NAME

X – a portable, network-transparent window system

SYNOPSIS

The X Window System is a network transparent window system which runs on a wide range of computing and graphics machines. It should be relatively straightforward to build the X Window System software distribution on most ANSI C and POSIX compliant systems. Commercial implementations are also available for a wide range of platforms.

The Open Group requests that the following names be used when referring to this software:

X
X Window System
X Version 11
X Window System, Version 11
X11

X Window System is a trademark of The Open Group.

DESCRIPTION

X Window System servers run on computers with bitmap displays. The server distributes user input to and accepts output requests from various client programs through a variety of different interprocess communication channels. Although the most common case is for the client programs to be running on the same machine as the server, clients can be run transparently from other machines (including machines with different architectures and operating systems) as well.

X supports overlapping hierarchical subwindows and text and graphics operations, on both monochrome and color displays. For a full explanation of the functions that are available, see the *Xlib - C Language X Interface* manual, the *X Window System Protocol* specification, the *X Toolkit Intrinsics - C Language Interface* manual, and various toolkit documents.

The number of programs that use *X* is quite large. Programs provided in the core X Window System distribution include: a terminal emulator, *xterm*; a window manager, *twm*; a display manager, *xdm*; a console redirect program, *xconsole*; a mail interface, *xmh*; a bitmap editor, *bitmap*; resource listing/manipulation tools, *appres*, *editres*; access control programs, *xauth*, *xhost*, and *iceauth*; user preference setting programs, *xrdb*, *xcmsdb*, *xset*, *xsetroot*, *xstdcmap*, and *xmodmap*; clocks, *xclock* and *oclock*; a font displayer, (*xfd*; utilities for listing information about fonts, windows, and displays, *xlsfonts*, *xwininfo*, *xlsclients*, *xdpyinfo*, *xlsatoms*, and *xprop*; screen image manipulation utilities, *xwd*, *xwud*, and *xmag*; a performance measurement utility, *x11perf*; a font compiler, *bdftopcf*; a font server and related utilities, *xfs*, *fsinfo*, *fslsfonts*, *fstobdf*; an X Image Extension exerciser, *xieperf*; a display server and related utilities, *Xserver*, *rgb*, *mkfontdir*; remote execution utilities, *rstart* and *xon*; a clipboard manager, *xclipboard*; keyboard description compiler and related utilities, *xkbcomp*, *xkbprint*, *xkbbell*, *xkbevd*, *xkbvleds*, and *xkbwatch*; a utility to terminate clients, *xkill*; an optimized X protocol proxy, *lbxproxy*; a firewall security proxy, *xfwp*; a proxy manager to control them, *proxymngr*; a utility to find proxies, *xfindproxy*; Netscape Navigator Plug-ins, *libxrx.so* and *libxrxnest.so*; an RX MIME-type helper program, *xrx*; and a utility to cause part or all of the screen to be redrawn, *xrefresh*.

Many other utilities, window managers, games, toolkits, etc. are included as user-contributed software in the X Window System distribution, or are available using anonymous ftp on the Internet. See your site administrator for details.

STARTING UP

There are two main ways of getting the X server and an initial set of client applications started. The particular method used depends on what operating system you are running and whether or not you use other window systems in addition to X.

xdm **(the X Display Manager)**

If you want to always have X running on your display, your site administrator can set your machine up to use the X Display Manager *xdm*. This program is typically started by the system at boot time and takes care of keeping the server running and getting users logged in. If you are running *xdm*, you will see a window on the screen welcoming you to the system and asking for your username and password. Simply type them in as you would at a normal terminal, pressing the Return key after each. If you make a mistake, *xdm* will display an error message and ask you to try again. After you have successfully logged in, *xdm* will start up your X environment. By default, if you have an executable file named *.xsession* in your home directory, *xdm* will treat it as a program (or shell script) to run to start up your initial clients (such as terminal emulators, clocks, a window manager, user settings for things like the background, the speed of the pointer, etc.). Your site administrator can provide details.

xinit **(run manually from the shell)**

Sites that support more than one window system might choose to use the *xinit* program for starting X manually. If this is true for your machine, your site administrator will probably have provided a program named "x11", "startx", or "xstart" that will do site-specific initialization (such as loading convenient default resources, running a window manager, displaying a clock, and starting several terminal emulators) in a nice way. If not, you can build such a script using the *xinit* program. This utility simply runs one user-specified program to start the server, runs another to start up any desired clients, and then waits for either to finish. Since either or both of the user-specified programs may be a shell script, this gives substantial flexibility at the expense of a nice interface. For this reason, *xinit* is not intended for end users.

DISPLAY NAMES

From the user's perspective, every X server has a *display name* of the form:

hostname:displaynumber.screennumber

This information is used by the application to determine how it should connect to the server and which screen it should use by default (on displays with multiple monitors):

hostname

The *hostname* specifies the name of the machine to which the display is physically connected. If the hostname is not given, the most efficient way of communicating to a server on the same machine will be used.

displaynumber

The phrase "display" is usually used to refer to collection of monitors that share a common keyboard and pointer (mouse, tablet, etc.). Most workstations tend to only have one keyboard, and therefore, only one display. Larger, multi-user systems, however, frequently have several displays so that more than one person can be doing graphics work at once. To avoid confusion, each display on a machine is assigned a *display number* (beginning at 0) when the X server for that display is started. The display number must always be given in a display name.

screennumber

Some displays share a single keyboard and pointer among two or more monitors. Since each monitor has its own set of windows, each screen is assigned a *screen number* (beginning at 0) when the X server for that display is started. If the screen number is not given, screen 0 will be used.

On POSIX systems, the default display name is stored in your DISPLAY environment variable. This variable is set automatically by the *xterm* terminal emulator. However, when you log into another machine on a network, you will need to set DISPLAY by hand to point to your display. For example,

```
% setenv DISPLAY myws:0
$ DISPLAY=myws:0; export DISPLAY
```

The *xon* script can be used to start an X program on a remote machine; it automatically sets the DISPLAY variable correctly.

Finally, most X programs accept a command line option of **-display** *displayname* to temporarily override the contents of DISPLAY. This is most commonly used to pop windows on another person's screen or as part of a "remote shell" command to start an xterm pointing back to your display. For example,

```
% xeyes -display joesws:0 -geometry 1000x1000+0+0
% rsh big xterm -display myws:0 -ls </dev/null &
```

X servers listen for connections on a variety of different communications channels (network byte streams, shared memory, etc.). Since there can be more than one way of contacting a given server, The *hostname* part of the display name is used to determine the type of channel (also called a transport layer) to be used. X servers generally support the following types of connections:

local

The hostname part of the display name should be the empty string. For example: *:0*, *:1*, and *:0.1*. The most efficient local transport will be chosen.

TCPIP

The hostname part of the display name should be the server machine's IP address name. Full Internet names, abbreviated names, and IP addresses are all allowed. For example: *x.org:0*, *expo:0*, *198.112.45.11:0*, *bigmachine:1*, and *hydra:0.1*.

DECnet

The hostname part of the display name should be the server machine's nodename, followed by two colons instead of one. For example: *myws::0*, *big::1*, and *hydra::0.1*.

ACCESS CONTROL

An X server can use several types of access control. Mechanisms provided in Release 6 are:

Host Access	Simple host-based access control.
MIT-MAGIC-COOKIE-1	Shared plain-text "cookies".
XDM-AUTHORIZATION-1	Secure DES based private-keys.
SUN-DES-1	Based on Sun's secure rpc system.
MIT-KERBEROS-5	Kerberos Version 5 user-to-user.

Xdm initializes access control for the server and also places authorization information in a file accessible to the user. Normally, the list of hosts from which connections are always accepted should be empty, so that only clients with are explicitly authorized can connect to the display. When you add entries to the host list (with *xhost*), the server no longer performs any authorization on connections from those machines. Be careful with this.

The file from which *Xlib* extracts authorization data can be specified with the environment variable **XAUTHORITY**, and defaults to the file **.Xauthority** in the home directory. *Xdm* uses **$HOME/.Xauthority** and will create it or merge in authorization records if it already exists when a user logs in.

If you use several machines and share a common home directory across all of the machines by means of a network file system, you never really have to worry about authorization files, the system should work correctly by default. Otherwise, as the authorization files are machine-independent, you can simply copy the files to share them. To manage authorization files, use *xauth*. This program allows you to extract records and insert them into other files. Using this, you can send authorization to remote machines when you login, if the remote machine does not share a common home directory with your local machine. Note that authorization information transmitted ‘‘in the clear’’ through a network file system or using *ftp* or *rcp* can be ‘‘stolen’’ by a network eavesdropper, and as such may enable unauthorized access. In many environments, this level of security is not a concern, but if it is, you need to know the exact semantics of the particular authorization data to know if this is actually a problem.

For more information on access control, see the *Xsecurity* manual page.

GEOMETRY SPECIFICATIONS

One of the advantages of using window systems instead of hardwired terminals is that applications don’t have to be restricted to a particular size or location on the screen. Although the layout of windows on a display is controlled by the window manager that the user is running (described below), most X programs accept a command line argument of the form **-geometry** *WIDTHxHEIGHT+XOFF+YOFF* (where *WIDTH*, *HEIGHT*, *XOFF*, and *YOFF* are numbers) for specifying a preferred size and location for this application’s main window.

The *WIDTH* and *HEIGHT* parts of the geometry specification are usually measured in either pixels or characters, depending on the application. The *XOFF* and *YOFF* parts are measured in pixels and are used to specify the distance of the window from the left or right and top and bottom edges of the screen, respectively. Both types of offsets are measured from the indicated edge of the screen to the corresponding edge of the window. The X offset may be specified in the following ways:

+XOFF The left edge of the window is to be placed *XOFF* pixels in from the left edge of the screen (i.e., the X coordinate of the window’s origin will be *XOFF*). *XOFF* may be negative, in which case the window’s left edge will be off the screen.

-XOFF The right edge of the window is to be placed *XOFF* pixels in from the right edge of the screen. *XOFF* may be negative, in which case the window’s right edge will be off the screen.

The Y offset has similar meanings:

+YOFF The top edge of the window is to be *YOFF* pixels below the top edge of the screen (i.e., the Y coordinate of the window’s origin will be *YOFF*). *YOFF* may be negative, in which case the window’s top edge will be off the screen.

-YOFF The bottom edge of the window is to be *YOFF* pixels above the bottom edge of the screen. *YOFF* may be negative, in which case the window’s bottom edge will be off the screen.

Offsets must be given as pairs; in other words, in order to specify either *XOFF* or *YOFF* both must be present. Windows can be placed in the four corners of the screen using the following specifications:

+0+0 upper left hand corner.

-0+0 upper right hand corner.

-0-0 lower right hand corner.

+0-0 lower left hand corner.

In the following examples, a terminal emulator is placed in roughly the center of the screen and a load average monitor, mailbox, and clock are placed in the upper right hand corner:

```
xterm -fn 6x10 -geometry 80x24+30+200 &
xclock -geometry 48x48-0+0 &
xload -geometry 48x48-96+0 &
xbiff -geometry 48x48-48+0 &
```

WINDOW MANAGERS

The layout of windows on the screen is controlled by special programs called *window managers*. Although many window managers will honor geometry specifications as given, others may choose to ignore them (requiring the user to explicitly draw the window’s region on the screen with the pointer, for example).

Since window managers are regular (albeit complex) client programs, a variety of different user interfaces can be built. The X Window System distribution comes with a window manager named *twm* which supports overlapping windows, popup menus, point-and-click or click-to-type input models, title bars, nice icons (and an icon manager for those who don't like separate icon windows).

See the user-contributed software in the X Window System distribution for other popular window managers.

FONT NAMES

Collections of characters for displaying text and symbols in X are known as *fonts*. A font typically contains images that share a common appearance and look nice together (for example, a single size, boldness, slant, and character set). Similarly, collections of fonts that are based on a common type face (the variations are usually called roman, bold, italic, bold italic, oblique, and bold oblique) are called *families*.

Fonts come in various sizes. The X server supports *scalable* fonts, meaning it is possible to create a font of arbitrary size from a single source for the font. The server supports scaling from *outline* fonts and *bitmap* fonts. Scaling from outline fonts usually produces significantly better results than scaling from bitmap fonts.

An X server can obtain fonts from individual files stored in directories in the file system, or from one or more font servers, or from a mixtures of directories and font servers. The list of places the server looks when trying to find a font is controlled by its *font path*. Although most installations will choose to have the server start up with all of the commonly used font directories in the font path, the font path can be changed at any time with the *xset* program. However, it is important to remember that the directory names are on the **server**'s machine, not on the application's.

