration file created by the client programmer

$XUSERFILESEARCHPATH
: application-specific configuration file created by the system administrator or user

XtermLog.*n*
: default log file for the terminal emulator xterm in the directory from which the terminal emulator is called

2.20.6 Environment variables

Detailed descriptions of all the environment variables, arranged in alphabetical order, are given in the chapter "Environment variables ".

DISPLAY
: The client reads the DISPLAY environment variable if the -display option is not specified in its command line. From this variable the client finds out the name of the X server and the display and screen on it.
The DISPLAY environment variable is a string in the format:
hostname:*display_number*.*screen_number*
The client terminates abnormally if the DISPLAY environment variable is not set and the -display option is not specified in the client command line.

HOMEidentifies the user's HOME directory.
: The client searches this directory for files.

TERMdenotes the type of terminal emulated.
: The terminal emulator sets the TERM variable according to the -tn option, the termName resource or the first entry in the terminfo database.

WINDOWID
: The terminal emulator sets the contents of the WINDOWID variable to the number of the window, which is unique in the context of the root window.

XAUTHORITY
: denotes the user-specific authorization file.

If the environment variable is not set, the client searches for the $HOME/.Xauthority file.

XFILESEARCHPATH
: specifies the path for configuration files.

XUSERFILESEARCHPATH
: specifies the path for configuration files.

2.21 x97801-480 - 97801-480 terminal emulator

This terminal emulator allows you to run programs that can only use alphanumeric devices of type 97801-480.
x97801-480[*option*]...
In cases where an option has both -*option* and +*option*, +*option* is usually used to reset the attribute specified by -*option* to its default value.

-ah ah - always highlight
See option "+ah"
See "alwaysHighlight" resource
This option causes the cursor to always be highlighted, even if the input focus is assigned to a different window.

Default:
Only a hollow cursor is displayed when the window loses the input focus.

+ah causes the cursor to be highlighted only if the input focus is assigned to the emulator window.

-b *number*
See "internalBorder" resource
number specifies (in pixels) the width of the internal border.
The internal border is the area between the text margin and the window border.

Default:
2 (pixels)

-bd *color*
bd - border color
See "borderColor" resource
color is the color of the window border.

Default:
black

-bg *color*
bg - background
See "background" resource
color is the color of the window background.

Default:
white

-bw *number*
bw - border width
See "borderWidth" resource
number is the width of the window border in pixels.
The window manager may ignore or adjust this value.

Default:
2 (pixels)

-cb cb - cut begin of line inhibit
See "cutToBeginningOfLine" resource
Affects line-by-line selection of text.
Text is copied to the buffer from the pointer position, not from the beginning of the line where the pointer is located.

Default:
The emulator copies text to the buffer from the beginning of the line.

+cb The terminal emulator copies text to the buffer from the beginning of the line.

-cc *position*:*index*[,...]
See "charClass" resource
Specifies how the terminal emulator handles characters when text is selected word by word.

position
identifies the characters which are to be grouped to form a character class in a terminal emulator. *position* can be a number or a value range in the form $number_1$-$number_2$.

index specifies the character class and thus how the characters selected by *position* are handled.

Default:
Default character class

-cn cn - cut newline inhibit
See "cutNewline" resource
Affects line-by-line selection of text.
The terminating newline character of the last line selected is not copied to the cut buffer.

Default:
The last line is also terminated by a newline character.

+cn When selecting text line by line, each line, including the last, is terminated by a newline character.

-cr *color*
See "cursorColor" resource
color is the color of the text cursor when the window has the input focus.

Default:
black

-cs8 See "eightbit" resource
Starts the terminal emulator in 7-bit mode.

Default:
The terminal emulator is started in 8-bit mode.

+cs8 Starts the terminal emulator in 8-bit mode.

-display [*hostname*]:*display_number*[.*screen_number*]

Identifies the X server, display and screen which the client is to contact.

hostname

symbolic name of the host or X terminal on which the X server which the client is to contact runs. *hostname* must be known in the transport protocol of the host on which the client resides.

display_number

number (unique to the X server) identifying the display for which the client is to manage I/O.

.screen_number

number (unique to the display) identifying the screen to which the client is to send output.

Default:

contents of the DISPLAY shell variable;
if DISPLAY is not set, the client terminates abnormally.

-e *command*

command is called in the emulator window instead of the shell. If you do not specify the -title option in the emulator call, the emulator uses the command name as the window and the icon name.
This option may only be specified as the last argument of the terminal emulator command line.

Default:

a shell is started in the terminal emulator window.

-fg *color*

fg - foreground
See "foreground" resource
color is the color of the window foreground.
This is the color in which text characters are displayed.

Default:

black

-g0 *char_set*

See option "-g1 char_set"
Causes the character set specified as *char_set* to be loaded into the loadable half of the G0 allocation area.
char_set can be replaced by:

char_set	Character set
INT	International
INTA	International A
GER	German
FAC	Facet

EURO	Euro
IBM	IBM
ISO1	ISO 8859-1, upper half
ISO2	ISO 8859-2, upper half
ISO3	ISO 8859-3, upper half
ISO4	ISO 8859-4, upper half
ISO5	ISO 8859-5, upper half
ISO7	ISO 8859-7, upper half
ISO9	ISO 8859-9, upper half
KLA	Brackets
MAT	Math

Default:
INTA

-g1 *char_set*
See option "-g0 char_set"
Causes the character set specified as *char_set* to be loaded into the loadable half of the G1 allocation area.
char_set can be replaced by:

char_set	Character set
INT	International
INTA	International A
GER	German
FAC	Facet
EURO	Euro
IBM	IBM
ISO1	ISO 8859-1, upper half
ISO2	ISO 8859-2, upper half
ISO3	ISO 8859-3, upper half
ISO4	ISO 8859-4, upper half
ISO5	ISO 8859-5, upper half
ISO7	ISO 8859-7, upper half
ISO9	ISO 8859-9, upper half
KLA	Brackets
MAT	Math

Default:
KLA

-geometry *width*x*height*+*x_off*+*y_off*
See "geometry" resource
Defines the size and position of the client window.
Specifications for the window size (*width*x*height*) are not evaluated: emulator windows always have a fixed size of 80 columns by 25 lines. You can use the escape sequences described in the operating system interfaces documentation to switch between 24 and 25 line mode. The terminal emulator is started in 24 line mode, with a blank 25th line. *x_off* and *y_off* determine the window position.

width the window width.
height
the window height.
x_off Displacement of the window along the x axis (horizontally) in pixels

+*x_off*
distance between left edge of window and screen
-*x_off*
distance between right edge of window and screen
y_off Displacement of the window along the y axis (vertically) in pixels

+*y_off*
distance between top edge of window and screen
-*y_off*
distance between bottom edge of window and screen

Default:
80x25+0+0
-help Writes a short description of the options to standard error.
This help text is also displayed if the client cannot interpret a command-line argument.
-iconic
Indicates that the mwm window manager is to start the client as an icon, not as a normal window.

Default:
The mwm window manager opens the normal window of the client.
-j j - jump scrolling
See "jumpScroll" resource
See option " -s"
See Mode menu (table "The "Mode" menu of x97801-480")
Text within the window is scrolled in blocks, not line by line.
Jump scrolling means that several lines are moved simultaneously out of the display at the upper edge of the screen. This speeds up scrolling through large amounts of data significantly, since it reduces the load on both the terminal emulator and the X server.

Default:

text in the emulator window is scrolled line by line.

+j The text in the emulator window is scrolled line by line.

-l l - logging
See "logging" resource
See option "-lf char_string"
Causes the emulator to keep a log.
The terminal emulator directs all output, including error output, both to the log file and to the screen.
You can use the -lf option to set up a log file of your own; if you do not do so, the terminal emulator uses the log file X97Log.*string*. *string* is an extension to the file name which makes it unique within the directory in which the file is stored. If you do not specify an absolute pathname, the log file is created in the directory where the emulator was started, or in the $HOME directory if you use the terminal emulator to start a login shell.

Default:
The terminal emulator does not keep a log.

+l The terminal emulator does not keep a log.

-lf *char_string*
lf - logfile
See "logFile" resource
See option "-l"
For *char_string* you can specify a filename *file* or set up a pipe to the command (in this case, the first character of *char_string* is the pipe symbol: |*command*).

-lf *file*
The emulator uses *file* as the log file instead of the default log file. *file* is created in the directory where the emulator was started.

-lf- |*command*
A pipe to *command* is created and *command* is executed by the shell.

Default:
X97Log.*string*
string is an extension to the file name which makes it unique within the directory in which the file is stored.

-ls ls - login shell
See "loginShell" resource
The terminal emulator starts a login shell and executes the .profile or .login file. The initial working directory of the login shell is the directory in which the terminal emulator is started.

Default:
the terminal emulator starts a subshell which is not a login shell.

+ls The terminal emulator starts a subshell which is not a login shell.

-mb mb - margin bell
See "marginBell" resource
See option "-nb number"

Enables the margin bell for the right text margin.
The margin bell sounds when the cursor reaches the text column which was specified using the -nb option or the corresponding resource.

Default:
the terminal emulator does not sound a margin bell.

+mb The terminal emulator does not sound a margin bell.

-mc milliseconds
mc - multi-click time
See "multiClickTime" resource
Defines the maximum time in milliseconds that may elapse between 2 mouse clicks for the two clicks to be interpreted as a multiple click.

Default:
250 (milliseconds)

-ms *color*
ms - mouse pointer color
See "pointerColor" resource
See option "-fg color"
color determines the color of the pointer.

Default:
black

-name *resource_link*
resource_link is the name of the application under which the client's window resources are to be found.
The client can thus be called under different names and utilize the resources for its client instance *resource_link*, yet still utilize the same resources for its client class *clientclass*.
resource_link should not contain dots (.) or asterisks (*), as these symbols have special meanings. Please refer to the section "Value assignment syntax " for more information.

Default:
x97801-480

-nb *number*
See option "-mb"
See resources "nMarginBell", "marginBell"
Specifies the column at which the margin bell is sounded.
number is the offset in columns from the right text margin. The margin bell is only sounded when enabled via the -mb option, the "Modes" menu or the marginBell resource.

Default:
10

-rv rv - reverse video

See "reverseVideo" resource
Inverts the foreground and background colors.

Default:
Foreground and background colors are not inverted.

-rw rw - reverse wraparound
See "reverseWrap" resource
See Mode menu (table "The "Mode" menu of x97801-480")
Enables the reverse wraparound function.
Reverse wraparound allows the cursor to back up from the left text margin (column 1) to the last column (right text margin) of the previous line. This is especially useful when editing long command lines.

Default:
the emulator does not support reverse wraparound.

+rw The text cursor cannot be backed up beyond the left-hand text margin.

-S*filedescriptor filename*
The terminal emulator is started in slave mode.
The terminal emulator uses a pseudo terminal for input and output. This pseudo terminal must have been allocated by the master which starts the emulator in slave mode, since the emulator does not perform any allocations in slave mode.
In slave mode, it is not possible to send signals to the process group of the terminal emulator via the menu.

filedescriptor
is the decimal number of the file descriptor which the master process used to open the master side of the pseudo terminal.
filename
is the absolute pathname of the slave side of the pseudo terminal used by the terminal emulator for input and output.
No default value.

-s Allows text to be scrolled asynchronously.
Positioning by scrolling large amounts of text is speeded up; during scrolling, the text visible in the window does not correspond to the scroll bar position.
-s is very useful when running the terminal emulator in extensive networks with long latency times.
Precondition: there must be a scroll bar.

Default:
synchronous scrolling

+s The emulator scrolls synchronously.
During scrolling, the text visible in the window corresponds to the scroll bar position.

-sb sb - scroll bar
See "saveLines" resource
See -sl option
The terminal emulator displays a vertical scroll bar.

The emulator saves the lines which are scrolled off the top of the window. The scroll bar can be used to restore scrolled lines to the window. The highlighted part of the scroll bar also shows what part of the whole text is being displayed within the window. You can use the -sl option or the saveLines resource to specify how many of the lines which have been scrolled off the window are saved.

Default:

no scroll bar

+sb No scroll bar is displayed.

-sf *filename*

sf - softkey file
See "softKeyFile" resource
Gives the name of the file containing the softkey definitions which are to be implemented when the terminal emulator is invoked.

Default:

$HOME/.X97SoftKeys

-si si - scroll to bottom on output inhibit
See option "-sb"
See "scrollInput" resource
When output is sent to the emulator window, this option inhibits positioning to the end of the scroll area unless it was already visible.
Precondition: There must be a scroll bar.

Default:

The emulator window is positioned to the end of the scroll area for output.

+s When output is sent to the emulator window, the window is positioned to the end of the scroll area.

-sk sk - scroll to bottom on key
See option "-sb"
See "scrollKey" resource
On input from the keyboard, the emulator window is positioned to the end of the scroll area (= end of text).
Precondition: There must be a scroll bar.

Default:

Positioning to the end of text is inhibited for keyboard input.

+sk On input from the keyboard, the window is not repositioned to the end of the scroll area (= end of text) if text has previously been scrolled down out of the window.

-sl *number*

sl - save lines
See "saveLines" resource
number specifies how many lines the emulator is to save.

Default:

24 (lines)

-title *char_string*

See option "-e command"
See "title" resource
char_string is the name which the mwm window manager displays in the title bar of the emulator window.

Default:

x97801-480 or the command name from the -e option.

-tm "*assignment*[*assignment*...]"

tm - tty modes
See "ttyModes" resource
This resource sets terminal driver parameters.
assignment is a value assignment for terminal attributes.

assignment

Each *assignment* consists of a keyword and the value for the terminal attribute in the form of *keyword value*; *value* is always exactly one character long.
Valid keywords: intr, quit, erase, kill, eof, eol
Control characters can be used with a leading ^ character; use ^? to specify the delete function.

Defaults:

intr	Delete
quit	^\
erase	^h
kill	^x
eof	^d
eol	^@

-tn *name*

See "termName" resource
name is the identifier of the terminal type which the terminal emulator assigns to the TERM variable.
The terminal type *name* must be defined in the terminfo database.

Default:

depends on which terminfo entries are supported.

-ut See "utmpInhibit" resource
The terminal emulator does not create a login entry in the system log file /etc/utmp.

Default:

The emulator creates a login entry.

+ut The emulator creates a login entry in the /etc/utmp file, provided that "root" is the owner of the emulator program file and that the "set-user-id" bit is set.

-vb vb - visual bell

See "visualBell" resource
On receiving CTRL-G the terminal emulator generates a visual warning instead of an acoustic one.
Window contents and background are displayed briefly in reverse video.

Default:
CTRL-G causes the terminal emulator to sound an acoustic warning.

+vb On receiving CTRL-G the terminal emulator sounds an acoustic warning.

-xrm "*resourceDefinition*:*value*"
Assigns a value to a client resource.
-xrm must be specified separately for each resource to be set in the client call. The "*resourceDefinition*:*value*" assignment is compared with the entries in the configuration files. For many resources there are corresponding command-line options. To address resources for which there are no specific options available in the client call you have to use the -xrm option.

resourceDefinition
Identifies the resource to which a value is to be assigned. For *resourceDefinition* you can give the name of the resource, the name of the client and of any widgets in the appropriate resource syntax (see section "Value assignment syntax ").

value Value to be assigned to the resource.

2.21.1 Resources

You can configure the client using its resource database. To do this, you must assign the required values to the resources. You can use either configuration files or the -xrm option in the client call to access resources.
The instance name of the client is x97801-480 and its class name X97801-480. Please note that you can use the -name *instancename* option to change the instance name of the client when you call it.

2.21.1.1 Resource overview

This subsection starts with overviews showing which resources can be used to configure which widgets, followed by brief descriptions of the resources listed alphabetically according to resource name.
The resources described in the resource list below are resources of the x97801-480 widget (class: X97801-480).

Resource	Value	Default
background	pixel	-
backgroundPixmap	pixmap	-
borderColor	pixel	-
bottomShadowC	pixel	-

olor		
bottomShadowPixmap	pixmap	-
foreground	pixel	-
topShadowColor	pixel	-
topShadowPixmap	pixmap	-
translations	string	-

Table 64: x97801-480 resources for the widgets: "vtmenubar" (class: XmRowColumn), "screen" (class: XmRowColumn), "mode" (class: XmRowColumn), "x97801" (class: XmRowColumn), "keyboard" (class: XmRowColumn)

Resource	Value	Default
background	pixel	-
backgroundPixmap	pixmap	-
borderColor	pixel	-
borderWidth	dimension	0
bottomShadowColor	pixel	-
bottomShadowPixmap	pixmap	-
foreground	pixel	-
topShadowColor	pixel	-
topShadowPixmap	pixmap	-

Table 65: x97801-480 resources for the widget vtscrollbar (class: XmScrollBar)

Resource	Value	Default
background	pixel	-
backgroundPixmap	pixmap	-
borderColor	pixel	-

Table 66: x97801-480 resources for the widget key programming (class: XmDialogShell)

2.21.1.2 Resource list

The resources available to this client are listed in alphabetical order below. You can also use them in the -xrm option.

allowSendEvents
Class: AllowSendEvents
Defines whether or not the terminal emulator evaluates key or button events sent to it by other clients.

Default:
False
Events sent by other clients are not evaluated.

alwaysHighlight
Class: AlwaysHighlight
See option "-ah"
Defines whether or not the text cursor is always highlighted, even if the input focus is assigned to a different window.

Default:
False
The cursor is only highlighted in the active window.

background
Class: Background
See option "-bg color"
Determines the color of the window background.

Default:
white

backgroundPixmap
Class: BackgroundPixmap
Specifies the pixmap to be used for the background of a window

Default:
depends on the visual class of the screen

blinkColor
Class: Foreground
See option -bc
Determines the color of "blinking" text.

Default:
red

blinkCursor
Class: Blinking
See "cursorBlinkTime" resource
Specifies whether or not the text cursor blinks.

Default:
True
The cursor blinks.

blockCursor
Class: BlockCursor
Specifies whether the text cursor appears as a rectangular block the size of a character or as an underscore the width of a character.

Default:
False
The text cursor appears as an underscore.

borderColor
Class: BorderColor
Specifies the color of the window border in the client area.

borderWidth
Class: BorderWidth
See option "-bw number"
Specifies the width of the client window border in pixels.

Default:
2 pixels

bottomShadowColor
Class: Foreground
Specifies the color for the bottom and right-hand 3-D window shadows.

Default:
depends on the visual class of the screen

bottomShadowPixmap
Class: BottomShadowPixmap
Specifies the pixmap to be used for the bottom shadow of windows.

Default:
depends on the visual class of the screen

charClass
Class: CharClass
See option "-cc position : index [,...]"
Determines which characters are grouped together as a character class.
You form a character class by assigning all characters of the desired class a common numeric value. As common numeric value for all elements of the character class, you can choose the decimal value of the position of any element of the class.
Default:
default character classes

Position	0	1	2	3	4	5	6	7
Character	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL
Character Class	32	1	1	1	1	1	1	1

Position	8	9	10	11	12	13	14	15
Character	BS	HT	NL	VT	NP	CR	SO	SI
Character Class	1	32	1	1	1	1	1	1
Position	16	17	18	19	20	21	22	23
Character	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB
Character Class	1	1	1	1	1	1	1	1
Position	24	25	26	27	28	29	30	31
Character	CAN	EM	SUB	ESC	FS	GS	RS	US
Character Class	1	1	1	1	1	1	1	1
Position	32	33	34	35	36	37	38	39
Character	SP	!	"	#	$	%	&	'
Character Class	32	33	34	35	36	37	38	39
Position	40	41	42	43	44	45	46	47
Character	(	)	*	+	,	-	.	/
Character Class	40	41	42	43	44	45	46	47
Position	48	49	50	51	52	53	54	55
Character	0	1	2	3	4	5	6	7
Character Class	48	48	48	48	48	48	48	48
Position	56	57	58	59	60	61	62	63
Character	8	9	:	;	<	=	>	?
Character Class	48	48	58	59	60	61	62	63
Position	64	65	66	67	68	69	70	71
Character	@	A	B	C	D	E	F	G
Character Class	64	48	48	48	48	48	48	48
Position	72	73	74	75	76	77	78	79
Character	H	I	J	K	L	M	N	O
Character	4	4	4	4	4	4	4	4

Class	8	8	8	8	8	8	8	8
Position	80	81	82	83	84	85	86	87
Character	P	Q	R	S	T	U	V	W
Character Class	48	48	48	48	48	48	48	48
Position	88	89	90	91	92	93	94	95
Character	X	Y	Z	[	\	]	^	_
Character Class	48	48	48	91	92	93	94	48
Position	96	97	98	99	100	101	102	103
Character	`	a	b	c	d	e	f	g
Character Class	96	48	48	48	48	48	48	48
Position	104	105	106	107	108	109	110	111
Character	h	i	j	k	l	m	n	o
Character Class	48	48	48	48	48	48	48	48
Position	112	113	114	115	116	117	118	119
Character	p	q	r	s	t	u	v	w
Character Class	48	48	48	48	48	48	48	48
Position	120	121	122	123	124	125	126	127
Character	x	y	z	{	³	}	~	DEL
Character Class	48	48	48	123	124	125	126	1
Position	128	129	130	131	132	133	134	135
Character	h80	h81	h82	h83	IND	NEL	h86	h87
Character Class	1	1	1	1	1	1	1	1
Position	136	137	138	139	140	141	142	143
Character	HTS	h89	h8a	h8b	h8c	RI	SS2	SS3

Character Class	1	32	1	1	1	1	1	1
Position	144	145	146	147	148	149	150	151
Character	DCS	h91	h92	h93	h94	h95	h96	h97
Character Class	1	1	1	1	1	1	1	1
Position	152	153	154	155	156	157	158	159
Character	h98	h99	h9a	CSI	ST	h9d	h9e	h9f
Character Class	1	1	1	1	1	1	1	1
Position	160	161	162	163	164	165	166	167
Character	SP	!	"	#	$	%	&	'
Character Class	0xa0	0xa1	0xa2	0xa3	0xa4	0xa5	0xa6	0xa7
Position	168	169	170	171	172	173	174	175
Character	(	)	*	+	,	-	.	/
Character Class	0xa8	0xa9	0xaa	0xab	0xac	0xad	0xae	0xaf
Position	176	177	178	179	180	181	182	183
Character	0	1	2	3	4	5	6	7
Character Class	0xb0	0xb1	0xb2	0xb3	0xb4	0xb5	0xb6	0xb7
Position	184	185	186	187	188	189	190	191
Character	8	9	:	;	<	=	>	?
Character Class	0xb8	0xb9	0xba	0xbb	0xbc	0xbd	0xbe	0xbf
Position	192	193	194	195	196	197	198	199
Character	`A	'A	^A	~A	"A	oA	AE	C
Character	48	48	48	48	48	48	48	48

Class								
Position	200	201	202	203	204	205	206	207
Character	`E	'E	^E	"E	`I	'I	^I	"I
Character Class	48	48	48	48	48	48	48	48
Position	208	209	210	211	212	213	214	215
Character	D	~N	`O	'O	^O	~O	"O	x
Character Class	48	48	48	48	48	48	48	0xd7
Position	216	217	218	219	220	221	222	223
Character	0	`U	'U	^U	"U	'Y	P	ss
Character Class	0xd8	48	48	48	48	48	48	48
Position	224	225	226	227	228	229	230	231
Character	`a	'a	^a	~a	"a	oa	he6	he7
Character Class	48	48	48	48	48	48	48	48
Position	232	233	234	235	236	237	238	239
Character	`e	'e	^e	"e	`i	'i	^i	"i
Character Class	48	48	48	48	48	48	48	48
Position	240	241	242	243	244	245	246	247
Character	hf0	~n	`o	'o	^o	~o	"o	%
Character Class	48	48	48	48	48	48	48	0xf7
Position	248	249	250	251	252	253	254	255
Character	0	`u	'u	^u	"u	'y	P	"y
Character Class	0xf	48	48	48	48	48	48	48



Table 67: x97801-480 character class

compose
Class: Compose
See "eightbit" resource
Specifies whether or not compose sequences are interpreted.

Default:
True
Compose sequences are interpreted.

cursorColor
Class: Foreground
See option " -cr color"
Determines the color of the text cursor.

Default:
black

cursorBlinkTime
Class: BlinkTime
See "blinkCursor" resource
Specifies the period of time (in milliseconds) during which the blinking cursor is visible.

Default:
400 milliseconds

cutNewline
Class: CutNewline
See option "-cn"
Specifies whether every line including the last line is terminated by a newline character when text is copied line by line to the cut buffer.

Default:
True
The last line is terminated by a newline character.

cutToBeginningOfLine
Class: CutToBeginningOfLine
See option "-cb"
Affects line-by-line copying of text to the cut buffer and specifies whether the selected line is copied to the buffer completely or only from the cursor position.

Default:
True
Selected lines are copied in full.

deadKey
Class: Compose

See "eightbit" resource
Specifies whether or not dead key character sequences are interpreted.

Default:
True
Dead key character sequences are interpreted.

defaultVirtualBindings
Class: DefaultVirtualBindings
Specifies the mapping of virtual buttons and keys to your keyboard.

Default:
dependent on X server

doHalfIntensity
Class: DoHalfIntensity
Determines how text with the text attribute "reduced intensity" is displayed.

Default:
True

for 2-n pixel plane screens:
Text with the text attribute "half-intensity" appears dimmed.
for 1 pixel plane screens:
Text with the text attribute "half-intensity" is displayed against a patterned background.

eightbit
Class: Eightbit
See option "-cs8"
Specifies whether or not the terminal emulator is started in 8-bit mode.

Default:
True
The terminal emulator is started in 8-bit mode.

fontList
Class: FontList
See "fontMapping ", "w0" resources
Font list
The x97801-480 uses, at the most, the first 16 fonts in the font list to simulate the 97801-480 terminal's character generator.
Startup time is shorter if you use only one ISO font. When using the ISO 8859-1 character set you reduce the font list to:
-sni-97480i1-medium-ot-normal--16-*-*-*-c-*-iso8859-1=chset0,
-sni-97480m1-medium-ot-normal--16-*-*-*-c-*--=chset7,
-sni-97480m2-medium-ot-normal--16-*-*-*-c-*--=chset8

Default:

-sni-97480i1-medium-ot-normal--16-*-*-*-c-*-iso8859-1=chset0
-sni-97480i2-medium-ot-normal--16-*-*-*-c-*-iso8859-2=chset1
-sni-97480i3-medium-ot-normal--16-*-*-*-c-*-iso8859-3=chset2
-sni-97480i4-medium-ot-normal--16-*-*-*-c-*-iso8859-4=chset3
-sni-97480i5-medium-ot-normal--16-*-*-*-c-*-iso8859-5=chset4
-sni-97480i7-medium-ot-normal--16-*-*-*-c-*-iso8859-7=chset5
-sni-97480i9-medium-ot-normal--16-*-*-*-c-*-iso8859-9=chset6
-sni-97480m1-medium-ot-normal--16-*-*-*-c-*--=chset7
-sni-97480m2-medium-ot-normal--16-*-*-*-c-*--=chset8

fontMapping
Class: FontMapping
See "fontList" resource
Specifies the mapping of characters used by the 97801-480 terminal emulator's character generator to glyphs in the font with the alias *alias*.
You must allocate the alias *alias* a font in the fontList resource (class: FontList).

foreground
Class: Foreground
See option "-fg color"
Window foreground color.
The foreground color is the color for text output.

Default:
black

functionKeyMapping
Class: FunctionKeyMapping
Controls double mapping of the function keys.

Default:
False
The terminal emulator does not assign a special meaning to the action of pressing the <Control> key and function key F<number> at the same time.

g0
Class: G0
See "fontMapping ", "g1", "g2", "g3", "w0", "w1" resources
Determines the mapping of the G0 allocation area for character sets to a particular character set.

Default:
INTA
International A character set

g1
Class: G1
See "fontMapping ", "g0", "g2", "g3", "w0", "w1" resources
Determines the mapping of the G1 allocation area for character sets to a particular character set.

Default:
KLA
brackets character set

g2
Class: G2
See "fontMapping ", "g0", "g1", "g3", "w0", "w1" resources
Determines the mapping of the G2 allocation area for character sets to a particular character set.

Default:
BLA
blanks character set

g3
Class: G3
See "fontMapping ", "g0", "g1", "g2", "w0", "w1" resources
Determines the mapping of the G3 allocation area for character sets to a particular character set.

Default:
ISO1
ISO 8859-1 character set, upper half

geometry
Class: Geometry
See option "-geometry width x height + x_off + y_off"
Defines the position of the window.
The value specified for this resource may be ignored by the window manager.
The client does not evaluate window height and width specifications.

Default:
80x25+0+0

iconGeometry
Class: IconGeometry
Specifies the size and position of the window's icon in pixels.
This resource is only effective if the icons are not managed in the icon box. The window manager can ignore these specifications.
No default value.

initString
Class: InitString
Specifies the character string which is sent to the terminal emulator when the terminal emulator is called.
This resource enables you to load the DRCS area when you call the terminal emulator, as described in the operating instructions for the 97801-4xx monitor.
No default value

internalBorder
Class: BorderWidth
See option "-b number"
Specifies the width of the area between the text margin and the window border in

pixels.

Default:

2 (pixels)

jumpScroll
Class: JumpScroll

See option "-j"

Specifies whether or not the text in the terminal emulator window is to be scrolled by line or in blocks where necessary (jump scrolling).

Default:

False

Text is scrolled line by line.

keySymMapping
Class: KeySymMapping

Modifies a key-to-keysym mapping.

Default:

The terminal emulator selects the mapping defined by the X server.

logFile
Class: LogFile

See "logging", "logInhibit" resources

See option "-lf char_string"

Specifies the file to which the terminal emulator writes the log.

This resource only takes effect if the resource logging (class: Logging) is assigned the value "True".

The file is created in the directory from which the emulator was started.

Default:

X97Log.*string*

string is an extension to the file, making it unique in the directory in which the file is stored.

logging
Class: Logging

See "logFile", "logInhibit" resources

See option "-l"

Specifies whether or not a terminal emulator logs the session.

If the value is "True", the emulator writes the log to the file specified by the logFile resource (class: LogFile).

Default:

False

The emulator does not log the terminal session.

logInhibit
Class: LogInhibit

See "logging", "logFile" resources

Specifies whether the menu item for the logging function is locked.

Default:

False
Logging is not inhibited.

loginShell
Class: LoginShell

See option "-ls"
Specifies whether the shell started in the emulator window is a login shell.

Default:

False
No login shell is started in the emulator window.

marginBell
Class: MarginBell

See option "-mb"
Specifies whether the emulator triggers an acoustic signal when the text cursor reaches the text column specified in the nMarginBell resource instance (class: Column).

Default:

False
The margin bell is not enabled.

menuAccelerator
Class: Accelerators

defines a sequence of keystrokes as a menu bar accelerator.
Entering the defined keystroke sequence causes the first menu item on the menu bar to be selected.

Default:

KMenuBar

multiClickTime
Class: MultiClickTime

See option "-mc milliseconds"
Specifies the maximum time (in milliseconds) that may elapse between 2 mouse clicks for the two clicks to be interpreted as a multiple click.

Default:

250 (milliseconds)

multiScroll
Class: MultiScroll

See option " -s"
Specifies whether text in the emulator window can be scrolled asynchronously.

Default:

False
Text is scrolled synchronously.

nMarginBell
Class: Column
See "marginBell" resource
See option "-nb number"
Specifies the column at which the margin bell is to ring.
This resource is only available if the marginBell resource (class: MarginBell) is assigned the value "True".

Default:
10

noMenubar
Class: NoMenu
Specifies whether or not a menu bar is visible in the terminal emulator window.

Default:
False
The menu bar is visible.

noState
Class: NoState
Specifies whether or not the status line is visible in the terminal emulator window.

Default:
False
The status line is visible.

pointerColor
Class: Foreground
See option "-ms color"
See "foreground" resource
Specifies the pointer color.

Default:
black

pointerShape
Class: Cursor
Specifies the shape of the pointer.

Default:
xterm (I-beam pointer)

reverseVideo
Class: ReverseVideo
See option "-rv"
Specifies whether the colors in the client window are to be inverted.

Default:

False

The colors are not inverted.

reverseWrap

Class: ReverseWrap

See option "-rw"

Specifies whether the reverse wraparound function is enabled. Reverse wraparound allows the cursor to back up from the leftmost column of one line to the rightmost column of the previous line.

Default:

False

Moving backwards from the left to the right margin is not possible.

saveLines

Class: SaveLines

See option "-sl number"

Specifies the number of lines that are to be saved when there is a scroll bar.

Default:

24 (lines)

scrollBar

Class: ScrollBar

See "saveLines" resource

See option "-sb"

Specifies whether a vertical scroll bar is to be displayed in the emulator window.

Default:

False

The emulator window does not have a scroll bar.

scrollInput

Class: ScrollCond

See option "-si"

Specifies whether the window is positioned to the end of the scroll area (= end of text) for output to the window if the end of the text was previously not visible.

Default:

True

The window is positioned to the end of the text.

scrollKey

Class: ScrollCond

See option "-sk"

Specifies whether pressing a key causes the window with the input focus is to be scrolled forward to the end of the text if the latter was previously not visible.

Default:

False

The window is not positioned to the end of the text for keyboard input.

showSoftKeyboard
Class: ShowSoftKeyboard
See "softKeyboardConfiguration" resource
Specifies whether to display a soft (i.e. mouse-operated) keyboard.

Default:
False
The soft keyboard is not displayed.

signalInhibit
Class: SignalInhibit
Specifies whether the terminal emulator menu inhibits the sending of signals to the process group of the shell and whether the associated terminal emulator menu items are locked.

Default:
False
The menu items for sending signals to the process group of the shell are selectable.

softKey
Class: SoftKey
See "softKeyFile", "eightbit" resources
Specifies whether or not softkeys are interpreted.
This resource is only available if the eightbit resource (class: Eightbit) has been assigned the value "True".
You will find more detailed information in the " "Programming function keys "" subsection of this client description.

Default:
True
Softkeys are interpreted.

softKeyboardConfiguration
Class: SoftKeyboardConfiguration
See "showSoftKeyboard" resource
Defines the keys for the terminal emulator's soft keyboard.

Default:
The terminal emulator uses the keyboard window appropriate to the X server.

softKeyFile
Class: SoftKeyFile
See "softKey", "eightbit" resources
Specifies the name of the file whose entries define the softkeys when the client is called.

Default:
$HOME/.X97SoftKeys

termName
Class: TermName
See option "-tn name"
This is the name of the terminal assigned to the TERM environment variable.
This name must be in the /usr/lib/terminfo database.
title
Class:Title
See option "-title char_string"
Specifies the name of the emulator window, which may be displayed by the window manager in the title bar.

Default:
x97801-480
topShadowColor
Class: Background
Specifies the color to be used for the top shadow of the window frame

Default:
depends on the visual class of the screen
topShadowPixmap
Class: TopShadowPixmap
Pixmap of the top and left 3D shadow for windows

Default:
depends on the visual class of the screen
translations
Class: Translations
specifies the key and button bindings for menus, selection functions and programmed strings
For more information on the syntax for the translations resource refer to the section " translations ".

Default:

Ctrl<Key>Help:	**programming()**	**\ n **
Ctrl<Key>F21:	read-mode(ON)	\ n \
Ctrl <Key> F22:	read-mode(OFF)	\ n \
<KeyPress>:	insert()	\ n

		\
<KeyPress>Retur n:	insert()	\ n \
~Meta Shift <Btn1Down>:	select-start()	\ n \
~Meta <Btn1Down>:	select-start()	\ n \
~Meta Shift <Btn1Motion>:	select-extend()	\ n \
~Meta <Btn1Motion>:	select-extend()	\ n \
~Meta ~Shift <Btn2Up>:	insert-selection(PRIMARY, CUT_BUFFER0)	\ n \
~Meta Shift <Btn2Up>:	insert-selection(PRIMARY, CUT_BUFFER0)	\ n \
~Ctrl ~Meta <Btn3Down>:	start-extend()	\ n \
~Meta <Btn3Motion>:	select-extend()	\ n \
~Meta ~Shift <BtnUp>:	select-end(PRIMARY, CUT_BUFFER0)	\ n \
~Meta Shift <BtnUp>:	select-end(PRIMARY, CUT_BUFFER0)	\ n \
<Btn2Down>:	ignore()	\ n \
<BtnDown>:	bell(0) \	

Table 68: Default: translations

ttyModes
Class: TtyModes

See option "-tm " assignment [assignment ...]""
Allows you to set terminal functions for a terminal emulator analogously to the stty command.