Bitmap font files are usually created by compiling a textual font description into binary form, using *bdftopcf*. Font databases are created by running the *mkfontdir* program in the directory containing the source or compiled versions of the fonts. Whenever fonts are added to a directory, *mkfontdir* should be rerun so that the server can find the new fonts. To make the server reread the font database, reset the font path with the *xset* program. For example, to add a font to a private directory, the following commands could be used:

```
% cp newfont.pcf ~/myfonts
% mkfontdir ~/myfonts
% xset fp rehash
```

The *xfontsel* and *xlsfonts* programs can be used to browse through the fonts available on a server. Font names tend to be fairly long as they contain all of the information needed to uniquely identify individual fonts. However, the X server supports wildcarding of font names, so the full specification

-adobe-courier-medium-r-normal--10-100-75-75-m-60-iso8859-1

might be abbreviated as:

--courier-medium-r-normal--*-100-*-*-*-*-iso8859-1*

Because the shell also has special meanings for * and *?*, wildcarded font names should be quoted:

```
% xlsfonts -fn '-*-courier-medium-r-normal--*-100-*-*-*-*-*-*'
```

The *xlsfonts* program can be used to list all of the fonts that match a given pattern. With no arguments, it lists all available fonts. This will usually list the same font at many different sizes. To see just the base scalable font names, try using one of the following patterns:

--*-*-*-*-*-0-0-0-0-*-0-*-*

--*-*-*-*-*-0-0-75-75-*-0-*-**
--*-*-*-*-*-0-0-100-100-*-0-*-**

To convert one of the resulting names into a font at a specific size, replace one of the first two zeros with a nonzero value. The field containing the first zero is for the pixel size; replace it with a specific height in pixels to name a font at that size. Alternatively, the field containing the second zero is for the point size; replace it with a specific size in decipoints (there are 722.7 decipoints to the inch) to name a font at that size. The last zero is an average width field, measured in tenths of pixels; some servers will anamorphically scale if this value is specified.

FONT SERVER NAMES

One of the following forms can be used to name a font server that accepts TCP connections:

tcp/*hostname*:*port*
tcp/*hostname*:*port*/*cataloguelist*

The *hostname* specifies the name (or decimal numeric address) of the machine on which the font server is running. The *port* is the decimal TCP port on which the font server is listening for connections. The *cataloguelist* specifies a list of catalogue names, with '+' as a separator.

Examples: *tcp/x.org:7100*, *tcp/198.112.45.11:7100/all*.

One of the following forms can be used to name a font server that accepts DECnet connections:

decnet/*nodename*::font$*objname*
decnet/*nodename*::font$*objname*/*cataloguelist*

The *nodename* specifies the name (or decimal numeric address) of the machine on which the font server is running. The *objname* is a normal, case-insensitive DECnet object name. The *cataloguelist* specifies a list of catalogue names, with '+' as a separator.

Examples: *DECnet/SRVNOD::FONT$DEFAULT*, *decnet/44.70::font$special/symbols*.

COLOR NAMES

Most applications provide ways of tailoring (usually through resources or command line arguments) the colors of various elements in the text and graphics they display. A color can be specified either by an abstract color name, or by a numerical color specification. The numerical specification can identify a color in either device-dependent (RGB) or device-independent terms. Color strings are case-insensitive.

X supports the use of abstract color names, for example, "red", "blue". A value for this abstract name is obtained by searching one or more color name databases. *Xlib* first searches zero or more client-side databases; the number, location, and content of these databases is implementation dependent. If the name is not found, the color is looked up in the X server's database. The text form of this database is commonly stored in the file *<XRoot>/lib/X11/rgb.txt*, where <XRoot> is replaced by the root of the X11 install tree.

A numerical color specification consists of a color space name and a set of values in the following syntax:

<color_space_name>:*<value>/.../<value>*

An RGB Device specification is identified by the prefix "rgb:" and has the following syntax:

rgb:*<red>/<green>/<blue>*

<red>, *<green>*, *<blue>* := *h* | *hh* | *hhh* | *hhhh*
h := single hexadecimal digits

Note that *h* indicates the value scaled in 4 bits, *hh* the value scaled in 8 bits, *hhh* the value scaled in 12 bits, and *hhhh* the value scaled in 16 bits, respectively. These values are passed directly to the X server, and are assumed to be gamma corrected.

The eight primary colors can be represented as:

black	rgb:0/0/0
red	rgb:ffff/0/0
green	rgb:0/ffff/0
blue	rgb:0/0/ffff
yellow	rgb:ffff/ffff/0
magenta	rgb:ffff/0/ffff
cyan	rgb:0/ffff/ffff
white	rgb:ffff/ffff/ffff

For backward compatibility, an older syntax for RGB Device is supported, but its continued use is not encouraged. The syntax is an initial sharp sign character followed by a numeric specification, in one of the following formats:

#RGB	(4 bits each)
#RRGGBB	(8 bits each)
#RRRGGGBBB	(12 bits each)
#RRRRGGGGBBBB	(16 bits each)

The R, G, and B represent single hexadecimal digits. When fewer than 16 bits each are specified, they represent the most-significant bits of the value (unlike the "rgb:" syntax, in which values are scaled). For example, #3a7 is the same as #3000a0007000.

An RGB intensity specification is identified by the prefix "rgbi:" and has the following syntax:

rgbi:*<red>*/*<green>*/*<blue>*

The red, green, and blue are floating point values between 0.0 and 1.0, inclusive. They represent linear intensity values, with 1.0 indicating full intensity, 0.5 half intensity, and so on. These values will be gamma corrected by *Xlib* before being sent to the X server. The input format for these values is an optional sign, a string of numbers possibly containing a decimal point, and an optional exponent field containing an E or e followed by a possibly signed integer string.

The standard device-independent string specifications have the following syntax:

CIEXYZ:*<X>*/*<Y>*/*<Z>*	(*none*, 1, *none*)
CIEuvY:*<u>*/*<v>*/*<Y>*	(~.6, ~.6, 1)
CIExyY:*<x>*/*<y>*/*<Y>*	(~.75, ~.85, 1)
CIELab:*<L>*/*<a>*/*<b>*	(100, *none*, *none*)
CIELuv:*<L>*/*<u>*/*<v>*	(100, *none*, *none*)
TekHVC:*<H>*/*<V>*/*<C>*	(360, 100, 100)

All of the values (C, H, V, X, Y, Z, a, b, u, v, y, x) are floating point values. Some of the values are constrained to be between zero and some upper bound; the upper bounds are given in parentheses above. The syntax for these values is an optional '+' or '-' sign, a string of digits possibly containing a decimal point, and an optional exponent field consisting of an 'E' or 'e' followed by an optional '+' or '-' followed by a string of digits.

For more information on device independent color, see the *Xlib* reference manual.

KEYBOARDS

The X keyboard model is broken into two layers: server-specific codes (called *keycodes*) which represent the physical keys, and server-independent symbols (called *keysyms*) which represent the letters or words that appear on the keys. Two tables are kept in the server for converting keycodes to keysyms:

modifier list

Some keys (such as Shift, Control, and Caps Lock) are known as *modifier* and are used to select different symbols that are attached to a single key (such as Shift-a generates a capital A, and Control-l generates a control character ^L). The server keeps a list of keycodes corresponding to the various modifier keys. Whenever a key is pressed or released, the server generates an *event* that contains the keycode of the indicated key as well as a mask that specifies which of the modifier keys are currently pressed. Most servers set up this list to initially contain the various shift, control, and shift lock keys on the keyboard.

keymap table

Applications translate event keycodes and modifier masks into keysyms using a *keysym table* which contains one row for each keycode and one column for various modifier states. This table is initialized by the server to correspond to normal typewriter conventions. The exact semantics of how the table is interpreted to produce keysyms depends on the particular program, libraries, and language input method used, but the following conventions for the first four keysyms in each row are generally adhered to:

The first four elements of the list are split into two groups of keysyms. Group 1 contains the first and second keysyms; Group 2 contains the third and fourth keysyms. Within each group, if the first element is alphabetic and the the second element is the special keysym *NoSymbol*, then the group is treated as equivalent to a group in which the first element is the lowercase letter and the second element is the uppercase letter.

Switching between groups is controlled by the keysym named MODE SWITCH, by attaching that keysym to some key and attaching that key to any one of the modifiers Mod1 through Mod5. This modifier is called the ''group modifier.'' Group 1 is used when the group modifier is off, and Group 2 is used when the group modifier is on.

Within a group, the modifier state determines which keysym to use. The first keysym is used when the Shift and Lock modifiers are off. The second keysym is used when the Shift modifier is on, when the Lock modifier is on and the second keysym is uppercase alphabetic, or when the Lock modifier is on and is interpreted as ShiftLock. Otherwise, when the Lock modifier is on and is interpreted as CapsLock, the state of the Shift modifier is applied first to select a keysym; but if that keysym is lowercase alphabetic, then the corresponding uppercase keysym is used instead.

OPTIONS

Most X programs attempt to use the same names for command line options and arguments. All applications written with the X Toolkit Intrinsics automatically accept the following options:

−display *display*

This option specifies the name of the X server to use.

−geometry *geometry*

This option specifies the initial size and location of the window.

−bg *color*, **−background** *color*

Either option specifies the color to use for the window background.

−bd *color*, **−bordercolor** *color*

Either option specifies the color to use for the window border.

−bw *number*, **−borderwidth** *number*

Either option specifies the width in pixels of the window border.

−fg *color*, **−foreground** *color*

Either option specifies the color to use for text or graphics.

–fn *font*, **-font** *font*
Either option specifies the font to use for displaying text.

–iconic
This option indicates that the user would prefer that the application's windows initially not be visible as if the windows had be immediately iconified by the user. Window managers may choose not to honor the application's request.

–name
This option specifies the name under which resources for the application should be found. This option is useful in shell aliases to distinguish between invocations of an application, without resorting to creating links to alter the executable file name.

–rv, **–reverse**
Either option indicates that the program should simulate reverse video if possible, often by swapping the foreground and background colors. Not all programs honor this or implement it correctly. It is usually only used on monochrome displays.

+rv
This option indicates that the program should not simulate reverse video. This is used to override any defaults since reverse video doesn't always work properly.

–selectionTimeout
This option specifies the timeout in milliseconds within which two communicating applications must respond to one another for a selection request.

–synchronous
This option indicates that requests to the X server should be sent synchronously, instead of asynchronously. Since *Xlib* normally buffers requests to the server, errors do not necessarily get reported immediately after they occur. This option turns off the buffering so that the application can be debugged. It should never be used with a working program.

–title *string*
This option specifies the title to be used for this window. This information is sometimes used by a window manager to provide some sort of header identifying the window.

–xnllanguage *language[_territory][.codeset]*
This option specifies the language, territory, and codeset for use in resolving resource and other filenames.

–xrm *resourcestring*
This option specifies a resource name and value to override any defaults. It is also very useful for setting resources that don't have explicit command line arguments.

RESOURCES

To make the tailoring of applications to personal preferences easier, X provides a mechanism for storing default values for program resources (e.g. background color, window title, etc.) Resources are specified as strings that are read in from various places when an application is run. Program components are named in a hierarchical fashion, with each node in the hierarchy identified by a class and an instance name. At the top level is the class and instance name of the application itself. By convention, the class name of the application is the same as the program name, but with the first letter capitalized (e.g. *Bitmap* or *Emacs*) although some programs that begin with the letter ''x'' also capitalize the second letter for historical reasons.

The precise syntax for resources is:

```
ResourceLine     = Comment | IncludeFile | ResourceSpec | <empty line>
Comment          = "!" {<any character except null or newline>}
IncludeFile      = "#" WhiteSpace "include" WhiteSpace FileName WhiteSpace
```

FileName	=	<valid filename for operating system>
ResourceSpec	=	WhiteSpace ResourceName WhiteSpace ":" WhiteSpace Value
ResourceName	=	[Binding] {Component Binding} ComponentName
Binding	=	"." | "*"
WhiteSpace	=	{<space> | <horizontal tab>}
Component	=	"?" | ComponentName
ComponentName	=	NameChar {NameChar}
NameChar	=	"a"−"z" | "A"−"Z" | "0"−"9" | "_" | "-"
Value	=	{<any character except null or unescaped newline>}

Elements separated by vertical bar (|) are alternatives. Curly braces ({...}) indicate zero or more repetitions of the enclosed elements. Square brackets ([...]) indicate that the enclosed element is optional. Quotes ("...") are used around literal characters.

IncludeFile lines are interpreted by replacing the line with the contents of the specified file. The word "include" must be in lowercase. The filename is interpreted relative to the directory of the file in which the line occurs (for example, if the filename contains no directory or contains a relative directory specification).

If a ResourceName contains a contiguous sequence of two or more Binding characters, the sequence will be replaced with single "." character if the sequence contains only "." characters, otherwise the sequence will be replaced with a single "*" character.

A resource database never contains more than one entry for a given ResourceName. If a resource file contains multiple lines with the same ResourceName, the last line in the file is used.

Any whitespace character before or after the name or colon in a ResourceSpec are ignored. To allow a Value to begin with whitespace, the two-character sequence “*space*” (backslash followed by space) is recognized and replaced by a space character, and the two-character sequence “*tab*” (backslash followed by horizontal tab) is recognized and replaced by a horizontal tab character. To allow a Value to contain embedded newline characters, the two-character sequence “\n” is recognized and replaced by a newline character. To allow a Value to be broken across multiple lines in a text file, the two-character sequence “*newline*” (backslash followed by newline) is recognized and removed from the value. To allow a Value to contain arbitrary character codes, the four-character sequence “*nnn*”, where each *n* is a digit character in the range of “0”−“7”, is recognized and replaced with a single byte that contains the octal value specified by the sequence. Finally, the two-character sequence “\\” is recognized and replaced with a single backslash.

When an application looks for the value of a resource, it specifies a complete path in the hierarchy, with both class and instance names. However, resource values are usually given with only partially specified names and classes, using pattern matching constructs. An asterisk (*) is a loose binding and is used to represent any number of intervening components, including none. A period (.) is a tight binding and is used to separate immediately adjacent components. A question mark (?) is used to match any single component name or class. A database entry cannot end in a loose binding; the final component (which cannot be "?") must be specified. The lookup algorithm searches the resource database for the entry that most closely matches (is most specific for) the full name and class being queried. When more than one database entry matches the full name and class, precedence rules are used to select just one. The full name and class are scanned from left to right (from highest level in the hierarchy to lowest), one component at a time. At each level, the corresponding component and/or binding of each matching entry is determined, and these matching components and bindings are compared according to precedence rules. Each of the rules is applied at each level, before moving to the next level, until a rule selects a single entry over all others. The rules (in order of precedence) are:

1. An entry that contains a matching component (whether name, class, or "?") takes precedence over entries that elide the level (that is, entries that match the level in a loose binding).

2. An entry with a matching name takes precedence over both entries with a matching class and entries that match using "?". An entry with a matching class takes precedence over entries that match using "?".

3. An entry preceded by a tight binding takes precedence over entries preceded by a loose binding.

Programs based on the X Tookit Intrinsics obtain resources from the following sources (other programs usually support some subset of these sources):

RESOURCE_MANAGER root window property
Any global resources that should be available to clients on all machines should be stored in the RESOURCE_MANAGER property on the root window of the first screen using the *xrdb* program. This is frequently taken care of when the user starts up X through the display manager or *xinit*.

SCREEN_RESOURCES root window property
Any resources specific to a given screen (e.g. colors) that should be available to clients on all machines should be stored in the SCREEN_RESOURCES property on the root window of that screen. The *xrdb* program will sort resources automatically and place them in RESOURCE_MANAGER or SCREEN_RESOURCES, as appropriate.

application-specific files
Directories named by the environment variable XUSERFILESEARCHPATH or the environment variable XAPPLRESDIR (which names a single directory and should end with a '/' on POSIX systems), plus directories in a standard place (usually under <XRoot>/lib/X11/, but this can be overridden with the XFILESEARCHPATH environment variable) are searched for for application-specific resources. For example, application default resources are usually kept in <XRoot>/lib/X11/app-defaults/. See the *X Toolkit Intrinsics - C Language Interface* manual for details.

XENVIRONMENT
Any user- and machine-specific resources may be specified by setting the XENVIRONMENT environment variable to the name of a resource file to be loaded by all applications. If this variable is not defined, a file named *$HOME*/.Xdefaults-*hostname* is looked for instead, where *hostname* is the name of the host where the application is executing.

−xrm *resourcestring*
Resources can also be specified from the command line. The *resourcestring* is a single resource name and value as shown above. Note that if the string contains characters interpreted by the shell (e.g., asterisk), they must be quoted. Any number of **−xrm** arguments may be given on the command line.

Program resources are organized into groups called *classes*, so that collections of individual resources (each of which are called *instances*) can be set all at once. By convention, the instance name of a resource begins with a lowercase letter and class name with an upper case letter. Multiple word resources are concatenated with the first letter of the succeeding words capitalized. Applications written with the X Toolkit Intrinsics will have at least the following resources:

background (class **Background)**
This resource specifies the color to use for the window background.

borderWidth (class **BorderWidth)**
This resource specifies the width in pixels of the window border.

borderColor (class **BorderColor)**
This resource specifies the color to use for the window border.

Most applications using the X Toolkit Intrinsics also have the resource **foreground** (class **Foreground**), specifying the color to use for text and graphics within the window.

By combining class and instance specifications, application preferences can be set quickly and easily. Users of color displays will frequently want to set Background and Foreground classes to particular defaults. Specific color instances such as text cursors can then be overridden without having to define all of the related resources. For example,

```
bitmap*Dashed:  off
XTerm*cursorColor:  gold
XTerm*multiScroll:  on
XTerm*jumpScroll:  on
XTerm*reverseWrap:  on
XTerm*curses:  on
XTerm*Font:  6x10
XTerm*scrollBar: on
XTerm*scrollbar*thickness: 5
XTerm*multiClickTime: 500
XTerm*charClass:  33:48,37:48,45-47:48,64:48
XTerm*cutNewline: off
XTerm*cutToBeginningOfLine: off
XTerm*titeInhibit:  on
XTerm*ttyModes:  intr ^c erase ^? kill ^u
XLoad*Background: gold
XLoad*Foreground: red
XLoad*highlight: black
XLoad*borderWidth: 0
emacs*Geometry:  80x65-0-0
emacs*Background:  rgb:5b/76/86
emacs*Foreground:  white
emacs*Cursor:  white
emacs*BorderColor:  white
emacs*Font:  6x10
xmag*geometry: -0-0
xmag*borderColor:  white
```

If these resources were stored in a file called *.Xresources* in your home directory, they could be added to any existing resources in the server with the following command:

```
% xrdb -merge $HOME/.Xresources
```

This is frequently how user-friendly startup scripts merge user-specific defaults into any site-wide defaults. All sites are encouraged to set up convenient ways of automatically loading resources. See the *Xlib* manual section *Resource Manager Functions* for more information.

EXAMPLES

The following is a collection of sample command lines for some of the more frequently used commands. For more information on a particular command, please refer to that command's manual page.

```
%  xrdb $HOME/.Xresources
%  xmodmap -e "keysym BackSpace = Delete"
%  mkfontdir /usr/local/lib/X11/otherfonts
%  xset fp+ /usr/local/lib/X11/otherfonts
```

```
% xmodmap $HOME/.keymap.km
% xsetroot -solid 'rgbi:.8/.8/.8'
% xset b 100 400 c 50 s 1800 r on
% xset q
% twm
% xmag
% xclock -geometry 48x48-0+0 -bg blue -fg white
% xeyes -geometry 48x48-48+0
% xbiff -update 20
% xlsfonts '*helvetica*'
% xwininfo -root
% xdpyinfo -display joesworkstation:0
% xhost -joesworkstation
% xrefresh
% xwd | xwud
% bitmap companylogo.bm 32x32
% xcalc -bg blue -fg magenta
% xterm -geometry 80x66-0-0 -name myxterm $*
% xon filesysmachine xload
```

DIAGNOSTICS

A wide variety of error messages are generated from various programs. The default error handler in *Xlib* (also used by many toolkits) uses standard resources to construct diagnostic messages when errors occur. The defaults for these messages are usually stored in *<XRoot>/lib/X11/XErrorDB*. If this file is not present, error messages will be rather terse and cryptic.

When the X Toolkit Intrinsics encounter errors converting resource strings to the appropriate internal format, no error messages are usually printed. This is convenient when it is desirable to have one set of resources across a variety of displays (e.g. color vs. monochrome, lots of fonts vs. very few, etc.), although it can pose problems for trying to determine why an application might be failing. This behavior can be overridden by the setting the *StringConversionsWarning* resource.

To force the X Toolkit Intrinsics to always print string conversion error messages, the following resource should be placed in the file that gets loaded onto the RESOURCE_MANAGER property using the *xrdb* program (frequently called *.Xresources* or *.Xres* in the user's home directory):

```
*StringConversionWarnings: on
```

To have conversion messages printed for just a particular application, the appropriate instance name can be placed before the asterisk:

```
xterm*StringConversionWarnings: on
```

SEE ALSO

XProjectTeam(1), XStandards(1), Xsecurity(1),

appres(1), bdftopcf(1), bitmap(1), editres(1), fsinfo(1), fslsfonts(1), fstobdf(1), iceauth(1), imake(1), lbxproxy(1), makedepend(1), mkfontdir(1), oclock(1), proxymngr(1), rgb(1), resize(1), rstart(1), smproxy(1), twm(1), x11perf(1), x11perfcomp(1), xauth(1), xclipboard(1), xclock(1), xcmsdb(1), xconsole(1), xdm(1), xdpyinfo(1), xfd(1), xfindproxy(1), xfs(1), xfwp(1), xhost(1), xieperf(1), xinit(1), xkbbell(1), xkbcomp(1), xbkevd(1), xkbprint(1), xkbvleds(1), xkbwatch(1), xkill(1), xlogo(1), xlsatoms(1), xlsclients(1), xlsfonts(1), xmag(1), xmh(1), xmodmap(1), xon(1), xprop(1), xrdb(1), xrefresh(1), xrx(1), xset(1), xsetroot(1), xsm(1), xstdcmap(1), xterm(1), xwd(1), xwininfo(1), xwud(1). Xserver(1), Xdec(1), XmacII(1), Xsun(1), Xnest(1), Xvfb(1), XF86_Acc(1), XF86_Mono(1), XF86_SVGA(1), XF86_VGA16(1), XFree86(1), kbd_mode(1), *Xlib − C Language X Interface*, and *X Toolkit Intrinsics − C Language Interface*

TRADEMARKS

X Window System is a trademark of The Open Group.

AUTHORS

A cast of thousands, literally. The Release 6.4 distribution is brought to you by The Open Group X Project Team. The names of all people who made it a reality will be found in the individual documents and source files. The The X Project Team staff members responsible for this release are: Arthur Barstow, Kaleb Keithley, Sekhar Makkapati, M. S. Ramesh, Jingping Ge, Ken Flowers, and Dave Knorr.

The X Window System standard was originally developed at the Laboratory for Computer Science at the Massachusetts Institute of Technology, and all rights thereto were assigned to the X Consortium on January 1, 1994. X Consortium, Inc. closed its doors on December 31, 1996. All rights to the X Window System have been assigned to The Open Group.

NAME

XStandards − X Consortium Standards and X Project Team Specifications

SYNOPSIS

The major goal of the X Consortium was to promote cooperation within the computer industry in the creation of standard software interfaces at all layers in the X Window System environment. The X Consortium produced standards - documents which defined network protocols, programming interfaces, and other aspects of the X environment. These standards continue to exist in The Open Group's X Project Team releases. The X Project Team produces specifications. Like X Consortium Standards, these are documents which define network protocols, programming interfaces, and other aspects of the X environment. Under the aegis of The Open Group, X Consortium standards, X Project Team specifications, and other specifications are the basis for portions of The Open Group's various CAE specifications.

The status of various standards, specifications, and the software in the X11R6.4 distribution, is explained below.