You can specify one or more assignments for *value*. Specifications consist of a keyword for the terminal attribute and the desired value.

utmpInhibit
Class: UtmpInhibit
See option "-ut"
Specifies whether the login entry for the emulator process is entered in the system log file /etc/utmp.

Default:
False
The emulator generates an entry.

visualBell
Class: VisualBell
See option "-vb"
Specifies whether the emulator generates a visual warning rather than an acoustic signal when CTRL-G is received.
The window contents are displayed briefly in reverse video.

Default:
False
An acoustic warning is sounded.

w0
Class: W0
See "fontMapping ", "g0", "w1" resources
Determines the mapping of the lower half of the character generator work area to an allocation area for character sets.

Default:
g0

w1
Class: W1
See "fontMapping ", "g3", "w0" resources
Determines the mapping of the upper half of the character generator work area to an allocation area for character sets.

Default:
g3

xnlLanguage
Class: XnlLanguage
Specifies the client's locale.

Default:
value of the LANG environment variable

2.21.2 Appearance and behavior

The x97801-480 program emulates almost all of the attributes of the 97801-480 terminal, with the following exceptions:

- Screen saver timeout is not supported.
  You cannot, therefore, enable and disable it using an escape sequence. You can only apply it to the entire root window.
- The CH.CODE LED on the keyboard is not supported; and you cannot enable and disable the function using escape sequences.
- Loading the compose key table:
  As long as storage space permits, the table of predefined compose key sequences is identical with that of the 97801-480 terminal. Unlike the terminal, the terminal emulator requests storage space dynamically.
- Loading the function key table:
  As long as storage space permits, the function key table is dynamically constructed. Unlike the terminal, the terminal emulator requests storage space dynamically.
- The "reduced-intensity" feature on monochrome screens is represented by displaying characters against a gray background. You can control this behavior using the doHalfIntensity resource.
- Keyboard clicking is not supported; you cannot, therefore, enable and disable it using an escape sequence.
- Single inverted block cursor:
  The single inverted block cursor is not supported. The corresponding escape sequence is therefore ignored.
- Loading the string keymap:
  Loading the string keymap is not supported, since 97801 key assignment tables are not supported, and the string keymap references the hardware code of a key. As an X client, the x97801-480 terminal emulator only handles keysyms. For more information on keysyms, please refer to the description of the xmodmap client (section "xmodmap - Displaying/modifying keyboard and pointer maps ").
- Loading key assignment tables:
  Loading key assignment tables is not supported. Key assignment tables are very hardware-specific. You can use the xmodmap client instead of key assignment tables for keyboard configuration.
- Loading the dead key table:
  As long as storage space permits, the table of predefined dead key sequences is identical with that of the 97801-480 terminal. Unlike the terminal, the terminal emulator requests storage space dynamically.
- Loading character generators:
  The x97801-480 terminal emulator supports the loading of character generator addresses for fonts whose characters are represented by an 8x16 bitmap only. Emulation of the loadable character generator in x97801-480 supports up to 256 loadable characters in the work area. The number of characters which may be loaded is also limited by the amount of storage space available.
  Wherever possible, you should use the fontList resource (class: FontList) and the fontMapping resource (class: FontMapping) instead of the loadable character generator.

The following sections describe the appearance and behavior of the default configuration of the terminal emulator with regard to input focus, escape sequences, internationalization, supported terminfo entries and character sets. The terminal emulator supports cut-and-paste functionality, as described in the section "Copying text ". You will find

information on how to configure the optional scroll bars in the section "Scroll bars ". The terminal emulator enables you to bind frequently used character strings to function keys. This is described in the section "Programming function keys ". You can also use dead key and compose key sequences to generate characters not on the keyboard. You will find information on this in the section " Dead keys and compose keys ". You can access the terminal emulator menus via a menu bar, provided that the noMenuBar resource (class: noMenu) is assigned the value "False". You can display the menu bar using the key sequence **[CTRL] [MENU] [HELP]**. For more information on the terminal emulator menus, please refer to the section "Menus".

2.21.2.1 Terminal emulator configuration

Input focus: If the input focus is assigne {'} d to the emulator window, the emulator represents this by visually highlighting the cursor (solid rectangle), if permitted by the window manager. Even if the keyboard is locked (using **[ESC] {!!!!!}**]), the cursor remains highlighted. You can use the alwaysHighlight resource of the AlwaysHighlight class to specify that the cursor should always be highlighted.
Escape sequences: In addition to the escape sequences for the emulated terminal as documented in the operating system interfaces description, you can also use the following escape sequences:

Define window and icon names	**[ESC]**]0; *name* \007
Define icon names	**[ESC]**]1; *name* \007
Define window names	**[ESC]**]2; *name* \007
Define filename of log file	**[ESC]**]46; *name* \007
Load allocation area G0 from the character generator	**[ESC]**(*char_set*
Load allocation area G1from the character generator	**[ESC]**)*char_set*
Load allocation area G2 from the character generator	**[ESC]***char_set*
Load allocation area G3 from the character generator	**[ESC]**+*char_set*

Table 69: Escape sequences

char_set can be replaced by:

char_set	character set
y	Blanks
K	German
x	DRCS character set

u	Euro
c	Facet
v	IBM
@	International
B	International A
w	Brackets
t	Math

Load allocation area G1 from the character generator	[ESC]-*char_set*
Load allocation area G2 from the character generator	[ESC].*char_set*
Load allocation area G3 from the character generator	[ESC]/*char_set*

Table 71: Escape sequences

char_set can be replaced by:

char_set	character set
A	ISO 8859-1
B	ISO 8859-2
C	ISO 8859-3
D	ISO 8859-4
F	ISO 8859-5
@	ISO 8859-7
T	ISO 8859-9

Table 72: Possible character sets

Load G1 to upper work area	[ESC]~
Load G2 to upper work area	[ESC]}
Load G3 to upper work area	[ESC]|
Load G0 to lower work area	<Shift in> corresponds to 0x0f
Load G1 to lower work area	<Shift out> corresponds to 0x0e
Load G2 to lower work area	[ESC]n

Load lower work area for display of next character only from G2	[ESC]N
Load lower work area for display of next character only from G2	<single Shift 2> corresponds 0x8e
Load lower work area from G3	[ESC]O
Load lower work area for display of next character only from G3	[ESC]O
Load lower work area for display of next character only from G3	<single Shift 3> corresponds 0x8f

Table 73: Escape sequences

International implementation: The terminal emulator supports handling of NLS message files and evaluates the LANG environment variable. Currently, however, there is only one message file and this is in German. For all other languages, the terminal emulator uses the English default texts embedded in the program code.
The message file includes the texts of the menu items and the messages which the emulator displays when you call it using the -help option or an invalid option.
Terminfo entries: The terminal emulator searches the terminfo database for the entries "x97801-480", "97801", "xansi", "ansi" and "dumb" in the given order.
Character sets: The terminal emulator supports the following character sets: Blanks, German, the character set in the DRCS area initialized with International A, European, Facet, IBM, International, International A, Brackets and Math, as well as the upper half of the character sets ISO 8859-1, ISO 8859-2, ISO 8859-3, ISO 8859-4, ISO 8859-5, ISO 8859-7 and ISO 8859-9.
Each of these character sets contains 128 different characters, as described in the operating system interfaces documentation. A total of up to 1024 different characters are available for use, 800 of which are predefined. These 1024 characters are stored in what is known as the character generator.
You can use the character sets by following the steps described below:

- Load the desired character set into an allocation area. An allocation area contains 128 characters. The terminal emulator provides you with four allocation areas: G0, G1, G2 and G3. You can load the character set into the appropriate area using the resources g0, g1, g2 and g3 or using the escape sequences described earlier.
- Load the selected allocation area into the work area. The work area contains 256 characters and can therefore accommodate two character sets, one in the lower half of the work area (W0) and one in the upper half of the work area (W1). You can load the allocation area into the appropriate work area using the w0 and w1 resources or using the escape sequences described earlier.
- The work area now contains 256 characters which have been predefined in the character generator. Specific display attributes, such as the appearance and size of the characters, reduced intensity, italics, etc. are defined by the font used to print the character. You can now use the fontMapping resource (class: FontMapping) to specify these attributes.

You can determine how characters are to appear by specifying an appropriate font for the font resource (class: Font) or the fontList resource (class: FontList). To ensure that each character is represented correctly in the work area, you must assign a corresponding image in a font to each of the maximum 1024 characters predefined in the character generator by assigning a suitable value to the fontMapping resource (class: FontMapping). The image of a character is also referred to as a glyph. Because only 256 glyphs per font are available to the terminal emulator, you must specify at least four fonts in the fontList resource (class: FontList) so that the entire character set of the character generator can be mapped.

2.21.2.2 Programming function keys

If the softKey resource (class: SoftKey) has the value "True", you can select the "Softkey Programming" option from the Keyboard menu in the menu bar. A new window appears on the screen in which the function keys that have already been programmed are displayed. You can edit this list by deleting individual entries for function keys, modifying an entry or adding new entries. Use the syntax described below to enter the new function key allocation to a character string in the input field at the bottom of the window.

- **Selecting a function key entry from the list:** If you click one of the entries on the list of currently programmed function keys using button 1, the entry is displayed in the input field at the bottom of the window and can be edited.
- **Enabling an entry:** A new allocation is not enabled until you click the "Apply" button.
- **Cancelling a modification:** If you want to delete an entry in the input field at the bottom of the window, click the "Cancel" button.
- **Ending the dialog:** If you want to quit the dialog for programming function keys, click the "Unmap" button.
- **Saving entries:** If you want to save the current list of assignments of character strings to function key for future sessions, click the "Store" button.

The syntax of a function key entry is as follows:
[*modifier*]...<key>F*number*: *char_string*

number
specifies the number of the function key

char_string
specifies the *char_string* you want to assign.
Leading blanks are not included in *char_string*. If *char_string* contains blanks, it must be enclosed in double quotes. The null string must be specified as "".
char_string can contain any character in the ASCII set, as well as the following special characters:

\E	[ESC]	0x1b
\n	newline	0x0a
\t	tabulator	0x09
\\	backslash "\"	0x5c
\ooo	character with octal value ooo	0xyy

%f*number*	expands to the string assigned to function key <F*number*>.
%F*number*	expands to the string assigned to function key <F*number*> with the [SHIFT] key pressed.
%c*number*	expands to the string assigned to function key <F*number*> with the [CTRL] key pressed.
%C*number*	expands to the string assigned to function key <F*number*> with [SHIFT] and [CTRL] pressed.
%%	percent sign "%"

Table 74: Special characters in the ASCII character set

Multilevel expansion is supported. Recursive expansion is not.

modifier

specifies the *modifier* you must press simultaneously with the function key to get *char_string*.
you can replace *modifier* with:

shift for the [shift] key
ctrl for the [ctrl] key
you can also specify more than one modifier.

2.21.2.3 Dead keys and compose keys

The terminal emulator in 8-bit mode enables you to use dead key and compose key sequences to generate characters not on the keyboard. You can, for example, use the eightbit resource (class: Eightbit) to set the 8-bit mode. While characters generated using compose keys are initiated by the compose key, characters generated using dead keys can be initiated by several different keys.
Dead keys: In 8-bit mode, the deadKey resource (class: Compose) must be assigned the value "True" if keyboard input using dead keys is to be interpreted by the terminal emulator. If you enter the first character of a dead key sequence, the state of the "Dead key" push button in the status line changes. Subsequent keyboard input is interpreted as part of a dead key sequence. If the key sequence does not correspond to one of the predefined dead key sequences listed below, an acoustic or visual warning is output, depending on the value of the visualBell resource (class: VisualBell). If, however, one of the predefined dead key sequences listed below is recognized by the terminal emulator, the combination character assigned to this dead key sequence is processed.
The table of predefined dead key sequences is the table of the 97801-480 terminal, provided that storage space permits. Unlike the terminal, the terminal emulator requests storage space dynamically. You can load this table using the following escape sequence:
<ESC>RE(combi_charcharacter1[character2[...[character8]...]]) \
[;(combi_charcharacter1[character2[...[character8]...]])]... \
[;character1[character2[...[character8]...]]]... \<*ESC*> \

combi_char
: two-digit hexadecimal number specifying the combination character to be processed when the subsequent characters are entered. The hexadecimal number is the address of the character in the work area of the terminal emulator.

character$_i$
: hexadecimal number specifying the i-th key in the dead key sequence. Extended to the terminal function range, a dead key sequence can consist of up to 8 characters, instead of 3. The hexadecimal number is the address of the character in the work area of the terminal emulator.

The following table of predefined dead key sequences contains entries in the form *character*$_1$*character*$_2$ *result* (*combi_char*) *number*

character$_i$
: see above

result
: representation of the character specified in hexadecimal form by *combi_char*

combi_char
: see above

number
: number of *character*$_i$ to be entered

^ ^ (5e) 2	`E È (c8) 2	˝o ő (d5) 2	,c ç[1] (e7) 2	^o ô (f4) 2
` ` (60) 2	´E É[1] (c9) 2	¨o ö[1] (d6) 2	`e è (e8) 2	˝o ő (f5) 2
˜ ˜ (7e) 2	^E Ê (ca) 2	`U Ù (d9) 2	´e é[1] (e9) 2	¨o ö[1] (f6) 2
¨ ¨[1] (a8) 2	¨E [1] (cb) 2	´U Ú[1] (da) 2	^e ê (ea) 2	`u ù (f9) 2
´ ´[1] (b4) 2	`I Ì (cc) 2	^U Û (db) 2	¨e [1] (eb) 2	´u ú[1] (fa) 2
¸ ¸[1] (b8) 2	´I Í[1] (cd) 2	¨U Ü[1] (dc) 2	`i ì (ec) 2	^u û (fb) 2
`A À (c0) 2	^I Î (ce) 2	´Y ý[1] (dd) 2	´i í[1] (ed) 2	¨u ü[1] (fc) 2
´A Á[1] (c1) 2	¨I Ï[1] (cf) 2	`a à (e0) 2	^i î (ee) 2	´y ý[1] (fd) 2
^A Â (c2) 2	˜N Ñ (d1) 2	´a à[1] (e1) 2	¨i ï[1] (ef) 2	¨y ÿ[1] (ff) 2
˜A Ã (c3) 2	`o Ò (d2) 2	^a â (e2) 2	˜n ñ (f1) 2	
¨A Ä[1] (c4) 2	´o Ó[1] (d3) 2	˜a ã (e3) 2	`o ò (f2) 2	
,C Ç[1] (c7) 2	^o Ô (d4) 2	¨a ä[1] (e4) 2	´o ó[1] (f3) 2	

Figure 1: ISO 8859-1dead key table sorted by dead key character

1) not in standard ASCII character set

Compose key: If you want the terminal emulator to interpret keyboard input commencing with the compose key, the compose resource (class: Compose) in 8-bit mode must have the value "True". When you press the key assigned to the keysym XK_Multi_key, the state of the "Compose" push button changes in the status line at the bottom of the emulator window and subsequent keyboard input is interpreted as part of a compose sequence. You can use the noState resource (class: NoState) to specify whether or not the status line is displayed. You can use the xmodmap client to obtain information on which key on the keyboard has been assigned the keysym XK_multi_key. You can also initiate a compose

sequence with the push button "Compose" instead of the using the key with the keysym XK_multi_key. The "Compose" push button also enables you to abort a compose sequence. If a key sequence is interpreted as a compose sequence, keyboard input is displayed in the status line of the emulator window.
If the key sequence does not correspond to one of the predefined compose sequences listed below, an acoustic or visual warning is output, depending on the value of the visualBell resource (class: VisualBell). If, however, the sequence is recognized as one of those listed below, the combination character assigned to this compose sequence is processed. The table of predefined compose sequences is the table of the 97801-480 terminal, provided that storage space permits. Unlike the terminal, the terminal emulator requests storage space dynamically. You can load this table using the following escape sequence:
<ESC>RD(*combi_charcharacter*$_1$[*character*$_2$[...[*character*$_8$]...]]) \
[;(*combi_charcharacter*$_1$[*character*$_2$[...[*character*$_8$]...]])]... \
[;*character*$_1$[*character*$_2$[...[*character*$_8$]...]]]... \ <ESC> \

combi_char
: two-digit hexadecimal number specifying the combination character to be processed when the subsequent characters are entered. The hexadecimal number is the address of the character in the work area of the terminal emulator.

character$_i$
: hexadecimal number specifying the (i+1)th key in the compose key sequence. Extended to the terminal function range, a compose sequence can consist of up to 8 characters, instead of 3. The hexadecimal number is the address of the character in the work area of the terminal emulator.

The following table of predefined compose sequences contains entries in the form
character$_1$*character*$_2$ *result* (*combi_char*) *number*

character$_i$
: see above

result
: representation of the character specified in hexadecimal form by *combi_char*

combi_char
: see above

number
: number of *character*$_i$ to be entered

++ ♯ (23) 2	-Y ¥ (a5) 2	rO Φ (ae) 2	A` À (c0) 2	O¯ Ō (d5) 2	`i ì (ec) 2
aa ((40) 2	y= ¥ (a5) 2	Or Φ (ae) 2	'A Á (c1) 2	"O Ö (d6) 2	i` ì (ec) 2
AA ((40) 2	=y ¥ (a5) 2	_ ¯ (af) 2	A' Á (c1) 2	O" Ö (d6) 2	'i í (ed) 2
((¦ (5b) 2	Y= ¥ (a5) 2	_ ¯ (af) 2	^A Â (c2) 2	xx × (d7) 2	i' í (ed) 2
// \ (5c) 2	=Y ¥ (a5) 2	..... ¯ (af) 2	A^ Â (c2) 2	** × (d7) 2	^i I (ee) 2
/< \ (5c) 2	| ¦ (a6) 2	^_ ¯ (af) 2	~A Ã (c3) 2	O/ Ø (d8) 2	i^ I (ee) 2
</ \ (5c) a	xo § (a7) a	_^ ¯ (af) a	A~ Ã (c3) a	/O Ø (d8) a	"i I (ef) a
))] (5d) 2	Dx § (a7) 2	o ° (b0) 2	"A Ä (c4) 2	`U Ù (d9) 2	i" I (ef) 2
(- ¦ (7D) 2	XO § (a7) 2	o ° (b0) 2	A" Ä (c4) 2	U` Ù (d9) 2	d- đ (f0) 2
-(¦ (7b) 2	DE § (a7) 2	0^ ° (b0) 2	Ao Å (c5) 2	'U Ú (da) 2	-d đ (f0) 2
^/ ¦ (7c) 2	x! § (a7) 2	^0 ° (b0) 2	oA Å (c5) 2	U' Ú (da) 2	~n ñ (f1) 2
/^ ¦ (7c) 2	!x § (a7) 2	♯ ° (b0) 2	A. Å[a] (c5) 2	^U Û (db) 2	n~ ñ (f1) 2
)-] (7d) 2	X! § (a7) 2	♯ ° (b0) 2	.A Å[1] (c5) 2	U^ Û (db) 2	`o ò (f2) 2
-)] (7d) 2	!E § (a7) 2	+- ± (b1) 2	AE Æ (c6) 2	"U Ü (dc) 2	o` ò (f2) 2
(a0) 2	X0 § (a7) 2	-+ ± (b1) 2	,C Ç (c7) 2	U" Ü (dc) 2	'o ó (f3) 2
!! ¡ (a1) a	Ox § (a7) a	a^ ² (b2) a	C, Ç (c7) a	'Y Ý (dd) a	'o ó (f3) a
c| ¢ (a2) 2	X0 § (a7) 2	^2 ² (b2) 2	`E È (c8) 2	Y' Ý (dd) 2	^o ô (f4) 2
|c ¢ (a2) 2	DE § (a7) 2	3^ ³ (b3) 2	E` È (c8) 2	TH Þ (de) 2	o~ õ (f5) 2
c/ ¢ (a2) 2	" " (a8) 2	^3 ³ (b3) 2	'E É (c9) 2	PT Þ (de) 2	~o õ (f5) 2
/c ¢ (a2) 2	xo © (a9) 2	' ´ (b4) 2	E' É (c9) 2	ss ß (df) 2	o~ õ (f5) 2
C| ¢ (a2) 2	xc © (a9) 2	' ´ (b4) 2	^E Ê (ca) 2	`a à (e0) 2	"o ö (f6) 2
|C ¢ (a2) 2	XO © (a9) 2	/u µ (b5) 2	E~ Ẽ (ca) 2	a` à (e0) 2	o" ö (f6) 2
C/ ¢ (a2) 2	XC © (a9) 2	/U µ (b5) 2	"E Ë[1] (cb) 2	'a á[1] (e1) 2	/ ÷ (f7) 2
/C ¢ (a2) 2	X0 © (a9) 2	p! ¶ (b6) 2	E" Ë[1] (cb) 2	a' á[1] (e1) 2	:- ÷ (f7) 2
1… ½ (a3) 2	xc © (a9) 2	!p ¶ (b6) 2	`I Ì (cc) 2	^a â (e2) 2	o/ ø (f8) 2
-1 ½ (a3) 2	X0 © (a9) 2	P! ¶ (b6) 2	I` Ì (cc) 2	a~ ã (e3) 2	/o ø (f8) 2
1= ½ (a3) 2	XC © (a9) 2	!P ¶ (b6) 2	'I Í[1] (cd) 2	~a ã (e3) 2	`u ù (f9) 2
=1 ½ (a3) 2	a- ª (aa) 2	. · (b7) 2	I' Í[1] (cd) 2	a~ ã (e3) 2	u` ù (f9) 2
L- ½ (a3) 2	-a ª (aa) 2	. · (b7) 2	^I Î (ce) 2	"a ä[1] (e4) 2	'u ú (fa) 2
-L ½ (a3) 2	A- ª (aa) 2	,, ¸ (b8) 2	I^ Î (ce) 2	a" ä[1] (e4) 2	u' ú (fa) 2
L= ½ (a3) 2	-A ª (aa) 2	,, ¸ (b8) 2	"I Ï (ce) 2	ao å[1] (e5) 2	^u û (fb) 2
=L ½ (a3) 2	<< « (ab) 2	1^ ¹ (b9) 2	I" Ï (ce) 2	oa å[1] (e5) 2	u^ û (fb) 2
xo ¤ (a4) 2	!- ¬ (ac) 2	^1 ¹ (b9) 2	D- Ð (d0) 2	ae æ (e6) 2	"u ü (fc) 2
ox ¤ (a4) 2	¨! ¬ (ac) 2	o… º (ba) 2	~D Ð (d0) 2	,c ç[1] (e7) 2	u" ü (fc) 2
XO ¤ (a4) 2	-- - (ad) 2	-o º (ba) 2	~N Ñ (d1) 2	C, Ç (e7) 2	'y ý (fd) 2
OX ¤ (a4) 2	- - (ad) 2	O- º (ba) 2	N~ Ñ (d1) 2	`e è (e8) 2	y' ý (fd) 2
x0 ¤ (a4) 2	- - (ad) 2	-O º (ba) 2	`O Ò (d2) 2	`e è (e8) 2	tp þ (fe) 2
0x ¤ (a4) 2	or © (ae) 2	>> » (bb) 2	O` Ò (d2) 2	'e é (e9) 2	pt p (fe) 2
X0 ¤ (a4) 2	ro © (ae) 2	14 ¼ (bc) 2	'O Ó (d3) 2	e' é (e9) 2	y" ÿ (ff) 2
0X ¤ (a4) 2	RO © (ae) 2	12 ½ (bd) 2	O' Ó (d3) 2	^e ê (ea) 2	"y ÿ (ff) 2
y- ¥ (a5) 2	OR © (ae) 2	34 ¾ (be) 2	^O Ô (d4) 2	e^ ê (ea) 2	
-y ¥ (a5) 2	oR © (ae) 2	?? ¿ (df) 2	O^ Ô (d4) 2	"e ë (eb) 2	
Y… ¥ (a5) 2	Ro © (ae) 2	`A À (c0) 2	~O Õ (d5) 2	e" ë (eb) 2	

a. not in standard ASCII character set

Figure 2: ISO 8859-1compose key table sorted by compose character

1) not in standard ASCII character set

2.21.2.4 Copying text

Using the mouse or puck buttons, text in an emulator window can be selected and copied either within the same window or into another emulator window. You use button 1 to select and copy text to a text buffer and button 2 to copy this text from the buffer to the cursor position in the active window. Please note that it is possible to change the binding of buttons to actions. Copied text is not examined for dead key sequences.

Selecting text: Position the pointer at the beginning of the text, then press button 1. Hold down button 1 and move the pointer to the end of the text to be selected. The selected text is highlighted. As soon as you release button 1, the terminal emulator copies the selected text to a global text buffer (referred to as the cut buffer).

Loading excessively large amounts of data into the buffer can cause a deadlock. If necessary, the terminal emulator displays an error message several times before you can continue. Note that the data in the text buffer may not be complete!

Copying text to a cut buffer makes it the primary selection. The cut buffer can be referenced via the name CUT_BUFFER0 or BPRIMARY.

Pasting text: Position the text cursor at the location where the selected text is to be pasted. Click button 2 to start copying the text from the global text buffer to the cursor position. The terminal emulator treats this text like entries made using the keyboard.

Speeding up text selection: With a double click of button 1 you enable word-by-word text selection, with a triple click line-by-line selection. You set the maximum time between clicks with the multiClickTime resource (class: MultiClickTime).

Line-by-line selection, i.e. selection following a triple click, can be affected by the "-cb" option or "cutToBeginningOfLine" resource.

Word-by-word selection, i.e. selection following a double click, takes into account the character classes defined using the charClass resource. It is especially useful to define your own character class if names made up of characters from a certain character set are to be selected as a whole.

For example, following the value assignment

*charClass: "33:48,37:48,45-47:48,64:48"

the characters !, %, -, ., / and & will be treated like alphabetic characters and figures.

Appending text: If you want to append text, you must not copy the last newline character into the text buffer when selecting the text. In line-by-line selection mode you can use the -cn option or the cutNewline resource to truncate the last newline character.

Changing the size of the selected area: You can use button 3 to change the size of the current selection. What effect the button has depends on the position of the cursor in relation to the edge of the selected text: the edge closest to the cursor is changed. If, for example, the cursor is closest to the right edge of the selection, the right edge of the selection is extended or reduced. If the cursor is outside of the selected area, the client assumes that you really meant the right or left edge, as appropriate, and restores the original selection, and then extends or reduces the selection by the selection unit last used. In other words, if selection was previously done word by word using button 1, the selection is extended/ reduced by words. Multiple-clicking using button 3 changes the selection mode as described for button 1.

2.21.2.5 Scroll bars

You can use the -sb option or the scrollBar resource (class: ScrollBar) to specify whether or not the terminal emulator displays a scroll bar on the window border. It is assumed in the following section that a scroll bar is displayed.
The highlighted part of the scroll bar indicates which portion of the text is displayed in the window. It corresponds to the position in the text as a whole. Using the scroll bar you can redisplay text which has been scrolled off the top of the window:
Scrolling text: Position the pointer in the highlighted part of the scroll bar, press and hold down button 1 while moving the pointer in the scroll bar.
Scrolling text page-by-page: Depending on whether you want to page forward or backward, position the pointer above or below the highlighted part of the scroll bar. Click button 1.
Using "scroll arrows" to scroll text: If you click button 1 on a scroll arrow, text is scrolled line-by-line. Text is not scrolled continuously when you keep button 1 pressed down on a scroll arrow.
Configuring scroll bars: You use the resources of the vtscrollbar widget (class: XmScrollbar) to customize your scroll bar.

2.21.3 Menus

The x97801-480 terminal emulator comes with a menu bar containing four pull-down menus: X97801, Mode, Display and the pull-down Keyboard menu. You can use the menus in the menu bar if the noMenuBar resource (class: noMenu) has the value "False". If the menu bar is not visible, you can use the menu bar with the key sequence **[CTRL] [MENU] [HELP].**
After providing information on the structure, contents and meaning of the menu items this section describes how to configure menus.

Menu items	Meaning
Logging	Enable/disable logging
Interrupt program	Send SIGINT signal
Terminate program	Send SIGTERM signal
Hangup program	Send SIGHUP signal (shell and emulator are terminated)
Kill program	Send SIGKILL signal (shell and emulator are terminated)
Unmap menu	Switch off menu bar
Quit	Close window and terminate client

Table 75: The "X97801" menu of x97801-480

Menu items	Accelerator	Meaning
Jump	**[ALT] [J]**	Scroll in blocks (instead of line by

Scroll		line)
Scrollbar	[ALT] [S]	Scroll bars
Scroll to bottom on **k**ey		Scroll to end when key is pressed
Scroll to bottom on in**p**ut		Scroll to end on input
Visual Bell	[ALT] [V]	Visual signal generated shortly before end of line is reached
Margin Bell	[ALT] [B]	Acoustic signal generated shortly before end of line is reached
Reverse Wraparo und	[ALT] [W]	Cursor moves backward from the first column in a line to the last column of the previous line
8 bit		Toggles between 7 and 8-bit mode
S**o**ft Reset		Reset emulator's input parser to initial state. If escape sequences have already been detected, branch to initial state of parser. The rest of the escape sequence is handled in normal mode.
F**u**ll Reset		Reset terminal modes and functions to initial state after start of x97801. Any modifications made during the session are cancelled, e.g. tabulators are reset (to every 8 columns). Wrap and smooth scroll are reset; contents of window are deleted.

Table 76: The "Mode" menu of x97801-480

Menu items	**Accele rator**	**Meaning**
Redraw	[ALT] [R]	Redraw window
Reverse **V**ideo		Foreground and background colors are inverted
Cursor blinking enabled		Enable blinking text cursor
B**l**ock cursor		Define cursor shape (block

		or underscore)
Ignore halfintensity		Text attribute "reduced intensity" ignored

Table 77: The "Display" menu of x97801-480

Menu items	**Accelerator**	**Meaning**
CH Code	**[ALT] [I]**	Enable/disable international character set; same effect as pressing **[CHCODE]**
Auto Repeat		Indicates "auto-repeat" status of X server; the effects of a change of status are not restricted to the active window
compose enabled		Enable compose key
dead keys enabled		Enable dead keys
softkey enabled		Enable function keys
Softkey **p**rogramming ...		Program function keys
Sho**w** Softkey window		Display mouse-operated keyboard

Table 78: The "Keyboard" menu of x97801-480

Configuring menus: The configuration of all menus is affected by the resources of x97801-480's vtmenubar widget (class: XmRowColumn).

Example:

X97801-480*vtmenubar*background: green
assigns the value "green" to the "background" resource instance of the menu bar "vtmenubar" for every client of the "X97801-480" class.

Individual menus can be configured using the resources of the x97801, mode, display and keyboard widgets, which are all members of the XmRowColumn class.

Example:

X97801-480*vtmenubar.display*background: green
assigns the value "green" to the "background" resource instance of the menu bar "vtmenubar" in the menu "Display" for every client of the "X97801-480" class.

2.21.4 Files

The files below are listed in alphabetical order. The resource configuration files are listed only in the form of the generalized file names $XENVIRONMENT, $XUSERFILESEARCHPATH and $XFILESEARCHPATH.

/etc/utmp
: system log file in which the terminal emulator creates a login entry when the -ut option is set.

$HOME/.login
: shell script executed when a login shell is started in the emulator window.

$HOME/.profile
: shell script executed when a login Bourne shell is started in the emulator window. You can, for example, define Bourne shell functions here.

$HOME/.X97SoftKeys
: specifies the name of the file whose entries are used to define the softkeys when the terminal emulator is invoked.

$HOME/.Xdefaults
: user-specific configuration file

$XAUTHORITY
: user-specific authorization file
The client uses this file as a user-specific authorization file. If XAUTHORITY is not set, the client uses the .Xauthority file in the user's HOME directory.

$XENVIRONMENT
: user-specific configuration file

$XFILESEARCHPATH
: application-specific configuration file created by the client programmer

$XUSERFILESEARCHPATH
: application-specific configuration file created by the system administrator or user

X97Log.*n*
: default log file for the terminal emulator x97801-480 in the directory from which the terminal emulator is called

2.21.5 Environment variables

Detailed descriptions of all the environment variables, arranged in alphabetical order, are given in the chapter "Environment variables ".

DISPLAY
: The client reads the DISPLAY environment variable if the -display option is not specified in its command line. From this variable the client finds out the name of the X server and the display and screen on it.
The DISPLAY environment variable is a string in the format:
hostname:*display_number*.*screen_number*
The client terminates abnormally if the DISPLAY environment variable is not set and the -display option is not specified in the client command line.

HOMEidentifies the user's HOME directory.
: The client searches this directory for files.

LANGis evaluated in the context of NLS language support for message texts and configuration files.

NLSPATH
: The client evaluates the NLSPATH environment variable as a search path for message files.
Thus the NLSPATH environment variable must include the search path /usr/lib/nls/msg/%l/x97801-480.cat. "%l" is replaced by that part of the locale definition which identifies the language.

TERMdenotes the type of terminal emulated.
: The terminal emulator sets the TERM variable according to the -tn option, the termName resource or the first entry in the terminfo database.

WINDOWID
: The terminal emulator sets the contents of the WINDOWID variable to the number of the window, which is unique in the context of the root window.

XAUTHORITY
: denotes the user-specific authorization file.
If the environment variable is not set, the client searches for the $HOME/.Xauthority file.

XBMLANGPATH
: defines a list of directories the client is to search for pixmaps and bitmaps .

XFILESEARCHPATH
: specifies the path for configuration files.

XUSERFILESEARCHPATH
: specifies the path for configuration files.

3 Client resources

Resources allow you to tailor the appearance and behavior of clients to meet your own individual requirements. Using the resources you can influence:

- the color and pattern of windows, window borders, menus and icons
- the font used in window frames, menus and icons
- focus policy for input devices and colormaps
- the size and position of window and icons.

The manual "[2]" "SINIX/windows User Environment; Guide for Experts and System Administrators" provides you with an introduction to configuring a client with resources. The chapter "Assigning values to resources " is a source of reference for configuring the user interface with resources.

Resource classes and resource instances

The concept of resource classes and resource instances provides an easy way of setting a whole group of resource instances at one time.

Each resource has a name or "instance". Each resource instance is allocated to a resource class. A resource class normally consists of several resource instances. The class names are generally different to the instance names in that most class names begin with an uppercase letter while instance names begin with a lowercase letter.

In general, the terms "resource instance" and "resource class" can be used instead of the term "resource". Should this not be the case at any point, the appropriate term will be used explicitly.

Description format

In the following pages you will find a description of the client resources. They are arranged in alphabetical order by resource class. The resource instances of each resource class are also described and arranged in alphabetical order.

The resource class to which the resource instance belongs is given in the running header. If you assign a value to a resource class, you implicitly assign this value as a default to all resource instances belonging to this resource class. If you do not explicitly assign a resource instance a value, but you do assign a value to the resource class to which the resource instance belongs, the value assigned to the resource class is also valid for the resource instance. The resource instance is also said to "inherit" its default value from its resource class. If you have assigned values to both the resource instance and the resource class, the value of the resource instance applies.

Under the heading of the **resource instance** there is a description of what the resource does if you use the syntax without optional specifications.

The syntax valid for the resource instance and the specified clients follows. You use this information to assign a value to a resource. The rules for syntax description given in the "Value assignment syntax " in the chapter "Assigning values to resources " apply.

In the description of the syntax, you can use the name of the resource instance or resource class instead of *resource*.

The component name furthest to the left in a syntax description generally identifies the client and is its instance name or class name. If you specify the client in a value assignment, you affect the resource *resource* for all objects of the user interface managed by the client *client*.

Example:

"mwm*background: green" affects the background of windows managed by the mwm client.

If further components are specified between the resource *resource* and the client *client*, you influence the resource *resource* for all components that inherit the value of the resource *resource* from this component.

Example:

"mwm*icon*background: green" affects the background of the icons of windows managed by the mwm client.

The values you can specify for the resource are described in the final subsection of each resource instance's description. The values are valid both for the resource instance and the resource class.

3.1 Accelerators

3.1.1 menuAccelerator

This resource defines a sequence of keystrokes as a menu bar accelerator.
Entering the defined keystroke sequence causes the first menu item on the menu bar to be selected.

Syntax

```
{display }
{mode    }
X97801-480[*{vtmenubar[*{x97801  }]}]*resource: value
{keyboard}
XPct*vtmenubar*resource: value
```

Value

```
[#replace ]
[[#override] \n]\
[#augment ]
[[!][~][@]modifier]...<action>key[key]...[,\
[[!][~][@]modifier]...<action>key[key]...]...
```

#replace

#replace enables you to specify that the following translation table is to fully replace the previously active one.

#override

#override enables you to specify that the following translation table is to replace the previously active one for those key or button sequences assigned a value in the previously active table.

#augment

#augment enables you to specify that the following translation table is to replace the previously active one for those key or button sequences not assigned a value in the previously active table.

!*modifier*

Prefixing *modifier* with ! means that no modifier other than *modifier* may be specified, to ensure that this entry in the translation table will become active for the current key or button sequence.

~modifier

Prefixing *modifier* with ~ means that this modifier *modifier* may not be specified, to ensure that this entry in the translation table will become active for the current key or button sequence.

@modifier

Prefixing *modifier* with @ means that all those modifiers containing the keysym *modifier* may be specified. Only then will this entry in the translation table become active for the current key or button sequence.
For more information on keysyms refer to the description of the xmodmap client in section "xmodmap - Displaying/modifying keyboard and pointer maps ".

modifier

can be replaced by:

Ctrl	CTRL key
Shift	Shift key
Meta	Meta key
Lock	LOCK key
Mod1	Modifier key 1
Mod2	Modifier key 2
Mod3	Modifier key 3
Mod4	Modifier key 4
Mod5	Modifier key 5

The xmodmap client will supply you with information on the mapping of the values for *modifier* to keys on the keyboard.

action

can be replaced by the following key or button actions:

Key	Press key
KeyDown	Press key
KeyUp	Release key
BtnDown	Press button
BtnUp	Release button
Btn*number*Down	Press button *number*, where *number* is between 1 and 5 inclusive
Btn*number*Up	Release button *number*, where *number* is between 1 and 5 inclusive
key	can be replaced by any character in the ISO Latin 1 character set. The xmodmap client will supply you with

	information on the keys your display supports.

3.2 AccessFile

3.2.1 accessFile

This resource defines the full pathname of the Xaccess file which controls access for XDMCP requests.
An entry in this file either grants a remote host direct access to the xdm service of the local host or defines a list of hosts accessible to a remote host.

Syntax

`DisplayManager*`*`resource`*`: `*`value`*

Value

name Full pathname of the file

3.3 Alignment

3.3.1 alignment

This resource governs the alignment of the logo in the xdm login window or host selection window.
This resource is evaluated only if the value of the showLogo resource is True.

Syntax

`XLogin*logo.`*`resource`*`: `*`value`*

Value

XmALIGNMENT_BEGINNING
: The logo is on the left-hand side of the login window.

XmALIGNMENT_CENTER
: The logo is in the center of the login window.

XmALIGNMENT_END
: The logo is on the right-hand side of the login window.

3.4 AllowIcons

3.4.1 allowIcons

This resource determines whether user login name icons are displayed in the login window on the local host. For reasons of security it may be advisable not to display icons.
The onlyLocalUser resource allows you to specify that only local user login names are to be displayed. In the file whose name is defined by the noIconUsers resource you can list individual user login names which are not to displayed as icons.

Syntax

`DisplayManager.`*`resource`*`: `*`value`*

Value

True User icons are displayed.
False User icons are not displayed.

3.5 AllowSendEvents

3.5.1 allowSendEvents

This resource specifies whether or not a terminal emulator is to evaluate keyboard or mouse events sent to it by other clients.

Syntax

`XPct[*xpct]*`*`resource`*`: `*`value`*

`X97801-480[*x97801-480]*`*`resource`*`: `*`value`*

Value

False Events sent by other clients are not evaluated.
True Events sent by other clients are evaluated.

3.6 AlwaysHighlight

3.6.1 alwaysHighlight

This resource specifies whether or not a terminal emulator is to highlight the text cursor even if a window is not the active window.

Syntax

```
XPct[*xpct]*resource: value
XTerm*vt100*resource: value
X97801-480[*x97801-480]*resource: value
```

Value

False In the terminal emulator, the text cursor is only highlighted in the active window.
True In the terminal emulator, the text cursor is highlighted in all windows.

3.7 AnimatedClientIcon

3.7.1 animatedClientIcon

specifies the icon image displayed when new messages are received.

Syntax

`XMessages.`*`resource`*`: `*`value`*

Value

name Full pathname of the icon file.

3.8 AuthComplain

3.8.1 authComplain

This resource determines whether a message is displayed in the xdm login window if user-specific authorization is active for the display but is not supported by the server. The message text can be defined using the unsecureGreeting resource.
For further information refer to the xdm client description in section "xdm - Display manager".

Syntax

`DisplayManager.DISPLAY.`*`resource`*`: `*`value`*

Value

True A message is displayed.
False A message is not displayed.

3.9 AuthDir

3.9.1 authDir

This resource defines the name of the directory in which the xdm client stores authorization information during session initialization.
For further information refer to the xdm client description in section "xdm - Display manager".

Syntax

`DisplayManager.`*`resource`*`: `*`value`*

Value

name Full pathname of the directory

3.10 AuthFile

3.10.1 authFile

This resource defines the name of the file xdm uses to pass authorization information to the X server in conjunction with the server option -auth.
The file should reside in a directory with restricted access, as destruction of this file paralyzes the authorization service and thus makes it impossible to set up connections to the server.
For further information refer to the xdm client description in section "xdm - Display manager".

Syntax

`DisplayManager.DISPLAY.`*`resource`*`: `*`value`*

Value

path File name

3.11 AuthName

3.11.1 authName

This resource defines the mechanism used when authorization is active.
Only MIT-MAGIC-COOKIE-1 is supported in the international version. In the US version, XDM-AUTHORIZATION-1 is supported as well. If the server and xdm communicate using the XDMCP protocol, the mechanism is determined automatically and the authName resource is ignored.
Authorization is activated with the authorize resource of class Authorize.
For further information refer to the xdm client description in section "xdm - Display manager".

Syntax

`DisplayManager.DISPLAY.`*`resource`*`: `*`value`*

Value

name name of the authorization mechanism

3.12 Authorize

3.12.1 authorize

This resource defines whether xdm uses an authorization mechanism based on the XDMCP protocol for local server connections.
The authName resource defines which mechanism is used.
If the value of the authorize resource is True and authorization is still not possible, the user is shown an error message in the xdm login window if the value of the authComplain resource is True.
With remote X servers which use the XDMCP protocol, the setting (True or False) of the authorize resource is not what decides whether or not authorization is active. In this case if an X server asks for authorization to be used when it requests xdm services, xdm will always provide authorization for that X server.
For further information refer to the xdm client description in section "xdm - Display manager".

Syntax

`DisplayManager.DISPLAY.`*`resource`*`:` *`value`*

Value

True The user-specific authorization mechanism is active.
False The user-specific authorization mechanism is not active.

3.13 AutoKeyFocus

3.13.1 autoKeyFocus

This resource determines the policy for allocating the input focus when the active window is closed or minimized.
This resource is only available if the keyboardFocusPolicy resource instance (class: KeyboardFocusPolicy) is assigned the value "explicit".

Syntax

`Mwm*`*`resource`*`: `*`value`*

Value

False You must click the window or icon that is to receive the input focus.
True If a window which has the input focus is no longer managed by mwm or is iconified, the input focus is set to the window which previously had the input focus.

3.14 AutoRaiseDelay

3.14.1 autoRaiseDelay

The time (in milliseconds) that elapses between a window receiving the input focus and mwm placing the window at the top of the stack.
This resource is only available if the focusAutoRaise resource instance in the FocusAutoRaise resource class is assigned the value "true" and the keyboardFocusPolicy resource instance in the resource class KeyboardFocusPolicy is assigned the value "pointer".

Syntax

`Mwm*`*`resource`*`: value`

Value

milliseconds
Time in milliseconds

3.15 AutoRescan

3.15.1 autoRescan

This resource controls whether xdm automatically rescans changes to configuration files affecting the configuration, the server and the access key.
You can send the SIGHUP signal to the main process (i.e. the xdm process with the lowest process ID) to force the client to rescan the files.

Syntax

`DisplayManager.`*`resource`*`: `*`value`*

Value

True Auto-rescanning is performed.
False Auto-rescanning is not performed.

3.16 AutoWrap

3.16.1 autoWrap

Defines whether the text cursor is positioned automatically in the first column of the next line when a character is output in the last column of the work area.
In addition, the value of the resource determines whether the display is scrolled by a line when the text cursor is in the last line of the work area.

Syntax

`XPct[*xpct]*`*`resource`*`: `*`value`*

Value

False The text cursor remains in the last column of the line until it is explicitly repositioned.
True The text cursor is automatically positioned in the first column of the next line, and the window is scrolled by one line.

3.17 Background

3.17.1 activeBackground

This resource specifies the color of the active window's background.
mwm uses this color only for the window border.

Syntax

```
{menu[*menu_name]     }
{icon[*defaults]      }
Mwm[*{                      }]*resource: value
{client[*title]       }
{feedback             }
```

Value

color Color name in the X server's color database

3.17.2 activeTopShadowColor

This resource specifies the color of the active window's top and left 3-D shadows.

Syntax

```
{menu[*menu_name]     }
{icon[*defaults]      }
Mwm[*{                      }]*resource: value
{client[*title]       }
{feedback             }
```

Value

color Color name in the X server's color database

3.17.3 background

This resource specifies the background color of a window.
The mwm client uses this color for inactive windows only.

Syntax

```
{menu[*menu_name]     }
{icon[*defaults]      }
Mwm[*{                      }]*resource: value
{client[*title]       }
{feedback             }
XTerm*vt100*resource: value
{display }
{mode    }
{vtmenubar     [*{x97801  }]}
X97801-480[*{                {keyboard} }]*resource: value
{vtscrollbar               }
{keyprogramming            }
```

```
XPct*resource: value
XMessages*resource: value
```

Value

color Color name in the X server's color database

3.17.4 iconImageBackground

This resource specifies the background color for icon images.

Syntax

```
{client_class   }
Mwm[*{client_instance}]*resource: value
{defaults       }
```

Value

color
Color name in the X server's database

3.17.5 iconImageTopShadowColor

This resource specifies the color of an icon image's top and left 3-D shadows.

Syntax

```
{client_class   }
Mwm[*{client_instance}]*resource: value
{defaults       }
```

Value

color Color name in the X server's database

3.17.6 matteBackground

This resource specifies the background color of the matte.
This resource is evaluted if the matteWidth resource is assigned a positive value, i.e. if there is a matte.

Syntax

```
{client_class   }
Mwm[*{client_instance}]*resource: value
{defaults       }
```

Value

color
Color name in the X server's color database

3.17.7 matteTopShadowColor

This resource specifies the color of the top and left 3-D shadows of the matte.
This resource is evaluated if the matteWidth resource is assigned a positive value, i.e. a matte exists.

Syntax

```
{client_class   }
Mwm[*{client_instance}]*resource: value
{defaults       }
```

Value

color
Color name in the X server's color database

3.17.8 topShadowColor

This resource specifies the color of a window's top and left 3-D shadows.
The mwm client uses this color for inactive windows only.

Syntax

```
{menu[*menu_name]     }
{icon[*defaults]      }
Mwm[*{                    }]*resource: value
{client[*title]       }
{feedback             }
XPct*resource: value
{display }
{modus   }
{vtmenubar  [*{x97801  }]}
X97801-480[*{             {keyboard}}]*resource: value
{vtscrollbar           }
```

Value

color Color name in the X server's color database

3.18 BackgroundPixmap

3.18.1 activeBackgroundPixmap

This resource specifies the pixmap of the window background of the active window. mwm uses this pixmap for the window frame only.

Syntax

```
{menu[*menu_name]}
{icon            }
Mwm[*{               }]*resource: value
{client[*title]  }
{feedback        }
```

Value

pixmap
: Filename of the pixmap in the directory specified by the bitmapDirectory resource (class: BitmapDirectory), or filename of the pixmap including the absolute path of the file.

background
: Pixmap in the color specified by the value assigned to the background resource instance (class: Background).

foreground
: Pixmap in the color specified by the value assigned to the foreground resource instance (class: Foreground).

25_foreground
: Color mix: 25% foreground and 75% background

50_foreground
: Color mix: 50% foreground and 50% background

75_foreground
: Color mix: 75% foreground and 25% background

horizontal
: Horizontal lines alternating in foreground and background

NULL
: No pixmap

vertical
: Vertical lines alternating in foreground and background

slant_right
: Diagonal lines (slanting to the right) alternating in foreground and background

slant_left
: Diagonal lines (slanting to the left) alternating in foreground and background

3.18.2 backgroundPixmap

This resource specifies the pixmap of the window's background.
The mwm client uses this pixmap for inactive windows only.

Syntax

```
{menu[*menu_name]}
{icon            }
```

```
Mwm[*{                }]*resource: value
{client[*title]  }
{feedback        }
XPct*resource: value
{display }
{mode    }
{vtmenubar     [*{x97801  }]}
X97801-480[*{               {keyboard} }]*resource: value
{vtscrollbar               }
{keyprogramming            }
```

Value

pixmap
: File name of the pixmap in the directory specified by the bitmapDirectory resource (class: BitmapDirectory), or file name of the pixmap including the absolute path of the file.

background
: Pixmap in the color specified by the value assigned to the background resource instance (class: Background).

foreground
: Pixmap in the color specified by the value assigned to the foreground resource instance (class: Foreground).

25_foreground
: Color mix: 25% foreground and 75% background

50_foreground
: Color mix: 50% foreground and 50% background

75_foreground
: Color mix: 75% foreground and 25% background

horizontal
: Horizontal lines alternating in foreground and background

NULL No pixmap

vertical
: Vertical lines alternating in foreground and background

slant_right
: Diagonal lines (slanting to the right) alternating in foreground and background

slant_left
: Diagonal lines (slanting to the left) alternating in foreground and background

3.19 Beep

3.19.1 beep

specifies whether an acoustic signal is to be emitted if messages arrive while xmessages is iconized.

Syntax

```
{ console  }
{ system   }
XMessages[*{          }]*resource: value
{ xerrors  }
{ common   }
```

Value

False No acoustic signal will be emitted.
True An acoustric signal will be emitted.

3.20 BitmapDirectory

3.20.1 bitmapDirectory

This resource specifies the directory the window manager is to examine for pixmaps and bitmaps if no absolute pathname is specified.
If mwm does not find the pixmap or bitmap there, it looks along the path $XBMLANGPATH.

Syntax

`Mwm*`*`resource`*`: value`

Value

name Pathname *name* with an absolute pathname or a pathname relative to the directory from which the client accessing the resource was called

3.21 Blinking

3.21.1 blinkCursor

This resource specifies whether or not the cursor is to blink.
You can use the cursorBlinkTime resource to specify how long the blinking text cursor is to remain visible before being made invisible again.

Syntax

```
XPct[*xpct]*resource: value
X97801-480[*x97801-480]*resource: value
```

Value

True The cursor blinks.
False The cursor does not blink.

3.22 BlinkTime

3.22.1 cursorBlinkTime

This resource specifies the period of time (in milliseconds) during which the blinking cursor is visible.
This resource is only availabe if the blinkCursor resource instance (class: Blinking) is assigned the value "True".

Syntax

```
XPct[*xpct]*resource: value
X97801-480[*x97801-480]*resource: value
```

Value

milliseconds
: can be set to any value between 100 and 1000 milliseconds. If you set a smaller value, the value is automatically set to 100 milliseconds. If you set a greater value, the value is set to 1000 milliseconds.

3.23 BlockCursor

3.23.1 blockCursor

This resource specifies whether the cursor is to be depicted as a rectangular block the size of a character or as an underscore the width of a character.

Syntax

XPct[*xpct]**resource*: *value*

X97801-480[*x97801-480]**resource*: *value*

Value

False The cursor is depicted as an underscore.
True The cursor is depicted as a block.

3.24 BlueColor

3.24.1 blueColor

color setting for color terminals

Syntax

`XPct*`*`resource`*`: value`

Value

color Color name in the X server's color database

3.25 BorderColor

3.25.1 borderColor

This resource specifies the color of the window border in the client area.

Syntax

```
XPct*resource: value
XTerm*resource: value
{display }
{mode    }
{vtmenubar     [*{x97801  }]}
X97801-480[*{                {keyboard} }]*resource: value
{vtscrollbar               }
{keyprogramming            }
```

Value

color Color name in the X server's color database

3.26 BorderWidth

3.26.1 borderWidth

This resource specifies the width (in pixels) of the window border in the client area.

Syntax

```
XPct[*{vtscrollbar}]*resource: value
XTerm*resource: value
X97801-480[*{vtscrollbar}]*resource: value
```

Value

pixels
: Width of the window border in pixels

3.26.2 internalBorder

Specifies the width (in pixels) of the area between the text margin and the window border in the client area.

Syntax

```
XPct[*xpct]*resource: value
XTerm[*vt100]*resource: value
X97801-480[*x97801-480]*resource: value
```

Value

pixel Width of the border in pixels

3.27 BottomShadowPixmap

3.27.1 activeBottomShadowPixmap

This resource specifies the pixmap of the active window's bottom and right 3-D shadows.

Syntax

```
{menu[*menu_name]}
{icon            }
Mwm[*{                }]*resource: value
{client[*title]  }
{feedback        }
```

Value

pixmap
: Filename of the pixmap in the directory specified by the bitmapDirectory resource (class: BitmapDirectory), or filename of the pixmap including the absolute path of the file.

background
: Pixmap in the color specified by the value assigned to the background resource instance (class: Background).

foreground
: Pixmap in the color specified by the value assigned to the foreground resource instance (class: Foreground).

25_foreground
: Color mix: 25% foreground and 75% background

50_foreground
: Color mix: 50% foreground and 50% background

75_foreground
: Color mix: 75% foreground and 25% background

horizontal
: Horizontal lines alternating in foreground and background

NULL No pixmap

vertical
: Vertical lines alternating in foreground and background

slant_right
: Diagonal lines (slanting to the right) alternating in foreground and background

slant_left
: Diagonal lines (slanting to the left) alternating in foreground and background

3.27.2 bottomShadowPixmap

This resource specifies the pixmap of a window's bottom and right 3-D shadows
The mwm client uses this pixmap for inactive windows only.

Syntax

```
{menu[*menu_name]}
{icon            }
Mwm[*{                }]*resource: value
```

```
{client[*title]  }
{feedback        }
XPct*resource: value
{display }
{mode    }
{vtmenubar  [*{x97801  }]}
X97801-480[*{             {keyboard}}]*resource: value
{vtscrollbar            }
```

Value

pixmap
: File name of the pixmap in the directory specified by the bitmapDirectory resource (class: BitmapDirectory), or file name of the pixmap including the absolute path of the file.

background
: Pixmap in the color specified by the value assigned to the background resource instance (class: Background).

foreground
: Pixmap in the color specified by the value assigned to the foreground resource instance (class: Foreground).

25_foreground
: Color mix: 25% foreground and 75% background

50_foreground
: Color mix: 50% foreground and 50% background

75_foreground
: Color mix: 75% foreground and 25% background

horizontal
: Horizontal lines alternating in foreground and background

NULL No pixmap

vertical
: Vertical lines alternating in foreground and background

slant_right
: Diagonal lines (slanting to the right) alternating in foreground and background

slant_left
: Diagonal lines (slanting to the left) alternating in foreground and background

3.27.3 iconImageBottomShadowPixmap

This resource specifies the pixmap of an icon image's bottom and right 3-D shadows.

Syntax

```
{menu[*menu_name]}
{icon            }
Mwm[*{client[*title]  }]*resource: value
{feedback        }
{defaults        }
```

Value

pixmap
: Filename of the pixmap in the directory specified by the bitmapDirectory resource (class: BitmapDirectory), or filename of the pixmap including the absolute path of the file.

background
: Pixmap in the color specified by the value assigned to the background resource instance (class: Background).

foreground
: Pixmap in the color specified by the value assigned to the foreground resource instance (class: Foreground).

25_foreground
: Color mix: 25% foreground and 75% background

50_foreground
: Color mix: 50% foreground and 50% background

75_foreground
: Color mix: 75% foreground and 25% background

horizontal
: Horizontal lines alternating in foreground and background

NULL No pixmap

vertical
: Vertical lines alternating in foreground and background

slant_right
: Diagonal lines (slanting to the right) alternating in foreground and background

slant_left
: Diagonal lines (slanting to the left) alternating in foreground and background

3.27.4 matteBottomShadowPixmap

This resource specifies the pixmap of the bottom and right 3-D shadows of the matte. This resource is evaluated if the matteWidth resource is assigned a positive value, i.e. a matte exists.

Syntax

```
{menu[*menu_name]}
{icon            }
Mwm[*{client[*title]  }]*resource: value
{feedback        }
{defaults        }
```

Value

pixmap
: Filename of the pixmap in the directory specified by the bitmapDirectory resource (class: BitmapDirectory), or filename of the pixmap including the absolute path of the file.

background
: Pixmap in the color specified by the value assigned to the background resource instance (class: Background).

foreground

Pixmap in the color specified by the value assigned to the foreground resource instance (class: Foreground).

25_foreground
: Color mix: 25% foreground and 75% background

50_foreground
: Color mix: 50% foreground and 50% background

75_foreground
: Color mix: 75% foreground and 25% background

horizontal
: Horizontal lines alternating in foreground and background

NULL
: No pixmap

vertical
: Vertical lines alternating in foreground and background

slant_right
: Diagonal lines (slanting to the right) alternating in foreground and background

slant_left
: Diagonal lines (slanting to the left) alternating in foreground and background

3.28 BrownColor

3.28.1 brownColor

color setting for color terminals

Syntax

`XPct*`*`resource`*`: value`

Value

color Color name in the X server's color database

3.29 BrowserHasFocus

3.29.1 browserHasFocus

This resource determines governs whether the input focus is on the icon browser or on the login name entry box when the login window is first opened.

Syntax
`XLogin.Login.`*`resource`*`: `*`value`*

Value
True|False

3.30 ButtonBindings

3.30.1 buttonBindings

This resource specifies the name of a set of button bindings.
mwm looks for this set of button bindings in the file specified in the configFile resource instance (class: ConfigFile).
Refer to the "Key and button bindings" in the mwm description for more information.

Syntax

`Mwm*`*`resource`*`: `*`value`*

Value

name Name of a set of button bindings

3.31 C132

3.31.1 c132

This resource determines whether or not the DECCOLM escape sequence is accepted. The DECCOLM escape sequence is used to switch between 80 and 132 column mode. **[ESC]**[?31 switches you to 80 column mode and **[ESC]**[?3h to 132 column mode.

Syntax

`XTerm*vt100*`*`resource`*`: `*`value`*

Value

False The DECCOLM escape sequence is not accepted.
True The DECCOLM escape sequence is accepted.

3.32 CharClass

3.32.1 charClass

This resource specifies which characters in a terminal emulator are grouped together to form a character class.
Character classes are important when text is selected word by word. When you double-click the mouse button BSelect, successive characters belonging to the same character class are selected for the buffer.
You form a character class by assigning all characters of the desired class the same numeric value. The characters are specified by means of their decimal value in the character set table. You may select the decimal value of the position of any element of the class as the common numeric value.

Example:

The string

```
33:48,37:48,45-47:48,64:46
```

specifies that the exclamation mark ! (33), the percent sign % (37), the minus sign - (45), the period . (46) and the slash / (47) are to be in the same character class as normal characters and digits. They are therefore assigned the character 0 (48). In addition, the commercial at @ (64) is assigned to the character class to which the period (46) belongs. This also belongs to character class 48. These specifications are useful when selecting UNIX file names and electronic mail adresses word by word.

Syntax

XTerm*vt100**resource*: *value*

X97801-480[*x97801-480]**resource*: *value*

Value

character:*class*[,*character*:*class*]...

character
Decimal value of the position of the character (see table below). Single characters *character1* as well as character strings *character1-character22* may be substituted for *character*. Non-sequential value assignments must be separated by commas.

class Decimal ASCII value to which the characters of a class are assigned. The decimal ASCII value of any character of this class may be used.

In the case of xterm, you may specify values from 0 to 127 for *character code* and *charClass*; in the case of x97801-480, values from 0 to 255. These values are assigned to characters as follows:

NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT
0	1	2	3	4	5	6	7	8	9
NL	VT	NP	CR	SO	SI	DLE	DC1	DC2	DC3
10	11	12	13	14	15	16	17	18	19
DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS
20	21	22	23	24	25	26	27	28	29

RS	US	SP	!	"	#	$	%	&	'
30	31	32	33	34	35	36	37	38	39
(	)	*	+	,	-	.	/	0	1
40	41	42	43	44	45	46	47	48	49
2	3	4	5	6	7	8	9	:	;
50	51	52	53	54	55	56	57	58	59
<	=	>	?	@	A	B	C	D	E
60	61	62	63	64	65	66	67	68	69
F	G	H	I	J	K	L	M	N	O
70	71	72	73	74	75	76	77	78	79
P	Q	R	S	T	U	V	W	X	Y
80	81	82	83	84	85	86	87	88	89
Z	[	\	]	^	_	`	a	b	c
90	91	92	93	94	95	96	97	98	99
d	e	f	g	h	i	j	k	l	m
100	101	102	103	104	105	106	107	108	109
n	o	p	q	r	s	t	u	v	w
110	111	112	113	114	115	116	117	118	119
x	y	z	{	³	}	~	DEL	h80	h81
120	121	122	123	124	125	126	127	128	129
h82	h83	IND	NEL	h86	h87	HTS	h89	h8a	h8b
130	131	132	133	134	135	136	137	138	139
h8c	RI	SS2	SS3	DCS	h91	h92	h93	h94	h95
140	141	142	143	144	145	146	147	148	149
h96	h97	h98	h99	h9a	CSI	ST	h9d	h9e	h9f
150	151	152	153	154	155	156	157	158	159
SP	!	"	#	$	%	&	'	(	)
160	161	162	163	164	165	166	167	168	169
*	+	,	-	.	/	0	1	2	3
170	171	172	173	174	175	176	177	178	179
4	5	6	7	8	9	:	;	<	=
180	181	182	183	184	185	186	187	188	189
>	?	`A	'A	^A	~A	"A	oA	AE	C
190	191	192	193	194	195	196	197	198	199
`E	'E	^E	"E	`I	'I	^I	"I	D	~N
200	201	202	203	204	205	206	207	208	209
`O	'O	^O	~O	"O	x	0	`U	'U	^U
210	211	212	213	214	215	216	217	218	219
"U	'Y	P	ss	`a	'a	^a	~a	"a	oa

220	221	222	223	224	225	226	227	228	229
he6	he7	`e	'e	^e	"e	`i	'i	^i	"i
230	231	232	233	234	235	236	237	238	239
hf0	~n	`o	'o	^o	~o	"o	%	0	`u
240	241	242	243	244	245	246	247	248	249
'u	^u	"u	'y	P	"y				
250	251	252	253	254	255				

Table 81: Assignment of character codes and character classes

3.33 CharSetMapping

3.33.1 charSetMapping

This resource defines one or more translation tables for converting characters belonging to ISO-8859 character sets to characters belonging to the IBM character set (at present IBM Code Pages 437, 850, 852, 857, 866 and 869 are supported).
Keyboard input must be converted when working with the IBM character set, since the KeySyms supplied by the xmodmap table are always coded in the form of one of the ISO character sets (ISO-8859-*). You can use the charSetMode resource to specify which character set the terminal emulator uses.
There is a predefined translation table for all supported ISO character sets. This normally does not have to be changed.

Example:

The string

```
iso8859-1, ibm437 \
[ 0xcc : 0x49     \
0xa4 : 0xb0     \
0xb1 : 0xf1 ]   \
iso8859-2, ibm437 \
[ 0xc4 : 0x8e ]
```

maps the characters on the left, which belong to ISO character sets, to the corresponding characters on the right, which belong to the IBM 437 character set. In the example, three characters from the ISO 8859-1 character set are converted, followed by one belonging to ISO 8859-2.

In the case of characters for which no value is specified in the resource definition, the default applies.

Syntax

`XPct[*xpct]*`*`resource`*`:` *`value`*

Value

translation_table[*translation_table*]...

translation_table

Contains a translation table for converting characters belonging to an ISO character set to characters belonging to an IBM character set. Several translation tables can be defined.
translation_table has the following format:
character_set_pair [*list_of_character_code_pairs*]

character_set_pair

Contains the names of the character sets to be converted in the form:
character_set1,character_set2

character_set1

The name of the ISO character set. The possible values are:

ISO standard	Languages
iso8859-1	Danish, German, Faroese, Finnish, French, Icelandic, Italian, Irish, Dutch, Norwegian, Portuguese, Swedish, Spanish, English
iso8859-2	Albanian, German, Polish, Romanian, Serbo-Croat, Slovakian, Slovenian, Czech, Hungarian, English
iso8859-3	Afrikaans, German, Esperanto, French, Italian, Catalan, Maltese, Dutch, Spanish, Turkish, English
iso8859-4	Danish, German, Estonian, Finnish, French, Greenlandic, Lapp, Latvian, Lithuanian, Norwegian, Swedish, English
iso8859-5	Cyrillic characters, English
iso8859-7	Greek characters, English
iso8859-9	Variation of iso8859-1 with Turkish, without Icelandic and Faroese

character_set2
The IBM character set. Currently valid values: ibm437, ibm850, ibm852, ibm866 and 869.

[*list_of_character_code_pairs*]
The delimiters [and] enclose the list of character code pairs.
list_of_character_code_pairs is in the form:
ISO_character_code : *IBM_character_code*[,*ISO_character_code* : *IBM_character_code*]...

ISO_character_code
The hexadecimal character code of the character to which the IBM character is to be mapped

IBM_character_code
The decimal or hexadecimal character code of the IBM character to be mapped

3.34 CharSetMode

3.34.1 charSetMode

This resource specifies the character set mode in which the terminal emulator works.
The terminal emulator can use either the ISO or the IBM character set mode. You can select different character sets within the character set mode by means of the isoCharSet and ibmCharSet resources.
You can switch character set mode during runtime by means of an escape sequence.
The character set mode specified by means of the -charsetmode option also determines whether the terminal emulator uses the ISO or the IBM character classes. When you use an escape sequence to switch the character set mode during runtime, this does not affect which character class is used. This may produce unexpected behavior when text is selected word by word.
If the IBM character set mode is selected, the translation table defined by means of the charSetMapping resource is used for keyboard input.

Syntax

`XPct[*xpct]*`*`resource`*`: `*`value`*

Value

iso The terminal emulator uses the ISO character set mode and the ISO character classes.
ibm The terminal emulator uses the IBM character set mode and the IBM character classes.

3.35 CheckPathnames

3.35.1 checkPathnames

Specifies whether the session selection menu item defined under the name <sessionArgument> is to be displayed. The value of this resource defines a list of path names which must or must not exist in order for the session to appear in the selection list. The path names in the list are logically combined using the Boolean operators OR (|), AND (&) and NOT (!). If the expression evaluates to True, the session is displayed on the selection menu.

Syntax

```
xlogin*Session.<sessionArgument>.resource: value
```

Value

name [*NOT*]pathname[*AND*|*OR*[*NOT*]pathname]...
NOT !
AND &
OR |
The value must not contain any blanks. Blanks are significant, i.e. they form part of the path name.

Example

```
xlogin*Session.SINIXSession.checkPathnames: \
/usr/bin/X11/.xdeskdir/_xdesk&/usr/dt/bin/Xsession
```

The session named "SINIXSession" is displayed only if the two files /usr/bin/X11/.xdeskdir/_xdesk and /usr/dt/bin/Xsession exist.

3.36 Chooser

3.36.1 chooser

This resource defines the full pathname of the application which displays the host selection window. See section "Appearance and behavior" in the xdm description.
The host selection window offers the user a choice of icons from which to select the host whose xdm service is to be used for display management. xdm only starts this program, known as Chooser, if the server issues an indirect XDMCP request and the entry for that server in the X access file contains the keyword CHOOSER.
For further information refer to the xdm client description in Abschnitt "xdm - Display manager".

Syntax

`DisplayManager*`*`resource`*`: `*`value`*

Value

name Full pathname of the file

3.37 CleanText

3.37.1 cleanText

This resource determines the appearance of the area immediately surrounding text in title areas and feedback windows.

Syntax

`Mwm*`*`resource`*`: value`

Value

True Single-colored background
False Background as defined for this area

3.38 ClearCount

3.38.1 clearCount

This resource specifies the rhythm in which the client automatically deletes the beginning of previous output.
The savedLines resource of the SavedLines class allows you to specify how many lines at the end of the previous output are not to be deleted by the client.
The value of this resource is only interpreted when the clearText resource of the ClearText class has a value of true.

Syntax

```
XUtmp*resource: value
{ console  }
{ system   }
XMessages[*{          }]*resource: value
{ xerrors  }
{ common   }
```

Value

line_number
Number of lines after which the client begins to delete

3.39 ClearText

3.39.1 clearText

This resource specifies whether the client automatically deletes previous output.
The client deletes the previous output in a rhythm that you specify by means of the clearCount resource of the ClearCount class. The savedLines resource of the SavedLines class allows you to specify how many lines at the end of the previous output are not to be deleted by the client.

Syntax

```
{ console  }
{ system   }
XMessages[*{          }]*resource: value
{ xerrors  }
{ common   }
```

Value

True The client automatically deletes the beginning of previous output.
False The client deletes the beginning of previous output only if you request it.

3.40 ClientAutoPlace

3.40.1 clientAutoPlace

This resource determines the position of a window if you do not specify the position explicitly.

Syntax

`Mwm*`*`resource`*`: `*`value`*

Value

False The default position is used. The window manager will attempt to place the windows totally on-screen.
True The top left corner of the window is positioned.
The window manager positions the top left corners so that they are offset horizontally and vertically.
mwm attempts to place the windows totally on-screen.

3.41 ClientDecoration

3.41.1 clientDecoration

This resource controls the amount of window frame decoration.
clientDecoration is specified as a list of decoration elements, indicating their inclusion in or exclusion from the mwm window frame. If you put a minus sign before an element, that element is not required in the frame. If the first character in the list is a minus sign, mwm assumes that the frame is to include all the possible elements with the exception of those listed. If the first character in the list is not a minus sign (i.e. a plus sign or a letter), mwm assumes that the frame is to include only those elements which are explicitly listed.

Syntax

```
{client_class   }
Mwm[*{client_instance}]*resource: value
{defaults       }
```

Value

```
{ - }            [-]
{   }decoration[[ ]decoration]...
{[+]}            [+]
```

You can replace *decoration* with the following:

all all of the following objects
border
window border
maximize
maximize button
Note the value of the clientFunctions resource instance (class: ClientFunctions) with which the f.maximize function can be made unavailable.
menu menu button
minimize
minimize button
Note the value of the clientFunctions resource instance (class: ClientFunctions) with which the f.minimize function can be made unavailable.
none none of the listed objects
resizeh
resize borders
title title bar

Signs are interpreted as follows:

+ or no sign
mwm starts with no objects and adds the listed objects as decoration.
- mwm assumes all objects and subtracts the listed objects from that set.

3.42 ClientFunctions

3.42.1 clientFunctions

This resource specifies which mwm functions are applicable to the window.
The following mwm functions are applicable (or non-applicable): f.maximize, f.minimize, f.resize, f.move and f.kill. This resource does not apply to mwm functions not mentioned here.

Syntax

```
{client_class   }
Mwm[*{client_instance}]*resource: value
{defaults       }
```

Value

```
{ - }          [-]
{   }function[[ ]function]...
{[+]}          [+]
```

You can replace *function* with the following:

all all of the following mwm functions are applicable
close mwm function f.kill
maximize
mwm function f.maximize
minimize
mwm function f.minimize
move mwm function f.move
none none of the mwm functions mentioned is applicable
resize
mwm function f.resize

Signs are interpreted as follows:

+ or no sign
mwm starts with no functions and adds the listed functions.
- mwm assumes all functions and subtracts the listed functions from that set.