STANDARDS

The following documents are X Consortium standards:

X Window System Protocol
X Version 11, Release 6.4
Robert W. Scheifler

Xlib − C Language X Interface
X Version 11, Release 6.4
James Gettys, Robert W. Scheifler, Ron Newman

X Toolkit Intrinsics − C Language Interface
X Version 11, Release 6.4
Joel McCormack, Paul Asente, Ralph R. Swick, Donna Converse

Bitmap Distribution Format
Version 2.1
X Version 11, Release 6.4

Inter-Client Communication Conventions Manual
Version 2.0
X Version 11, Release 6.4
David Rosenthal, Stuart W. Marks

Compound Text Encoding
Version 1.1
X Version 11, Release 6.4
Robert W. Scheifler

X Logical Font Description Conventions
Version 1.5
X Version 11, Release 6.4
Jim Flowers, Stephen Gildea

X Display Manager Control Protocol
Version 1.0
X Version 11, Release 6.4
Keith Packard

X11 Nonrectangular Window Shape Extension
Version 1.0
X Version 11, Release 6.4
Keith Packard

X11 Input Extension Protocol Specification
Version 1.0
X Version 11, Release 6.4
George Sachs, Mark Patrick

X11 Input Extension Library Specification
X Version 11, Release 6.4
Mark Patrick, George Sachs

The X Font Service Protocol
Version 2.0
X Version 11, Release 6.4
Jim Fulton

PEX Protocol Specification
Version 5.1
Cheryl Huntington (architect), Paula Womack (editor)

PEXlib Specification and C Language Binding
Version 5.1
Jeff Stevenson

Inter-Client Exchange (ICE) Protocol
Version 1.0
X Version 11, Release 6.4
Robert Scheifler, Jordan Brown

Inter-Client Exchange (ICE) Library
Version 1.0
X Version 11, Release 6.4
Ralph Mor

X Session Management Protocol
Version 1.0
X Version 11, Release 6.4
Mike Wexler

X Session Management Library
Version 1.0
X Version 11, Release 6.4
Ralph Mor

The Input Method Protocol
Version 1.0
X Version 11, Release 6.4
Masahiko Narita, Hideki Hiura

X Synchronization Extension
Version 3.0
X Version 11, Release 6.4
Tim Glauert, Dave Carver, Jim Gettys, David P. Wiggins

X Image Extension, Protocol Reference Manual
Version 5.0
X Version 11, Release 6.4
Bob Shelley

XTEST Extension
Version 2.2
Kieron Drake

Big Requests Extension
Version 2.0
X Version 11, Release 6.4
Bob Scheifler

XC-MISC Extension
Version 1.1
X Version 11, Release 6.4
Bob Scheifler, Dave Wiggins

Double Buffer Extension
Version 1.0
Ian Elliott, David P. Wiggins

Record Extension Protocol
Version 1.13
Martha Zimet, Stephen Gildea

Record Extension Library
Version 1.13
Martha Zimet, Stephen Gildea

X Keyboard Extension Protocol
X Version 11, Release 6.4
Erik Fortune

X Keyboard Extension Library
X Version 11, Release 6.4
Amber J. Benson, Gary Aitken, Erik Fortune, Donna Converse,
George Sachs, and Will Walker

X Print Extension Protocol

X Version 11, Release 6.4

X Print Extension Library
X Version 11, Release 6.4

X Application Group Extension Protocol and Library
Version 1.0
X Version 11, Release 6.4
Kaleb Keithley

X Security Extension Protocol and Library
Version 4.0
X Version 11, Release 6.4
Dave Wiggins

X Proxy Manager Protocol
X Version 11, Release 6.4
Ralph Swick

LBX Extension Protocol and Library
X Version 11, Release 6.4
Keith Packard, Dave Lemke, Donna Converse, Ralph Mor, Ray Tice

Remote Execution MIME Type
Version 1.0
X Version 11, Release 6.4
Arnaud Le Hors

SPECIFICATIONS

The following documents are X Project Team specifications:

Colormap Utilization Policy and Extension
Version 1.0
Kaleb Keithley

Extended Visual Information Extension
Version 1.0
Peter Daifuku

X Display Power Management (DPMS) Extension Protocol and Library
Version 1.0
Rob Lembree

DRAFT STANDARDS

The following documents are currently draft standards of the X Consortium.

X Image Extension Library
Public Review Draft
Gary Rogers

PEX Protocol Specification

Version 5.2
Jeff Stevenson (architect), Jane Sczechowski (editor)

PEXlib Specification and C Language Binding
Version 5.2
Karl Schultz

INCLUDE FILES

The following include files are part of the Xlib standard.

<X11/cursorfont.h>
<X11/keysym.h>
<X11/keysymdef.h>
<X11/X.h>
<X11/Xatom.h>
<X11/Xcms.h>
<X11/Xlib.h>
<X11/Xlibint.h>
<X11/Xproto.h>
<X11/Xprotostr.h>
<X11/Xresource.h>
<X11/Xutil.h>
<X11/X10.h>

The following include files are part of the X Toolkit Intrinsics standard.

<X11/Composite.h>
<X11/CompositeP.h>
<X11/Constraint.h>
<X11/ConstrainP.h>
<X11/Core.h>
<X11/CoreP.h>
<X11/Intrinsic.h>
<X11/IntrinsicP.h>
<X11/Object.h>
<X11/ObjectP.h>
<X11/RectObj.h>
<X11/RectObjP.h>
<X11/Shell.h>
<X11/ShellP.h>
<X11/StringDefs.h>
<X11/Vendor.h>
<X11/VendorP.h>

The following include file is part of the Nonrectangular Window Shape Extension standard.

<X11/extensions/shape.h>

The following include files are part of the X Input Extension standard.

<X11/extensions/XI.h>
<X11/extensions/XInput.h>
<X11/extensions/XIproto.h>

The following include files are part of the PEXlib standard.

<X11/PEX5/PEX.h>
<X11/PEX5/PEXlib.h>
<X11/PEX5/PEXlibint.h>
<X11/PEX5/PEXproto.h>
<X11/PEX5/PEXprotost.h>

The following include files are part of the ICElib standard.

<X11/ICE/ICE.h>
<X11/ICE/ICEconn.h>
<X11/ICE/ICElib.h>
<X11/ICE/ICEmsg.h>
<X11/ICE/ICEproto.h>
<X11/ICE/ICEutil.h>

The following include files are part of the SMlib standard.

<X11/SM/SM.h>
<X11/SM/SMlib.h>
<X11/SM/SMproto.h>

The following include file is part of the Synchronization standard.

<X11/extensions/sync.h>

The following include files are part of the XIElib draft standard.

<X11/extensions/XIE.h>
<X11/extensions/XIElib.h>
<X11/extensions/XIEproto.h>
<X11/extensions/XIEprotost.h>

The following include file is part of the XTEST standard.

<X11/extensions/XTest.h>

The following include file is part of the Double Buffer Extension standard.

<X11/extensions/Xdbe.h>

The following include file is part of the Record Library standard.

<X11/extensions/record.h>

The following include files are part of the X Keyboard Extension Library standard.

<X11/XKBlib.h>
<X11/extensions/XKB.h>
<X11/extensions/XKBproto.h>
<X11/extensions/XKBstr.h>
<X11/extensions/XKBgeom.h>

The following include files are part of the X Print Extension Library standard.

<X11/extensions/Print.h>
<X11/extensions/Printstr.h>

The following include files are part of the X Application Group Extension Library standard.

<X11/extensions/Xag.h>
<X11/extensions/Xagstr.h>

The following include files are part of the X Security Extension Library standard.

<X11/extensions/security.h>
<X11/extensions/securstr.h>

The following include files are part of the LBX Extension library standard.

<X11/extensions/XLbx.h>
<X11/extensions/lbxbuf.h>
<X11/extensions/lbxbufstr.h>
<X11/extensions/lbxdeltastr.h>
<X11/extensions/lbximage.h>
<X11/extensions/lbxopts.h>
<X11/extensions/lbxstr.h>
<X11/extensions/lbxzlib.h>

The following include files are part of the Colormap Utilization Policy and Extention specification.

<X11/extensions/Xcup.h>
<X11/extensions/Xcupstr.h>

The following include files are part of the Extended Visual Information specification.

<X11/extensions/XEVI.h>
<X11/extensions/XEVIstr.h>

The following include files are part of the X Display Management Signaling Extension specification.

<X11/extensions/dpms.h>
<X11/extensions/dpmsstr.h>

NON STANDARDS

The X11R6.4 distribution contains *sample* implementations, not *reference* implementations. Although much of the code is believed to be correct, the code should be assumed to be in error wherever it conflicts with the specification.

The only X Consortium standards are the ones listed above. No other documents, include files, or software in X11R6.4 carry special status within the X Consortium. For example, none of the following are standards: internal interfaces of the sample server; the MIT-SHM extension; the Athena Widget Set; the Xmu library; the Xau library; the RGB database; the X Locale database; the fonts distributed with X11R6.4; the applications distributed with X11R6.4; the include files <X11/XWDFile.h>, <X11/Xfuncproto.h>, <X11/Xfuncs.h>, <X11/Xosdefs.h>, <X11/Xos.h>, <X11/Xos_r.h>, <X11/Xwinsock.h>, and <X11/Xthreads.h>; the bitmap files in <X11/bitmaps>.

The Multi-Buffering extension was a draft standard of the X Consortium but has been superseded by DBE as a standard.

X REGISTRY

The X Consortium maintained a registry of certain X-related items, to aid in avoiding conflicts and to aid in sharing of such items.

The registry is published as part of the X Consortium software release. The latest version may also be available by sending a message to xstuff@x.org. The message can have a subject line and no body, or a single-line body and no subject, in either case the line looking like:

    send docs registry

The X Registry and the names in it are not X Consortium standards.

NAME

XProjectTeam − X Project Team information

SYNOPSIS

Release 6.4 of X Version 11 was brought to you by The Open Group X Project Team.

DESCRIPTION

The Open Group X Project Team was created as the successor to the X Consortium, Inc., after the X Consortium ceased operations. The purpose of The X Project Team is to foster development, evolution, and maintenance of the X Window System.

The X Consortium was an independent, not-for-profit Delaware membership corporation. It was formed in 1993 as the successor to the MIT X Consortium.

The X Window System was created in the mid-1980s at the Massachusetts Institute of Technology. In 1988, MIT formed a member-funded consortium to provide the technical and administrative leadership necessary to support further development of the X Window System. In 1992, MIT and the membership decided it was in their best interests to move the consortium out of MIT and create an independent, standalone organization. All rights to the X Window System were assigned by MIT to X Consortium, Inc. on January 1, 1994. X Consortium, Inc. closed its doors on December 31, 1996. All rights to the X Window System have been assigned to the Open Group.

X PROJECT TEAM STAFF

Arthur Barstow
Kaleb Keithley
Sekhar Makkapati
M. S. Ramesh
Ken Flowers
Dave Knorr

ADDRESS

To reach The Open Group public Wide World Web server, use http://www.opengroup.org/ and http://www.opengroup.org/desktop/x/.

To reach The X Project Team public ftp machine, use anonymous ftp at ftp://ftp.x.org/

FULL MEMBERS

Astec
Attachmate
BARCO Chromatics
CliniComp International
CRM
Digital
Fujitsu
Hewlett-Packard
Hitachi
Hummingbird Communications
IBM
Insignia
Jupiter Systems
Metro Link
NCD
NetManage

Peritek
Seaweed Systems
Sequent Computer Systems
Shiman Associates
Silicon Graphics
Siemens Nixdorf
Societe Axel
SunSoft
Vigra - Visicom Laboratories
WRQ
Xi Graphics

NAME

Xsecurity – X display access control

SYNOPSIS

X provides mechanism for implementing many access control systems. The sample implementation includes five mechanisms:

Host Access	Simple host-based access control.
MIT-MAGIC-COOKIE-1	Shared plain-text "cookies".
XDM-AUTHORIZATION-1	Secure DES based private-keys.
SUN-DES-1	Based on Sun's secure rpc system.
MIT-KERBEROS-5	Kerberos Version 5 user-to-user.

ACCESS SYSTEM DESCRIPTIONS

Host Access

Any client on a host in the host access control list is allowed access to the X server. This system can work reasonably well in an environment where everyone trusts everyone, or when only a single person can log in to a given machine, and is easy to use when the list of hosts used is small. This system does not work well when multiple people can log in to a single machine and mutual trust does not exist. The list of allowed hosts is stored in the X server and can be changed with the *xhost* command. When using the more secure mechanisms listed below, the host list is normally configured to be the empty list, so that only authorized programs can connect to the display.

MIT-MAGIC-COOKIE-1

When using MIT-MAGIC-COOKIE-1, the client sends a 128 bit "cookie" along with the connection setup information. If the cookie presented by the client matches one that the X server has, the connection is allowed access. The cookie is chosen so that it is hard to guess; *xdm* generates such cookies automatically when this form of access control is used. The user's copy of the cookie is usually stored in the *.Xauthority* file in the home directory, although the environment variable **XAUTHORITY** can be used to specify an alternate location. *Xdm* automatically passes a cookie to the server for each new login session, and stores the cookie in the user file at login.

The cookie is transmitted on the network without encryption, so there is nothing to prevent a network snooper from obtaining the data and using it to gain access to the X server. This system is useful in an environment where many users are running applications on the same machine and want to avoid interference from each other, with the caveat that this control is only as good as the access control to the physical network. In environments where network-level snooping is difficult, this system can work reasonably well.

XDM-AUTHORIZATION-1

Sites in the United States can use a DES-based access control mechanism called XDM-AUTHORIZATION-1. It is similar in usage to MIT-MAGIC-COOKIE-1 in that a key is stored in the *.Xauthority* file and is shared with the X server. However, this key consists of two parts - a 56 bit DES encryption key and 64 bits of random data used as the authenticator.

When connecting to the X server, the application generates 192 bits of data by combining the current time in seconds (since 00:00 1/1/1970 GMT) along with 48 bits of "identifier". For TCP/IP connections, the identifier is the address plus port number; for local connections it is the process ID and 32 bits to form a unique id (in case multiple connections to the same server are made from a single process).