3.43 ClientIcon

3.43.1 ClientIcon

defines the icon used by the window manager to represent the client.

Syntax

`XMessages.`*`resource`*`:` *`value`*

Value

name Full pathname of the icon file

3.44 ColormapFocusPolicy

3.44.1 colormapFocusPolicy

This resource indicates the colormap focus policy to be used.

Syntax

`Mwm*`*`resource`*`: value`

Value

keyboard
: mwm always uses the colormap of the client whose window has the keyboard input focus.

explicit
: mwm always uses the colormap of the client in whose window a predefined selection action was performed. This predefined action might be pressing a key or button or selecting a menu item. For more information refer to the Abschnitt "Key and button bindings" in the mwm description.

pointer
: mwm always uses the colormap of the client whose window has the mouse input focus.

3.45 Compose

3.45.1 compose

This resource specifies whether or not compose sequences are to be interpreted.
This resource is only available if the eightbit resource instance (class: Eightbit) is assigned the value "True".

Syntax

`X97801-480[*x97801-480]*`*`resource: value`*

Value

True Compose sequences are interpreted.
The key to which the keysym XK_Multi_Key has been assigned starts a compose sequence. The xmodmap client will tell you to which of your keyboard keys the XK_Multi_Key keysym has been assigned.
False Compose sequences are not interpreted.

3.45.2 deadKey

This resource specifies whether or not dead key sequences are to be interpreted.
This resource is only available if the eightbit resource instance (class: Eightbit) is assigned the value "True".

Syntax

`X97801-480[*x97801-480]*`*`resource: value`*

Value

True Dead key sequences are interpreted.
False Dead key sequences are not interpreted.

3.46 Column

3.46.1 nMarginBell

This resource specifies the column at which the terminal emulator is to ring the margin bell. This resource is only available if the marginBell resource instance (class: MarginBell) is assigned the value "True".
The visualBell resource of class VisualBell governs whether the signal is acoustic or visual.

Syntax

`XPct[*xpct]*`*`resource`*`: `*`value`*

`XTerm[*vt100]*`*`resource`*`: `*`value`*

`X97801-480[*x97801-480]*`*`resource`*`: `*`value`*

Value

column
: specifies the column at which the margin bell is to ring; this is the distance from the right text margin expressed in columns.

3.47 Columns

3.47.1 columns

Number of columns in the text field

Syntax

`XMessages*`*`resource`*`: `*`value`*

Value

no_of_columns
Number of columns

3.48 ConfigFile

3.48.1 configFile

This resource specifies the path name of the mwm configuration file defining key and button bindings.
The entries in the file are separated by blanks, tabs or newlines. Blank lines are ignored. Comments are skipped. Comments are

- all characters in a line after the first unquoted hash sign (#).
- a whole line with an exclamation point (!) as its first character.

Entries can be placed in single quotes (') to be prevent interpretation of special characters. Entries in single quotes can be placed in double quotes ("). To prevent interpretation of a single special character you can precede it with a backslash (\). If a line ends with a backslash (\), mwm interprets the next line as a continuation of it.

Syntax

`Mwm*`*`resource`*`: value`

Value

~/name
: Filename *name* including the pathname relative to $HOME/$LANG, or $HOME if LANG is not set.

name
: Filename *name* including the absolute pathname, or a relative pathname to be interpreted relative to the current working directory.

3.49 Console

3.49.1 console

specifies whether all messages written to the system console are to be displayed..

Syntax

`XMessages*`*`resource`*`: `*`value`*

Value

False Messages are not displayed.
True Messages are displayed.

3.50 Cpp

3.50.1 cpp

This resource specifies the directory containing the C preprocessor used by the xrdb client.

Syntax

`DisplayManager*`*`resource`*`: `*`value`*

Value

name Absolute pathname

3.51 Csm

3.51.1 csm

specifies whether messages written to /dev/console are to be redirected to /dev/csm. Interpreted only if console is set to true and on system platforms which support /dev/csm.

Syntax

`XMessages*`*`resource`*`: `*`value`*

Value

False Messages are not redirected.
True Messages are redirected.

3.52 Curses

3.52.1 curses

This resource determines whether or not xterm outputs tabulator characters at the beginning of continuation lines when the "more" command is executed and the lines are longer than the width of the screen.

Syntax

`XTerm*vt100*`*`resource`*`: `*`value`*

Value

False Leading tabs are output.
True Leading tabs are not output.

3.53 Cursor

3.53.1 pointerShape

This resource specifies the shape of the pointer.

Syntax

```
XPct[*xpct]*resource: value
XTerm[*vt100]*resource: value
X97801-480[*x97801-480]*resource: value
```

Value

arrow Pointer in the shape of an arrowhead
xterm Pointer in the shape of a pipe (I)

3.54 CutNewline

3.54.1 cutNewline

This resource determines behavior when selecting text by lines and copying it to the cut buffer.

Syntax

`XPct[*xpct]*`*`resource`*`: `*`value`*

`X97801-480[*x97801-480]*`*`resource`*`: `*`value`*

Value

False The last line selected does not end with a newline character.
True The last line selected ends with a newline character.

3.55 CutToBeginningOfLine

3.55.1 cutToBeginningOfLine

This resource determines behavior when selecting text by lines and copying it to the cut buffer.

Syntax

`XPct[*xpct]*`*`resource`*`: `*`value`*

`X97801-480[*x97801-480]*`*`resource`*`: `*`value`*

Value

False A line is selected starting at the column in which the pointer is located.
True A line is always selected from the beginning of the line.

3.56 CyanColor

3.56.1 cyanColor

color setting for color terminals

Syntax

`XPct*`*`resource`*`: value`

Value

color Color name in the X server's color database

3.57 DaemonMode

3.57.1 daemonMode

Determines whether the client is started as a background process.
If the value of daemonMode is True, the client runs in the background. It makes itself independent of the parent process by using *fork* to spawn a new process. It also closes all file descriptors and is thus no longer associated with the controlling terminal.

Syntax

`DisplayManager.`*`resource`*`: `*`value`*

Value

True The client runs in the background after startup.
False The client runs in the foreground after startup.

3.58 Daemon

3.58.1 daemon

Determines whether the client is started as a background process.
If the value of daemon is True, the client runs in the background.

Syntax

`XMessages.`*`resource`*`:` *`value`*

Value

True The client runs in the background after startup.
False The client runs in the foreground after startup.

3.59 DebugLevel

3.59.1 debugLevel

This resource specifies whether or not the client runs in debug mode.
It is thus possible to have error messages displayed on the screen for diagnostic purposes and disable the daemonMode resource which makes xdm run in the background.

Syntax

`DisplayManager.`*`resource`*`: `*`value`*

Value

>0 The client runs in the foreground. Error messages are displayed on the screen.
0 The client runs in the background. Error messages are not displayed on the screen.

3.60 DefaultHost

3.60.1 defaultHost

Specifies a default host that is highlighted and selectable in the icon browser once the chooser window comes up.

Syntax

`Chooser*`*`resource`*`: `*`value`*

Value

Host name

3.61 DefaultIcon32

3.61.1 defaultIcon32

This resource specifies the icon and the icon mask to be used to represent the login names in the xdm login window or the hosts in the xdm host selection window.
This icon and the matching icon mask is used if

- icons are to be allowed.
  This is the case if the value of the allowIcons resource is True.
- the size of the icon in pixels is to be 24x24 (XPM format) or 32x32 (GIF format).
  This is the case if the value of the smallIconSize resource is True.
- the user manager or the host manager has not defined a custom icon for a particular user or host.

The icons must be in XPM, GIF or XBM format.

Syntax

`XLogin.Login.`*`resource`*`: `*`value`*
`Chooser*`*`resource`*`: `*`value`*

Value

name Name of the icon file

3.62 DefaultIcon64

3.62.1 defaultIcon64

This resource specifies the icon an icon mask to be used to represent the login names in the xdm login window or the hosts in the xdm host selection window.
This icon and the matching icon mask is used if

- icons are to be allowed.
  This is the case if the value of the allowIcons resource is True.
- the size of the icon in pixels is to be 48x48 (XPM format) or 64x64 (GIF format).
  This is the case if the value of the smallIconSize resource is False.
- the user manager or the host manager has not defined a custom icon for a particular user or host.

The icons must be in XPM, GIF or XBM format.

Syntax

`XLogin.Login.`*`resource`*`: `*`value`*
`Chooser*`*`resource`*`: `*`value`*

Value

name Name of the icon file

3.63 DefaultUser

3.63.1 defaultUser

Specifies a default user login name that is highlighted and selectable in the icon browser once the login window comes up.

Syntax
`XLogin.Login.`*`resource`*`: `*`value`*

Value
Login name

3.64 DefaultVirtualBindings

3.64.1 defaultVirtualBindings

This resource specifies the mapping of virtual buttons and keys to keys on your keyboard.

Syntax

`Mwm*`*`resource`*`: value`
`XPct[*xpct]*`*`resource`*`: value`
`X97801-480[*x97801-480]*`*`resource`*`: value`

Value

virtualKey:[*modifier*]...<Key>*key*[\n\
virtualKey:[*modifier*]...<Key>*key*]...

virtualKey
can be replaced by:

osfActivate	osfLeft
osfAddMode	osfMenu
osfBackSpace	osfMenuBar
osfBeginLine	osfPageDown
osfClear	osfPageUp
osfCopy	osfPaste
osfCut	osfPrimaryPaste
osfDelete	osfQuickPaste
osfDown	osfRight
osfEndLine	osfSelect
osfCancel	osfUndo
osfHelp	osfUp
osfInsert	

modifier
can be replaced by:

Ctrl	CTRL key
Shift	Shift key
Meta	Meta key
Lock	LOCK key
Mod1	Modifier key

	1
Mod2	Modifier key 2
Mod3	Modifier key 3
Mod4	Modifier key 4
Mod5	Modifier key 5

xmodmap will supply you with information on the mapping of the values for *modifier* to keys on the keyboard.

key You can use any character in the ISO Latin 1 character set.
xmodmap will supply you with information on which keys your display supports.

3.65 DeiconifyKeyFocus

3.65.1 deiconifyKeyFocus

This resource determines the input focus policy when an icon is deiconified. This resource is only available if the keyboardFocusPolicy resource instance (class: KeyboardFocusPolicy) is assigned the value "explicit".

Syntax

`Mwm*`*`resource`*`: `*`value`*

Value

False The input focus remains with the window or icon that currently has it.
True When an icon is deiconified, it receives the input focus.

3.66 DoHalfIntensity

3.66.1 doHalfIntensity

This resource determines whether text is displayed with the text attribute "reduced intensity".
This resource is only available if the terminal emulator is running on a display with a single color plane, i.e. monochrome display.

Syntax

`X97801-480[*x97801-480]*`*`resource: value`*

Value

True for 2-n pixel plane screens:
Text with the text attribute "half-intensity" appears dimmed.
for 1 pixel plane screens:
Text with the text attribute "half-intensity" is displayed against a patterned background.

False Text with the text attribute "reduced intensity" is displayed without a patterned background.

3.67 DoubleClickTime

3.67.1 doubleClickTime

This resource specifies the maximum amount of time that can elapse between two clicks in order for them to be interpreted as a double-click and not as two separate clicks on the same button.

Syntax

`Mwm*`*`resource`*`: value`

Value

milliseconds
: Time in milliseconds

3.68 Eightbit

3.68.1 eightbit

This resource determines whether the terminal emulator is started in 8-bit mode.

Syntax

`X97801-480[*x97801-480]*resource: value`

Value

True The terminal emulator is started in 8-bit mode.
False The terminal emulator is started in 7-bit mode.

3.69 EnableWarp

3.69.1 enableWarp

This resource specifies whether or not the pointer is positioned in the middle of the selected window when you use the cursor keys to change the size or position of windows.

Syntax

`Mwm*`*`resource`*`: `*`value`*

Value

True The pointer is positioned in the middle of the selected window.
False The pointer retains its current position.

3.70 EnforceKeyFocus

3.70.1 enforceKeyFocus

Determines whether the policy requiring a window to be explicitly assigned the input focus applies to all windows.
This applies especially to "globally active" input windows. An example of a globally active window is a scroll bar.

Syntax

`Mwm*`*`resource`*`: `*`value`*

Value

False You must click the window or icon that is to have the input focus.
True The input focus is temporarily set to the window or icon that is active, regardless of the input focus, for the duration of input.

3.71 ErrorLogFile

3.71.1 errorLogFile

This resource indicates the pathname of the file in which client error messages are logged. Errors assignable to a user session are written to the appropriate $HOME/.xerrors file.

Syntax

`DisplayManager.`*`resource`*`: `*`value`*

Value

name Filename *name* including the full pathname.

3.72 ExitOnFail

3.72.1 exitOnFail

Specifies wether the client is to exit when unable to redirect system console message output.

Syntax

`XMessages*`*`resource`*`: `*`value`*

Value

False The client is not to exit.
True The client is to exit.

3.73 ExportList

3.73.1 exportList

This resource defines a list of environment variables to be exported to the xdm configuration files Xsetup, Xstartup, Xsession and Xreset in addition to the environment variables always exported to these files.

Syntax

`DisplayManager.`*`resource`*`: `*`value`*

Value

list List of environment variables in the form:
environment_variable[*environment_variable*]...

3.74 FadeNormalIcon

3.74.1 fadeNormalIcon

This resource indicates whether an icon in the icon box is grayed out (faded) when the associated window has been opened.

Syntax

`Mwm*`*`resource`*`: `*`value`*

Value

False An icon whose associated window has been opened is not grayed out.
True An icon whose associated window has been opened is grayed out.

3.75 Fail

3.75.1 fail

This resource defines the text of the message which is displayed when user authentication fails.

Syntax

`XLogin.Login.`*`resource`*`: `*`value`*

Value

string
: message text

3.76 FailTimeout

3.76.1 failTimeout

This resource specifies the time (in seconds) that the failure message is displayed. This is the error message displayed in the xdm login window when a user authentication fails.

Syntax

`XLogin.Login.`*`resource`*`: `*`value`*

Value

sec Time in seconds

3.77 FailsafeClient

3.77.1 failsafeClient

This resource specifies the program to which the client falls back if the default session fails to execute.
The default session is the session that should be started for the user according to the specifications in the file Xsession or $HOME/.xsession.
The program is executed without arguments, but uses the same environment variables as the default session would have used.

Syntax

`DisplayManager.DISPLAY.`*`resource`*`: `*`value`*

Value

name Filename *name* including the pathname relative to $HOME.
name Filename *name* including the absolute pathname.

3.78 FeedbackGeometry

3.78.1 feedbackGeometry

This resource specifies the position of the window in which the client during an action outputs the current result of the action.
mwm gives you feedback of this kind when you use the controls in the window frame to change the position or size of a window.

Syntax

```
Mwm*resource: value
```

Value

```
{+}        {+}
{ }x-offset{ }y-offset
{-}        {-}
```

x-offset
: Horizontally measured distance of the window from the edge of the screen, in pixels
+ sign Distance from the left edge of the screen
- sign Distance from the right edge of the screen

y-offset
: Vertically measured distance of the window from the edge of the screen, in pixels
+ sign Distance from the top of the screen
- sign Distance from the bottom of the screen

3.79 File

3.79.1 file

Specifies the name of a file which is to be monitored if the xerrors resource is set. Any messages which clients write to standard output and standard error after xmessages startup are redirected to this file.

Syntax

`XMessages.`*`resource`*`: `*`value`*

Value

name Full pathname of the file

3.80 FocusAutoRaise

3.80.1 focusAutoRaise

This resource specifies whether mwm raises a client window to the top of the stack, thus making it completely visible, when the window is assigned the keyboard input focus.

Syntax

```
{client_class   }
Mwm[*{client_instance}]*resource: value
{defaults       }
```

Value

False A window assigned the input focus is not automatically raised to the top of the stack.
True The window manager automatically moves a window to the top of the stack when it gets the input focus.

3.81 Font

3.81.1 boldFont

This resource specifies the font to be used for bold text in windows and icons.

Syntax

`XTerm*vt100*`*`resource`*`: value`

Value

fontname
: Name of a font the X server can find in its font path. The xlsfonts client lists the applicable font names for the X server.

3.81.2 failFont

This resource specifies the font used to display the authentication failure message.

Syntax

`XLogin.Login.`*`resource`*`: value`

Value

fontname
: Name of a font the X server can find in its font path. The xlsfonts client lists the applicable font names for the X server.

3.81.3 font

This resource specifies the font to be used for text in windows and icons.

Syntax

`XLogin.Login.`*`resource`*`: value`

`XTerm*vt100*`*`resource`*`: value`

Value

fontname
: Name of a font the X server can find in its font path. The xlsfonts client lists the applicable font names for the X server.

3.81.4 greetFont

This resource specifies the font to be used for the welcome message in the xdm login window or host selection window.

Syntax

`Chooser*`*`resource`*`: value`

`XLogin.Login.`*`resource`*`:` *`value`*

Value

fontname
: Name of a font the X server can find in its font path. The xlsfonts client lists the applicable font names for the X server.

3.81.5 promptFont

This resource specifies the font used to display prompts.

Syntax

`XLogin.Login.`*`resource`*`:` *`value`*

Value

fontname
: Name of a font the X server can find in its font path. The xlsfonts client lists the applicable font names for the X server.

3.82 FontList

3.82.1 fontList

Font list
mwm uses the first font in the list for the title area in the mwm window frame. The characters in this font should be characters for text.
xpct uses the font list for character output in the menu bar.
xmessages uses the font list for character output.
The x97801-480 uses a maximum of the first 16 fonts in the list to emulate the 97801-480 terminal character generator. The x97801-480 terminal emulator evaluates this resource only if no value has been assigned to the font resource of the Font class.

Syntax

```
CommunicationManager*resource: value
{menu[*menu_name]}
{icon            }
Mwm[*{               }]*resource: value
{client[*title]  }
{feedback        }
Xpct*vtmenubar*resource: value
XMessages.resource: value
X97801-480[*x97801-480]*resource: value
```

Value
fontname[=*alias*][,*fontname*[=*alias*]]...

fontname
: Name of a font the X server can find in its font path. The xlsfonts client lists the applicable font names for the X server.

alias You can allocate every font *fontname* an alias *alias*.

3.82.2 ibmFontList

This resource defines a list of fonts for the display of the terminal emulator when an IBM character set is selected.
The charSetMode resource determines whether an IBM or ISO character set is used. The ibmCharSet resource determines which IBM character set is used. At present IBM Code Page 437, 850, 852, 857, 866 and 869 character sets are supported. The font list therefore cannot have more than one entry.
The font used must be the same size as the fonts in the ISO font list. Proportional fonts cannot be used.
If the specified font does not exist, a replacement font is used instead (e.g. fixed).

Syntax

```
XPct[*xpct]*resource: value
```

Value
fontname[=*alias*][,*fontname*[=*alias*]]...

fontname
Font name of a font that the X server finds in its font path.
alias You can assign a name (*alias*) to each font (*fontname*).

3.82.3 isoFontList

This resource defines a list of fonts for the display of the terminal emulator when an ISO character set is selected.
The charSetMode resource determines whether an IBM or ISO character set is used. The ibmCharSet resource determines which ISO character set is used. A font can be specified for each ISO character set.
The font used must be the same size as the fonts in the IBM font list. Proportional fonts cannot be used.
If the specified font does not exist, a replacement font is used instead (e.g. fixed).

Syntax
`XPct[*xpct]*resource: value`

Value
fontname[=*alias*][,*fontname*[=*alias*]]...

fontname
Font name of a font that the X server finds in its font path.
alias You can assign a name (*alias*) to each font (*fontname*).

3.83 FontMapping

3.83.1 fontMapping

This resource specifies the mapping of characters used by the 97801-480 terminal emulator's character generator to glyphs in the font with the alias *alias*.
You must assign a font to the alias *alias* in the fontList resource instance (class: FontList).

Syntax

`X97801-480[*x97801-480]*resource: value`

Value

$character_{1,1}$[-$character_{2,1}$]=$alias_1$:$glyph_{1,1}$[-$glyph_{2,1}$][,
$character_{1,2}$[-$character_{2,2}$]=$alias_2$:$glyph_{1,2}$[-$glyph_{2,2}$]]...
The terminal emulator's character generator maps every *character* to a glyph *glyph* in the font with alias *alias*. The character generator thus represents the character *character* as a glyph *glyph* in the font with alias *alias*.
In the following, *character* is the character with the unique address *character* for this character in the character generator. Similarly, *glyph* is the glyph with the unique address *glyph* for this glyph in a font.
The terminal emulator's character generator uses the characters $0 \leq character < 1024$. For further information on the character generator addresses, please refer to the sections dealing with character sets in your operating system interfaces documentation.
If *character* is between $character_{1,j}$ and $character_{2,j}$, the character generator maps this character to the glyph $glyph = glyph_{1,j} + (character - character_{1,j})$ in the font $alias_j$, provided that $glyph \leq glyph_{2,j}$. If $glyph > glyph_{2,j}$, it maps this character to the glyph $glyph = glyph_{2,j}$ in the font with the alias $alias_j$.

$character_{1,j}$
: Lowest address of the character that the 97801-480 terminal emulator's character generator should map to a character in the font with the alias $alias_i$.

$character_{2,j}$
: Highest address of the character that the 97801-480 terminal emulator's character generator should map to a character in the font with the alias $alias_i$. If you omit $character_{2,j}$, the character generator uses $character_{2,j} = character_{1,j}$

$alias_j$
: A font alias that has been assigned to a font in the fontList resource instance (class: FontList).

$glyph_{1,j}$
: Lowest address of a character in the font with the alias $alias_j$ that is allocated the address of a character in the character generator. You can use only the numbers 0 to 255 as address $glyph_{1,j}$.

$glyph_{2,j}$
: Highest address of a character in the font with the alias $alias_j$ that is allocated the address of a character in the character generator. If you omit $glyph_{2,j}$, the character

generator uses
$glyph_{2,j} = glyph_{1,j} + (character_{2,j} - character_{1,j})$.
You can use only the numbers 0 to 255 as address $glyph_{2,j}$.

3.84 Foreground

3.84.1 activeBottomShadowColor

This resource specifies the color of the active window's bottom and right 3-D shadows.

Syntax

```
{menu[*menu_name]     }
{icon[*defaults]      }
Mwm[*{                      }]*resource: value
{client[*title]       }
{feedback             }
```

Value

color Color name in the X server's color database.

3.84.2 activeForeground

This resource specifies the foreground color of the active window.

Syntax

```
{menu[*menu_name]     }
{icon[*defaults]      }
Mwm[*{                      }]*resource: value
{client[*title]       }
{feedback             }
```

Value

color Color name in the X server's color database

3.84.3 blinkColor

This resource specifies the color to be used instead of the attribute "blinking".

Syntax

```
XPct[*xpct]*resource: value
X97801-480[*x97801-480]*resource: value
```

Value

color
Color name in the X server's color database

3.84.4 bottomShadowColor

This resource specifies the color of a window's bottom and right 3-D shadows. The mwm client uses this color for inactive windows only.

Syntax

```
{menu[*menu_name]     }
{icon[*defaults]      }
Mwm[*{                     }]*resource: value
{client[*title]       }
{feedback             }
XPct*resource: value
{display }
{mode    }
{vtmenubar  [*{x97801  }]}
X97801-480[*{              {keyboard} }*resource: value
{vtscrollbar             }
```

Value

color Color name in the X server's color database

3.84.5 cursorColor

This resource specifies the color to be used for the text cursor.

Syntax

```
XPct[*xpct]*resource: value
XTerm*vt100*resource: value
X97801-480[*x97801-480]*resource: value
```

Value

color Color name in the X server's color database

3.84.6 failColor

This resource specifies the color used to display the failure message.
This is the error message displayed in the login window when a user authentication fails.

Syntax

```
XLogin.Login.resource:  value
```

Value

color Color name in the X server's color database

3.84.7 foreground

This resource specifies the color to be used for a window's foreground.
The foreground color is the color for text output.
The mwm client uses this color for inactive windows only.

Syntax

```
{menu[*menu_name]     }
```

```
Mwm[*{icon[*defaults]     }]*resource: value
{                    }
{client[*title]      }
{feedback            }
XLogin.Login.resource: value
XPct*resource: value
XTerm*vt100*resource: value
XMessages*resource: value
{display }
{mode    }
{vtmenubar  [*{x97801  }]}
X97801-480[*{            {keyboard} }]*resource: value
{vtscrollbar            }
```

Value

color Color name in the X server's color database

3.84.8 greetColor

This resource specifies the color of the login greeting.

Syntax

```
Chooser*resource: value
XLogin.Login.resource: value
```

Value

color
Color name in the X server's color database

3.84.9 iconImageBottomShadowColor

This resource specifies the color of an icon image's bottom and right 3-D shadows.

Syntax

```
{client_class   }
Mwm[*{client_instance}]*resource: value
{defaults       }
```

Value

color
Color name in the X server's color database

3.84.10 iconImageForeground

This resource specifies the foreground color of icon images.

Syntax

```
{client_class   }
Mwm[*{client_instance}]*resource: value
{defaults       }
```

Value

color
Color name in the X server's color database

3.84.11 matteBottomShadowColor

This resource specifies the color of the bottom and right 3-D shadows of the matte.
This resource is evaluated if the matteWidth resource is assigned a positive value, i.e. a matte exists.

Syntax

```
{client_class   }
Mwm[*{client_instance}]*resource: value
{defaults       }
```

Value

color Color name in the X server's color database

3.84.12 matteForeground

This resource specifies the foreground color of the matte.
This resource is evaluated if the matteWidth resource is assigned a positive value, i.e. a matte exists

Syntax

```
{client_class   }
Mwm[*{client_instance}]*resource: value
{defaults       }
```

Value

color Color name in the X server's color database

3.84.13 pointerColor

This resource specifies the color to be used for the pointer.

Syntax

```
XPct[*xpct]*resource: value
XTerm[*vt100]*resource: value
X97801-480[*x97801-480]*resource: value
```

Value

color Color name in the X server's database

3.84.14 promptColor

This resource specifies the color used to display prompts.

Syntax

`XLogin.Login.`*`resource`*`: `*`value`*

Value

color Color name in the X server's color database

3.85 FunctionKeyMapping

3.85.1 functionKeyMapping

Controls double mapping of the function keys.

Syntax

`X97801-480*`*`resource`*`: `*`value`*

Value

True The terminal emulator interprets the action of pressing the <Control> key and a function key F<number> at the same time as the function key press F<number + 10>. Example: <Control> + <F2> is interpreted as <F12>.

False The terminal emulator does not assign a special meaning to the action of pressing the <Control> key and function key F<number> at the same time.

3.86 FrameBorderWidth

3.86.1 frameBorderWidth

This resource specifies the width (in pixels) of a window border without resize handles. The border width includes the 3-D shadows.
It is a good idea to make sure the value specified for this resource is compatible with the resource instance resizeBorderWidth (class: ResizeBorderWidth).

Syntax

`Mwm*`*`resource`*`: value`

Value

pixels
: Width of the window border in pixels

3.87 G0

3.87.1 g0

This resource specifies the mapping of the G0 allocation area for character sets to a particular character set.
You thus specify which character set can be used in the work area of the character generator. The character set determines the mapping of the character codes to characters and thus which characters you are able to use. The appearance of these characters is, however, not specified. The appearance of characters is specified by the fontMapping resource (class: FontMapping).

Syntax

`X97801-480[*x97801-480]*`*`resource`*`: `*`value`*

Value

char_set
char_set can be replaced by:

BLA Blanks
DRCS The character set loaded in the DRCS area. In the default configuration this is the character set International A.
EURO Euro
FAC Facet
GER German
IBM IBM
INT International
INTA International A
ISO1 ISO 8859-1, upper half
ISO2 ISO 8859-2, upper half
ISO3 ISO 8859-3, upper half
ISO4 ISO 8859-4, upper half
ISO5 ISO 8859-5, upper half
ISO7 ISO 8859-7, upper half
ISO9 ISO 8859-9, upper half
KLA Brackets
MAT Math
For further information on character sets, please refer to the index of the SINIX Interfaces manual, under "character sets".

3.88 G1

3.88.1 g1

This resource specifies the mapping of the G1 allocation area for character sets to a particular character set.
You thus specify which character set can be used in the work area of the character generator. The character set determines the mapping of the character codes to characters and thus which characters you are able to use. The appearance of these characters is, however, not specified. The appearance of characters is specified by the fontMapping resource (class: FontMapping).

Syntax

`X97801-480[*x97801-480]*`*`resource: value`*

Value

char_set
char_set can be replaced by:

BLA Blanks
DRCS The character set loaded in the DRCS area. In the default configuration this is the character set International A.
EURO Euro
FAC Facet
GER German
IBM IBM
INT International
INTA International A
ISO1 ISO 8859-1, upper half
ISO2 ISO 8859-2, upper half
ISO3 ISO 8859-3, upper half
ISO4 ISO 8859-4, upper half
ISO5 ISO 8859-5, upper half
ISO7 ISO 8859-7, upper half
ISO9 ISO 8859-9, upper half
KLA Brackets
MAT Math
For further information on character sets, please refer to the index of the SINIX Interfaces manual, under "character sets".

3.89 G2

3.89.1 g2

This resource specifies the mapping of the G2 allocation area for character sets to a particular character set.
You thus specify which character set can be used in the work area of the character generator. The character set determines the mapping of the character codes to characters and thus which characters you are able to use. The appearance of these characters is, however, not specified. The appearance of characters is specified by the fontMapping resource (class: FontMapping).

Syntax

`X97801-480[*x97801-480]*`*`resource: value`*

Value

char_set
char_set can be replaced by:
BLA
Blanks

DRCSThe character set loaded in the DRCS area. In the default configuration this is the character set International A.
EUROEuro
FAC Facet
GER German
IBM IBM
INT International
INTA International A
ISO1 ISO 8859-1, upper half
ISO2 ISO 8859-2, upper half
ISO3 ISO 8859-3, upper half
ISO4 ISO 8859-4, upper half
ISO5 ISO 8859-5, upper half
ISO7 ISO 8859-7, upper half
ISO9 ISO 8859-9, upper half
KLA Brackets
MAT Math
For further information on character sets, please refer to the index of the SINIX Interfaces manual, under "character sets".

3.90 G3

3.90.1 g3

This resource specifies the mapping of the G3 allocation area for character sets to a particular character set.
You thus specify which character set can be used in the work area of the character generator. The character set determines the mapping of the character codes to characters and thus which characters you are able to use. The appearance of these characters is, however, not specified. The appearance of characters is specified by the fontMapping resource (class: FontMapping).

Syntax

`X97801-480[*x97801-480]*`*`resource: value`*

Value

char_set
char_set can be replaced by:
BLA
Blanks

DRCSThe character set loaded in the DRCS area. In the default configuration this is the character set International A.
EUROEuro
FAC Facet
GER German
IBM IBM
INT International
INTA International A
ISO1 ISO 8859-1, upper half
ISO2 ISO 8859-2, upper half
ISO3 ISO 8859-3, upper half
ISO4 ISO 8859-4, upper half
ISO5 ISO 8859-5, upper half
ISO7 ISO 8859-7, upper half
ISO9 ISO 8859-9, upper half
KLA Brackets
MAT Math
For further information on character sets, please refer to the index of the SINIX Interfaces manual, under "character sets".

3.91 Geometry

3.91.1 geometry

This resource indicates the position and size of a window.
The X97801-480 and xpct clients do not interpret specifications indicating the width and height of a window.
The window manager can ignore the value specified for this resource.

Syntax

```
XPct[*xpct]*resource: value
XTerm*vt100*resource: value
X97801-480[*x97801-480]*resource: value
XMessages*resource: value
```

Value

```
{+}        {+}
[widthxheight][{ }x-offset{ }y-offset]
{-}        {-}
```

width Width of the window in pixels
height
Height of the window in pixels
x-offset
Horizontal offset (in pixels) of the window from the edge of the screen
+ Offset from left edge of screen
- Offset from right edge of screen
y-offset
Vertical offset (in pixels) of the window from the edge of the screen
+ Offset from top of screen
- Offset from bottom of screen

3.91.2 tekGeometry

This resource indicates the position and size of the tek4014 widget (class: Tek4014).
The window manager can override the value specified for this resource.

Syntax

```
XTerm*vt100*resource: value
```

Value

```
{+}        {+}
[widthxheight{ }x-offset{ }y-offset]
{-}        {-}
```

width Width of the tek4014 widget in pixels
height
Height of the tek4014 widget in pixels
x-offset

Horizontal offset (in pixels) of the tek4014 widget from the edge of the screen
+ Offset from left edge of screen
- Offset from right edge of screen

y-offset

Vertical offset (in pixels) of the tek4014 widget from the edge of the screen
+ Offset from top of screen
- Offset from bottom of screen

3.92 GrabServer

3.92.1 grabServer

This resource grabs the server while a user's login name and password are being read in. The client grabs the server and the keyboard while reading in the login name and password. The server can be grabbed for the entire duration of this process or only until the keyboard is locked.
The grabTimeout resource is used to specify how long the client will wait for the grab to succeed. If the grab fails, xdm restarts the server and the session.
Definitions in the xdm Xsetup file cannot be executed unless the value of this resource is False.

Syntax

`DisplayManager.DISPLAY.`*`resource`*`: `*`value`*

Value

True The server is grabbed for the entire duration of login name/password reading.
False The server is grabbed only until the keyboard is locked.

3.93 GrabTimeout

3.93.1 grabTimeout

This resource specifies the maximum time the client will wait for a server and keyboard grab to succeed.
In order to compensate for the security shortcomings of the X protocol, xdm grabs the server and the keyboard while reading in the user's login and password. The grab may fail if another client has already grabbed the server and the network latencies are very high. Before setting the timeout to a high value, please consider the fact that a user could be fooled by a look-alike window on the screen. Failure to grab the server within the specified time results in xdm terminating the X server and the connection between xdm and the X server being rebuilt.

Syntax

`DisplayManager*`*`resource`*`: `*`value`*

Value

seconds
Time in seconds

3.94 GreenColor

3.94.1 greenColor

color setting for color terminals

Syntax

`XPct*`*`resource`*`: value`

Value

color Color name in the X server's color database

3.95 Greeting

3.95.1 greeting

This resource defines the text of the welcome message displayed when the user has logged in.

Syntax

`Chooser*`*`resource`*`: value`
`XLogin.Login.`*`resource`*`: value`

Value

string
Character string defining the message text.

3.96 GreyColor

3.96.1 greyColor

color setting for color terminals

Syntax

`XPct*`*`resource`*`: value`

Value

color Color name in the X server's color database

3.97 Header

3.97.1 header

Defines a title line for messages appearing in the message window. This title line appears whenever the message source changes.

Syntax

```
{ console  }
{ system   }
XMessages[*{          }]*resource: value
{ xerrors  }
{ common   }
```

Value

string
: Text string appearing in the title line in the message window

3.98 Height

3.98.1 height

This resource specifies the height of a window (in pixels).

Syntax

```
Chooser*resource: value
XLogin.Login.resource: value
XTerm*tek4014*resource: value
```

Value

pixels
: Height of the Tektronix emulator window in pixels

3.99 Hostname

3.99.1 hostname

This resource specifies the string displayed as the hostname in the login window. A string longer more than 32 characters long is truncated to 32 characters.
The value of the showHostname resource instance specifies whether or not a hostname is displayed.

Syntax

`XLogin.Login.`*`resource`*`: `*`value`*

Value

string
Name of the host

3.100 IbmCharClass

3.100.1 ibmCharClass

This resource specifies which characters of the IBM character set are grouped together in a character class.
Only the IBM Code Page 437 character set is currently supported.
Character classes are important when text is selected word by word. When you double-click the mouse button BSelect, successive characters of the same character class are selected for the text buffer.
The terminal emulator uses the IBM character classes when the character set mode is set to IBM by means of the charSetMode resource or -charsetmode option. The character class selected continues to be used even if you use an escape sequence to switch character set mode during runtime. This may produce unexpected behavior when text is selected word by word.
You create a character class by assigning a common numerical value to all characters of a particular class. The individual characters are specified by their decimal values in the character set table. You can select the decimal value of any character in the class as the common numerical value.
Example: The string
`33:48,37:48,45-47:48,64:46`
specifies that the exclamation mark ! (33), the percent sign % (37), the minus sign - (45), the period . (46) and the slash / (47) are to be in the same character class as normal characters and digits. They are therefore assigned the character 0 (48). In addition, the commerical at @ (64) is assigned to the character class to which the period (46) belongs. This also belongs to character class 48. These specifications are useful when selecting UNIX file names and electronic mail addresses word by word.

Syntax

`XPct[*xpct]*`*`resource`*`: `*`value`*

Value

character_code:*class_character*[,*character_code*:*class_character*]...

character_code
: Decimal character code of the character. You can use either a single character (*character_code1*) or a string of characters (*character_code1-character_code22*) for *character_code*. Several non-successive value assignments are separated by commas.

class_character
: Decimal character code of a character to which the characters of a class are assigned. You can select the decimal character code of any character in the class.

3.101 IbmCharSet

3.101.1 ibmCharSet

This resource specifies the character set with which the terminal emulator works when the character set mode is set to IBM.
You set the character set mode using the charSetMode resource.
At present IBM Code Pages 437, 850, 852, 857, 866 and 869 are supported.

Syntax

`XPct[*xpct]*`*`resource`*`: `*`value`*

Value

character set
Name of the IBM character set. Possible values:

ibm437
IBM character set Code Page 437 (default)

ibm850
IBM character set Code Page 850

ibm852
IBM character set Code Page 852

ibm857
IBM character set Code Page 857

ibm866
IBM character set Code Page 866

ibm869
IBM character set Code Page 869

3.102 IconAutoPlace

3.102.1 iconAutoPlace

This resource determines the position of the icons on the screen.

Syntax

`Mwm*`*`resource`*`: value`

Value

False The position of the icons corresponds to the position of the iconified screen.
True mwm determines the position of icons on the screen in the area of the screen defined by the value of the iconPlacement resource.

3.103 IconBackground

3.103.1 iconBackground

This resource defines the background color for icons in the login window or host selection window.
This resource is evaluated only if the icon is stored in XBM format.

Syntax

`XLogin.Login.`*`resource`*`: `*`value`*
`Chooser*`*`resource`*`: `*`value`*

Value

color Color name in the X server's color database

3.104 IconBoxGeometry

3.104.1 iconBoxGeometry

This resource indicates the position and the size of the icon box in number of icons.
The icons are arranged horizontally if the width specified is greater than the height and the resource instance iconPlacement (class: IconPlacement) has not already been assigned a value explicitly.
The icons are arranged vertically if the height specified is greater than the width and the resource instance iconPlacement (class: IconPlacement) has not already been assigned a value explicitly.

Syntax

```
Mwm*resource: value
```

Value

```
{+}        {+}
[widthxheight{ }x-offset{ }y-offset]
{-}        {-}
```

width Width of the icon box in number of icons

height

Height of the icon box in number of icons

x-offset

Horizontal offset (in pixels) of the icon box from the edge of the screen

+ Offset from left edge of screen
- Offset from right edge of screen

y-offset

Vertical offset (in pixels) of the icon box from the edge of the screen

+ Offset from top of screen
- Offset from bottom of screen

3.105 IconBoxName

3.105.1 iconBoxName

This resource specifies the icon box name that mwm uses to look up resources for the icon box.
Please note the difference between this resource and the resource instance iconBoxTitle (class: IconBoxTitle).

Syntax

`Mwm*`*`resource`*`: `*`value`*

Value

name Icon box name used for looking up resources

3.106 IconBoxSBDisplayPolicy

3.106.1 iconBoxSBDisplayPolicy

This resource determines the strategy according to which the window manager displays scroll bars in the icon box.

Syntax

`Mwm*`*`resource`*`: value`

Value

all mwm displays both a horizontal and a vertical scroll bar in the icon box.

horizontal
: mwm displays a single, horizontal scroll bar; the icon box is arranged vertically, even if otherwise specified in the iconBoxGeometry resource instance.

vertical
: mwm displays a single, vertical scroll bar; the icon box is arranged horizontally, even if otherwise specified in the iconBoxGeometry resource instance.

3.107 IconBoxTitle

3.107.1 iconBoxTitle

This resource specifies the name that is to be used in the title area of the icon box.

Syntax

`Mwm*`*`resource`*`: value`

Value

name Name of the used in the title area of the icon box

3.108 IconClick

3.108.1 iconClick

This resource specifies whether the window menu is posted and remains posted when an icon is clicked.
This resource instance is only available if the icons are not managed in the icon box.

Syntax

`Mwm*`*`resource`*`: `*`value`*

Value

False The window menu is only posted for as long as the button remains pressed.
True The window menu is posted and left posted when an icon is clicked.

3.109 IconDecoration

3.109.1 iconDecoration

This resource defines general icon decoration, which is the result combining the attributes/values listed below:

activelabel
: When the icon clicked on, the text in the icon label is displayed in full (see label below).

imageThe icon image is displayed.

label The icon label is displayed; if necessary, the label will be right-truncated to the width of the icon.

Syntax

`Mwm*`*`resource`*`: value`

Value

label only the icon label

imageonly the icon image

label image
: icon label and icon image

activelabel label image
: the icon label of an active icon is not truncated

3.110 IconForeground

3.110.1 iconForeground

This resource defines the foreground color for icons in the login window or host selection window.
This resource is evaluated only if the icon is stored in XBM format.

Syntax

`XLogin.Login.`*`resource`*`: `*`value`*
`Chooser*`*`resource`*`: `*`value`*

Value

color Color name in the X server's color database

3.111 IconGeometry

3.111.1 iconGeometry

This resource specifies the position and size (in pixels) of an icon.
This resource instance is only available if the icons are not managed in the icon box. mwm can override the value specified.

Syntax

```
XPct[*xpct]*resource: value
XTerm*resource: value
X97801-480[*x97801-480]*resource: value
```

Value

```
{+}        {+}
[widthxheight{ }x-offset{ }y-offset]
{-}        {-}
```

width Width of the icon in pixels

height
Height of the icon in pixels

x-offset
Horizontal offset (in pixels) of the icon from the edge of the screen
+ Offset from left edge of screen
- Offset from right edge of screen

y-offset
Vertical offset (in pixels) of the icon from the edge of the screen
+ Offset from top of screen
- Offset from bottom of screen

3.112 IconImage

3.112.1 iconImage

This resource specifies the filename of the bitmap file for an icon image. The pathname of the bitmap file may also be specified.
If this resource is enabled and if the value of the resource useIconClient is "False", no values for an icon image assigned by other clients take effect. The mwm client evaluates the client-specific resource useClientIcon to determine whether to use a client-supplied pixmap or a user-supplied pixmap for the icon image.

Syntax

```
{client_class   }
Mwm[*{client_instance}]*resource: value
{defaults       }
```

Value

~/name
Filename *name* including the pathname relative to $HOME.
name Filename *name* including the absolute pathname.

3.113 IconImageMaximum

3.113.1 iconImageMaximum

This resource specifies the maximum size (in pixels) of an icon image.
The size of an icon image is the size of the image added to the width of the icon frame. The maximum supported size is 128 x 128 (pixels). If the specified values are higher, the icon image is truncated to the right and bottom so that it has the maximum size.

Syntax

`Mwm*`*`resource`*`: value`

Value

*width*x*height*
: Width x height of the icon image in pixels

3.114 IconImageMinimum

3.114.1 iconImageMinimum

This resource specifies the minimum size (in pixels) of an icon image.
The size of an icon image is the size of the image added to the width of the icon frame. The minimum supported size is 16 x 16 (pixels). If the specified values are lower, mwm ignores iconImageMinimum.

Syntax

`Mwm*`*`resource`*`: value`

Value

*width*x*height*
: Width x height of the icon image in pixels

3.115 IconLabelFont

3.115.1 iconLabelfont

This resource defines the font for labels on icons displayed in the xdm login window or host selection window.

Syntax

`XLogin.Login.`*`resource`*`: `*`value`*

`Chooser*`*`resource`*`: `*`value`*

Value

fontname

Name of a font the X server can find in its font path. The xlsfonts client lists the applicable font names for the X server.

3.116 IconPath

3.116.1 iconPath

This resource defines the user-specific list of files the client is to search for the icon image for an object that it manages.
After searching the user-specific list, the client searches the systemspecific list of files defined by means of the systemIconPath resource.
xdm uses the resource to define the system-specific list of files containing icon images for the output of the login window and host selection window.
The definition of the iconPath resource applies only if the XDESK_ICON_PATH environment variable is not defined. If the resource is assigned a value, the client will only continue to find the icon images for existing objects if the existing search path is part of the new search path.

Syntax

```
Chooser*resource: value
HostManager*resource: value
CommunicationManager*resource: value
NFSManager*resource: value
SoftwareManager*resource: value
VolumeManager*resource: value
UserManager*resource: value
XDeskmaster*resource: value
XLogin.Login.resource: value
```

Value

filename[:*filename*]...

List of absolute file names *filename* each separated by a colon (:).
filename can contain symbolic names. The client replaces:

%N	with the name of the file containing the icon image,
%I	with the value of the iconSet resource,
%L	with the locale as defined by the LANG environment variable,
%l	with the part of the locale which defines the language,
%t	with the part of the locale which defines the territory,
%c	with the part of the locale which defines the codeset.

3.117 IconPlacement

3.117.1 iconPlacement

This resource specifies the icon placement scheme to be used.
This resource instance is only available if the resource instance iconAutoPlace (class: IconAutoPlace) is assigned the value "True".

Syntax
`Mwm*`*`resource: value`*

Value
layout [tight]
layout can be replaced by:

left top
: in rows from left to right starting at the top of the screen

right top
: in rows from right to left starting at the top of the screen

left bottom
: in rows from left to right starting at the bottom of the screen

right bottom
: in rows from right to left starting at the bottom of the screen

bottom left
: in columns from bottom to top starting at the left of the screen

bottom right
: in columns from bottom to top starting at the right of the screen

top left
: in columns from top to bottom starting at the left of the screen

top right
: in columns from top to bottom starting at the right of the screen

tight
: The icons are packed together without space between them.

3.118 IconPlacementMargin

3.118.1 iconPlacementMargin

This resource specifies the distance (in pixels) between the edge of the screen and the first icon placed.

Syntax

`Mwm*`*`resource`*`: value`

Value

pixels
: Number of pixels between the edge of the screen and the first icon placed

3.119 IconSet

3.119.1 iconSet

defines the set of icon images the client uses for the objects that it manages.
A set of icon images consists of files in either of two formats: GIF format and bitmap format. GIF (Graphics Interchange Format) is the standard interchange format for color images in the X Window System. In bitmap format it is not possible to store multicolor images. Bitmap format supports only one color plane; in other words, each pixel can only be set either to foreground color or to background color.
xdm uses the set of icon images to display the user-specific icons in the login window and to display the host-specific icons in the host selection window.

Syntax

`Chooser*`*`resource`*`: `*`value`*
`HostManager*`*`resource`*`: `*`value`*
`CommunicationManager*`*`resource`*`: `*`value`*
`NFSManager*`*`resource`*`: `*`value`*
`SoftwareManager*`*`resource`*`: `*`value`*
`VolumeManager*`*`resource`*`: `*`value`*
`UserManager*`*`resource`*`: `*`value`*
`XDeskmaster*`*`resource`*`: `*`value`*
`XLogin.Login.`*`resource`*`: `*`value`*
`XMessages.`*`resource`*`: `*`value`*

Value

mono Icons for a monitor that supports one color plane.
color Icons for a monitor that supports more than one color plane.
directory
Directory (without path) containing a set of icon images. The clients will find icon images you have created only if you include the path to this directory in the iconPath resource.

3.120 Init

3.120.1 init

This resource defines the full pathname of the Xinit file which runs under root before a each session is opened.
The usual pathname is /usr/lib/X11/xdm/Xinit.

Syntax

`DisplayManager*`*`resource`*`: `*`value`*

Value

name Full pathname of the program

3.121 InitString

3.121.1 initString

This resource specifies the character string which is output to the terminal emulator when the terminal emulator is called.
This resource enables you to load the DRCS area when you call the terminal emulator, as described in the operating instructions for the 97801-4xx terminal.

Syntax

`X97801-480[*x97801-480]*`*`resource: value`*

Value

character[*character*]...
In addition to the characters A to Z and a to z, *character* can also be replaced by, :

\\	the character \
\E	the ESC character
\n	the newline character
\t	the tab character
\ooo	the character with octal value ooo

3.122 InteractivePlacement

3.122.1 interactivePlacement

This resource controls the initial placement of new windows on the screen.

Syntax

`Mwm*`*`resource`*`: value`

Value

False The window manager determines initial placement of new windows.
True You determine the position of the top lefthand corner of the window with the mouse using the resize cursor.

3.123 IsoCharClass

3.123.1 isoCharClass

This resource specifies which characters of ISO character sets are grouped together in a character class.
Character classes are important when text is selected word by word. When you double-click the mouse button BSelect, successive characters of the same character class are selected for the text buffer.
The terminal emulator uses the ISO character classes when the character set mode is set to ISO by means of the charSetMode resource or -charsetmode option. The character class selected continues to be used even if you switch character set mode during runtime by means of an escape sequence.
You create a character class by assigning a common numerical value to all characters of a particular class. The individual characters are specified by their decimal values in the character set table. You can select the decimal value of any character in the class as the common numerical value.
Example: The string

```
33:48,37:48,45-47:48,64:46
```

specifies that the exclamation mark ! (33), the percent sign % (37), the minus sign - (45), the period . (46) and the slash / (47) are to be in the same character class as normal characters and digits. They are therefore assigned the character 0 (48). In addition, the commercial at @ (64) is assigned to the character class to which the period (46) belongs. This also belongs to character class 48. These specifications are useful when selecting UNIX file names and electronic mail addresses word by word.

Syntax

`XPct[*xpct]*`*`resource`*`: `*`value`*

Value

character_code:*class_character*[,*character_code*:*class_character*]...

character_code
: Decimal character code of the character. You can use either a single character (*character_code1*) or a string of characters (*character_code1-character_code22*) for *character_code*. Several non-successive value assignments are separated by commas.

class_character
: Decimal character code of a character to which the characters of a class are assigned. You can select the decimal character code of any character in the class.

3.124 IsoCharSet

3.124.1 isoCharSet

This resource specifies the character set with which the terminal emulator works when the character set mode is set to ISO.
You set the character set mode using the charSetMode resource.

Syntax

`XPct[*xpct]*`*`resource: value`*

Value

character_set
The name of the ISO character set. The possible values are:

ISO standard	Languages
iso8859-1	Danish, German, Faroese, Finnish, French, Icelandic, Italian, Irish, Dutch, Norwegian, Portuguese, Swedish, Spanish, English
iso8859-2	Albanian, German, Polish, Romanian, Serbo-Croat, Slovakian, Slovenian, Czech, Hungarian, English
iso8859-3	Afrikaans, German, Esperanto, French, Italian, Catalan, Maltese, Dutch, Spanish, Turkish, English
iso8859-4	Danish, German, Estonian, Finnish, French, Greenlandic, Lapp, Latvian, Lithuanian, Norwegian, Swedish, English
iso8859-5	Cyrillic characters, English
iso8859-7	Greek characters, English
iso8859-9	Variation of iso8859-1 with Turkish, without Icelandic and Faroese

Table 85: Languages of the ISO character set

3.125 JumpScroll

3.125.1 jumpScroll

This resource specifies whether text in the terminal emulator window is to be scrolled by line or in blocks where necessary (jump scrolling).

Syntax

```
XPct[*xpct]*resource: value
XTerm[*vt100]*resource: value
X97801-480[*x97801-480]*resource: value
```

Value

False Text is scrolled one line at a time.
True Text is scrolled in blocks where necessary.

3.126 KeyBindings

3.126.1 keyBindings

This resource specifies the name of a set of key bindings.
mwm searches for this set of key bindings in the file specified in the configFile resource (class: ConfigFile).
Please see the Abschnitt "Key and button bindings" in the mwm description for more details.

Syntax

`Mwm*`*`resource`*`: `*`value`*

Value

name Name of a set of key bindings

3.127 KeyboardFocusPolicy

3.127.1 keyboardFocusPolicy

This resource determines the keyboard input focus policy.

Syntax

`Mwm*`*`resource`*`: value`

Value

explicit
: mwm sets the keyboard input focus only to a window given the input focus by means of a predefined action. This predefined action can be a key press, a button press or a click on a menu item. You will find information on defining these actions in the Abschnitt "Key and button bindings" in the mwm description. The default select action is pressing button 1 on the mouse.

pointer
: mwm sets the keyboard input focus to the window in which the pointer is located.

3.128 KeyFile

3.128.1 keyFile

xdm and the terminal require a common key for XDMCP authorization using the XDM-AUTHORIZATION-1 mechanism. This resource specifies the file which contains the encryption values.

Syntax

`DisplayManager.display.`*`resource`*`: `*`value`*

Value

pathname

3.129 KeySymMapping

3.129.1 keySymMapping

This resource modifies a key-to-keysym mapping.
When the terminal emulator is processing a keysym, it checks to see if the keySymMapping resource has been used to modify the keysym mapping. It makes allowance for any modifier keys that may have been pressed. If it finds the keysym, it converts $keysymname_{before}$ to $keysymname_{after}$ (together with the optional modifier keys).
If you set this resource, you change the mapping only for the client which is evaluating the resource. The other clients are unaffected.

Syntax

`X97801-480*`*`resource`*`: value`

Value

[$modifier_{before,1}$]$keysymname_{before,1}$:[$modifier_{after,1}$]$keysymname_{after,1}$\n\
[$modifier_{before,2}$]$keysymname_{before,2}$:[$modifier_{after,2}$]$keysymname_{after,2}$\n\
.
[$modifier_{before,k}$]$keysymname_{before,k}$:[$modifier_{after,k}$]$keysymname_{after,k}$\n\
.
[$modifier_{before,m}$]$keysymname_{before,m}$:[$modifier_{after,m}$]$keysymname_{after,m}$

$modifier_{...}$
: is one of the modifier keys Shift, Altand Control.

$keysymname_{...}$
: is a key symbol (keysym). The xmodmap client displays a list of the valid keysyms.

3.130 LabelString

3.130.1 labelString

Specifies the label of an entry in the session option menu with the right-hand value <sessionArgument>. Also refer to "SessionArgument".
Specifies the label of an entry in the language option menu with the right-hand value <languageArgument>. Note that each period '.' in <languageArgument> must be replaced with an underscore '_'. Also refer to "LanguageArgument".

Syntax

```
xlogin*Session.<sessionArgument>.resource: value
xlogin*Language.<languageArgument>.resource: value
```

Value

string
label in the option menu.

Example
If the xdm-config file defines the sessionEntries resource as follows:

```
DisplayManager*sessionEntries:
        User Session=UserSession,Default Session=
        DefaultSession,SINIX Desktop=SINIXSession,
        Light Session=LightSession,Failsafe Session=
        failsafe
```

you can redefine one of the labels with the following assignment:
xlogin*Session.UserSession.labelString: Private Session

3.131 LanguageArgument

3.131.1 languageArgument

Defines the LANG variable which is to be the default for a session. If the option menu 'Language' contains an entry with this variable, the entry appears as the label of the option button.
If the resource is not specified, the /etc/default/language file is evaluated.

Syntax

`DisplayManager.display.`*`resource`*`: `*`value`*
`Xlogin*Language.`*`resource`*`:labelString`

Value

name Contents of the LANG variable

3.132 LanguageEntries

3.132.1 languageEntries

Indicates a list of language options from which a user can select a language with option button 'Language' before logging on to the system.

Syntax

`DisplayManager.display.`*`resource`*`:` *`value`*

Value

name=language[_territory[.character_set]][,name=language[_territory[.character_set]]]...

name Name of the language option, label in the options menu

language_territory.character_set

Configuration for the variable LANG following selection of the entry; corresponds to the name of a directory under /usr/lib/locale

3.133 LanguageSwitch

3.133.1 languageSwitch

Specifies whether the option button 'Language' is displayed in the login window of the xdm.

Syntax

`DisplayManager.display.`*`resource`*`: `*`value`*

Value

False The option button is not displayed.
True The option button is displayed.

3.134 LightBlueColor

3.134.1 lightblueColor

color setting for color terminals

Syntax

`XPct*`*`resource`*`: value`

Value

color Color name in the X server's color database

3.135 LightCyanColor

3.135.1 lightCyanColor

color setting for color terminals

Syntax

`XPct*`*`resource`*`: value`

Value

color Color name in the X server's color database

3.136 LightGreenColor

3.136.1 lightGreenColor

color setting for color terminals

Syntax

`XPct*`*`resource`*`: value`

Value

color Color name in the X server's color database

3.137 LightMagentaColor

3.137.1 lightMagentaColor

color setting for color terminals

Syntax

`XPct*`*`resource`*`: value`

Value

color Color name in the X server's color database

3.138 LightRedColor

3.138.1 lightRedColor

color setting for color terminals

Syntax

`XPct*`*`resource`*`: value`

Value

color Color name in the X server's color database

3.139 LightWhiteColor

3.139.1 lightWhiteColor

color setting for color terminals

Syntax

`XPct*`*`resource`*`: value`

Value

color Color name in the X server's color database

3.140 LimitResize

3.140.1 limitResize

This resource specifies whether you can resize a window to a size greater than that specified in
- the resource instance maximumClientSize (class: MaximumClientSize)
- the general description of client window attributes
- the default maximum size assigned a client window by mwm

Syntax

`Mwm*`*`resource`*`: `*`value`*

Value

False The maximum size may be exceeded.
True The maximum size may not be exceeded.

3.141 LockPidFile

3.141.1 lockPidFile

This resource prevents xdm processes running on the same host from hindering one another. This is done by means of a locking mechanism.
The process id(s) is read from the file specified for the pidFile resource.

Syntax

`DisplayManager.`*`resource`*`:` *`value`*

Value

True The locking mechanism is used.
False The locking mechanism is not used.

3.142 LogFile

3.142.1 logFile

This resource specifies the name of the file to which a terminal emulator writes the session log.
This resource is only available if the resource instance logging (class: Logging) is assigned the value "True" or the resource instance logInhibit (class: LogInhibit) is assigned the value "False" and logging is enabled interactively.

Syntax

`XPct[*xpct]*`*`resource`*`: `*`value`*

`XTerm[*vt100]*`*`resource`*`: `*`value`*

`X97801-480[*x97801-480]*`*`resource`*`: `*`value`*

Value

filename
: Name of the log file. This file is created in the directory from which the terminal emulator was started.

3.143 Logging

3.143.1 logging

This resource determines whether or not a terminal emulator logs the session.
If a terminal emulator is to log a session, it writes the log to the file specified for the resource instance logFile (class: LogFile).

Syntax

`XPct[*xpct]*`*`resource`*`: `*`value`*

`XTerm[*vt100]*`*`resource`*`: `*`value`*

`X97801-480[*x97801-480]*`*`resource`*`: `*`value`*

Value

False The terminal emulator does not log the session.
True The terminal emulator logs the session.

3.144 LogInhibit

3.144.1 logInhibit

This resource specifies whether or not the logging of a session in a terminal emulator can be enabled and disabled via the menu.

Syntax

```
XPct[*xpct]*resource: value
XTerm[*vt100]*resource: value
X97801-480[*x97801-480]*resource: value
```

Value

False Logging can be enabled/disabled via the menu.
True Logging cannot be enabled or disabled via the menu.

3.145 LoginLogo

3.145.1 loginLogo

This resource defines the logo for the xdm login window or host selection window.
The logo is displayed if the value of the showLogo resource is True. The logo must be in the form of a bitmap (XBM format) or a pixmap (GIF format).
This resource should be defined in the Xresources file specified by the value of the 'resources' resource.

Syntax

`Chooser*`*`resource`*`: value`
`XLogin.Login.`*`resource`*`: value`

Value

name Full pathname of the file

3.146 LoginShell

3.146.1 loginShell

This resource specifies whether the shell to be started in the terminal emulator window is a login shell.
The initial working directory of the login shell is the directory in which the terminal emulator is started.

Syntax

```
XPct[*xpct]*resource: value
XTerm[*vt100]*resource: value
X97801-480[*x97801-480]*resource: value
```

Value

False The terminal emulator starts a normal shell, not a login shell.
True The terminal emulator starts a login shell.

3.147 LowerOnIconify

3.147.1 lowerOnIconify

This resource specifies whether a window's icon automatically drops to the bottom of a window stack or whether it retains its original position in the window stack when it is iconified.

Syntax

`Mwm*`*`resource`*`: value`

Value

False The icon retains its original position in the window stack.
True The icon is placed at the bottom of the window stack.

3.148 MagentaColor

3.148.1 magentaColor

color setting for color terminals

Syntax

`XPct*`*`resource`*`: value`

Value

color Color name in the X server's color database

3.149 Mail

3.149.1 mailPath

This resource sets the MAIL environment variable for the duration of the session. The MAIL environment variable defines the directory in which the user's mail is stored.

Syntax

`DisplayManager.DISPLAY.`*`resource`*`: `*`value`*

Value

name Full pathname of the directory

3.150 MapchanFile

3.150.1 mapchanFile

Defines the Mapchan file for the key mapping for keyboard inputs.

Syntax

`Xpct[*xpct]*`*`resource`*`: `*`value`*

Value

[*path*]*filename*

path is the path in which the Mapchan file is searched on startup. If the path is not specified, the following directories are searched for the Mapchan file:
1. $HOME/mapchan
2. /usr/lib/mapchan

filename
the name of the Mapchan file. If a Mapchan file was found and activated on startup of the client, the entry "key mapping" in the "keyboard" pulldown menu becomes sensitive and the button "key mapping" is actuated. If the Mapchan file has not been specified or not been found, the entry "key mapping" does not become sensitive.

3.151 MarginBell

3.151.1 marginBell

This resource specifies whether the terminal emulator is to ring the margin bell when the text cursor reaches a certain column. The column is specified in the nMarginBell resource instance (class: Column).

Syntax

```
XPct[*xpct]*resource: value
XTerm[*vt100]*resource: value
X97801-480[*x97801-480]*resource: value
```

Value

False The terminal emulator does not ring the margin bell.
True The terminal emulator rings the margin bell.

3.152 MatteWidth

3.152.1 matteWidth

This resource specifies the width of the matte (in pixels).

Syntax

```
{client_class   }
Mwm[*{client_instance}]*resource: value
{defaults       }
```

Value

pixels
: Width of the matte in pixels

3.153 MaximumClientSize

3.153.1 maximumClientSize

This resource specifies the size of a window if it is to be maximized, or the direction in which it is maximized.

Syntax

```
{client_class   }
Mwm[*{client_instance}]*resource: value
{defaults       }
```

Value

*width*x*height*
: Width times height of the window in units of the client

horizontal
: The window is enlarged until its horizontal dimensions fill the screen.

vertical
: The window is enlarged until its vertical dimensions fill the screen.

3.154 MaximumMaximumSize

3.154.1 maximumMaximumSize

This resource specifies the maximum size of a client window (in pixels).

Syntax

`Mwm*`*`resource`*`: value`

Value

*width*x*height*
: Width x height of the client window in pixels

3.155 MenuBorder

3.155.1 menuBorder

This resource specifies the width of the menu border (in pixels).

Syntax

`XTerm*menu*`*`resource`*`: `*`value`*

Value

pixels
: Width of the border in pixels

3.156 MenuFont

3.156.1 menuFont

This resource specifies the name of the font to be used for menu items.

Syntax

`XTerm*menu*`*`resource`*`: `*`value`*

Value

name Name of the font

3.157 MenuPad

3.157.1 menuPad

This resource specifies the distance between the menu border and menu items in pixels.

Syntax

`XTerm*menu*`*`resource`*`: `*`value`*

Value

pixels
: Distance between the menu border and menu items in pixels

3.158 Monitor

3.158.1 monitor

Xmessages periodically attempts to open the error log file defined in the resource file once the former file's status has changed. It continues attempting to do so until no more error messages can be displayed.

Syntax

`XMessages.`*`resource`*`: `*`value`*

Value

True error log file is monitored
False error log file is not monitored

3.159 Monochrome

3.159.1 monochrome

Specifies whether the client comes up in monochrome mode, in which case all escape sequences relating to color settings are ignored.

Syntax

`XPct*`*`resource`*`: value`

Value

True The client starts in monochrome mode.
False The client does not start in monochrome mode.

3.160 MoveOpaque

3.160.1 moveOpaque

Specifies how a window is to be displayed while being moved.

Syntax

`Mwm*`*`resource`*`: value`

Value

True The window is displayed with its contents.
False The window is displayed as a thin rectangular frame.

3.161 MoveThreshold

3.161.1 moveThreshold

This resource specifies the distance (in pixels) that the pointer must be moved with the button held down for the movement to be interpreted as a dragging operation.

Syntax

`Mwm*`*`resource`*`: value`

Value

pixels
: Number of pixels the pointer must be moved before the movement is interpreted as dragging

3.162 MultiClickTime

3.162.1 multiClickTime

This resource specifies the maximum time (in milliseconds) that may elapse between the mouse clicks that make up a multiple click.

Syntax

`Mwm*`*`resource`*`: value`
`XPct[*xpct]*`*`resource`*`: value`
`XTerm*`*`resource`*`: value`
`X97801-480[*x97801-480]*`*`resource`*`: value`

Value

milliseconds
Time in milliseconds

3.163 MultiScreen

3.163.1 multiScreen

This resource specifies whether or not mwm is to manage all the screens on a display.

Syntax

`Mwm*`*`resource`*`: value`

Value

True mwm manages all the screens on the display.
False mwm manages only one screen.

3.164 MultiScroll

3.164.1 multiScroll

This resource indicates whether or not text can be scrolled asynchronously.
If text is scrolled asynchronously, the screen is not always up to date when scrolling through large amounts of text.

Syntax

```
XPct[*xpct]*resource: value
XTerm*vt100*resource: value
X97801-480[*x97801-480]*resource: value
```

Value

False Text can only be scrolled synchronously.
True Text can be scrolled asynchronously.

3.165 NamePrompt

3.165.1 namePrompt

This resource specifies the string displayed to prompt the user for a login name.
xrdb strips trailing white space from the resource value; any blanks you want to have at the end of the string must be escaped with backslashes (\).

Syntax

`XLogin.Login.`*`resource`*`:` *`value`*

Value

string
String displayed on the screen to prompt the user for input

3.165.2 passwdPrompt

This resource specifies the string displayed to prompt the user for a password.
xrdb strips trailing white space from the resource value; any blanks you want to have at the end of the string must be escaped with backslashes (\).

Syntax

`XLogin.Login.`*`resource`*`:` *`value`*

Value

string
Character string displayed on the screen in order to prompt the user for input

3.166 NoIconUsers

3.166.1 noIconUsers

This resource defines the name of the file that contains a list of login names which are not to be displayed as icons in the xdm login window.
For reasons of security it may be advisable not to display all the icons.
The file is in the following format:
- one user login name per line, starting in column 1.
- the hash sign # in column 1 identifies the line as a comment line.

If icons are to be displayed, the value of the allowIcons resource must be True.
The onlyLocalUser resource allows you to specify that only local user login names are to be displayed.

Syntax

`DisplayManager.`*`resource`*`: `*`value`*

Value

name Full pathname of the file

3.167 NoMenu

3.167.1 noMenubar

This resource specifies whether or not a menu bar is displayed in the terminal emulator window.

Syntax

`XPct[*xpct]*`*`resource`*`: `*`value`*

`X97801-480[*x97801-480]*`*`resource`*`: `*`value`*

Value

True No menu bar is displayed.
You cannot access the menus in the menu bar.
False The menu bar is displayed.

3.168 NoState

3.168.1 noState

This resource specifies whether or not a status line is displayed in the terminal emulator window.
The status line displays the bit mode, compose and dead key status, as well as previously entered compose and dead key sequences.

Syntax

`X97801-480[*x97801-480]*`*`resource`*`: `*`value`*

Value

True No status line is displayed.
False A status line is displayed.

3.169 Notify

3.169.1 notify

Specifies whether the icon imahe should change when messages arrive. Operates only if popup is set to false.

Syntax

```
{ console  }
{ system   }
XMessages[*{          }]*resource: value
{ xerrors  }
{ common   }
```

Value

False The icon image is not changed.
True The icon image is changed.

3.170 OnceOnly

3.170.1 onceOnly

Specifies that an acoustic signal is emitted only the first time a new message is received.

Syntax

```
{ console  }
{ system   }
XMessages[*{          }]*resource: value
{ xerrors  }
{ common   }
```

Value

False An acoustic signal is emitted every time a new message is received.
True An acoustic signal is emitted only the first time a new message is received.

3.171 OnlyIconLabel

3.171.1 onlyIconLabel

This resource governs whether the icons in the xdm login window or host selection window are to be displayed in the form of image plus label or just label.
This resource is evaluated for the login window only if the value of the allowIcons resource is True.

Syntax

`Chooser*`*`resource`*`: `*`value`*
`XLogin.Login.`*`resource`*`: `*`value`*

Value

True Only the label is displayed.
False The image and the label are displayed.

3.172 OnlyLocalUser

3.172.1 onlyLocalUser

This resource determines whether icons are to be displayed in the xdm login window for all users with a login name on the local host or only for local users. A login name is held to be local if:

- the user has a home directory on the local host
- the home directory belongs to the user
- the user has a valid password which has not been disabled.

If icons are to be displayed, the value of the allowIcons resource must be True.
In the NoIconUsers file (see the noIconUsers resource in the section "NoIconUsers ") you can list individual user login names which are not to displayed as icons.
For further information on the xdm login window refer to the xdm client description in section "xdm - Display manager".

Syntax

`DisplayManager.`*`resource`*`: `*`value`*

Value

True Icons are not displayed for users without a home directory.
False Icons are displayed for all users.

3.173 OpenDelay

3.173.1 openDelay

This resource specifies the length of the pause (in seconds) between a client's successive attempts to open a server.
The openDelay, openRepeat, openTimeout and startAttempts resources control the behavior of the client when server a cannot be opened.

Syntax

`DisplayManager*`*`resource`*`: `*`value`*

Value

seconds
Time in seconds

3.174 OpenRepeat

3.174.1 openRepeat

This resource specifies the number of attempts a client makes to open the server. The openDelay, openRepeat, openTimeout and startAttempts resources control the behavior of the client when server a cannot be opened.
Once the client has made the number of attempts specified for openRepeat, it aborts, restarts the server, and again attempts to set up a connection.

Syntax

`DisplayManager*`*`resource`*`: `*`value`*

Value

number
: Number of attempts to open the server

3.175 OpenTimeout

3.175.1 openTimeout

This resource specifies the period of time (in seconds) during which the client can attempt to open the server, i.e. the maximum period of time available for the connect system call.
The openDelay, openRepeat, openTimeout and startAttempts resources control the behavior of the client when server a cannot be opened.
Once the period of time specified in openTimeout has elapsed, the client aborts, restarts the server and again attempts to set up a connection.

Syntax

`DisplayManager*`*`resource`*`: `*`value`*

Value

seconds
: Time in seconds for an attempt to open the server

3.176 Osm

3.176.1 osm

Specifies whether messages written to /dev/osm are to be displayed.

Syntax

`XMessages*`*`resource`*`: `*`value`*

Value

False The client does not display kernel messages from /dev/osm. .
True The client displays kernel messages from /dev/osm .

3.177 PassButtons

3.177.1 passButtons

This resource specifies whether messages from the server relating to button press events are passed to the client in whose window the button was pressed after the window manager has performed the window manager function associated with the message.
Messages inform clients whether the button is being pressed, released, clicked or used for dragging.

Syntax

`Mwm*`*`resource`*`: value`

Value

False If you press a button that is bound to a window manager function, the function is executed but no message is passed on to the client.
True If you press a button that is bound to a window manager function, the function is executed and the message is passed on to the client.

3.178 PassSelectButton

3.178.1 passSelectButton

This resource specifies whether messages from the server relating to button 1 events are passed to the client in whose window the event took place after the window manager has performed the window manager function associated with the message.
The messages inform clients of whether the selection button was pressed, released, clicked or used to drag an object.
This resource is only available if the resource instance keyboardFocusPolicy (class: KeyboardFocusPolicy) is assigned the value "explicit".