This 192 bit packet is then encrypted using the DES key and sent to the X server, which is able to verify if the requestor is authorized to connect by decrypting with the same DES key and validating the authenticator and additional data. This system is useful in many environments where host-based access control is inappropriate and where network security cannot be ensured.

SUN-DES-1

Recent versions of SunOS (and some other systems) have included a secure public key remote procedure call system. This system is based on the notion of a network principal; a user name and NIS domain pair. Using this system, the X server can securely discover the actual user name of the requesting process. It involves encrypting data with the X server's public key, and so the identity of the user who started the X server is needed for this; this identity is stored in the *.Xauthority* file. By extending the semantics of "host address" to include this notion of network principal, this form of access control is very easy to use.

To allow access by a new user, use *xhost*. For example,

```
xhost keith@ ruth@mit.edu
```

adds "keith" from the NIS domain of the local machine, and "ruth" in the "mit.edu" NIS domain. For keith or ruth to successfully connect to the display, they must add the principal who started the server to their *.Xauthority* file. For example:

```
xauth add expo.lcs.mit.edu:0 SUN-DES-1 1.expo.lcs.mit.edu@our.domain.edu
```

This system only works on machines which support Secure RPC, and only for users which have set up the appropriate public/private key pairs on their system. See the Secure RPC documentation for details. To access the display from a remote host, you may have to do a *keylogin* on the remote host first.

MIT-KERBEROS-5

Kerberos is a network-based authentication scheme developed by MIT for Project Athena. It allows mutually suspicious principals to authenticate each other as long as each trusts a third party, Kerberos. Each principal has a secret key known only to it and Kerberos. Principals includes servers, such as an FTP server or X server, and human users, whose key is their password. Users gain access to services by getting Kerberos tickets for those services from a Kerberos server. Since the X server has no place to store a secret key, it shares keys with the user who logs in. X authentication thus uses the user-to-user scheme of Kerberos version 5.

When you log in via *xdm*, *xdm* will use your password to obtain the initial Kerberos tickets. *xdm* stores the tickets in a credentials cache file and sets the environment variable *KRB5CCNAME* to point to the file. The credentials cache is destroyed when the session ends to reduce the chance of the tickets being stolen before they expire.

Since Kerberos is a user-based authorization protocol, like the SUN-DES-1 protocol, the owner of a display can enable and disable specific users, or Kerberos principals. The *xhost* client is used to enable or disable authorization. For example,

```
xhost krb5:judy krb5:gildea@x.org
```

adds "judy" from the Kerberos realm of the local machine, and "gildea" from the "x.org" realm.

THE AUTHORIZATION FILE

Except for Host Access control, each of these systems uses data stored in the *.Xauthority* file to generate the correct authorization information to pass along to the X server at connection setup. MIT-MAGIC-COOKIE-1 and XDM-AUTHORIZATION-1 store secret data in the file; so anyone who can read the file can gain access to the X server. SUN-DES-1 stores only the identity of the principal who started the server (1.*host-name@domain* when the server is started by *xdm*), and so it is not useful to anyone not authorized to connect to the server.

Each entry in the *.Xauthority* file matches a certain connection family (TCP/IP, DECnet or local connections) and X display name (hostname plus display number). This allows multiple authorization entries for different displays to share the same data file. A special connection family (FamilyWild, value 65535) causes an entry to match every display, allowing the entry to be used for all connections. Each entry additionally contains the authorization name and whatever private authorization data is needed by that authorization type to generate the correct information at connection setup time.

The *xauth* program manipulates the *.Xauthority* file format. It understands the semantics of the connection families and address formats, displaying them in an easy to understand format. It also understands that SUN-DES-1 and MIT-KERBEROS-5 use string values for the authorization data, and displays them appropriately.

The X server (when running on a workstation) reads authorization information from a file name passed on the command line with the −*auth* option (see the *Xserver* manual page). The authorization entries in the file are used to control access to the server. In each of the authorization schemes listed above, the data needed by the server to initialize an authorization scheme is identical to the data needed by the client to generate the appropriate authorization information, so the same file can be used by both processes. This is especially useful when *xinit* is used.

MIT-MAGIC-COOKIE-1
: This system uses 128 bits of data shared between the user and the X server. Any collection of bits can be used. *Xdm* generates these keys using a cryptographically secure pseudo random number generator, and so the key to the next session cannot be computed from the current session key.

XDM-AUTHORIZATION-1
: This system uses two pieces of information. First, 64 bits of random data, second a 56 bit DES encryption key (again, random data) stored in 8 bytes, the last byte of which is ignored. *Xdm* generates these keys using the same random number generator as is used for MIT-MAGIC-COOKIE-1.

SUN-DES-1
: This system needs a string representation of the principal which identifies the associated X server. This information is used to encrypt the client's authority information when it is sent to the X server. When *xdm* starts the X server, it uses the root principal for the machine on which it is running (1.*hostname@domain*, e.g., "unix.expire.lcs.mit.edu@our.domain.edu"). Putting the correct principal name in the *.Xauthority* file causes Xlib to generate the appropriate authorization information using the secure RPC library.

MIT-KERBEROS-5
: Kerberos reads tickets from the cache pointed to by the *KRB5CCNAME* environment variable, so does not use any data from the *.Xauthority* file. An entry with no data must still exist to tell clients that MIT-KERBEROS-5 is available.

Unlike the *.Xauthority* file for clients, the authority file passed by xdm to a local X server (with "**−auth** *filename*", see xdm(1)) does contain the name of the credentials cache, since the X server will not have the *KRB5CCNAME* environment variable set. The data of the MIT-KERBEROS-5 entry is the credentials cache name and has the form ''UU:FILE:*filename*'', where *filename* is the name of the credentials cache file created by xdm. Note again that this form is *not* used by clients.

FILES

.Xauthority

SEE ALSO

X(1), xdm(1), xauth(1), xhost(1), xinit(1), Xserver(1)

NAME

imake − C preprocessor interface to the make utility

SYNOPSIS

imake [**−D***define*] [**−I***dir*] [**−T***template*] [**−f** *filename*] [**−C** *filename*] [**−s** *filename*] [**−e**] [**−v**]

DESCRIPTION

Imake is used to generate *Makefiles* from a template, a set of *cpp* macro functions, and a per-directory input file called an *Imakefile*. This allows machine dependencies (such as compiler options, alternate command names, and special *make* rules) to be kept separate from the descriptions of the various items to be built.

OPTIONS

The following command line options may be passed to *imake*:

−D*define* This option is passed directly to *cpp*. It is typically used to set directory-specific variables. For example, the X Window System uses this flag to set *TOPDIR* to the name of the directory containing the top of the core distribution and *CURDIR* to the name of the current directory, relative to the top.

−I*directory*
This option is passed directly to *cpp*. It is typically used to indicate the directory in which the *imake* template and configuration files may be found.

−T*template*
This option specifies the name of the master template file (which is usually located in the directory specified with *−I*) used by *cpp*. The default is *Imake.tmpl*.

−f *filename*
This option specifies the name of the per-directory input file. The default is *Imakefile*.

−C *filename*
This option specifies the name of the .c file that is constructed in the current directory. The default is *Imakefile.c*.

−s *filename*
This option specifies the name of the *make* description file to be generated but *make* should not be invoked. If the *filename* is a dash (−), the output is written to *stdout*. The default is to generate, but not execute, a *Makefile*.

−e This option indicates the *imake* should execute the generated *Makefile*. The default is to leave this to the user.

−v This option indicates that *imake* should print the *cpp* command line that it is using to generate the *Makefile*.

HOW IT WORKS

Imake invokes *cpp* with any *−I* or *−D* flags passed on the command line and passes the name of a file containing the following 3 lines:

```
#define IMAKE_TEMPLATE "Imake.tmpl"
#define INCLUDE_IMAKEFILE <Imakefile>
#include IMAKE_TEMPLATE
```

where *Imake.tmpl* and *Imakefile* may be overridden by the *−T* and *−f* command options, respectively.

The IMAKE_TEMPLATE typically reads in a file containing machine-dependent parameters (specified as *cpp* symbols), a site-specific parameters file, a file defining variables, a file containing *cpp* macro functions for generating *make* rules, and finally the *Imakefile* (specified by INCLUDE_IMAKEFILE) in the current directory. The *Imakefile* uses the macro functions to indicate what targets should be built; *imake* takes care of generating the appropriate rules.

Imake configuration files contain two types of variables, imake variables and make variables. The imake variables are interpreted by cpp when *imake* is run. By convention they are mixed case. The make variables are written into the *Makefile* for later interpretation by *make.* By convention make variables are upper case.

The rules file (usually named *Imake.rules* in the configuration directory) contains a variety of *cpp* macro functions that are configured according to the current platform. *Imake* replaces any occurrences of the string "@@" with a newline to allow macros that generate more than one line of *make* rules. For example, the macro

```
#define      program_target(program, objlist)                @@\
program:     objlist                                          @@\
             $(CC) -o $@ objlist $(LDFLAGS)
```

when called with *program_target(foo, foo1.o foo2.o)* will expand to

```
foo:         foo1.o foo2.o
             $(CC) -o $@ foo1.o foo2.o $(LDFLAGS)
```

Imake also replaces any occurrences of the word "XCOMM" with the character "#" to permit placing comments in the Makefile without causing "invalid directive" errors from the preprocessor.

Some complex *imake* macros require generated *make* variables local to each invocation of the macro, often because their value depends on parameters passed to the macro. Such variables can be created by using an *imake* variable of the form **XVARdef***n*, where *n* is a single digit. A unique *make* variable will be substituted. Later occurrences of the variable **XVARuse***n* will be replaced by the variable created by the corresponding **XVARdef***n*.

On systems whose *cpp* reduces multiple tabs and spaces to a single space, *imake* attempts to put back any necessary tabs (*make* is very picky about the difference between tabs and spaces). For this reason, colons (:) in command lines must be preceded by a backslash (\).

USE WITH THE X WINDOW SYSTEM

The X Window System uses *imake* extensively, for both full builds within the source tree and external software. As mentioned above, two special variables, *TOPDIR* and *CURDIR,* are set to make referencing files using relative path names easier. For example, the following command is generated automatically to build the *Makefile* in the directory *lib/X/* (relative to the top of the sources):

```
% ../.././config/imake -I../.././config \
        -DTOPDIR=../../. -DCURDIR=./lib/X
```

When building X programs outside the source tree, a special symbol *UseInstalled* is defined and *TOPDIR* and *CURDIR* are omitted. If the configuration files have been properly installed, the script *xmkmf*(1) may be used.

INPUT FILES

Here is a summary of the files read by *imake* as used by X. The indentation shows what files include what other files.

Imake.tmpl	generic variables
site.def	site-specific, BeforeVendorCF defined
*.cf	machine-specific
*Lib.rules	shared library rules
site.def	site-specific, AfterVendorCF defined
Imake.rules	rules
Project.tmpl	X-specific variables
*Lib.tmpl	shared library variables
Imakefile	
Library.tmpl	library rules
Server.tmpl	server rules
Threads.tmpl	multi-threaded rules

Note that *site.def* gets included twice, once before the **.cf* file and once after. Although most site customizations should be specified after the **.cf* file, some, such as the choice of compiler, need to be specified before, because other variable settings may depend on them. The first time *site.def* is included, the variable BeforeVendorCF is defined, and the second time, the variable AfterVendorCF is defined. All code in *site.def* should be inside an #ifdef for one of these symbols.

FILES

Imakefile.c	temporary input file for cpp
/tmp/Imf.XXXXXX	temporary Makefile for -s
/tmp/IIf.XXXXXX	temporary Imakefile if specified Imakefile uses # comments
/lib/cpp	default C preprocessor

SEE ALSO

make(1), xmkmf(1)
S. I. Feldman, *Make — A Program for Maintaining Computer Programs*

ENVIRONMENT VARIABLES

The following environment variables may be set, however their use is not recommended as they introduce dependencies that are not readily apparent when *imake* is run:

IMAKEINCLUDE
If defined, this specifies a ‘‘−I’’ include argument to pass to the C preprocessor. E.g., ‘‘−I/usr/X11/config’’.

IMAKECPP
If defined, this should be a valid path to a preprocessor program. E.g., ‘‘/usr/local/cpp’’. By default, *imake* will use /lib/cpp.

IMAKEMAKE
If defined, this should be a valid path to a make program, such as ‘‘/usr/local/make’’. By default, *imake* will use whatever *make* program is found using *execvp(3).* This variable is only used if the ‘‘−e’’ option is specified.

AUTHOR

Todd Brunhoff, Tektronix and MIT Project Athena; Jim Fulton, MIT X Consortium

NAME

makedepend − create dependencies in makefiles

SYNOPSIS

makedepend [**−D***name***=***def*] [**−D***name*] [**−I***includedir*] [**−Y***includedir*] [**−a**] [**−f***makefile*] [**−o***objsuffix*] [**−p***objprefix*] [**−s***string*] [**−w***width*] [**−v**] [**−m**] [**−−** *otheroptions* **−−**] *sourcefile* . . .