Syntax

`Mwm*`*`resource`*`: value`

Value

False If you perform an action with the select button that is bound to a window manager function, the function is executed but no message is passed to the client.
True If you perform an action with the select button that is bound to a window manager function, the function is executed and a message passed to the client.

3.179 PidFile

3.179.1 pidFile

This resource specifies the name of a file in which the process id of the main xdm process, i.e. the xdm process with the lowest process id, is stored in ASCII format.

Syntax

`DisplayManager.`*`resource`*`: `*`value`*

Value

name Full pathname of the file

3.180 PingInterval

3.180.1 pingInterval

This resource specifies the interval (in minutes) at which the client checks whether a remote display is still active.
The client sends Xsync requests to remote displays at regular intervals to check whether the displays respond.
The pingTimeout resource specifies the maximum time to wait for the display to respond.
If the display does not respond, the session is terminated.

Syntax

`DisplayManager*`*`resource`*`: `*`value`*

Value

minutes
Time in minutes

3.181 PingTimeout

3.181.1 pingTimeout

This resource specifies the maximum time for a remote display to respond to an Xsync request.
The client sends Xsync requests to remote displays at regular intervals to check whether the display responds. The pingInterval resource specifies the interval between checks.
If the display does not respond, the session is terminated.

Syntax

`DisplayManager*`*`resource`*`: `*`value`*

Value

min Time in minutes

3.182 Popup

3.182.1 popup

Specifies wether the client automatically opens its text window when messages arrive.

Syntax

```
{ console  }
{ system   }
XMessages[*{          }]*resource: value
{ xerrors  }
{ common   }
```

Value

False The text window is not opened automatically.
True The text window is opened automatically.

3.183 Position

3.183.1 x

This resource specifies the horizontal distance, or x offset, between the window and the edge of the screen (in pixels).

Sign +:
Offset from left edge of screen
Sign -:
Offset from right edge of screen

Syntax

`Chooser*`*`resource`*`: `*`value`*
`XLogin.Login.`*`resource`*`: `*`value`*

Value

number
Offset from edge of screen in pixels

3.183.2 y

This resource specifies the vertical distance, or y offset, between the window and the edge of the screen (in pixels).

Sign +:
Offset from upper edge of screen
Sign -:
Offset from lower edge of screen

Syntax

`Chooser*`*`resource`*`: `*`value`*
`XLogin.Login.`*`resource`*`: `*`value`*

Value

number
Offset from edge of screen in pixels

3.184 PositionIsFrame

3.184.1 positionIsFrame

This resource indicates whether position information applies to the position of the mwm window frame or to the position of the client area managed by the client.

Syntax

`Mwm*`*`resource`*`: value`

Value

False Position information applies to the position of the client area.
True Position information applies to the position of the mwm window frame.

3.185 PositionOnScreen

3.185.1 positionOnScreen

This resource specifies how a window is clipped at the edge of the screen when a window is displayed which is larger than the screen.

Syntax

`Mwm*`*`resource`*`: value`

Value

False Windows are placed as specified without regard to the screen borders even if they are then totally off-screen.

True Windows are placed so that they are totally on-screen. If this is not possible because, for example, the window is larger than the screen, as much of the window is displayed as possible, starting at the top left corner.

3.186 QuitTimeout

3.186.1 quitTimeout

This resource specifies the amount of time (in milliseconds) that mwm waits between sending the client a message that it is to save itself and ceasing to process client responses. This resource is only available if the client has been programmed accordingly. See also mwm function " f.kill" in the section "Appearance and behavior" of the mwm description.

Syntax

`Mwm*`*`resource`*`: `*`value`*

Value

milliseconds
Time in milliseconds

3.187 RaiseKeyFocus

3.187.1 raiseKeyFocus

This resource specifies whether the window which is normalized and raised to the top of the window stack by means of the mwm function f.normalize_and_raise receives the input focus.
This resource is only available if the keyBoardFocusPolicy resource instance (class: KeyboardFocusPolicy) is assigned the value "Explicit".

Syntax

`Mwm*`*`resource`*`: `*`value`*

Value

False The input focus is not changed.
True The window which is normalized and raised to the top of the window stack with the mwm function f.normalize_and_raise receives the input focus.

3.188 RandomFile

3.188.1 randomFile

This resource defines the name of the file used to generate the authorization key.

Syntax

`DisplayManager*`*`resource`*`: `*`value`*

Value

name Full pathname of the file

3.189 RedColor

3.189.1 redColor

color setting for color terminals

Syntax

`XPct*`*`resource`*`: value`

Value

color Color name in the X server's color database

3.190 RemoveDomainname

3.190.1 removeDomainname

This resource specifies whether xdm strips the domain name from the names of hosts communicating with it via the XDMCP protocol.

Syntax

`DisplayManager.`*`resource`*`: `*`value`*

Value

True The domain name is stripped from the host name.
False The domain name is not stripped from the host name.

3.191 RequestPort

3.191.1 requestPort

This resource defines the number of the port at which xdm accepts XDMCP requests on the basis of the UDP protocol.
XDMCP is entered in the /etc/services file as a service with a defined port number; so the port number should not be changed unless essential for diagnostic purposes.

Syntax

`DisplayManager.`*`resource`*`: `*`value`*

Value

portNumber
: Number of the port at which xdm accepts XDMCP requests on the basis of the UDP protocol.

3.192 Reset

3.192.1 reset

This resource defines the full pathname of the Xreset file which runs under root after each session terminates.
The usual pathname is /usr/lib/X11/xdm/reset.

Syntax

`DisplayManager*`*`resource`*`: `*`value`*

Value

name Full pathname of the program

3.193 ResetForAuth

3.193.1 resetForAuth

This resource governs whether the xdm display manager sends a SIGHUP signal to the server after changes to the authorization file, thus causing it to reboot and implement the changes.
Some servers read authorization information only when rebooted, not when setting up a connection to an xdm service.

Syntax

`DisplayManager.DISPLAY.`*`resource`*`:` *`value`*

Value

True The server is rebooted.
False The server is not rebooted.

3.194 ResetSignal

3.194.1 resetSignal

This resource defines the number of the signal to use to reset the server.

Syntax

`DisplayManager.DISPLAY.`*`resource`*`: `*`value`*

Value

number
: Signal number

3.195 ResizeBorderWidth

3.195.1 resizeBorderWidth

This resource specifies the width (in pixels) of the resize handles, including the 3-D shadows.
It is a good idea to make sure that the value specified here is compatible with the value specified for the resource instance frameBorderWidth (class: FrameBorderWidth).

Syntax

`Mwm*`*`resource`*`: value`

Value

pixels
: Width of the window border in pixels

3.196 ResizeCursors

3.196.1 resizeCursors

This resource is used determine whether the resize cursors are displayed when the pointer is located on the resize borders.

Syntax

`Mwm*`*`resource`*`: value`

Value

False The resize cursors are not displayed when the pointer is on the resize borders.
True The resize cursors are displayed when the pointer is on the resize borders.

3.197 Resources

3.197.1 resources

Defines the name of the Xresources file that the xrdb client loads as the resource database for the root window of the display.
The Xresources file may contain resource definitions for the setup file and the xdm login window and host selection window. The description of the xdm client contains further information on the various resources that can be used in this file.

Syntax

`DisplayManager*`*`resource`*`: `*`value`*

Value

name Search path

3.198 ReverseVideo

3.198.1 reverseVideo

This resource indicates whether background and foreground colors are inverted.

Syntax

```
XPct[*xpct]*resource: value
XTerm[*vt100]*resource: value
X97801-480[*x97801-480]*resource: value
```

Value

False The background is displayed in the background color, the foreground in the foreground color.
True The background is displayed in the foreground color, the foreground in the background color.

3.199 ReverseWrap

3.199.1 reverseWrap

This resource specifies whether, for the xterm client, you can move the text cursor backwards from the first column in one line to the last column of the previous line.

Syntax

```
XPct[*xpct]*resource: value
XTerm*resource: value
X97801-480[*x97801-480]*resource: value
```

Value

False Moving backwards from the first column of a line does not move you to the last column of the same line.

True Moving backwards from the first column of a line moves you to the last column of the previous line.

3.200 RootBackground

3.200.1 rootBackground

Specifies the color for the root window.

Syntax

`XLogin.Login.`*`resource`*`: `*`value`*

Value

color color name or RGB specification

3.201 RootBackgroundPixmap

3.201.1 rootBackgroundPixmap

Specifies a pixmap for tiling the root window. The pixmap must be in XBM, GIF or XPM format.

Syntax

`XLogin.Login.`*`resource`*`: `*`value`*

Value

pixmap
name of a pixmap files

3.202 RootDesktop

3.202.1 rootDesktop

This resource specifies the placement of desktools of the SINIX/windows desktop.

Syntax

`XDeskmaster*`*`resource`*`: `*`value`*

Value

True The desktools of the SINIX/windows desktop are placed in the root window.
False The desktools of the SINIX/windows desktop are placed in a window.

3.203 Rows

3.203.1 rows

This resource defines the number of lines in the text field

Syntax

`XMessages*`*`resource`*`: `*`value`*

Value

no_of_lines
Number of lines

3.204 SavedLines

3.204.1 savedLines

This resource defines the number of lines of the previous output that the client does not delete.

Syntax

```
{ console  }
{ system   }
XMessages[*{          }]*resource: value
{ xerrors  }
{ common   }
```

Value

no_of_lines
Number of lines that the client does not delete.

3.205 SaveLines

3.205.1 saveLines

This resource specifies the number of lines a terminal emulator saves when there is a scroll bar.

Syntax

```
XPct[*xpct]*resource: value
XTerm[*vt100]*resource: value
X97801-480[*x97801-480]*resource: value
```

Value

lines Number of lines to be saved

3.206 SaveOnExit

3.206.1 saveOnExit

This resource specifies whether the client saves its status when terminating.

Syntax

```
CommunicationManager*resource: value
HostManager*resource: value
NFSManager*resource: value
SoftwareManager*resource: value
VolumeManager*resource: value
UserManager*resource: value
XDeskmaster*resource: value
```

Value

never The client never saves its status when terminating.
always
The client always saves its status when terminating.
ask The client asks the user when terminating whether it should save its status.

3.207 SaveUnder

3.207.1 saveUnder

This resource indicates whether the X server is to save components beneath the mwm window frame if these components are obscured by other windows.
This resource is only available in a client window if

- the X server provides the functionality for save unders and
- the client requests the X server to save components beneath its client area when it is obscured by other windows.

Syntax

```
{menu[*menu_name]}
{icon            }
Mwm[*{               }]*resource: value
{client[*title]  }
{feedback        }
```

Value

False The server does not save any window frame components when the window is obscured by other windows.

True If a client has the server save the contents of a window, window frame components are also saved.

3.208 Screens

3.208.1 screens

This resource defines names for the screens managed by mwm.
These names are used by mwm to allocate screen-specific resources, using the following syntax:
mwm**screen_name*[*widgetname]*resource: value

Syntax

`Mwm*resource: value`

Value

screen_name$_1$[*screen_name*$_2$]...

screen_name$_i$

Screen name *screen_name*$_i$ is the name of the screen with the screen number i.

If mwm is managing only one screen, it uses the first name on the list; this name is assigned to the screen entered in the option -display.
If mwm is managing several screens, the names will be assigned in the order in which the screens are given, starting with screen 0. If the X server for the display is managing more screens than names have been defined, mwm assigns these the first screen name.

3.209 ScrollBar

3.209.1 scrollBar

This resource specifies whether a vertical scroll bar is to be displayed in the terminal emulator window.
The saveLines resource allows you to specify how many lines are buffered by the terminal emulator.

Syntax

```
XPct[*xpct]*resource: value
XTerm[*vt100]*resource: value
X97801-480[*x97801-480]*resource: value
```

Value

False No scroll bar is displayed.
True A scroll bar is displayed.

3.210 ScrollCond

3.210.1 scrollInput

This resource specifies whether the window is positioned to the end of the scroll area (= end of text) for output to the window if the end of the text was previously not visible.

Syntax

`XPct[*xpct]*`*`resource`*`: `*`value`*
`XTerm[*vt100]*`*`resource`*`: `*`value`*
`X97801-480[*x97801-480]*`*`resource`*`: `*`value`*

Value

False The window is not scrolled forward.
True The window is scrolled forward.

3.210.2 scrollKey

This resource specifies whether pressing a key causes the window with the input focus to be scrolled forward to the end of the text if the latter was previously not visible.

Syntax

`XPct[*xpct]*`*`resource`*`: `*`value`*
`XTerm[*vt100]*`*`resource`*`: `*`value`*
`X97801-480[*x97801-480]*`*`resource`*`: `*`value`*

Value

False The window is not scrolled forward.
True The window is scrolled forward.

3.211 Servers

3.211.1 servers

Either defines the full pathname of the Xservers file listing the names of the X servers managed by xdm, or specifies a single server entry (see Abschnitt "Appearance and behavior" in the xdm description).
X servers which call up the client via the X Display Manager Control Protocol (XDMCP) must not be listed in this file.

Syntax

`DisplayManager.`*`resource`*`: `*`value`*

Value

name Full pathname of the file listing the server entries
entry Server entry, consisting of display name, display class, display type and the command used to start the server (local servers)

3.212 Session

3.212.1 session

Defines the full pathname of the Xsession file containing system-wide and user-specific session management definitions.
The file is usually /usr/lib/X11/xdm/Xsession.
The configuration data for the user's session are taken from the file $HOME/.xsession, if such a file exists.

Syntax

`DisplayManager*`*`resource`*`: `*`value`*

Value

name Full pathname of the file

3.213 SessionArgument

3.213.1 sessionArgument

This resource defines which menu item on the session selection option menu is selected by default when the login window comes up.
The resource definition for the login window in the Xresources file (syntax: Xlogin...) overrides the resource definition in the xdm-config file (syntax: DisplayManager...).

Syntax

```
DisplayManager*resource: value
DisplayManager[.display_class].resource: value
DisplayManager[.display].resource: value
Xlogin[.Login].resource: value
Xlogin*Language.resource: value
Xlogin*Session.resource: value
```

Value

name is selected. You must define *name* as a menu item label in the resource definition for sessionEntries.

3.214 SessionEntries

3.214.1 sessionEntries

This resource defines the items on the session selection option menu in the xdm login window.
The sessionArgument resource defines the item which is selected when the login window comes up.

Syntax

```
DisplayManager*resource: value
DisplayManager[.display_class].resource: value
DisplayManager[.display].resource: value
```

Value

name=*sessionArgument*[, *name*=*sessionArgument*]...
Each *name*=*sessionArgument* pair defines a menu item:

name is displayed as the menu item label on the option menu.
sessionArgument
is passed as an argument on invocation of the shell script in the Xsession file. The Xsession file must be prepared for each *sessionArgument* used.

3.215 SessionPrompt

3.215.1 sessionPrompt

This resource determines the text of the session selection prompt.

Syntax

```
DisplayManager*resource: value
DisplayManager[.displayclass].resource: value
DisplayManager[.display].resource: value
```

Value

menu item label

3.216 SessionSwitch

3.216.1 sessionSwitch

This resource determines whether the session selection option menu appears in the xdm login window.

Syntax

```
DisplayManager*resource: value
DisplayManager[.display_class].resource: value
DisplayManager[.display].resource: value
```

Value

True The option menu is displayed.
False The option menu is not displayed.

3.217 Setup

3.217.1 setup

This resource defines the full pathname of the Xsetup file. The Xsetup file runs under root after the login window has been displayed. The file can be used to customize the appearance of the screen and open other windows.
The usual pathname for the file is /usr/lib/X11/xdm/Xsetup.

Syntax

`DisplayManager.DISPLAY.`*`resource`*`:` *`value`*

Value

name Full pathname of the file

3.218 ShowFeedback

3.218.1 showFeedback

This resource controls when mwm displays feedback information.

Syntax

`Mwm*`*`resource`*`: `*`value`*

Value

```
{ - }         [-]
{   }context[[ ]context]...
{[+]}         [+]
```

context can be replaced by the following:

all Feedback for all mwm functions listed below

behavior
Confirm behavior switch

kill Confirm receipt of KILL signal

move Show current window position during move operations

none No feedback information for the mwm functions listed

placement
Show size and position of window during interactive placement

quit Confirm mwm termination

resize
Show current window size during resize operations

restart
Confirm mwm restart

Signs are interpreted as follows:

+ or no sign
- preceding first list item: mwm assumes no context
- preceding other list items: the named context is added

-
- preceding first list item: mwm assumes all contexts
- preceding other list items: the named context is deleted

3.219 ShowIcons

3.219.1 showIcons

This resource governs whether user login name icons are displayed in the xdm login window.
The user can select an icon and hence a login name with the mouse instead of entering the login name at the keyboard.
This resource is evaluated only if the value of the allowIcons resouuce is True.
The resource definition for the login window in the Xresources file (syntax: Xlogin...) overrides the resource definition in the xdm-config file (syntax: DisplayManager...).

Syntax

```
DisplayManager*resource: value
DisplayManager[.display_class].resource: value
DisplayManager[.display].resource: value
XLogin.Login.resource: value
```

Value

True The icons are displayed.
False The icons are not displayed.

3.220 ShowLogo

3.220.1 showLogo

This resource governs whether the logo defined by the loginLogo resource is displayed in the xdm login window or host selection window.

Syntax

`XLogin.Login.`*`resource`*`: `*`value`*

`Chooser*`*`resource`*`: `*`value`*

Value

True The logo is displayed.
False The logo is not displayed.

3.221 ShowSoftKeyboard

3.221.1 showSoftKeyboard

This resource specifies whether to display a soft (i.e. mouse-operated) keyboard.
From the user's viewpoint a soft keyboard is an extra input device in the form of a keyboard displayed on the screen together with the terminal emulator. You use the mouse to press the keys, and the terminal emulator processes this input just like input from an ordinary keyboard. From the X server's viewpoint, though, the soft keyboard is simply part of a client, not an extra input device.
The terminal emulator depicts the soft keyboard as defined by the softKeyboardConfiguration resource (class: SoftKeyboardConfiguration).

Syntax

`X97801-480*`*`resource`*`: `*`value`*

Value

True The soft keyboard is displayed.
False The soft keyboard is not displayed.

3.222 SignalInhibit

3.222.1 signalInhibit

This resource relates to terminal emulator menus and specifies whether the sending of signals to the process group of the shell is inhibited and whether the associated menu items are locked.

Syntax

```
XTerm[*vt100]*resource: value
XPct[*xpct]*resource: value
X97801-480[*x97801-480]*resource: value
```

Value

False The menu entries for sending signals to the shell process group may be selected.
True The menu entries for sending signals to the shell process group may not be selected.

3.223 SmallIconSize

3.223.1 smallIconSize

This resource governs whether the size in pixels of the icons representing the user login names in the login window and the icons representing the hosts in the host selection window is 32x32 or 64x64.
This resource is evaluated for the login window only if the value of the allowIcons resource is True.

Syntax

`Chooser*`*`resource`*`: value`
`XLogin.Login.`*`resource`*`: value`

Value

True The icon size in pixels is 32x32.
False The icon size in pixels is 64x64.

3.224 SNIExtensions

3.224.1 showHostname

This resource determines whether the xdm login window displays the name of the host after the welcome message.
The hostname displayed is specified by the hostname resource instance.

Syntax

`XLogin.Login.`*`resource`*`: `*`value`*

Value

True The host name is displayed.
False The host name is not displayed.

3.225 SoftKey

3.225.1 softKey

This resource specifies whether softkeys are interpreted.
The softkeys in the file specified for the softKeyFile resource instance (class: SoftKeyFile) are interpreted.
This resource is only available if the eightbit resource instance (class: Eightbit) is assigned the value "True".

Syntax

`X97801-480[*x97801-480]*`*`resource: value`*

Value

True Softkeys are interpreted. Using a menu, you can open a dialog box in which you can define softkeys.
False Softkeys are not interpreted.

3.226 SoftKeyboardConfiguration

3.226.1 softKeyboardConfiguration

Defines the keys for the terminal emulator's soft keyboard.

Syntax

X97801-480**resource*: *value*

Value

The value of this resource is composed of a number of lines, all but the last ending with "\n\". Thus assignments take the following form:

```
X97801-480*softKeyboardConfiguration: \n\
softkeyboard_key_or_separator1 \n\
softkeyboard_key_or_separator2 \n\
.
softkeyboard_key_or_separatork \n\
.
softkeyboard_key_or_separatorm
```

The value of *softkeyboard_key_or_separator*$_k$ is formed according to the syntax indicated by the following transformation rules:

The left-hand side of a rule consists of a single unique symbol. The symbol "production" is the starting point for a transformation producing a valid value for *softkeyboard_key_or_separator*$_k$.

The right-hand side consists of the possible transformations of the left-hand side into symbols which cannot be further resolved, in other words terminal symbols. Terminal symbols are keywords in single quotes, e.g. 'keyword', or freely selectable strings shown here in italics, e.g.: *freely_selectable_string*.

The right-hand side may also include the following operators:

a|b either a or b.

[*example*]

the bracketed symbol, in this case *example*, may appear once or not at all.

In accordance with these conventions the transformation rules read:

```
production                      = softkeyboard_key_with_label
| softkeyboard_key_with_bitmap
| functionless_key_with_label
| separator
softkeyboard_key_with_label     = 'Button' widget_name_of_key labels : actions
softkeyboard_key_with_bitmap    = 'BitmapButton' widget_name_for_key bitmap_name
: actions
functionless_key_with_label     = 'Label' widget_name_for_key label
separator                       = 'Separator' separator_type
labels                          = [unshift_label [shift_label [alt_label
[shift_alt_label]]]]
actions                         = [unshift_action [shift_action [alt_action
[shift_alt_action]]]]
separator_type                  = 'pixelspace' positive_integer
| 'charspace' positive_integer
```

```
| 'nextline' [ positive_integer ]
| 'top_align_to' widget_name_for_key
| 'left_align_to' widget_name_for_key
unshift_action                  = action
shift_action                    = action
alt_action                      = action
shift_alt_action                = action
action                          = 'f.key' SNIKey    | 'r.key' SNIKey
| 'f.keysym' Keysym | 'r.keysym' Keysym
| 'f.string' string | 'r.string' string
SNIKey                          = SNIPrint       | SNIMode       | SNIHelp
| SNICompose
| SNIStart    | SNIEnd        | SNIMenu
| SNIBackTab  | SNIClear
| SNIDelChar  | SNIDelWord    | SNIDelLine
| SNIInsChar  | SNIInsWord    | SNIInsLine
| SNINextPage | SNIPrevPage   | SNIHome       | SNICursorEnd
| SNICursorUp | SNICursorDown | SNICursorLeft | SNICursorRight
| SNIRowBegin | SNIRowEnd     | SNIRowLeft    | SNIRowRight
| SNIF1 ... SNIF27
| SNIShiftF1 ... SNIShiftF27
| SNINone
```

You define the keys of the soft keyboard as follows:
First you establish the key type, using one of the keywords 'Button', 'BitmapButton' or 'Label': a soft keyboard key with labels as identifiers, a soft keyboard key with a bitmap as identifier, or a key which is not associated with any action but simply acts as an extra area for displaying text in conjunction with the labels of soft keyboard keys. The keyword is followed by the widget name you select for the key. The terminal emulator uses this widget name when generating the widget. This allows you to specify additional resources for the keys: fontList, background, backgroundPixmap, borderColor, borderPixmap, borderWidth. You do so using the resource syntax:
x97801-480**widget_name_for_key***resourceSpecification*: *value*.
If you define a key with the keyword 'Button', you next specify the labels for the key: for the key without additional modifiers, together with the <Shift> modifier, with <Alt>, and with <Shift> and <Alt>. If you specify labels for the key together with <Shift> or with <Shift> and <Alt>, the label will be in two-line format. If you want blanks to appear in a label, you must enclose the entire string in double quotes. If you want a label to include double quotes or colons (normally used to separate a label from the associated action), you must escape them by preceding them with two backslashes (\\).
A label may include the following special characters:

\\E escape
\\n newline
\\ backslash "\"
\ooo character with octal value ooo

If you define a key with the keyword 'BitmapButton', the widget name must be followed by the name of a bitmap file. The bitmap file is searched for in the directories assigned to the XBMLANGPATH environment variable.
For every key you define with the keyword 'Button' or 'BitmapButton', you then specify the actions to be executed when that key is clicked on with the mouse. You successively specify the associated input events: click on key without additional modifier, together with

<Shift>, together with <Alt>, and together with <Shift> and <Alt>. For each action you can choose one of two function types: the functions of the f.*actiontype* group perform the action once only, i.e. send the character just once, while the functions of the r.*actiontype* group after a delay of 750 ms automatically repeat the action every 100 ms for as long as the mouse button is held down on the soft keyboard key. Select one of the following actions:

- f.key *SNIKey* (or r.key *SNIKey*) sends the application the Siemens Nixdorf-specific escape sequence which has the symbolic name *SNIKey*. Thus you can send the application the following: SNIClear (CE key on numeric keypad), SNIRowBegin (broken left arrow), SNIRowEnd (broken right arrow), SNIRowLeft (double left arrow), SNIRowRight (double right arrow), SNICursorEnd (arrow pointing down to the right), SNIF1 ... SNIF27, SNIShiftF1 ... SNIShiftF27 (function keys F1 through F27 with or without <Shift> modifier), or SNINone (generates a null string).
- f.keysym *XKeySym* (or r.keysym *XKeySym*) sends the terminal emulator the X Window System keysym *XKeySym*. The terminal emulator processes the keysym and passes the result to the application. The xmodmap client supplies information about the keymap table of the X server you are using.
- f.string *string* (or r.string *string*) sends the application the string *string*.

Defining a key with the keyword 'Label' allows you to display additional textual information on the soft keyboard. You can define just one label and no action for each key of this type. The keys of the soft keyboard are displayed from left to right, key by key in the order in which they appear in the resource specification. You can influence the layout by defining separators with the keyword 'Separator'. The keyword is followed by further parameters defining the appearance of the spacing:

- 'pixelspace' *positive_integer* is interpreted by the terminal emulator as a horizontal forward space in pixels;
- 'charspace' *positive_integer* is interpreted by the terminal emulator as a horizontal forward space in character widths in the current font,
- 'nextline' [*positive_integer*] is interpreted by the terminal emulator as a linefeed plus return to start of line. You can optionally define the spacing from the preceding line as *positive_integer* pixels. Unless otherwise instructed, the terminal emulator aligns subsequent lines with the bottom edge of the highest key on the line above.
- 'top_align_to' *widget_name_for_key* is interpreted by the terminal emulator to mean that it should align the next line below the key with the name *widget_name_for_key*. Unless otherwise instructed, the terminal emulator aligns subsequent lines with the bottom edge of the highest key on the line above.
- 'left_align_to' *widget_name_for_key* is interpreted by the terminal emulator to mean that it should position the next key with its left edge next to the key with the name *widget_name_for key*. Unless otherwise instructed, the terminal emulator positions subsequent keys with their left edge aligned with the preceding key.

Example:

```
x97801-480*softKeyboardConfiguration: \n\
Separator pixelspace 5 \n\
Button button1 InsChar: f.key SNIInsChar \n\
Button button2 InsLine: f.key SNIInsLine \n\
Separator charspace 4 \n\
Button button3 DelChar: f.key SNIDelChar \n\
Button button4 DelLine: f.key SNIDelLine \n\
Separator nextline \n\
```

```
Separator left_align_to button1 \n\
Button button5 " End   " " Start ": f.keysym Cancel f.keysym Start \n\
Button button6 " Print " "": f.keysym Print \n\
Separator left_align_to button3 \n\
Button button7 " Help  " " Menu  ": f.keysym Help f.keysym Menu \n\
Separator nextline 10 \n\
Separator left_align_to button1 \n\
Label label1 Commands\\: \\n\
Separator nextline \n\
Separator left_align_to button1 \n\
Button button8 "List" "files": f.string "ls -C" \n\
Button button9 "List" "processes": f.string ps
```

This soft keyboard configuration generates the following soft keyboard:

3.227 SoftKeyFile

3.227.1 softKeyFile

This resource specifies the name of the file whose entries are used to define softkeys when the terminal emulator is called.
The file specified by this resource is always read, regardless of the value assigned to the resource instances softKey (class: SoftKey) and eightbit (class: Eightbit).
The syntax for entries in this file is described in the section "Programming function keys " in the description of the x97801-480 client.

Syntax

`X97801-480[*x97801-480]*`*`resource: value`*

Value

filename
: Name of the file whose entries are used to define softkeys when the terminal emulator is called.

3.228 StandardIconSize

3.228.1 standardIconSize

Size of the icons when the client is started

Syntax

```
HostManager*resource: value
CommunicationManager*resource: value
NFSManager*resource: value
SoftwareManager*resource: value
VolumeManager*resource: value
UserManager*resource: value
XDeskmaster*resource: value
```

Value

Big When started, the client assigns each icon the size in pixels defined in the XDCL configuration file for that icon. The size is the extension of the value of the "name" element of that instance of the XDCL class "image" that is defined in the instance associated with the icon in the XDCL class "icon". The extension may have no more than three digits.

Small When started, the client uses icons with half the height and width of the value of "Big".

Auto The client determines the size of the icons automatically depending on the resolution of the monitor.

3.229 StartAttempts

3.229.1 startAttempts

This resource specifies the number of attempts the client makes to open the server. The openDelay, openRepeat, openTimeout and startAttempts resources control the behavior of the client when server a cannot be opened.
Once the number of attempts specified by openRepeat have been made, or when the time specified by openTimeout for an attempt has elapsed, xdm terminates, restarts the server and again attempts to set up a connection. startAttempts specifies how often this entire process is to be repeated. If the value specified is 0, an infinite number of attempts will be made.

Syntax

`DisplayManager*`*`resource`*`:` *`value`*

Value

number
The number of attempts

3.230 Startup

3.230.1 startup

Defines the full pathname of the Xstartup file which runs after the user logs in. The usual name is /usr/lib/X11/xdm/Xstartup.

Syntax

`DisplayManager*`*`resource`*`: `*`value`*

Value

name Full pathname of the program

3.231 StartupKeyFocus

3.231.1 startupKeyFocus

This resource specifies whether a window is assigned the keyboard input focus when it is made visible.
This resource is only available if the resource instance keyboardFocusPolicy (class: KeyboardFocusPolicy) is assigned the value "explicit".

Syntax

`Mwm*`*`resource`*`: `*`value`*

Value

False The window that has the keyboard input focus when a new window is made visible retains the input focus.
True A new window that is made visible is assigned the keyboard input focus.

3.232 StripNonPrint

3.232.1 stripNonPrint

Specifies whether non-printing characters are to be stripped.

Syntax

```
{ console  }
{ system   }
XMessages[*{          }]*resource: value
{ xerrors  }
{ common   }
```

Value

False Non-printing characters are not stripped.
True Non-printing characters are stripped.

3.233 SunFunctionKeys

3.233.1 sunFunctionKeys

This resource specifies whether or not Sun escape codes are generated for the function keys instead of standard escape sequences.

Syntax

`XTerm*`*`resource`*`: `*`value`*

Value

False Standard escape sequences are generated.
True Sun escape codes are generated.

3.234 SystemIconPath

3.234.1 systemIconPath

This resource specifies the system-specific part of the list of files the client searches for the icon image for an object it manages.
The client begins by searching the list of files defined by means of the iconPath resource.
The definition of the systemIconPath resource applies only if the XDESK_SYSTEM_ICON_PATH environment variable is not defined.
If a value is assigned to the resource, the client will find the icon images for existing objects only if the existing search path is part of the new search path.

Syntax

`HostManager*`*`resource`*`: value`
`CommunicationManager*`*`resource`*`: value`
`NFSManager*`*`resource`*`: value`
`SoftwareManager*`*`resource`*`: value`
`VolumeManager*`*`resource`*`: value`
`UserManager*`*`resource`*`: value`
`XDeskmaster*`*`resource`*`: value`
`XMessages*`*`resource`*`: value`

Value

filename[:*filename*]...
List of absolute file names *filename* each separated by a colon (:).
filename can contain symbolic names. The client replaces:

%N	with the name of the file containing the icon image,
%I	with the value of the iconSet resource,
%L	with the locale as defined by the LANG environment variable,
%l	with the part of the locale which defines the language,
%t	with the part of the locale which defines the territory,
%c	with the part of the locale which defines the codeset.

3.235 SystemPath

3.235.1 systemPath

xdm sets the PATH environment variable for the Xstartup and Xreset files to the value of this resource. The default value for this resource is specified in the system configuration file and is normally
/etc:/bin:/usr/bin:/usr/bin/X11:/usr/ucb
To make it more difficult to break into the system using Trojan horses penetration schemes, this entry deliberately excludes a dot (.) for the current working directory.

Syntax

`DisplayManager*`*`resource`*`: `*`value`*

Value

list List of directories

3.236 SystemShell

3.236.1 systemShell

This resource sets the SHELL environment variable for the Xstartup and Xreset files to the value of this resource.

Syntax

`DisplayManager*`*`resource`*`: value`

Value

name Contents of the SHELL environment variable.

3.237 TekInhibit

3.237.1 tekInhibit

This resource specifies whether the Tektronix emulator can be called by the xterm client.

Syntax

`XTerm*vt100*`*`resource`*`: `*`value`*

Value

False Tektronix emulator can be called.
True Tektronix emulator cannot be called.

3.238 TekStartup

3.238.1 tekStartup

This resource specifies whether the Tektronix emulator is started when xterm is called.

Syntax

`XTerm*vt100*`*`resource`*`: `*`value`*

Value

False Vt100 emulator is started.
True Tektronix emulator is started.

3.239 TermName

3.239.1 termName

This resource specifies the name of the terminal to be assigned to the TERM environment variable.
The name of the terminal must be defined in the directory /usr/lib/terminfo.

Syntax

```
XPct*resource: value
XTerm*resource: value
X97801-480*resource: value
```

Value

name Name of the terminal

3.240 TerminateServer

3.240.1 terminateServer

This resource specifies whether the X server is to be terminated when a session terminates. Normally, it is simply reset. This option enables you to limit the amount of time the server is run.
This is possible on local displays only.

Syntax

`DisplayManager*`*`resource`*`: `*`value`*

Value

True The server is terminated.
False The server is not terminated.

3.241 TermSignal

3.241.1 termSignal

This resource defines the number of the signal to use to terminate the server.

Syntax

`DisplayManager.DISPLAY.`*`resource`*`: `*`value`*

Value

zahl Signal number

3.242 Thickness

3.242.1 thickness

This resource specifies the width of the scroll bar in pixels.

Syntax

`XTerm*scrollbar*`*`resource`*`: `*`value`*

Value

pixels
: Width of the scroll bar in pixels

3.243 Title

3.243.1 title

This resource specifies the name of the window, which may be displayed by the window manager in the title bar.

Syntax

```
XPct*resource: value
XTerm*resource: value
X97801-480*resource: value
{ console  }
{ system   }
XMessages[*{          }]*resource: value
{ xerrors  }
{ common   }
```

Value

name Name of the window

3.244 TopShadowPixmap

3.244.1 activeTopShadowPixmap

This resource defines the pixmap for the active window's top and left 3-D shadows.

Syntax

```
{menu[*menu_name]}
{icon            }
Mwm[*{client[*title]  }]*resource: value
{feedback        }
{defaults        }
```

Value

pixmap
: Filename of the pixmap in the directory specified by the bitmapDirectory resource (class: BitmapDirectory), or filename of the pixmap including the absolute path of the file.

background
: Pixmap in the color specified by the value assigned to the background resource instance (class: Background).

foreground
: Pixmap in the color specified by the value assigned to the foreground resource instance (class: Foreground).

25_foreground
: Color mix: 25% foreground and 75% background

50_foreground
: Color mix: 50% foreground and 50% background

75_foreground
: Color mix: 75% foreground and 25% background

horizontal
: Horizontal lines alternating in foreground and background

NULL No pixmap

vertical
: Vertical lines alternating in foreground and background

slant_right
: Diagonal lines (slanting to the right) alternating in foreground and background

slant_left
: Diagonal lines (slanting to the left) alternating in foreground and background

3.244.2 iconImageTopShadowPixmap

This resource specifies the pixmap of an icon image's top and left 3-D shadows.

Syntax

```
{menu[*menu_name]}
{icon            }
Mwm[*{client[*title]  }]*resource: value
{feedback        }
```

```
{defaults        }
```

Value

pixmap
: Filename of the pixmap in the directory specified by the bitmapDirectory resource (class: BitmapDirectory), or filename of the pixmap including the absolute path of the file.

background
: Pixmap in the color specified by the value assigned to the background resource instance (class: Background).

foreground
: Pixmap in the color specified by the value assigned to the foreground resource instance (class: Foreground).

25_foreground
: Color mix: 25% foreground and 75% background

50_foreground
: Color mix: 50% foreground and 50% background

75_foreground
: Color mix: 75% foreground and 25% background

horizontal
: Horizontal lines alternating in foreground and background

NULL No pixmap

vertical
: Vertical lines alternating in foreground and background

slant_right
: Diagonal lines (slanting to the right) alternating in foreground and background

slant_left
: Diagonal lines (slanting to the left) alternating in foreground and background

3.244.3 matteTopShadowPixmap

This resource specifies the pixmap of the matte's top and left 3-D shadows.
This resource is evaluated if the matteWidth resource is assigned a positive value, i.e. a matte exists

Syntax

```
{menu[*menu_name]}
{icon            }
Mwm[*{client[*title]  }]*resource: value
{feedback        }
{defaults        }
```

Value

pixmap
: Filename of the pixmap in the directory specified by the bitmapDirectory resource (class: BitmapDirectory), or filename of the pixmap including the absolute path of the file.

background
: Pixmap in the color specified by the value assigned to the background resource instance (class: Background).

foreground
: Pixmap in the color specified by the value assigned to the foreground resource instance (class: Foreground).

25_foreground
: Color mix: 25% foreground and 75% background

50_foreground
: Color mix: 50% foreground and 50% background

75_foreground
: Color mix: 75% foreground and 25% background

horizontal
: Horizontal lines alternating in foreground and background

NULL No pixmap

vertical
: Vertical lines alternating in foreground and background

slant_right
: Diagonal lines (slanting to the right) alternating in foreground and background

slant_left
: Diagonal lines (slanting to the left) alternating in foreground and background

3.244.4 topShadowPixmap

This resource specifies the pixmap of a window's top and left 3-D shadows.
The mwm client uses this pixmap for inactive windows only.

Syntax

```
{menu[*menu_name]}
{icon            }
Mwm[*{                }]*resource: value
{client[*title]  }
{feedback        }
XPct*resource: value
{display }
{mode    }
{vtmenubar  [*{x97801  }]}
X97801-480[*{            {keyboard}}]*resource: value
{vtscrollbar           }
```

Value

pixmap
: File name of the pixmap in the directory specified by the bitmapDirectory resource (class: BitmapDirectory), or file name of the pixmap including the absolute path of the file.

background
: Pixmap in the color specified by the value assigned to the background resource instance (class: Background).

foreground
: Pixmap in the color specified by the value assigned to the foreground resource instance (class: Foreground).

25_foreground
: Color mix: 25% foreground and 75% background

50_foreground
: Color mix: 50% foreground and 50% background

75_foreground
: Color mix: 75% foreground and 25% background

horizontal
: Horizontal lines alternating in foreground and background

NULL
: No pixmap

vertical
: Vertical lines alternating in foreground and background

slant_right
: Diagonal lines (slanting to the right) alternating in foreground and background

slant_left
: Diagonal lines (slanting to the left) alternating in foreground and background

3.245 TransientDecoration

3.245.1 transientDecoration

This resource controls the amount of decoration on the frames of transient windows (cf. "ClientDecoration " resource).
transientDecoration is specified as a list of decoration elements.

Syntax

```
Mwm*resource: value
```

Value

```
{ - }            [-]
{   }decoration[[ ]decoration]...
{[+]}            [+]
```

decoration can be replaced with the following:

all All of the following decorations
border
Window border
maximize
Maximize button
Note the value of the transientFunctions resource instance (class: TransientFunctions) with which the f.maximize function can be made unavailable.
menu Menu button
minimize
Minimize button
Note the value of the transientFunctions resource instance (class: TransientFunctions) with which the f.minimize function can be made unavailable.
none None of the decorations listed
resizeh
Border with resize handles
title Title bar
Signs are interpreted as follows:

+ or no sign
Starting with no decoration, all the decorations listed are added.
- Starting with all decorations, the decorations listed are removed from that set.

3.246 TransientFunctions

3.246.1 transientFunctions

This resource specifies which mwm functions are applicable to a transient window. The following mwm functions are applicable or (non-applicable): f.maximize, f.minimize, f.resize, f.move, f.kill. This resource does not apply to mwm functions not mentioned here. transientFunctions is specified as a list of functions.

Syntax

```
Mwm*resource: value
```

Value

```
{ - }          [-]
{   }function[[ ]function]...
{+}            [+]
```

function can be replaced with the following:

all All the following mwm functions are applicable
close mwm function f.kill
maximize
mwm function f.maximize
minimize
mwm function f.minimize
move mwm function f.move
none None of the mwm functions mentioned are applicable
resize
mwm function f.resize

Signs are interpreted as follows:

\+ or no sign
Starting with no mwm functions, all the mwm functions listed are added.
\- Starting with all mwm functions, the mwm functions listed are removed from this set.

3.247 Translations

3.247.1 translations

This resource specifies the key and button bindings for menus, selection functions and programmed strings.

Syntax

```
{loginName }
XLogin*{          }.resource: value
{passwdName}
XTerm*vt100*resource: value
XPct*resource: value
X97801-480*resource: value
```

Value

```
[#replace ]
[[#override] \n]\
[#augment ]
[[!][~][@]modifier]...<action>key[key]...: function[ \n\
[[!][~][@]modifier]...<action>key[key]...: function]...
```

#replace
: #replace enables you to specify that the following translation table is to fully replace the previously active one.

#override
: #override enables you to specify that the following translation table is to replace the previously active one for those key or button sequences assigned a value in the previously active table.

#augment
: #augment enables you to specify that the following translation table is to replace the previously active one for those key or button sequences not assigned a value in the previously active table.

!*modifier*
: Prefixing *modifier* with ! means that no modifier other than *modifier* may be specified, to ensure that this entry in the translation table will become active for the current key or button sequence.

~*modifier*
: Prefixing *modifier* with ~ means that this modifier *modifier* may not be specified, to ensure that this entry in the translation table will become active for the current key or button sequence.

@*modifier*
: Prefixing *modifier* with @ means that all those modifiers containing the keysym *modifier* may be specified. Only then will this entry in the translation table become active for the current key or button sequence.
For further information on keysyms, please refer to the xmodmap client description in section "xmodmap - Displaying/modifying keyboard and pointer maps ".

modifier
: can be replaced by the following:

Ctrl	CTRL key
Shift	Shift key
Meta	Meta key
Lock	LOCK key
Mod1	Modifier key 1
Mod2	Modifier key 2
Mod3	Modifier key 3
Mod4	Modifier key 4
Mod5	Modifier key 5

xmodmap will supply you with information on the mapping of the values for *modifier* to keys on the keyboard.

action

can be replaced by the following key or button actions:

Key	press key
KeyDown	press key
KeyUp	release key
BtnDown	press button
BtnUp	release button
Btn*number* Down	press button *number*; for *number* you may specify 1 to 5
Btn*number* Up	release button *number*; for *number* you may specify 1 to 5

key can be replaced by any character in the character set ISO Latin 1.
You can use xmodmap to obtain information on the keys supported by your keyboard.

function

can be replaced by the following for the xdm client:

abort-display()
Terminates and locks the server. This is not accessible in the default configuration. It can be used to terminate xdm when the system is switched off.

abort-session()
Terminates and restarts the server.

allow-all-access()
Prevents access control in the server.

finish-input()
If the input focus is in the login name input field, login name input is finished. If the input focus is in the password input field, password input is finished. If the login name and the password are valid, xdm starts a session; if not, it displays an error message.

restart-session()
Resets the X server and starts a new session. This can be used when you want to test modified resources, or when the display has been overwritten with system messages.

set-session-argument()
Denotes a single-word argument that is to be passed when the Xsession file is started.

function
can be replaced by the following for the xterm, xpct and x97801-4 clients:

bell([*volume*]**)**
Generates an acoustic signal. *volume* allows you to specify the volume increment above or below base volume.

ignore()
Key or button event is ignored.

insert()
The keys are processed in the normal manner, i.e. the ASCII character code corresponding to the keysym in the keymap table is inserted into the input stream.

insert-selection(*name***[,***name***]...)**
name is the name of a selection buffer. The terminal emulator evaluates the list from left to right and replaces the key or key sequence with the contents of the first non-empty buffer *name* and inserts the contents of the buffer *name* into the input stream.
You can replace *name* with BPRIMARY, SECONDARY, CUT_BUFFER0, ..., CUT_BUFFER7. *name* is case sensitive.

keymap(*name***)**
Defines a new resource for the terminal emulator.
name is added as a prefix to the character string keymap, thus creating the name *name*keymap. This new resource can be used to dynamically define a new translation table; *name* is case sensitive.
If you specify the value None as *name*, the default translation table is restored.

mode-menu()
Posts either the "Modes" or the "Tektronix" menu, depending on which button is pressed.

programming()
Displays the menu bar.
This function applies to the xpct and x97801-480 clients only.

read-mode(*parameter***)**
Switches PAGE mode on and off.
parameter
ON switches PAGE mode on.
OFF switches PAGE mode off.
This function applies to the xpct and x97801-480 clients only.

select-start()
Deselects any previously selected text and begins selecting new text.

select-end(*name***[,***name***]...)**
Ends text selection.

name is the name of a selection or the name of a cut buffer into which the text is to be copied. *xterm* assumes ownership of all the selections named and will copy the text into each of the cut buffers. *name* is case sensitive.

select-extend()

Continues selecting text (from the previous starting position).

start-extend()

Begins extending the selection from the farthest (left or right) corner.

string(*char_string***)**

The key or key sequence is bound to *char_string*, i.e. *char_string* is inserted into the input stream.

If *char_string* contains blanks or special characters, it must be enclosed in quotes. If *char_string* starts with the character *0x*, the characters that follow are interpreted as hexadecimal characters and the corresponding characters inserted in the normal way.

3.248 TtyModes

3.248.1 ttyModes

This resource sets terminal attributes for a terminal emulator.
It causes attributes of the terminal driver to be set on the principle of the stty command. If the call to the terminal emulator includes the -L option, the program ignores any ttyModes specifications.

Syntax

XPct*_resource_: _value_
XTerm*_resource_: _value_
X97801-480[*x97801-480]*_resource_: _value_

Value

function ^*character*[*function* ^*character*]...
You assign *function* the control character ^*character*, which you can then use to initiate this function from the keyboard. The character "^" stands for the key to which the modifier "Ctrl" is assigned. You may replace *character* with any character on the keyboard.
You may use any of the following for *function*:
intr, quit, erase, kill, eof, eol
You will find the meaning of these functions in the description of the tty interface in your operating system interfaces documentation.

3.249 UnsecureGreeting

3.249.1 unsecureGreeting

This resource defines the text displayed in the xdm login window if user-specific authorization is active for the display but is not supported by the server.

Syntax

`XLogin[*Login]*`*`resource`*`: `*`value`*

Value

string
Message text string

3.250 Update

3.250.1 update

Specifies the interval in seconds at which the log file is to be polled.

Syntax

`XMessages*`*`resource`*`: value`

Value

number
Interval in seconds.

3.251 UseClientIcon

3.251.1 useClientIcon

This resource specifies whether an icon image supplied by the client takes precedence over a user-supplied icon image.

Syntax

```
{client_class   }
Mwm[*{client_instance}]*resource: value
{defaults       }
```

Value

False A user-supplied icon image takes precedence over a client-supplied icon image.
True A client-supplied icon image takes precedence over a user-supplied icon image.

3.252 UseIconBox

3.252.1 useIconBox

This resource specifies whether or not an icon box is managed by mwm.

Syntax

`Mwm*`*`resource`*`: `*`value`*

Value

False There is no icon box.
True An icon box exists and is managed by mwm.

3.253 UsePPosition

3.253.1 usePPosition

This resource specifies whether mwm adopts the default position of a window from the client when you have not specified a position.

Syntax

```
{client_class   }
Mwm[*{client_instance}]*resource: value
{defaults       }
```

Value

on A position you have specified has priority over that of the client.
off A position you have specified does not have priority over that of the client.
nonzero
A position you have specified has priority over that of the client providing it is not the origin (0,0) of the screen.

3.254 UserAuthDir

3.254.1 userAuthDir

This resource defines the name of the directory in which the xdm client creates the user's .Xauthority authorization file if it cannot do so in the home directory. It also defines the value of the XAUTHORITY environment variable.
For further information refer to the xdm client description in section "xdm - Display manager".

Syntax

`DisplayManager.DISPLAY.`*`resource`*`: `*`value`*

Value

name Name of the directory

3.255 UserPath

3.255.1 userPath

This resource sets the environment variable PATH for the session to the value specified for this resource.
The value should be a colon-separated list of directories. The default value is set by the system configuration file and is normally:
:/usr/bin:/usr/bin/X11:/sbin:/usr/sbin:/opt/bin:/usr/dbin:/usr/ldbin

Syntax

`DisplayManager*`*`resource`*`: `*`value`*

Value

list List of directories

3.256 UseStandardColormap

3.256.1 useStandardColormap

defines the behavior of the Client, if more colors are used than the display supports

Syntax

`XPct*`*`resource`*`: value`

Value

False An attempt is made to use colors as close as possible to the StandardColormap.

True A dedicated colormap is created and the required colors are generated in it. If a dedicated colormap is allocated, the color settings for the windows of other clients will also be changed. When the input focus is moved to another window, the color settings are restored to their original values.

3.257 Utmp

3.257.1 utmp

Specifies whether a pseudo terminal is to be allocated and entries made in the /etc/utmp system log file and the /etc/wtmp file. The user will then be known to commands such as *who* and *finger* and will also receive output from the *wall* command, because the client provides a window in which to display output directed to the pseudo terminal.

Syntax

`XMessages*`*`resource`*`: `*`value`*

Value

False The Client does not make any entries in the system log-files /etc/utmp and /etc/wtmp.
True The Client makes entries in the system log-files /etc/utmp and /etc/wtmp.

3.258 UtmpInhibit

3.258.1 utmpInhibit

This resource specifies whether the terminal emulator writes a login entry in the system log file /etc/utmp.

Syntax

`XPct*`*`resource`*`: value`
`XTerm*`*`resource`*`: value`
`X97801-480[*x97801-480]*`*`resource`*`: value`

Value

False The terminal emulator writes a record in the system log file /etc/utmp.
True The terminal emulator does not write a record in the system log file /etc/utmp.

3.259 Verbose

3.259.1 verbose

Defines whether a title line is to appear in the first line of the xmessages window.

Syntax

`XMessages*`*`resource`*`: `*`value`*

Value

False No title line os displayed.
True A title line is displayed.

3.260 VisualBell

3.260.1 visualBell

This resource specifies whether xterm should generate a visual bell rather than an acoustic one when CTRL-G is received.

Syntax

```
XPct[*xpct]*resource: value
XTerm[*vt100]*resource: value
X97801-480[*x97801-480]*resource: value
```

Value

False An acoustic signal is generated.
True A visual signal is generated.

3.261 Volume

3.261.1 volume

Defines the volume of the acoustic signal as a percentage of the maximum

Syntax

```
{ console  }
{ system   }
XMessages[*{          }]*resource: value
{ xerrors  }
{ common   }
```

Value

number
: Volume of the acoustic signal as a percentage of the maximum.

3.262 W0

3.262.1 w0

This resource determines the mapping of the lower half of the character generator work area to an allocation area for character sets.
You can thus specify the character set you wish to use. In the character set, you determine the mapping of character codes to characters.

Syntax

`X97801-480[*x97801-480]*`*`resource`*`: `*`value`*

Value

g0 The allocation area G0 is loaded to the lower half of the character generator work area. For further information, please refer to the resource g0 (class: g0, see section "G0 ").

g1 The allocation area G1 is loaded to the lower half of the character generator work area. For further information, please refer to the resource g1 (class: g1, see section "G1 ").

g2 The allocation area G2 is loaded to the lower half of the character generator work area. For further information, please refer to the resource g2 (class: g2, see section "G2 ").

g3 The allocation area G3 is loaded to the lower half of the character generator work area. For further information, please refer to the resource g3 (class: g3, see section "G3 ").

3.263 W1

3.263.1 w1

This resource determines the mapping of the upper half of the character generator work area to an allocation area for character sets.
You can thus specify the character set you wish to use. In the character set, you determine the mapping of character codes to characters.

Syntax

`X97801-480[*x97801-480]*`*`resource`*`: `*`value`*

Value

g0 The allocation area G0 is loaded to the upper half of the character generator work area. For further information, please refer to the resource g0 (class: g0, see section "G0 ").

g1 The allocation area G1 is loaded to the upper lower half of the character generator work area. For further information, please refer to the resource g1 (class: g1, see section "G1 ").

g2 The allocation area G2 is loaded to the upper half of the character generator work area. For further information, please refer to the resource g2 (class: g2, see section "G2 ").

g3 The allocation area G3 is loaded to the upper half of the character generator work area. For further information, please refer to the resource g3 (class: g3, see section "G3 ").

3.264 WMenuButtonClick

3.264.1 wMenuButtonClick

This resource specifies whether the window menu is posted and remains posted when you click the menu button.

Syntax

`Mwm*`*`resource`*`: `*`value`*

Value

False When you click the menu button, the window menu is posted and remains posted until you release the mouse button.

True When you click the menu button, the window menu is posted and remains posted even after you release the mouse button.

3.265 WMenuButtonClick2

3.265.1 wMenuButtonClick2

This resource specifies whether or not the mwm function f.kill is executed when you double-click the menu button.

Syntax

`Mwm*`*`resource`*`: `*`value`*

Value

False The mwm function f.kill is not executed when you double-click the menu button.
True The mwm function f.kill is executed when you double-click the menu button.

3.266 Width

3.266.1 width

This resource specifies the width of a window in pixels.

Syntax

```
Chooser*resource: value
XLogin.Login.resource: value
XTerm*tek4014*resource: value
```

Value

pixels
: Width of the Tektronix emulator window in pixels

3.267 WindowMenu

3.267.1 windowMenu

This resource specifies the name of the menu requested by the menu button on the window border.
Please refer to the section "Menus" in the mwm description for more information.

Syntax

```
{client_class   }
Mwm[*{client_instance}]*resource: value
{defaults       }
```

Value

name Name of the menu

3.268 Xerrors

3.268.1 xerrors

Specifies whether the log file identified by the file resource is to be monitored.

Syntax

`XMessages*`*`resource`*`: `*`value`*

Value

False The log file is not monitored.
True The log file is monitored.

3.269 XnlLanguage

3.269.1 xnlLanguage

This resource specifies the client's locale.
If this resource is not assigned a value, the clients use the value of the LANG environment variable.

Syntax

```
Mwm*resource: value
XPct[*xpct]*resource: value
XTerm*resource: value
X97801-480[*x97801-480]*resource: value
```

Value

language[*_territory*][*.char_set*]

language
: *language* specifies that part of the locale definition which identifies the language.

territory
: *territory* specifies that part of the locale definition which identifies the territory.

char_set
: *char_set* specifies that part of the locale definition which identifies the character set.

The syntax of the xnlLanguage resource value corresponds to the syntax of the LANG environment variable. For further information on the LANG environment variable, please refer to the reference manual for the operating system commands.

3.270 Xrdb

3.270.1 xrdb

This resource specifies the name of the program used to load the resources.

Syntax

`DisplayManager*`*`resource`*`: `*`value`*

Value

name Full pathname of the program

3.271 YellowColor

3.271.1 yellowColor

color setting for color terminals

Syntax

`XPct*`*`resource`*`: value`

Value

color Color name in the X server's color database

4 Assigning values to resources

You can assign values to resources by specifying options in the client call or by making entries in a configuration file. You will find the syntax for assigning values to resources in the section "Value assignment syntax ".

Assigning values to resources in the client call
When you call a client, you can use options to assign values to its resources. You use the -fg option to assign a value to the foreground resource, for example. A value assigned to a resource by means of an option in the client call takes precedence over a value assigned in a configuration file. The special option -xrm allows you to assign a value to each of the client's resources in the client call. Please note that when you use the -xrm option, you must specify the resource in just as much detail as when you assign values to it in files. Only then does a value assigned by means of the -xrm option take precedence over a value assigned to the resource in a configuration file.

Assigning values to resources in a configuration file
Generally, you use resources when you want to configure the appearance and behavior of a client long term. To do this, you assign a value to the resource in a configuration file. The section "Configuration files and their syntax " tells you which configuration files are available for resources and the syntax that must be observed in these files.

4.1 Configuration files and their syntax

Each client evaluates a number of configuration files for resources:
- the user-specific configuration file $XENVIRONMENT;
- the resource database of the X server;
- the client-specific configuration file $XUSERFILESEARCHPATH for users;
- the client-specific configuration file $XFILESEARCHPATH.

There are generally several default values for a file name, depending on various environment variables. The configuration files are described in the section "File hierarchy ". This section also describes what happens when a default is not found in a configuration file, and the client makes a default available in its binary file instead.

Syntax
The following syntax applies in all configuration files: each line of a configuration file can contain either a comment, an #include statement, or a value assignment to a resource.
- Comment: clients interpret blank lines and lines beginning with an exclamation mark (!) as comments and do not process them further. A command can consist of almost any characters; new-line characters and the NULL character are not permitted.
- #include statement: lines beginning with #include followed by a valid file name are replaced by the contents of the associated file. The file name is interpreted relative to the directory of the configuration file.
- Value assignment to a resource: the syntax of these lines is described in the section "Value assignment syntax ".

A value assigned to a resource in one configuration file can override a value assigned in another configuration file or later in the same file. Which of the values assigned to a

resource takes precedence over others assigned to the same resource, thus overriding them, depends on the file in which it is located and the accuracy with which the components for which the resource is to be assigned a value are specified.
If a configuration file contains several value assignments to a resource, which differ only in the value of the resource, the last one in the file applies.
The client evaluates the configuration files in the sequence described in the section "File hierarchy ". For each of these files the client uses the rules described in the section "Value assignment hierarchy within files ".

4.2 File hierarchy

The order in which the files are described in this section reflects their priority as far as value assignments are concerned. The value assignments made in a file override value assignments for the same resource (which use the same syntax) made in any of the files that follow the description of the file in question.
The following is the case in hand: A resource has been assigned a value in more than one file. You want to find out which resource value is valid for a client. This section describes the order in which the client prioritizes the files.
All of the following files are located on the host on which the client runs, unless explicitly stated otherwise.

4.2.1 User-specific configuration file $XENVIRONMENT

Default
$HOME/.Xdefaults-*hostname*

Environment variables
XENVIRONMENT; HOME

Explanation of the file
The only value assignments that can override the value assignments made in this file are those specified
- using options in the client call
- in the files described in the following sections if the value assignments in those files take precedence as defined in the section "Value assignment hierarchy within files ".

With this file there are two different cases to consider:
The XENVIRONMENT environment variable is set:
The client accesses the file $XENVIRONMENT.
The XENVIRONMENT environment variable is not set:
The client accesses the file $HOME/.Xdefaults-*hostname*. *hostname* is the host on which the client is running.

Contents of the file
In this file the ordinary user can assign user-specific resource values which are to be made available to all clients. In order for you to be able to assign host-specific resource values in a file, you should
1. create a file called *X-display_hostname* for each host *display_hostname* at which you are working on a display.
2. assign the XENVIRONMENT environment variable the current *X-display_hostname* each time you change the host on which you are working.

Clients
All clients have access to the value assignments in this file.

4.2.2 X server resource database

Default
$HOME/.Xdefaults

Environment variables
HOME

Explanation of the file
The only value assignments that can override the value assignments made in this file are those specified
- using options in the client call
- in the files already described
- in the files described in the following sections, if the value assignments in those files take precedence as defined in the section "Value assignment hierarchy within files ".

There are two different cases to consider:
The server has already created a resource database:
The server creates a database for the resources and their values if explicitly requested to do so by a client. Each time a client is called the server adds resources to this database as instructed by the client. This allows the server to inform the client of existing resource values for the display and terminal managed by the server when requested to do so.
Normally, when a client is started, it copies the resource database from the server and then uses this copy. This means that resource queries can be answered without putting an additional load on the network, but only using the values valid when the database was copied.
You can use the xrdb client to modify the database on the X server.
The client does not evaluate the $HOME/.Xdefaults file if the X server already has a database.
The server has not yet created a resource database:
The server only creates a database for resources and their values if a client explicitly requests it to do so. If no database exists, the client evaluates the $HOME/.Xdefaults file in order to determine a value for the resource.
You can use the xrdb client to query the database on the X server.

Contents of the file
In the X server's database you can assign user-specific resource values which are to be made available to all clients and which do not depend on the host on which the client is running.
Note that the client evaluates the $HOME/.Xdefaults file only if the X server has not created a resource database.

Clients
All clients have access to the value assignments in the server database.

4.2.3 Client-specific configuration file $XUSERFILESEARCHPATH for users

Default
- if XAPPLRESDIR is set:
  $XAPPLRESDIR/%L/%N%C
  $XAPPLRESDIR/%l/%N%C
  $XAPPLRESDIR/%N%C
  $HOME/%N%C
  $XAPPLRESDIR/%L/%N
  $XAPPLRESDIR/%l/%N

$XAPPLRESDIR/%N
$HOME/%N

- if XAPPLRESDIR is not set:
  $HOME/%L/%N%C
  $HOME/%l/%N%C
  $HOME/%N%C
  $HOME/%L/%N
  $HOME/%l/%N
  $HOME/%N

Environment variables
XUSERFILESEARCHPATH; XAPPLRESDIR; LANG; HOME

Explanation of the file
The only value assignments that can override the value assignments made in this file are those specified

- using options in the client call
- in the files already described
- in the files described in the following sections, if the value assignments in those files take precedence as defined in the section "Value assignment hierarchy within files ".

As a user you can assign the XUSERFILESEARCHPATH environment variable a value using the syntax described in the chapter "Environment variables ".
With this file there are a number of different cases to consider:
The XUSERFILESEARCHPATH environment variable is set:
The client replaces environment variables and symbolic names in the XUSERFILESEARCHPATH environment variable with concrete values. The client then reads the value assignments from the file specified by the leftmost filename which denotes an existing file.
If none of the specified files exists, the client behaves as though the XUSERFILESEARCHPATH environment variable were not set.
The XUSERFILESEARCHPATH environment variable is not set but the XAPPLRESDIR environment variable is set:
The client does not replace any environment variables or symbolic names in XAPPLRESDIR with concrete values. It does, however, create a new name using the XAPPLRESDIR environment variable:
`$XAPPLRESDIR/%L/%N%C:$XAPPLRESDIR/%l/%N%C:$XAPPLRESDIR/%N%C:$HOME/%N%C:`
`$XAPPLRESDIR/%L/%N:$XAPPLRESDIR/%l/%N:$XAPPLRESDIR/%N:$HOME/%N`
The client reads the value assignments from the file with the leftmost filename which denotes an existing file.
The XUSERFILESEARCHPATH environment variable is not set and the XAPPLRESDIR environment variable is not set:
The client creates a new name with the HOME environment variable:
`$HOME/%L/%N%C:$HOME/%l/%N%C:$HOME/%N%C:$HOME/%L/%N:$HOME/%l/%N:$HOME/%N`
The client reads the value assignments from the file with the leftmost filename which denotes an existing file.

Contents of the file
In this file the ordinary user can assign the user- and client-specific resources that are to be made available to a client instance of class `%N` and which do not depend on which host your

display is on, or on which host the client is running.

Clients
All client instances belonging to client class %N have access to the value assignments in this file. All clients use this file.

4.2.4 Client-specific configuration file $XFILESEARCHPATH

Default:
/usr/lib/X11/%L/app-defaults/%N%C%S
/usr/lib/X11/%l/app-defaults/%N%C%S
/usr/lib/X11/app-defaults/%N%C%S
/usr/lib/X11/%L/app-defaults/%N%S
/usr/lib/X11/%l/app-defaults/%N%S
/usr/lib/X11/app-defaults/%N%S

Environment variables
LANG

Explanation of the file
The only value assignments that can override the value assignments made in this file are those specified
- using options in the client call
- in the files already described
- in the files described in the following sections, if the value assignments in those files take precedence as defined in the section "Value assignment hierarchy within files ".

As system administrator you can assign the XFILESEARCHPATH environment variable a value using the syntax described in the chapter "Environment variables ".
With this file there are two different cases to consider:
The XFILESEARCHPATH environment variable is set:
The client replaces environment variables and symbolic names in the XUSERFILESEARCHPATH environment variable with concrete values. The client then reads the value assignments from the file specified by the leftmost filename which denotes an existing file.
If none of the specified files exists, the client behaves as though the XUSERFILESEARCHPATH environment variable were not set.
The XUSERFILESEARCHPATH environment variable is not set:
The client evaluates the following path:
```
/usr/lib/X11/%L/app-defaults/%N%C%S:/usr/lib/X11/%l/app-defaults/%N%C%S:
/usr/lib/X11/app-defaults/%N%C%S:/usr/lib/X11/%L/app-defaults/%N%S:
/usr/lib/X11/%l/app-defaults/%N%S:/usr/lib/X11/app-defaults/%N%S
```
The client reads the value assignments from the file specified by the leftmost filename which denotes an existing file.

Contents of the file
In this file the system administrator can assign the client-specific resource values that are to be made available to a client instance of class %N and which do not depend on which host your display is on, or on which host the client is running on.

Clients

All client instances belonging to client class %N have access to the value assignments in this file. All clients use this file.

4.2.5 Client defaults

If no entry exists in a file for a particular resource, the client uses the value defined as its default value.

4.3 Value assignment syntax

You can assign a value to a resource by means of options in the client call or entries in a configuration file. If you assign a value to a resource using the -xrm option in the client call or in a configuration file, you must comply with the syntax explained in this section.
The syntax of a value assignment *value_assignment* is described below. The syntax description consists of a number of transformation rules, each consisting of a left and right side. The two sides are separated by a symbol (::=).

```
value_assignment    ::= white_space resourceDefinition white_space ':'
white_space value
white_space         ::= { 'blank' ³ 'tabulator' }
resourceDefinition  ::= [ binding ] { component binding } componentName
binding             ::= '*' ³ '.'
component           ::= '?' ³ componentName
componentName       ::= character { character }
character           ::= 'a'-'z' ³ 'A'-'Z' ³ '0'-'9' ³ '_' ³ '-'
value               ::= { 'any character except NULL or a
newline that has not been invalidated' }
```

The left side consists of a single unique symbol. The right side consists of the possible transformation of the left side into symbols that either occur on the left side or are cannot be further resolved. If they cannot be further resolved, they are referred to as terminal symbols and placed in single quotes. The right side of a rule can also contain the following operators:

| OR operator. a|b means: either a or b.
{} repetition operator indicating that the contents of these brackets occur one or more times or not at all.
[] optional operator indicating that the contents of these brackets occur once or not at all.

Terminal symbols are:

- the names of components, referred to in the syntax as *componentName*. You can also use a question mark (?) as *componentName* for all components except the last one. The question mark replaces a component name without you having to know it.
  You can obtain the valid component names from the documentation of the client for which you want to assign a value to a resource. The component name furthest to the left of each syntax description in the chapter "Client resources" generally identifies the client and is its instance or class name.
  componentName can consist of uppercase and lowercase letters (A-Z and a-z), digits (0-9), the underscore (_) and the minus sign (-).-
- the values of resources, referred to in the syntax as *value*.
  You can obtain the permissible values for a resource from the chapter "Client resources".
  value can include almost all characters. Newlines that have not been invalidated by a preceding backslash (\) and the NULL character are exceptions.
  To obtain a blank as the beginning of *value*, enter the two-character string of a backslash followed by a blank (\ë); to obtain a tab, enter a backslash followed by a tab. To obtain a newline in *value*, enter the two-character string of a backslash followed by n (\n). To allow *value* to be entered over more than one line (to make it easier to read, for example), enter the two-character string of a backslash followed by a newline. To obtain any one-byte character in *value*, enter the four-character string of a backslash followed by three digits

between 0 and 7 (e.g. \000) specifying the octal value of the character. Finally, to obtain a backslash in *value*, enter the two-character string of two successive backslashes (\\).

- the bindings between components, referred to in the syntax as *binding*.
  binding is either an asterisk (*) for a loose binding between two component names or a period (.) for a tight binding between two component names.
  A loose binding with an asterisk replaces any number (including 0) of component names in the value assignment, whereas the period (.) bindings the names of two successive components. This allows you to make partly qualified value assignments to resources and use patterns.
  If there are two or more successive tight bindings, they are replaced at evaluation by a single tight binding. If, on the other hand, one of the successive bindings is a loose binding, the successive bindings are replaced by a single loose binding.
- white space between syntax elements, referred to in the syntax as *white_space*.
  white_space is either a blank or a tab.

4.4 Value assignment hierarchy within files

The client applies the following rules in the given order to determine which entry specifying a resource value takes precedence in a file if several entries for the same resource exist in that file.

- A value assignment which uses a tight binding (a dot ".") overrides value assignments containing a loose binding (asterisk "*").
  Therefore you should use an asterisk for more general specifications and the dot in exceptional cases.
  Example: mwm*icon.background: red
  overrides mwm*icon*background: green
- A value assignment which uses an instance name overrides value assignments which use class names, regardless of whether they are client names, resource names or any other specified components.
  Therefore you should assign resource values using class names for more general specifications and the name of instances for exceptional cases.
  Example: mwm*icon.background: red
  overrides mwm*icon.Background: green for the background resource instance.
- A value assignment which uses a class name overrides value assignments with a question mark (?) as a component name, regardless of whether they are client names or any other specified components.
  You should therefore only use the question mark if you want to specify a single component but do not know its name.
  Example: mwm*Icon.background: red
  overrides mwm*?.Background: green for Icon components.
- A value assignment containing more components in the resource syntax overrides value assignments containing fewer components.
  Therefore you should use as few components as possible to assign resource values in a more general context and as many components as possible for exceptional cases.
  Example: mwm*icon.background: red
  overrides mwm*background: green in value assignments to icons used by mwm.
- A value assignment containing a greater number of components specified from the left, i.e. starting with the client, overrides value assignments with a smaller number of components specified from the left. Exception: A value assignment in which a client is specified overrides value assignments in which no client is specified.
  Example: mwm*background: red
  overrides *background: green
- A value assignment which has no binding specified in front of the first component is handled in the same way as value assignments beginning with a tight binding (".").
  Example: mwm*background: red
  is handled the same way as .mwm*background: red

The more exact, i.e. detailed, the syntax used to assign a value to a resource is, the higher the priority of the value assignment.

5 Environment variables

In this chapter you will find alphabetically arranged descriptions of the environment variables used by clients.

5.1 DISPLAY

Each X server supports exactly one display, which has a unique name. The display name is the value assigned to the DISPLAY environment variable. The display name has the following format:
[*hostname*]:*display_number*[.*screen_number*]
A client evaluates the DISPLAY environment variable to find out which X server, display and screen it is to use if you have not explicitly specified a value for the display name with the -display option.
SINIX/windows supports the TCP transport protocol.

hostname
: *hostname* is the name of the host to which the display is physically connected. This *hostname* must be valid for the transport protocol which is being used. With regard to the transport protocol being used, the local host needs to know the names of hosts that can be reached using the protocol plus a services identifier xserver*display_number*.
If no hostname is specified, the most effective communication path to an X server on the local host will be used.
Valid hostnames are:
names of hosts that the local host is able to associate with the transport protocol being used.

:*display_number*
: A display can comprise one or more screens controlled by a single keyboard and a single pointing device, i.e. mouse, hand-held cursor or graphics tablet. As a host can support more than one display concurrently, each display is assigned a *display_number* as soon as the X server for the display has been started. Numbering is sequential, starting with 0. This makes it possible to uniquely identify each display. *display_number* must always be specified in this environment variable.