DESCRIPTION

The **makedepend** program reads each *sourcefile* in sequence and parses it like a C-preprocessor, processing all *#include, #define, #undef, #ifdef, #ifndef, #endif, #if, #elif* and *#else* directives so that it can correctly tell which *#include,* directives would be used in a compilation. Any *#include,* directives can reference files having other *#include* directives, and parsing will occur in these files as well.

Every file that a *sourcefile* includes, directly or indirectly, is what **makedepend** calls a *dependency.* These dependencies are then written to a *makefile* in such a way that **make(1)** will know which object files must be recompiled when a dependency has changed.

By default, **makedepend** places its output in the file named *makefile* if it exists, otherwise *Makefile.* An alternate makefile may be specified with the **−f** option. It first searches the makefile for the line

```
# DO NOT DELETE THIS LINE −− make depend depends on it.
```

or one provided with the **−s** option, as a delimiter for the dependency output. If it finds it, it will delete everything following this to the end of the makefile and put the output after this line. If it doesn't find it, the program will append the string to the end of the makefile and place the output following that. For each *sourcefile* appearing on the command line, **makedepend** puts lines in the makefile of the form

```
sourcefile.o: dfile ...
```

Where *sourcefile.o* is the name from the command line with its suffix replaced with ".o", and *dfile* is a dependency discovered in a *#include* directive while parsing *sourcefile* or one of the files it included.

EXAMPLE

Normally, **makedepend** will be used in a makefile target so that typing "make depend" will bring the dependencies up to date for the makefile. For example,

```
SRCS = file1.c file2.c ...
CFLAGS = −O −DHACK −I../foobar −xyz
depend:
        makedepend −− $(CFLAGS) −− $(SRCS)
```

OPTIONS

The program will ignore any option that it does not understand so that you may use the same arguments that you would for **cc(1).**

−D*name***=***def* or **−D***name*

Define. This places a definition for *name* in **makedepend's** symbol table. Without *=def* the symbol becomes defined as "1".

−I*includedir*

Include directory. This option tells **makedepend** to prepend *includedir* to its list of directories to search when it encounters a *#include* directive. By default, **makedepend** only searches the standard include directories (usually /usr/include and possibly a compiler-dependent directory).

−Y*includedir*
Replace all of the standard include directories with the single specified include directory; you can omit the *includedir* to simply prevent searching the standard include directories.

−a Append the dependencies to the end of the file instead of replacing them.

−f*makefile*
Filename. This allows you to specify an alternate makefile in which **makedepend** can place its output. Specifying "−" as the file name (i.e., **−f−**) sends the output to standard output instead of modifying an existing file.

−o*objsuffix*
Object file suffix. Some systems may have object files whose suffix is something other than ".o". This option allows you to specify another suffix, such as ".b" with *−o.b* or ":obj" with *−o:obj* and so forth.

−p*objprefix*
Object file prefix. The prefix is prepended to the name of the object file. This is usually used to designate a different directory for the object file. The default is the empty string.

−s*string*
Starting string delimiter. This option permits you to specify a different string for **makedepend** to look for in the makefile.

−w*width*
Line width. Normally, **makedepend** will ensure that every output line that it writes will be no wider than 78 characters for the sake of readability. This option enables you to change this width.

−v Verbose operation. This option causes **makedepend** to emit the list of files included by each input file on standard output.

−m Warn about multiple inclusion. This option causes **makedepend** to produce a warning if any input file includes another file more than once. In previous versions of **makedepend** this was the default behavior; the default has been changed to better match the behavior of the C compiler, which does not consider multiple inclusion to be an error. This option is provided for backward compatibility, and to aid in debugging problems related to multiple inclusion.

−− *options* **−−**
If **makedepend** encounters a double hyphen (−−) in the argument list, then any unrecognized argument following it will be silently ignored; a second double hyphen terminates this special treatment. In this way, **makedepend** can be made to safely ignore esoteric compiler arguments that might normally be found in a CFLAGS **make** macro (see the **EXAMPLE** section above). All options that **makedepend** recognizes and appear between the pair of double hyphens are processed normally.

ALGORITHM

The approach used in this program enables it to run an order of magnitude faster than any other "dependency generator" I have ever seen. Central to this performance are two assumptions: that all files compiled by a single makefile will be compiled with roughly the same −*I* and −*D* options; and that most files in a single directory will include largely the same files.

Given these assumptions, **makedepend** expects to be called once for each makefile, with all source files that are maintained by the makefile appearing on the command line. It parses each source and include file exactly once, maintaining an internal symbol table for each. Thus, the first file on the command line will take an amount of time proportional to the amount of time that a normal C preprocessor takes. But on subsequent files, if it encounters an include file that it has already parsed, it does not parse it again.

For example, imagine you are compiling two files, *file1.c* and *file2.c,* they each include the header file *header.h,* and the file *header.h* in turn includes the files *def1.h* and *def2.h.* When you run the command

```
makedepend  file1.c  file2.c
```

makedepend will parse *file1.c* and consequently, *header.h* and then *def1.h* and *def2.h.* It then decides that the dependencies for this file are

```
file1.o:  header.h  def1.h  def2.h
```

But when the program parses *file2.c* and discovers that it, too, includes *header.h,* it does not parse the file, but simply adds *header.h, def1.h* and *def2.h* to the list of dependencies for *file2.o.*

SEE ALSO

cc(1), make(1)

BUGS

makedepend parses, but does not currently evaluate, the SVR4 #predicate(token-list) preprocessor expression; such expressions are simply assumed to be true. This may cause the wrong *#include* directives to be evaluated.

Imagine you are parsing two files, say *file1.c* and *file2.c,* each includes the file *def.h.* The list of files that *def.h* includes might truly be different when *def.h* is included by *file1.c* than when it is included by *file2.c.* But once **makedepend** arrives at a list of dependencies for a file, it is cast in concrete.

AUTHOR

Todd Brunhoff, Tektronix, Inc. and MIT Project Athena

NAME

lndir − create a shadow directory of symbolic links to another directory tree

SYNOPSIS

lndir [**−silent**] [**−ignorelinks**] *fromdir* [*todir*]

DESCRIPTION

The *lndir* program makes a shadow copy *todir* of a directory tree *fromdir,* except that the shadow is not populated with real files but instead with symbolic links pointing at the real files in the *fromdir* directory tree. This is usually useful for maintaining source code for different machine architectures. You create a shadow directory containing links to the real source, which you will have usually mounted from a remote machine. You can build in the shadow tree, and the object files will be in the shadow directory, while the source files in the shadow directory are just symlinks to the real files.

This scheme has the advantage that if you update the source, you need not propagate the change to the other architectures by hand, since all source in all shadow directories are symlinks to the real thing: just cd to the shadow directory and recompile away.

The *todir* argument is optional and defaults to the current directory. The *fromdir* argument may be relative (e.g., ../src) and is relative to *todir* (not the current directory).

Note that RCS, SCCS, CVS and CVS.adm directories are not shadowed.

If you add files, simply run *lndir* again. New files will be silently added. Old files will be checked that they have the correct link.

Deleting files is a more painful problem; the symlinks will just point into never never land.

If a file in *fromdir* is a symbolic link, *lndir* will make the same link in *todir* rather than making a link back to the (symbolic link) entry in *fromdir.* The **−ignorelinks** flag changes this behavior.

OPTIONS

−silent Normally *lndir* outputs the name of each subdirectory as it descends into it. The **−silent** option suppresses these status messages.

−ignorelinks

Causes the program to not treat symbolic links in *fromdir* specially. The link created in *todir* will point back to the corresponding (symbolic link) file in *fromdir*. If the link is to a directory, this is almost certainly the wrong thing.

This option exists mostly to emulate the behavior the C version of *lndir* had in X11R6. Its use is not recommended.

DIAGNOSTICS

The program displays the name of each subdirectory it enters, followed by a colon. The **−silent** option suppresses these messages.

A warning message is displayed if the symbolic link cannot be created. The usual problem is that a regular file of the same name already exists.

If the link already exists but doesn't point to the correct file, the program prints the link name and the location where it does point.

BUGS

The *patch* program gets upset if it cannot change the files. You should never run *patch* from a shadow directory anyway.

You need to use something like

find todir −type l −print | xargs rm

to clear out all files before you can relink (if fromdir moved, for instance). Something like

find . \! −type d −print

will find all files that are not directories.

NAME

makeg – make a debuggable executable

SYNOPSIS

makeg [*make-options* ...] [*targets* ...]

DESCRIPTION

The *makeg* script runs *make,* passing it variable settings to create a debuggable target when used with a Makefile generated by *imake.* For example, it arranges for the C compiler to be called with the **–g** option.

ENVIRONMENT

MAKE The *make* program to use. Default "make".

GDB Set to a non-null value if using the *gdb* debugger on Solaris 2, which requires additional debugging options to be passed to the compiler.

SEE ALSO

make (1), imake (1)

NAME

makestrs – makes string table C source and header(s)

SYNOPSIS

makestrs [-f source] [-abioptions ...]

DESCRIPTION

The *makestrs* command creates string table C source files and headers. If *-f source* is not specified *makestrs* will read from *stdin.* The C source file is always written to *stdout.* *makestrs* creates one or more C header files as specified in the source file. The following options may be specified: *-sparcabi, -intelabi, -functionabi, -arrayperabi,* and *-defaultabi.* *-sparcabi* is used on SPARC platforms conforming to the SPARC Compliance Definition, i.e. SVR4/Solaris. *-intelabi* is used on Intel platforms conforming to the System V Application Binary Interface, i.e. SVR4. *-earlyR6abi* may be used in addition to *-intelabi* for situations where the vendor wishes to maintain binary compatiblity between X11R6 public-patch 11 (and earlier) and X11R6 public-patch 12 (and later). *-functionabi* generates a functional abi to the string table. This mechanism imposes a severe performance penalty and it's recommended that you not use it. *-arrayperabi* results in a separate array for each string. This is the default behavior if makestrs was compiled with -DARRAYPERSTR (it almost never is). *-defaultabi* forces the generation of the "normal" string table even if makestrs was compiled with -DARRAYPERSTR. Since makestrs is almost never compiled with -DARRAYPERSTR this is the default behavior if no abioptions are specified.

SYNTAX

The syntax for string-list file is (items in square brackets are optional):

```
#prefix <text>
#feature <text>
#externref <text>
#externdef [<text>]
[#ctempl <text>]
#file <filename>
#table <tablename>
[#htempl]
<text>
<text>
[#table <tablename>
<text>
<text>
#table <tablename>
[#file <filename>
```

In words you may have one or more #file directives. Each #file may have one or more #table directives. The #prefix directive determines the string that makestr will prefix to each definition. The #feature directive determines the string that makestr will use for the feature-test macro, e.g. X[TM]STRINGDEFINES. The #externref directive determines the string that makestr will use for the extern clause, typically this will be "extern" but Motif wants it to be "externalref" The #externdef directive determines the string that makestr will use for the declaration, typically this will be the null string (note that makestrs requires a trailing space in this case, i.e. "#externdef "), and Motif will use "externaldef(_xmstrings). The #ctmpl directive determines the name of the file used as a template for the C source file that is generated Each #file <filename> directive will result in a corresponding header file by that name containing the appropriate definitions as specified by command line options. A single C source file containing the declarations for the definitions in all the headers will be printed to stdout. The #htmpl directive determines the name of the file used as a template for the C header file that is generated. Each #table <tablename> directive will be processed in accordance with the ABI. On most platforms all tables will be catenated into a single table with the name of the first table for that file. To conform to the Intel ABI separate tables will be generated with the names indicated. The template files specified by the #ctmpl and #htmpl directives are processed by copying line for line from the template file to the appropriate output file. The line containing the string *<<<STRING_TABLE_GOES_HERE>>>* is not copied to the output file. The appropriate data is then copied to the output file and then the remainder of the template file is copied to the output file.

BUGS

makestrs is not very forgiving of syntax errors. Sometimes you need a trailing space after # directives, other times they will mess you up. No warning messages are emitted.

SEE ALSO

SPARC Compliance Definition 2.2., SPARC International Inc., 535 Middlefield Road, Suite 210, Menlo Park, CA 94025 System V Application Binary Interface, Third Edition, ISBN 0-13-100439-5 UNIX Press, PTR Prentice Hall, 113 Sylvan Avenue, Englewood Cliffs, NJ 07632 System V Application Binary Interface, Third Edition, Intel386 Architecture Processor Supplement ISBN 0-13-104670-5 UNIX Press, PTR Prentice Hall, 113 Sylvan Avenue, Englewood Cliffs, NJ 07632 System V Application Binary Interface, Third Edition, SPARC Architecture Processor Supplement ISBN 0-13-104696-9 UNIX Press, PTR Prentice Hall, 113 Sylvan Avenue, Englewood Cliffs, NJ 07632

NAME

mkdirhier − makes a directory hierarchy

SYNOPSIS

mkdirhier directory ...

DESCRIPTION

The *mkdirhier* command creates the specified directories. Unlike *mkdir* if any of the parent directories of the specified directory do not exist, it creates them as well.

SEE ALSO

mkdir(1)

NAME

xmkmf − create a Makefile from an Imakefile

SYNOPSIS

xmkmf [-a] [*topdir* [*curdir*]]

DESCRIPTION

The *xmkmf* command is the normal way to create a *Makefile* from an *Imakefile* shipped with third-party software.

When invoked with no arguments in a directory containing an *Imakefile,* the *imake* program is run with arguments appropriate for your system (configured into *xmkmf* when X was built) and generates a *Makefile.*

When invoked with the −*a* option, *xmkmf* builds the *Makefile* in the current directory, and then automatically executes ‘‘make Makefiles’’ (in case there are subdirectories), ‘‘make includes’’, and ‘‘make depend’’ for you. This is the normal way to configure software that is outside the X Consortium build tree.

If working inside the X Consortium build tree (unlikely unless you are an X developer, and even then this option is never really used), the *topdir* argument should be specified as the relative pathname from the current directory to the top of the build tree. Optionally, *curdir* may be specified as a relative pathname from the top of the build tree to the current directory. It is necessary to supply *curdir* if the current directory has subdirectories, or the *Makefile* will not be able to build the subdirectories. If a *topdir* is given, *xmkmf* assumes nothing is installed on your system and looks for files in the build tree instead of using the installed versions.

SEE ALSO

imake(1)

NAME

Xserver – X Window System display server

SYNOPSIS

X [option ...]

DESCRIPTION

X is the generic name for the X Window System display server. It is frequently a link or a copy of the appropriate server binary for driving the most frequently used server on a given machine.

STARTING THE SERVER

The X server is usually started from the X Display Manager program *xdm(1)*. This utility is run from the system boot files and takes care of keeping the server running, prompting for usernames and passwords, and starting up the user sessions.

Installations that run more than one window system may need to use the *xinit(1)* utility instead of *xdm*. However, *xinit* is to be considered a tool for building startup scripts and is not intended for use by end users. Site administrators are **strongly** urged to use *xdm*, or build other interfaces for novice users.

The X server may also be started directly by the user, though this method is usually reserved for testing and is not recommended for normal operation. On some platforms, the user must have special permission to start the X server, often because access to certain devices (e.g. /dev/mouse) is restricted.

When the X server starts up, it typically takes over the display. If you are running on a workstation whose console is the display, you may not be able to log into the console while the server is running.

OPTIONS

All of the X servers accept the following command line options:

:*displaynumber*
: the X server runs as the given *displaynumber*, which by default is 0. If multiple X servers are to run simultaneously on a host, each must have a unique display number. See the DISPLAY NAMES section of the *X(1)* manual page to learn how to specify which display number clients should try to use.

–a *number*
: sets pointer acceleration (i.e. the ratio of how much is reported to how much the user actually moved the pointer).

–ac
: disables host-based access control mechanisms. Enables access by any host, and permits any host to modify the access control list. Use with extreme caution. This option exists primarily for running test suites remotely.

–audit *level*
: Sets the audit trail level. The default level is 1, meaning only connection rejections are reported. Level 2 additionally reports all successful connections and disconnects. Level 4 enables messages from the SECURITY extension, if present, including generation and revocation of authorizations and violations of the security policy. Level 0 turns off the audit trail. Audit lines are sent as standard error output.

–auth *authorization-file*
: Specifies a file which contains a collection of authorization records used to authenticate access. See also the *xdm* and *Xsecurity* manual pages.

bc disables certain kinds of error checking, for bug compatibility with previous releases (e.g., to work around bugs in R2 and R3 xterms and toolkits). Deprecated.

−bs disables backing store support on all screens.

−c turns off key-click.

c *volume* sets key-click volume (allowable range: 0-100).

−cc *class* sets the visual class for the root window of color screens. The class numbers are as specified in the X protocol. Not obeyed by all servers.

−co *filename*
sets name of RGB color database. The default is <XRoot>/lib/X11/rgb, where <XRoot> refers to the root of the X11 install tree.

−config *filename*
reads more options from the given file. Options in the file may be separated by newlines if desired. If a '#' character appears on a line, all characters between it and the next newline are ignored, providing a simple commenting facility. The **−config** option itself may appear in the file.

−core causes the server to generate a core dump on fatal errors.

−dpi *resolution*
sets the resolution of the screen, in dots per inch. To be used when the server cannot determine the screen size from the hardware.

−deferglyphs *whichfonts*
specifies the types of fonts for which the server should attempt to use deferred glyph loading. *whichfonts* can be all (all fonts), none (no fonts), or 16 (16 bit fonts only).

−f *volume*
sets feep (bell) volume (allowable range: 0-100).

−fc *cursorFont*
sets default cursor font.

−fn *font* sets the default font.

−fp *fontPath*
sets the search path for fonts. This path is a comma separated list of directories which the X server searches for font databases.

−help prints a usage message.

−I causes all remaining command line arguments to be ignored.

−kb disables the XKEYBOARD extension if present.

−p *minutes*
sets screen-saver pattern cycle time in minutes.

−pn permits the server to continue running if it fails to establish all of its well-known sockets (connection points for clients), but establishes at least one.

−r turns off auto-repeat.

r turns on auto-repeat.

−s *minutes*
sets screen-saver timeout time in minutes.

−su disables save under support on all screens.

−t *number*
sets pointer acceleration threshold in pixels (i.e. after how many pixels pointer acceleration should take effect).

−terminate
causes the server to terminate at server reset, instead of continuing to run.

−to *seconds*
sets default connection timeout in seconds.

−tst disables all testing extensions (e.g., XTEST, XTrap, XTestExtension1, RECORD).

tty*xx* ignored, for servers started the ancient way (from init).

v sets video-off screen-saver preference.

−v sets video-on screen-saver preference.

−wm forces the default backing-store of all windows to be WhenMapped. This is a backdoor way of getting backing-store to apply to all windows. Although all mapped windows will have backing store, the backing store attribute value reported by the server for a window will be the last value established by a client. If it has never been set by a client, the server will report the default value, NotUseful. This behavior is required by the X protocol, which allows the server to exceed the client's backing store expectations but does not provide a way to tell the client that it is doing so.

−x *extension*
loads the specified extension at init. This is a no-op for most implementations.

[+-]xinerama
enable(+) or disable(-) XINERAMA extension. Default is disabled.

SERVER DEPENDENT OPTIONS

Some X servers accept the following options:

−ld *kilobytes*
sets the data space limit of the server to the specified number of kilobytes. A value of zero makes the data size as large as possible. The default value of −1 leaves the data space limit unchanged.

−lf *files* sets the number-of-open-files limit of the server to the specified number. A value of zero makes the limit as large as possible. The default value of −1 leaves the limit unchanged.

−ls *kilobytes*
sets the stack space limit of the server to the specified number of kilobytes. A value of zero makes the stack size as large as possible. The default value of −1 leaves the stack space limit unchanged.

−logo turns on the X Window System logo display in the screen-saver. There is currently no way to change this from a client.

nologo turns off the X Window System logo display in the screen-saver. There is currently no way to change this from a client.

XDMCP OPTIONS

X servers that support XDMCP have the following options. See the *X Display Manager*

Control Protocol specification for more information.

–query *host-name*
Enable XDMCP and send Query packets to the specified host.

–broadcast
Enable XDMCP and broadcast BroadcastQuery packets to the network. The first responding display manager will be chosen for the session.

–indirect *host-name*
Enable XDMCP and send IndirectQuery packets to the specified host.

–port *port-num*
Use an alternate port number for XDMCP packets. Must be specified before any –query, –broadcast or –indirect options.

–class *display-class*
XDMCP has an additional display qualifier used in resource lookup for display-specific options. This option sets that value, by default it is "MIT-Unspecified" (not a very useful value).

–cookie *xdm-auth-bits*
When testing XDM-AUTHENTICATION-1, a private key is shared between the server and the manager. This option sets the value of that private data (not that it is very private, being on the command line!).

–displayID *display-id*
Yet another XDMCP specific value, this one allows the display manager to identify each display so that it can locate the shared key.

XKEYBOARD OPTIONS

X servers that support the XKEYBOARD extension accept the following options:

–xkbdir *directory*
base directory for keyboard layout files

–xkbmap *filename*
keyboard description to load on startup

[+-]accessx
enable(+) or disable(-) AccessX key sequences

–ar1 *milliseconds*
sets the length of time in milliseconds that a key must be depressed before autorepeat starts

–ar2 *milliseconds*
sets the length of time in milliseconds that should elapse between autorepeat-generated keystrokes

Many servers also have device-specific command line options. See the manual pages for the individual servers for more details.

SECURITY EXTENSION OPTIONS

X servers that support the SECURITY extension accept the following option:

–sp *filename*
causes the server to attempt to read and interpret filename as a security policy file with the format described below. The file is read at server startup and reread at each server reset.

The syntax of the security policy file is as follows. Notation: "*" means zero or more occurrences of the preceding element, and "+" means one or more occurrences. To interpret <foo/bar>, ignore the text after the /; it is used to distinguish between instances of <foo> in the next section.

<policy file> ::= <version line> <other line>*

<version line> ::= <string/v> ’\n’

<other line > ::= <comment> | <access rule> | <site policy> | <blank line>

<comment> ::= # <not newline>* ’\n’

<blank line> ::= <space> ’\n’

<site policy> ::= sitepolicy <string/sp> ’\n’

<access rule> ::= property <property/ar> <window> <perms> ’\n’

<property> ::= <string>

<window> ::= any | root | <required property>

<required property> ::= <property/rp> | <property with value>

<property with value> ::= <property/rpv> = <string/rv>

<perms> ::= [<operation> | <action> | <space>]*

<operation> ::= r | w | d

<action> ::= a | i | e

<string> ::= <dbl quoted string> | <single quoted string> | <unqouted string>

<dbl quoted string> ::= <space> " <not dqoute>* " <space>

<single quoted string> ::= <space> ’ <not squote>* ’ <space>

<unquoted string> ::= <space> <not space>+ <space>

<space> ::= [’ ’ | ’\t’]*

Character sets:

<not newline> ::= any character except ’\n’
<not dqoute> ::= any character except "
<not squote> ::= any character except ’
<not space> ::= any character except those in <space>

The semantics associated with the above syntax are as follows.

<version line>, the first line in the file, specifies the file format version. If the server does not recognize the version <string/v>, it ignores the rest of the file. The version string for the file format described here is "version-1" .

Once past the <version line>, lines that do not match the above syntax are ignored.

<comment> lines are ignored.

<sitepolicy> lines are currently ignored. They are intended to specify the site policies used by the XC-QUERY-SECURITY-1 authorization method.

<access rule> lines specify how the server should react to untrusted client requests that affect the X Window property named <property/ar>. The rest of this section describes the interpretation of an <access rule>.

For an <access rule> to apply to a given instance of <property/ar>, <property/ar> must be on a window that is in the set of windows specified by <window>. If <window> is any, the rule applies to <property/ar> on any window. If <window> is root, the rule applies to <property/ar> only on root windows.

If <window> is <required property>, the following apply. If <required property> is a <property/rp>, the rule applies when the window also has that <property/rp>, regardless of its value. If <required property> is a <property with value>, <property/rpv> must also have the value specified by <string/rv>. In this case, the property must have type STRING and format 8, and should contain one or more null-terminated strings. If any of the strings match <string/rv>, the rule applies.

The definition of string matching is simple case-sensitive string comparison with one elaboration: the occurence of the character '*' in <string/rv> is a wildcard meaning "any string." A <string/rv> can contain multiple wildcards anywhere in the string. For example, "x*" matches strings that begin with x, "*x" matches strings that end with x, "*x*" matches strings containing x, and "x*y*" matches strings that start with x and subsequently contain y.

There may be multiple <access rule> lines for a given <property/ar>. The rules are tested in the order that they appear in the file. The first rule that applies is used.

<perms> specify operations that untrusted clients may attempt, and the actions that the server should take in response to those operations.

<operation> can be r (read), w (write), or d (delete). The following table shows how X Protocol property requests map to these operations in the X Consortium server implementation.

GetProperty	r, or r and d if delete = True
ChangeProperty	w
RotateProperties	r and w
DeleteProperty	d
ListProperties	none, untrusted clients can always list all properties

<action> can be a (allow), i (ignore), or e (error). Allow means execute the request as if it had been issued by a trusted client. Ignore means treat the request as a no-op. In the case of GetProperty, ignore means return an empty property value if the property exists, regardless of its actual value. Error means do not execute the request and return a BadAtom error with the atom set to the property name. Error is the default action for all properties, including those not listed in the security policy file.