.*screen_number*
: A display can comprise one or more screens controlled by a single keyboard and a single pointing device, i.e. mouse, hand-held cursor or graphics tablet. In order to be able to select a certain screen, each screen on a display is assigned a *screen_number* as soon as the X server for the display has been started. Numbering is sequential, starting with 0. This makes it possible to clearly identify each screen. If no *screen_number* is specified, screen 0 is used.

The DISPLAY environment variable is evaluated by all clients.

5.2 HOME

Clients use the HOME environment variable to find user-specific configuration files. They search for these files in the identified directory or its subdirectories.

5.3 LANG

Clients use the LANG environment variable to find language-specific configuration files.

5.4 MAIL

The MAIL environment variable is assigned the absolute path name of the file to which the mail command writes the user's mail.
The MAIL environment variable is evaluated by the xdm and xdmm clients.

5.5 MWMSHELL

The MWMSHELL environment variable is evaluated by the mwm function f.exec in order to execute a command. If MWMSHELL is not set, mwm evaluates the SHELL environment variable.

5.6 NLSPATH

The NLSPATH environment variable defines the search path for language-specific message files.
The value of the NLSPATH environment variable is a string in the format:
pathname$_1$[[:*pathname*$_2$]...]
pathname$_i$ is the pathname of a message file. The client reads the messages from the message file with the leftmost pathname denoting an existing file.
The pathnames may also include the following symbolic names:

- %l
  is replaced by that part of the locale definition which identifies the language.
- %N
  is replaced by the client name as given in the descriptions in this manual.

If you want to use the percent sign % as part of a name, you need to double it (%%). If the locale is undefined, or if parts of the locale definition are missing, the client substitutes the null string for undefined symbolic names. In compliance with X/Open requirements, two or more adjacent slashes "/" are converted to a single slash.
The clients which evaluate the NLSPATH environment variable are xpct, x97801-480, xdesk and xdfm.

5.7 SHELL

The SHELL environment variable is evaluated by the mwm function f.exec in order to execute a command. mwm evaluates SHELL only if MWMSHELL is not set.

5.8 TERM

The value assigned to the TERM environment variable is the terminal type valid for the window involved. Terminal emulators automatically set this variable. The following values are supported:

- by xpct: xpct, xansi, ansi, dumb
- by xterm: xterm, vt102, vt100, ansi, dumb

The TERM environment variable is evaluated by xpct and xterm.

5.9 TRASH

Specifies the directory in which the desktop and file manager creates the directory for trash. The name of the directory for trash begins with a period (.) and is therefore visible in $TRASH only if these file and directory names are displayed. If the variable is not set, the client evaluates the HOME environment variable.

5.10 TZ

The TZ variable contains timezone information. The date command and most of the library functions which reference the time of day use TZ to determine the timezone and to convert universal time UTC (Universal Time Coordinated) to local time and vice versa.
The TZ environment variable is evaluated by xdm. It has the following syntax:
Format 1*: std offset1*[*dst*[*offset2*][;*start*[/*time*],*end*[/*time*]]]
Format 2*: std offset1*[*dst*[*offset2*][,*start*[/*time*],*end*[/*zeit*]]]
Format 3: *zonename*
The individual components of the value of TZ follow one another without intervening blanks, e.g. MET-1 (Format 1 or 2) or :US/Pacific (Format 3). The meaning of each component is as follows:

Format 1

std Standard local time zone
std consists of at least three letters, either uppercase or lowercase.
Note: *std* has no effect on the calculation of local time. It simply defines the name of the time zone.

offset1
Indicates the value that must be added to the local time in order to arrive at Coordinated Universal Time (UTC, also known as Greenwich Meantime, or GMT).
offset1 has the following format:
[±]*hh*[:*mm*[:*ss*]]

± If it is preceded by a plus sign, the time zone is west of the prime (or Greenwich) meridian. If it is preceded by a minus sign, the time is east of the prime meridian.
Default:
same effect as +.

hh Hours (0-24)

mm Minutes (0-59)

ss Seconds (0-59)

Example:
Central European Time (CET) is UTC + 1 hour: CET=UTC+1, i.e. UTC=CET-1, where -1 corresponds to offset1.

dst Summer timezone (daylight saving time; same format as *std*). As with *std*, *dst* simply defines the name of the summer timezone.
Default:
there is no conversion to summer time.

offset2
Indicates the value that must be added to the local summer time in order to arrive at UTC. *offset2* has the same format as *offset1*.
Default:
summer time = standard time + 1 hour

start[/*time*],*end*[/*time*]
start[/*time*] defines when to change from standard time to summer time; *end*[/*time*]

specifies when to change back. *start* and *end* specify the day on which the change is made; *time* indicates when this happens (in current local time).
start and *end* are given in the form of a number *n*, where *n* stands for the *n*th day of the Julian Calendar and is greater than 0 but less than 365. Leap days are counted, so in this format it is possible to refer to February 29.

time has the format: *hh*[:*mm*[:*ss*]]
(See "")
Default:
time = 02:00:00
Default:
The change from standard time to summer time occurs on the last Sunday in March; summer time is changed back to standard time on the last Sunday in September (at 2am in each case).

Format 2
Format 2 is like Format 1 with the addition of more flexible syntax for specifying when summer time starts and ends. Note that the only difference in the syntax synopsis is that Format 2 has a comma instead of a semicolon before *start*[/*time*],*end*[/*time*].

std see "Format 1"
offset1
see "Format 1"
dst see "Format 1"
offset2
see "Format 1"
start[/*time*],*end*[/*time*]
start[/*time*] defines when to change from standard time to summer time; *end*[/*time*] specifies when to change back. *start* and *end* specify the day on which the change is made; *time* indicates when this happens (in current local time).

start and *end* can have any of the following formats: J*n*, *n*, M*m*.*w*.*d*
J*n*
stands for the *n*th day of the Julian calendar, where 1 < *n* < 365. Leap days are not counted. In other words, February 28 is always day 59, and March 1 is always day 60. It is impossible to refer to February 29 in this format.
n
stands for the *n*th day of the Julian Calendar where 0 < *n* < 365. Leap days are counted, and it is possible to refer to February 29.
M*m*.*w*.*d*
stands for the *d*th day of week *w* of month *m* (1 < *m* < 12, 1 < *w* < 5, 0 < *d* < 6). Day zero of a week is Sunday. Week 1 of a month is the week in which the first day of the month falls. When a value of 5 is specified for a week (M*m*.5.*d*), the "last *d* day in month *m*" is implied. For example, M9.5.6 refers to the last Saturday in September.
time has the format: *hh*[:*mm*[:*ss*]]
(see Format 1, "")
Default:

time = 02:00:00

Default:

The change from standard time to summer time occurs on the last Sunday in April; summer time is changed back to standard time on the last Sunday in October (at 2am in each case).

Format 3

zonename

zonename is the name of a predefined timezone with a predefined change to summer time. The accepted values for *zonename* are the names of the files under /usr/lib/locale/TZ. These files are created with the system administration command zic.

Tip:

A syntax error in the TZ environment variable triggers a default response. If *std* or *offset1* is unidentifiable (Format 1 or 2), or if the named file does not exist under /usr/lib/locale/TZ (Format 3), the clock reverts to GMT. In other cases, only the valid part of the variable is interpreted, and the rest is ignored.

5.11 USER

The USER environment variable is evaluated by xdm and xdmm to identify the user.

5.12 WINDOWID

The WINDOWID environment variable is set by the clients xterm and xpct when they are invoked. Its value is the number which uniquely identifies the xterm or xpct client window within a display.

5.13 XAUTHORITY

The $XAUTHORITY file contains user-specific authorization information. This information must be available to all clients started by a user during a session so that they can access the server.
The xdm client writes the authorization information to the $XAUTHORITY file at the beginning of a user session. If the value of this environment variable is not set, the client uses the .Xauthority file in the user's home directory.
The file has the following format:
display protocol key

display
identifies the display for which authorization information is to be processed as described in the DISPLAY environment variable.

protocol
is the name of the authorization protocol. The only authorization mechanism that the international version supports is MIT-MAGIC-COOKIE-1; the US version also supports XDM-AUTHORIZATION-1.

key is the authorization key in hexadecimal form. The authorization key is generated by xdm, the display manager, at the beginning of each user session, written to the authorization file, and made known to the server. It is the access key for all clients the user starts, and is valid for the duration of the session only.

5.14 XAPPLRESDIR

In the file $XAPPLRESDIR the ordinary user can assign the user- and client-specific resources that are to be made available to a client instance of class %N and which do not depend on the host your display is on, or on which host the client is running on. The XAPPLRESDIR environment variable is evaluated only when the XUSERFILESEARCHPATH environment variable is not set.
The client does not replace any environment variables or symbolic names in XAPPLRESDIR. It does, however, create a new name using the environment variable XAPPLRESDIR:

```
$XAPPLRESDIR/%L/%N:$XAPPLRESDIR/%l/%N:$XAPPLRESDIR/%N:$HO ME/%N
```

The client reads the value assignment from the file with the leftmost filename denoting an existing file.
All client instances belonging to the client class %N use this file. For further information on this file, please refer to the section "Configuration files and their syntax ".

5.15 XBMLANGPATH

The XBMLANGPATH environment variable defines a list of directories the client is to search for pixmaps and bitmaps.
The list of directories may include symbolic names. The client expands:

- %B to the name of the file containing the pixmap or bitmap,
- %N to the client's class *client_class*,
- %L to the value of the xnlLanguage resource (class: XnlLanguage).
  If there is no value assigned to this resource, the client expands %L to the value of the LANG environment variable. %L stands for the locale you are working in.
- %l to that part of the locale definition which identifies the language,
- %t to that part of the locale definition which identifies the territory,
- %c to that part of the locale definition which identifies the codeset.

If the XBMLANGPATH environment variable is not set but XAPPLRESDIR is, the pixmap or bitmap is searched for under the following names and in the following sequence:
%B
$XAPPLRESDIR/%L/bitmaps/%N/%B
$XAPPLRESDIR/%l/bitmaps/%N/%B
$XAPPLRESDIR/bitmaps/%N/%B
$XAPPLRESDIR/%L/bitmaps/%B
$XAPPLRESDIR/%l/bitmaps/%B
$XAPPLRESDIR/bitmaps/%B
$HOME/bitmaps/%B
$HOME/%B
/usr/lib/X11/%L/bitmaps/%N/%B
/usr/lib/X11/%l/bitmaps/%N/%B
/usr/lib/X11/bitmaps/%N/%B
/usr/lib/X11/%L/bitmaps/%B
/usr/lib/X11/%l/bitmaps/%B
/usr/lib/X11/bitmaps/%B
/usr/include/X11/bitmaps/%B
If neither XBMLANGPATH nor XAPPLRESDIR is set, the pixmap or bitmap is searched for under the following names and in the following sequence:
%B
$HOME/%L/bitmaps/%N/%B
$HOME/%l/bitmaps/%N/%B
$HOME/bitmaps/%N/%B
$HOME/%L/bitmaps/%B
$HOME/%l/bitmaps/%B
$HOME/bitmaps/%B
$HOME/%B
/usr/lib/X11/%L/bitmaps/%N/%B
/usr/lib/X11/%l/bitmaps/%N/%B
/usr/lib/X11/bitmaps/%N/%B
/usr/lib/X11/%L/bitmaps/%B
/usr/lib/X11/%l/bitmaps/%B
/usr/lib/X11/bitmaps/%B
/usr/include/X11/bitmaps/%B

5.16 XDESK_CFG_PATH

The XDESK_CFG_PATH environment variable defines the list of directories the xdesk, xdfm, xdhm, xdmm and xdum clients are to search for their .xdesk.cfg configuration file. The items in the list are separated from each other by colons (:). If the variable is not set, the client evaluates the HOME environment variable.

5.17 XDESK_ICON_PATH

The XDESK_ICON_PATH environment variable outputs a user-specific list of files containing icon images for objects managed by the xdesk, xdfm, xdhm, xdmm and xdum clients. The items in the list are separated by colons (:). If this environment variable is set, it overrides the value of the iconPath resource of class IconPath. After searching the user-specific list, the client searches the system-specific list of files defined by means of the XDESK_SYSTEM_ICON_PATH environment variable.
The list of files may include symbolic names. The client replaces

- %N with the name of the file containing the icon image,
- %I with the value of the iconSet resource,
- %L with the locale as defined by the LANG environment variable,
- %l with the part of the locale which defines the language,
- %t with the part of the locale which defines the territory,
- %c with the part of the locale which defines the codeset.

5.18 XDESK_SAVE_DIR

The XDESK_SAVE_DIR environment variable defines the directory in which the xdesk, xdfm, xdhm, xdmm and xdum clients store their save files. If the variable is not set, the client evaluates the HOME environment variable.

5.19 XDESK_SYSTEM_ICON_PATH

The XDESK_ICON_PATH environment variable outputs a system-specific list of files containing icon images for objects managed by the xdesk, xdfm, xdhm, xdmm and xdum clients. If this environment variable is set, it overrides the value of the systemIconPath resource of class SystemIconPath.
The client begins by searching the list of files defined by means of the XDESK_ICON_PATH environment variable.
The list of files may include symbolic names. The client replaces

- %N with the name of the file containing the icon image,
- %I with the value of the iconSet resource,
- %L with the locale as defined by the LANG environment variable,
- %l with the part of the locale which defines the language,
- %t with the part of the locale which defines the territory,
- %c with the part of the locale which defines the codeset.

5.20 XDMM_TRASH

The XDMM_TRASH environment variable specifies the directory in which the client creates the directory for mail trash. If the variable is not set, the client uses the directory $HOME/.xdmmtrash.

5.21 XENVIRONMENT

In the file $XENVIRONMENT you can assign the user-specific resource values that are to be made available to all clients.

5.22 XFILESEARCHPATH

In this file the system administrator can assign the client-specific resource values that are to be made available to a client instance of class %N and which do not depend on which host your display is on, or on which host the client is running on.
The system administrator can assign a string with the following syntax to the XFILESEARCHPATH environment variable:
filename$_1$[[:*filename*$_2$]...]
This string is made up of a series of filenames *filename*$_n$, each separated from the other by a colon (:). The client reads the value assignment from the file with the leftmost filename denoting an existing file.
As *filename*$_n$ you may specify the absolute or relative pathname; environment variables may also be used. It is also possible for you to use the following symbolic names:

- %c
  The client replaces "%c" with that part of the locale definition which identifies the character set. The locale is defined by %L.
- %L
  The client replaces "%L" with the value of the xnlLanguage resource instance (class: XnlLanguage). If this resource instance has not been assigned a value, the client replaces "%L" with the value of the LANG environment variable. This environment variable determines the locale in which you work.
- %l
  The client replaces "%l" with that part of the locale definition which identifies the language. The locale is defined by %L.
- %N
  The client replaces "%N" with the client class *client_class* of the client that queried the value of a resource.
- %t
  The client replaces "%t" with that part of the locale definition which identifies the territory. The locale is defined by %L.

If you want to use the percent sign % as part of a name, you need to double it (%%). If the locale is undefined, or if parts of the locale definition are missing, the client substitutes the null string for undefined symbolic names. In compliance with X/Open requirements, two or more adjacent slashes "/" are converted to a single slash.
For example: The resource instance xnlLanguage has not been assigned a value, and the environment variable LANG has been assigned the value En_US.ASCII. The x97801-480 client performs the following replacements:

- %L with En_US.ASCII
- %l with En
- %N with X97801-480
- %t with US
- %c with ASCII
- %% with %

All client instances belonging to client class %N use this file. For further information on this file, please refer to the section "Configuration files and their syntax ".

5.23 XMBINDDIR

The XMBINDDIR environment variable defines the path for the xmbind.alias file, from which the client obtains the name of the file from which it reads the virtual key and button mapping. If the variable is not set, the client searches for the file in the /usr/lib/Xm/bindings directory.

5.24 XUSERFILESEARCHPATH

In the file $XUSERFILESEARCHPATH the ordinary user can assign the user- and client-specific resources that are to be made available to a client instance of the class %N and which do not depend on which host your display is on, or on which host the client is running on.
As an ordinary user, you can assign a string with the following syntax as the value for the XUSERFILESEARCHPATH environment variable:
filename$_1$[[:*filename*$_2$]...]
This string is made up of a series of filenames *filename*$_n$, each separated from the other by a colon (:). The client reads the value assignment from the file with the leftmost filename which denotes an existing file.
As *filename*$_n$ you may specify the absolute or relative pathname; environment variables may also be used. It is also possible for you to use the following symbolic names:

- %c
  The client replaces "%c" with that part of the locale definition which identifies the character set (codeset). The locale is defined by %L.
- %L
  The client replaces "%L" with the value of the xnlLanguage resource instance (class: XnlLanguage). If this resource instance has not been assigned a value, the client replaces "%L" with the value of the LANG environment variable. This environment variable determines the locale in which you work.
- %l
  The client replaces "%l" with that part of the locale definition which identifies the language. The locale is defined by %L.
- %N
  The client replaces "%N" with the client class *client_class* of the client that queried the value of a resource.
- %t
  The client replaces "%t" with that part of the locale definition which identifies the territory. The locale is defined by %L.

If you want to use the percent sign % as part of a name, you need to double it (%%). If the locale is undefined, or if parts of the locale definition are missing, the client substitutes the null string for undefined symbolic names. In compliance with X/Open requirements, two or more adjacent slashes "/" are converted to a single slash.
For example: the resource instance xnlLanguage has not been assigned a value, and the environment variable LANG has been assigned the value En_US.ASCII. The x97801-480 client performs the following replacements:

- %L with En_US.ASCII
- %l with En
- %N with X97801-480
- %t with US
- %c with ASCII
- %% with %

All client instances belonging to the client class %N use this file. For further information on this file, please refer to the section "Configuration files and their syntax ".

6 Related publications

[1] SINIX/windows
Übersicht zur Dokumentation

[2] SINIX/windows User Environment
Guide for Experts and System Administrators
(SINIX Desktop)
System Administrator Guide

[3] SINIX/windows User Environment
Introduction to Handling and Configuration
(SINIX Desktop)
User Guide

[4] SINIX/windows User Environment
XDCL Desktop Configuration Language
(SINIX Desktop)
Reference Manual

[5] SINIX/windows User Environment
Desktop Kornshell User's Guide
(Common Desktop Environment)

[6] SINIX/windows User Environment
Guide for Users
(Common Desktop Environment)
User Guide

[7] SINIX/windows User Environment
TED Enhancements
(Common Desktop Environment)

[8] SINIX/windows User Environment
Guide for Experts and System Administrators
(Common Desktop Environment)
System Administrator Guide

[9] SINIX/windows Development
Product Overview

[10] SINIX/windows Development
OSF/Motif
Style Guide Release 1.2

[11] SINIX/windows Development
OSF/Motif
Programmer's Reference, Release 1.2

[12] SINIX/windows Development
OSF/Motif
Programmer's Guide, Release 1.2

[13] SINIX/windows Development
OSF/Motif
X Server Reference Manual

[14] SINIX/windows Development
X Window System, Xt Toolkit Intrinsics

Reference Manual
[15] SINIX/windows Development
X Window System X11, Xlib
Programming Manual
[16] SINIX/windows Development
X Window System, Xlib
Reference Manual
[17] SINIX/windows Development
X Window System X11 Rel.5 (SINIX), X Input Extension
Programmer's Reference
[18] SINIX/windows Development
X Window System, X Protocol
Reference Manual
[19] SINIX/windows Development
Widget Set
Programmer's Reference
[20] SINIX/windows Development
Programmer's Guide
(Common Desktop Environment)
[21] SINIX/windows Development
Programmer's Overview
(Common Desktop Environment)
[22] SINIX/windows Development
Application Builder User's Guide
(Common Desktop Environment)
[23] SINIX/windows Development
Help System Author's and Programmer's Guide
(Common Desktop Environment)
[24] SINIX/windows Development
Internationalization Programmer's Guide
(Common Desktop Environment)
[25] SINIX/windows Development
Style Guide and Certification Checklist
(Common Desktop Environment)
[26] SINIX/windows Development
Tooltalk Messaging Overview
(Common Desktop Environment)
[27] SINIX/windows Development
Tooltalk Programmer's Guide
(Common Desktop Environment)

Ordering manuals

The manuals listed above and the corresponding order numbers can be found in the Siemens Nixdorf *List of Publications*. New publications are described in the *Druckschriften Neuerscheinungen (New Publications)*.
You can arrange to have both of these sent to you regularly by having your name placed on the appropriate mailing list. Please apply to your local office, where you can also order the manuals.

Glossary

Symbol 3D appearance

Motif-based interface normally display the visible objects on the screen in 3D form, as long as these objects are managed by the window manager. Three dimensional representation is provided by all the window manager resources that apply to the shadow of an object, e.g. the bottomShadowColor resource instance of the class Foreground.

A accelerator key

You can define a key sequence that can be used to select a certain menu item. Instead of posting a menu in a window and selecting an item from it, you can press the defined combination of keys and thus have faster access.

access control list

You allocate each server a list of the hosts whose users are permitted to access that server once it has been started. In addition to the users on the hosts, users on the local host have access to the local server. This list is maintained in the file /etc/X*displaynumber*.hosts, where *displaynumber* is the number of the display on the local host managed by the server. When the server is started, it reads the appropriate file for a given display and makes a copy of this file. You can use xhost to access this copy.

active window

An active window is a window which has the keyboard input focus.

ancestor

An ancestor of a window A is a window B to which window A is assigned. A is the child of its ancestor B. If A is assigned directly to B, B is the parent of A. If, however, A is assigned to window C and window C to window B, then C is the parent of A; C and B are ancestors of A.

You can only see a window within its ancestors. Every window (except the root window) has exactly one parent. The root window is an ancestor of all the windows on a screen.

authorization

Authorization is a mechanism designed to check whether a client is allowed to access the X server in the course of a user's session. The X server grants access only to X clients which can supply a defined session key. The xdm display manager uses this authorization mechanism: it generates the session key, tells the X server what it is and stores it in the user's authorization file. The authorization mechanism used to generate the session key is MIT-MAGIC-COOKIE-1 in the international version, XDM-AUTHORIZATION-1 in the US version.

B backing store

A backing store is an area of storage in the server.

The server can manage background storage in which the contents of a window are saved if the window is obscured by another window. This means that the server is able to restore the window contents once the window is exposed without the help of the client. Not every server always supports the backing store function.

bitmap
A bitmap is a pixmap of depth one.

bit plane
The *n*th bit of all the pixels in a pixmap make up the *n*th bit plane of the pixmap. The number of bit planes per window depends on the number of bits per pixel, the depth of the pixmap. All bit planes can be displayed on a screen if the screen provides at least the same number of color planes.

button
There are two kinds of button: First, there are buttons on the mouse, i.e. mouse buttons. Second, there are small squares on window frames which you click in order to perform a certain function, also referred to as window buttons

button binding
A button binding is the association of a mouse button with a window manager function. When defining a binding, you can specify which button action is to invoke the mwm function, e.g. pressing a button, releasing a button or dragging the mouse with a button held down. You may also specify modifiers which must be pressed along with the mouse button in order to execute the mwm function.

button event
A button event is an event which the server sends to a client to inform it that a mouse button has been pressed or released in a certain window or that the mouse has been moved while a mouse button was being held down.

C

character
The alphanumeric characters, special characters and graphic elements of a character set, without a definition of their representation in terms of width, height and line thickness, are called characters. The representation of a character is referred to as a glyph.
In general, the terms font and character set are synonymous, as are the terms glyph and character. It is necessary to make a distinction only in the context of the representation of a character.

character code
The character code is the code for a character as used internally by the machine.

character set
Alphanumeric characters, special characters and graphic elements (e.g. for the pointer) are grouped together in a character set under one name, the name of the character set. The alphanumeric characters, special characters and graphic elements of a character set, without a definition of their representation in terms of width, height, line thickness and other metric information, are called characters. The representation of a character is referred to as a glyph.
In general, the terms font and character set are synonymous, as are the terms glyph and character. It is necessary to make a distinction only in the context of the representation of a character.

child

A child of a window A is a window B, which is assigned directly to window A. Window A is the parent of window B. A window can have more than one child. All the offspring of a parent are stacked in a certain order on the screen, this is the stacking order. All the offspring of window A, and all their offspring, etc. on down the line, are descendants of A.

client

A client is an application program which exchanges data with a server on the same machine on which the server is running or on another machine. In order to do this the client sets up a connection to the server, e.g. a TCP connection. The server handles all connections as separate connections, even if a client has established several connections to the server.

client area

The client area is a rectangular area within a client window.

If a client requests a new window and this window is to be managed by mwm, mwm places a request for a new window with the server. The window comprises a border area, the mwm window frame including the window decoration and a rectangular area within the window, i.e. the client area.

The mwm window frame makes it possible for you to resize the window, move the window and perform other similar actions. The client area is the area to which the client directs output. The client area normally has an internal border, which ensures a predefined distance between the text and graphics in the client area and its border. A distinction is made between the internal border in the client area and the matte and the mwm window frame, both of which are managed by mwm.

client window

There are some clients which do not require windows, but normally a client expects input and output in a window. It therefore requests that the server open a window. Each window that the client opens for input or output is a client window which belongs to this client. Client windows which are offspring of the root window are referred to as primary windows; all other windows are secondary windows.

color plane

The *n*th bit of all the pixels of a screen make up the *n*th color plane on the screen. The maximum number of color planes on a screen depends on the hardware involved. The number of color planes available on a screen determines the colors (and gray scales) that can be displayed.

colormap

The colormap of a window determines the colors available to the window associated with the colormap. It also determines the color into which a pixel value is converted. The colormaps supported by the server depend on the selected visual class.

A colormap is made up of columns and rows. The columns are assigned the three colors red, green and blue. Each line is a color entry and generates a certain color on the screen. The visual class determines what a color entry looks like and how it is accessed.

Depending on the hardware, it is possible to define more than one colormap. Each client can, however, regardless of hardware limitations, define any number of virtual colormaps. The window manager can load virtual colormaps. If the storage space allocated to the

colormaps has been exhausted and a new colormap is loaded, an existing colormap will be deleted, provided the screen hardware allows this.

colormap focus

The colormap focus determines which colormap the server uses. The colormap focus is set to the colormap whose associated window is the active window, i.e. has the input focus. If more than one colormap has been defined, the client can set the focus to a certain colormap.

D descendant

A descendant of a window A is a window B, which is assigned to window A. A is the ancestor of B. If window A is the parent of window B, then B is the child of A. If, however, window B is assigned to window C and window C to window A, then B is the child of C, and windows B and C are descendants of A.

You can only see a window within its ancestors. Every window (except the root window) is a descendant of the root window.

display

A display is made up of one or more screens, a keyboard and a mouse. As a server only manages one display, the term display is often used as a synonym for server. The DISPLAY environment variable contains all the information required to uniquely identify a server in a network: the hostname, the display number and the screen number.

The mouse is used here as a collective term for input devices which can be used to quickly indicate a certain position on the screen to the host. Another example of an input device is a graphics tablet. A pointer marks the current position of the mouse on the screen. The text cursor, on the other hand, marks the position in a window at which input from the keyboard is accepted. The pointer and the text cursor can be various shapes, depending on where they are on the screen.

The mouse has buttons which you can use to select objects, e.g. items from a menu. The keyboard has keys.

E event

An event is a message from the server to a client about an occurrence. This message is normally transmitted asynchronously. An event provides information about a button or key event and about the window which had the input focus at the time of the event. Each client decides which events it will process and receives only those events explicitly requested from the server. The client can request the server to pass on events to other clients.

F font

Alphanumeric characters, special characters and graphic elements (for the pointer, for example) are grouped together under one name, the font name. The alphanumeric characters, special characters and graphic elements of a font are called glyphs. A glyph is the representation of a character. A glyph contains information on width, height, line thickness and other metric information regarding a glyph. A font contains additional metric information to determine interglyph and interline spacing.

In general, the terms font and character set are synonymous, as are the terms glyph and character. It is necessary to make distinctions only in the context of the representation of a

character.

The xlsfonts client provides a list of all the fonts available to the current display.

In the X Window System, fonts are always provided by the server. The clients can request a certain font from the server, but the server will not necessarily be able to access that font. If the server receives a request for a font, it will try to load the font from one of the directories specified in the font path. See the server description for further information on the font path.

foreground

The foreground of a window is either single-colored or patterned, depending on whether you have specified a color or a pixmap for the window foreground. Text and graphics are displayed in the foreground color or pattern against the window background.

G glyph

A glyph is the representation of a character. A glyph is an element of a font, i.e. an alphanumeric character, a special character or a graphic element. Unlike characters, glyphs have a fixed width, height and line thickness.

In general, the terms font and character set are synonymous, as are the terms glyph and character. It is necessary to make distinctions only in the context of the representation of a character.

gray out

If something is "grayed out", this indicates that this object is not currently available.

In the default configuration, those parts of the windows which you cannot currently access are grayed out. For example, menu items which would not perform a meaningful action in the current context are grayed out.

I icon

An icon is a small window which represents another window but takes up less space on the screen than the window itself.

In the case of the window manager mwm, an icon is made up of an icon image and an icon label.

icon box

The icon box is a window in which icons can be collected. If you use mwm's icon box, a window can exist concurrently in two states: as an icon in the icon box and as a normal window.

input focus

Windows can be assigned the keyboard input focus. The window which has the input focus and the client which manages this window thus receive, from the server, all information regarding keyboard input.

A window which has the keyboard input focus is referred to as the active window.

K key

Each key on the keyboard is assigned a certain keycode on the server; this keycode cannot be changed. Using the xmodmap client, you can assign this keycode a keysym. The

keysym is what the client actually interprets as input. The keycode can be modified by means of the modifier keys. A keycode can be assigned to a different keysym by pressing a modifier key at the same time as another key on the keyboard. xmodmap can also be used to define the assignment of modifiers to keysyms.

key binding
A key binding is the association of a key with a window manager function. When defining a key binding, you can specify modifiers which must be pressed at the same time as a particular key in order for the mwm function to be invoked.

key event
A key event is an event which is sent by the server to a client to inform the client that a certain key has been pressed in a certain window and which modifiers were pressed at the same time.

M

matte
The matte is part of the window decoration. It is located between the client area in a client window and the resize border, which is also part of the window decoration. You can configure the matte using the appropriate resources.

menu
A menu is a window containing a list of menu items. You can click a menu item using the mouse, thus invoking the function defined for the menu item.

mnemonic
A mnemonic is a single letter of the alphabet which appears somewhere in a menu item and uniquely identifies that item within the menu; it appears underscored in the menu item name. When a menu is posted, you can select an item from it by pressing the key corresponding to its mnemonic.

P

parent
A parent of window A is window B, to which window A is assigned directly. If A is the child of B, then B is the parent of A. If, however, window A is assigned to window C and window C to window B, then C is the parent of A, and C and B are ancestors of A.
You can only see a window within its ancestors. Each window (except the root window) has exactly one parent. The root window is an ancestor of all the windows on a screen.

pixmap
A pixmap is a two-dimensional arrangement of pixels, the smallest addressable element of a screen. Each pixel in a pixmap has the same number of bits, e.g. n. The pixmap is then said to have the depth n, or the pixmap has n bit planes. These bit planes can be displayed on a screen if the screen provides at least n color planes: one color plane for each bit plane. If you have n bit planes, 2^n colors are available for each pixel.
You can use pixmaps for icons, pointers, background and foreground patterns, and so forth.

primary window
Primary windows are the client windows which are children of the root window. All other client windows are secondary windows.

R

resource

Resources provide you with a way of creating your own user interface according to your own personal tastes. You can use resources to influence the size, position, color and pattern of windows, the window decoration, the 3-D shadows, the font used in windows and input processing.

root window

Each screen on a display has a root window. All other windows on this screen are descendants of the root window.

S

save under

A *save under* is an area of storage in the server.

The server can manage this type of storage area for windows. The contents of a window are saved in this storage area if the window is obscured by another window. This means that the server is able restore the window contents once the window is exposed without the help of the client. Not every server always supports the save under function.

scroll area

The scroll area represents the visible part of a client's input and output to a window. A client can save more than the visible portion in the scroll buffer. It is possible to configure some clients in such a way that you can use a scroll bar to view the contents of the scroll buffer.

scroll bar

A scroll bar is a widget with which you can position the scroll area over a certain position in the scroll buffer. In windows containing text, you will often find a vertical scroll bar. A scroll bar is made up of a narrow rectangle with scroll arrows at either end and a slider. You can use the scroll arrows to move the scroll area in a certain direction by clicking the appropriate scroll arrow. The slider indicates the position of the scroll area relative to the entire scroll buffer.

secondary window

Secondary windows are all the client windows which are not primary windows, i.e. are not children of the root window.

server

A server is a program that manages a display and all the connections between the display and the clients. Because a server manages only one display, the term display is often used as a synonym for server. The server handles all connections as separate connections, even if a client has established several connections to the server. The server provides the window mechanism, distributes graphics requests received from the clients to the correct window on the correct screen and delivers input to the client which manages the currently active window.

stacking order

The offspring of a parent are stacked from top to bottom in the window stack. The uppermost window is completely visible and obscures the other windows in part or entirely.

You can change the stacking order by placing a window on top of the stack or on the bottom explicitly.

T **transient window**

A transient window is a temporary window belonging to a client which, depending on the context in which it was opened, may be the only client window or even the only window on the screen that accepts input. Because it prevents input to a potentially large group of windows, such windows usually have only a very short lifespan. Menus are an example of transient windows.

V **visual class**

The visual class of a window or icon determines the ways in which colors can be displayed. Whether these can be implemented depends on the screen. There are six visual classes: StaticGray, StaticColor, TrueColor, GrayScale, PseudoColor, DirectColor.

The visual classes have a hierarchical structure: "StaticGray" is derived from "GrayScale"; "GrayScale" and "StaticColor" are derived from "PseudoColor"; "PseudoColor" and "TrueColor" are derived from "DirectColor".

In the class "GrayScale" the values for red, green and blue in every colormap are the same, resulting in a grayscale. The grayscales can be modified.

The visual class "GrayScale" can be regarded as a degenerated subclass of "PseudoColor". Unlike "GrayScale", the values for red, green and blue for the "PseudoColor" visual class can be different from one another.

The visual class "PseudoColor" can be regarded as a degenerate subclass of "DirectColor". Unlike "PseudoColor" where the value of an entire pixel is assigned a combination of values for red, green and blue, the values for red, green and blue in the case of "DirectColor" can not only be different from one another, but the different parts of a pixel can be assigned a value, each independent of the other. The first part of each pixel can be used for defining a value for red, the second part for green and the third part for blue.

"StaticGray" is the static variant of "GrayScale", "StaticColor" is the static variant of "PseudoColor", and "TrueColor" the static variant of "DirectColor". Colormap entries cannot be modified for the static variants of the visual classes.

W **widget**

A widget is a window with built-in functionality. A scroll bar is an example of a widget.

window

A window is a rectangular area on a screen. It possesses window attributes which determine the background, appearance, size and position of the window. You can assign many but not all of the window attributes a value via the resources, thus customizing the user interface.

A window is always the child of another window, unless it is the root window of a screen. This results in a window hierarchy which determines the visibility of a window and influences event handling.

A window is normally managed by two clients, the client that created the window and the window manager. The window manager manages the window attributes, the client manages the client area.

window decoration

The amount of decoration on a window determines the functionality of its mwm window frame, which is provided for your use by the window manager. You can define the functionality by means of mwm resources. Window decoration includes the resize borders, matte, maximize button, menu button, minimize button and the window's title area.

The maximize button, minimize button and the resize border allow you to change the size of a window. Clicking the menu button posts the window menu. The matte enhances the appearance of the client window and the title area contains the name of the window.

window manager

The window manager is a client which manages all the windows on a display. It allows you, among other things, to resize or move windows, change the stacking order, and to iconify or normalize windows. The window manager does not, however, have any influence on the client area of a window. The client area is managed by the client alone.

Depending on the layout guidelines upon which the window manager's behavior is based, the window manager may override or modify window layout values specified by the client.

X X Display Manager Control Protocol (XDMCP)

XDMCP is a protocol governing communications between X servers and display managers. Using this protocol an X server can send a management request to a display manager on a remote host. In addition, the display manager uses this protocol to inform the X server of the authorization key generated for the authorization mechanism.

Index