An <action> applies to all <operation>s that follow it, until the next <action> is encountered. Thus, irwad means ignore read and write, allow delete.

GetProperty and RotateProperties may do multiple operations (r and d, or r and w). If different actions apply to the operations, the most severe action is applied to the whole request; there is no partial request execution. The severity ordering is: allow < ignore < error. Thus, if the <perms> for a property are ired (ignore read, error delete), and an untrusted client attempts GetProperty on that property with delete = True, an error is returned, but the property value is not. Similarly, if any of the properties in a RotateProperties do not allow both read and write, an error is returned without changing any property values.

Here is an example security policy file.

```
version-1

# Allow reading of application resources, but not writing.
property RESOURCE_MANAGER          root       ar iw
property SCREEN_RESOURCES          root       ar iw

# Ignore attempts to use cut buffers.  Giving errors causes apps to crash,
# and allowing access may give away too much information.
property CUT_BUFFER0               root       irw
property CUT_BUFFER1               root       irw
property CUT_BUFFER2               root       irw
property CUT_BUFFER3               root       irw
property CUT_BUFFER4               root       irw
property CUT_BUFFER5               root       irw
property CUT_BUFFER6               root       irw
property CUT_BUFFER7               root       irw

# If you are using Motif, you probably want these.
property _MOTIF_DEFAULT_BINDINGS   root       ar iw
property _MOTIF_DRAG_WINDOW        root       ar iw
property _MOTIF_DRAG_TARGETS       any        ar iw
property _MOTIF_DRAG_ATOMS         any        ar iw
property _MOTIF_DRAG_ATOM_PAIRS    any        ar iw

# The next two rules let xwininfo -tree work when untrusted.
property WM_NAME                   any        ar

# Allow read of WM_CLASS, but only for windows with WM_NAME.
# This might be more restrictive than necessary, but demonstrates
# the <required property> facility, and is also an attempt to
# say "top level windows only."
property WM_CLASS                  WM_NAME    ar

# These next three let xlsclients work untrusted.  Think carefully
# before including these; giving away the client machine name and command
# may be exposing too much.
property WM_STATE                  WM_NAME    ar
property WM_CLIENT_MACHINE         WM_NAME    ar
property WM_COMMAND                WM_NAME    ar

# To let untrusted clients use the standard colormaps created by
```

```
# xstdcmap, include these lines.
property RGB_DEFAULT_MAP          root        ar
property RGB_BEST_MAP             root        ar
property RGB_RED_MAP              root        ar
property RGB_GREEN_MAP            root        ar
property RGB_BLUE_MAP             root        ar
property RGB_GRAY_MAP             root        ar

# To let untrusted clients use the color management database created
# by xcmsdb, include these lines.
property XDCCC_LINEAR_RGB_CORRECTION          rootar
property XDCCC_LINEAR_RGB_MATRICES            rootar
property XDCCC_GRAY_SCREENWHITEPOINT          rootar
property XDCCC_GRAY_CORRECTION    root        ar

# To let untrusted clients use the overlay visuals that many vendors
# support, include this line.
property SERVER_OVERLAY_VISUALS   root        ar

# Dumb examples to show other capabilities.

# oddball property names and explicit specification of error conditions
property "property with spaces"   'property with "'aw er ed

# Allow deletion of Woo-Hoo if window also has property OhBoy with value
# ending in "son".  Reads and writes will cause an error.
property Woo-Hoo                  OhBoy = "*son"ad
```

NETWORK CONNECTIONS

The X server supports client connections via a platform-dependent subset of the following transport types: TCPIP, Unix Domain sockets, DECnet, and several varieties of SVR4 local connections. See the DISPLAY NAMES section of the *X(1)* manual page to learn how to specify which transport type clients should try to use.

GRANTING ACCESS

The X server implements a platform-dependent subset of the following authorization protocols: MIT-MAGIC-COOKIE-1, XDM-AUTHORIZATION-1, SUN-DES-1, and MIT-KERBEROS-5. See the *Xsecurity(1)* manual page for information on the operation of these protocols.

Authorization data required by the above protocols is passed to the server in a private file named with the **–auth** command line option. Each time the server is about to accept the first connection after a reset (or when the server is starting), it reads this file. If this file contains any authorization records, the local host is not automatically allowed access to the server, and only clients which send one of the authorization records contained in the file in the connection setup information will be allowed access. See the *Xau* manual page for a description of the binary format of this file. See *xauth(1)* for maintenance of this file, and distribution of its contents to remote hosts.

The X server also uses a host-based access control list for deciding whether or not to accept connections from clients on a particular machine. If no other authorization mechanism is being used, this list initially consists of the host on which the server is running as well as any machines listed in the file */etc/X***n***.hosts*, where **n** is the display number of the server. Each line of the file should contain either an Internet hostname (e.g. expo.lcs.mit.edu) or a DECnet hostname in double colon format (e.g. hydra::). There should be no leading or trailing spaces on any lines. For example:

```
joesworkstation
corporate.company.com
star::
bigcpu::
```

Users can add or remove hosts from this list and enable or disable access control using the *xhost* command from the same machine as the server.

If the X FireWall Proxy (*xfwp*) is being used without a sitepolicy, host-based authorization must be turned on for clients to be able to connect to the X server via the *xfwp*. If *xfwp* is run without a configuration file and thus no sitepolicy is defined, if *xfwp* is using an X server where xhost + has been run to turn off host-based authorization checks, when a client tries to connect to this X server via *xfwp*, the X server will deny the connection. See *xfwp(1)* for more information about this proxy.

The X protocol intrinsically does not have any notion of window operation permissions or place any restrictions on what a client can do; if a program can connect to a display, it has full run of the screen. X servers that support the SECURITY extension fare better because clients can be designated untrusted via the authorization they use to connect; see the *xauth(1)* manual page for details. Restrictions are imposed on untrusted clients that curtail the mischief they can do. See the SECURITY extension specification for a complete list of these restrictions.

Sites that have better authentication and authorization systems might wish to make use of the hooks in the libraries and the server to provide additional security models.

SIGNALS

The X server attaches special meaning to the following signals:

SIGHUP This signal causes the server to close all existing connections, free all resources, and restore all defaults. It is sent by the display manager whenever the main user's main application (usually an *xterm* or window manager) exits to force the server to clean up and prepare for the next user.

SIGTERM
This signal causes the server to exit cleanly.

SIGUSR1
This signal is used quite differently from either of the above. When the server starts, it checks to see if it has inherited SIGUSR1 as SIG_IGN instead of the usual SIG_DFL. In this case, the server sends a SIGUSR1 to its parent process after it has set up the various connection schemes. *Xdm* uses this feature to recognize when connecting to the server is possible.

FONTS

The X server can obtain fonts from directories and/or from font servers. The list of directories and font servers the X server uses when trying to open a font is controlled by the *font path*. The default font path is "<XRoot>/lib/X11/fonts/misc/, <XRoot>/lib/X11/fonts/Speedo/, <XRoot>/lib/X11/fonts/Type1/, <XRoot>/lib/X11/fonts/75dpi/, <XRoot>/lib/X11/fonts/100dpi/" . where <XRoot> refers to the root of the X11 install tree. The font path can be set with the **–fp** option or by *xset(1)* after the server has started.

FILES

/etc/X**n**.hosts	Initial access control list for display number **n**
<XRoot>/lib/X11/fonts/misc, <XRoot>/lib/X11/fonts/75dpi, <XRoot>/lib/X11/fonts/100dpi	Bitmap font directories
<XRoot>/lib/X11/fonts/Speedo, <XRoot>/lib/X11/fonts/Type1	Outline font directories
<XRoot>/lib/X11/fonts/PEX	PEX font directories
<XRoot>/lib/X11/rgb.txt	Color database
/tmp/.X11-unix/X**n**	Unix domain socket for display number **n**
/tmp/rcX**n**	Kerberos 5 replay cache for display number **n**
/usr/adm/X**n**msgs	Error log file for display number **n** if run from *init(8)*
<XRoot>/lib/X11/xdm/xdm-errors	Default error log file if the server is run from *xdm(1)* Note: <XRoot> refers to the root of the X11 install tree.

SEE ALSO

General information: X(1)

Protocols: *X Window System Protocol, The X Font Service Protocol, X Display Manager Control Protocol*

Fonts: bdftopcf(1), mkfontdir(1), xfs(1), xlsfonts(1), xfontsel(1), xfd(1), *X Logical Font Description Conventions*

Security: Xsecurity(1), xauth(1), Xau(1), xdm(1), xhost(1), xfwp(1) *Security Extension Specification*

Starting the server: xdm(1), xinit(1)

Controlling the server once started: xset(1), xsetroot(1), xhost(1)

Server-specific man pages: Xdec(1), XmacII(1), Xsun(1), Xnest(1), Xvfb(1), XF86_Accel(1), XF86_Mono(1), XF86_SVGA(1), XF86_VGA16(1), XFree86(1)

Server internal documentation: *Definition of the Porting Layer for the X v11 Sample Server*

AUTHORS

The sample server was originally written by Susan Angebranndt, Raymond Drewry, Philip Karlton, and Todd Newman, from Digital Equipment Corporation, with support from a large cast. It has since been extensively rewritten by Keith Packard and Bob Scheifler, from MIT. Dave Wiggins took over post-R5 and made substantial improvements.

Name

Xdec – X server for Digital RISC machines

Syntax

Xdec [*options*]

Description

The command starts the X server. The command supports the following hardware configurations:

DECstation 2100 Monochrome or Color Workstations
DECstation 3100 Monochrome or Color Workstations
DECstation 5000/100/200 CX or MX Single or Multiscreen Workstations

This server should run on reasonable one bit or eight bit Ultrix/RISC TURBOchannel displays of any resolution, if correct device driver support is present. The server queries the device driver interface to determine if a suitable display device is installed, and if so, configures the server appropriately. The command that executes the server is specified together with its command line options in the file or using xdm(1) and, therefore, is automatically run when your system is started in multiuser mode. Optionally, you can create an file containing device-dependent command line options (separated by spaces) and use it to start the server. Command line options specified in the command starting the X server override those specified in the file.

Start the server in bug compatibility mode (with the **bc** option) to remain bug-for-bug compatible with previous releases of the server.

Options

In addition to the normal server options described in the *Xserver(1)* manual page, *Xdec* accepts the following device-dependent, vendor-specific options. When the server is run on multiscreen capable platforms, selected device-dependent options take an optional screen-specification argument. Omitting the screen-specification argument defines the parameter for all available screens.

–btn *num*
Specifies the number of buttons on the pointer device. The default is three for a mouse device and four for a tablet device.

–bp[*screen*] *color*
Sets the color of black pixels for the screen. The *color* argument can be a named color from the database or a number sign (#) followed by a hexidecimal number.

–class[*screen*] *visual class*
Sets the visual class for the root window of the screen. Possible values are and

–dpi[*screen*] *num*
Sets the dots per inch for the x and y coordinates.

–dpix[*screen*] *num*
Sets the dots per inch for the x coordinates.

–dpiy[*screen*] *num*
Sets the dots per inch for the y coordinates. By default on multiscreen systems, the server presumes the left hand most screen is screen zero, screen one to its right, screen two yet further to its right, and so on. The cursor will track from the right hand edge of screen zero to the left hand edge of screen one, from the right hand edge of screen one to the left hand edge of screen two. This arrangement can be modified in various ways, to support other physical arrangements of monitors.

−edge_bottom*scr1 scr2*
: Attaches the bottom edge of the screen specified by *scr1* to the screen specified by *scr2*.

−edge_left*scr1 scr2*
: Attaches the left edge of the screen specified by *scr1* to the screen specified by *scr2*.

−edge_right*scr1 scr2*
: Attaches the right edge of the screen specified by *scr1* to the screen specified by *scr2*.

−edge_top*scr1 scr2*
: Attaches the top edge of the screen specified by *scr1* to the screen specified by *scr2*.

−wp[*screen*] *color*
: Sets the color of white pixels for the screen. The syntax for *color* is the same as for the argument to the **−bp** option.

−tb *n*
: Opens for graphics tablet communications.

−pcm *n*
: Opens for Protocol Control Module (PCM) communications. The two free serial ports on the DECstation correspond to and Dial boxes and button boxes must be connected through these two ports.

Restrictions

If options not listed in this reference page are used, the server may fail. Using invalid options for the X server in the file may cause the workstation to behave as if the X server is hung.

Multiscreen configurations can contain either two- or three-color frame buffer display devices or monochrome frame buffer display devices.

Color and monochrome frame buffer display devices can be installed in the same workstation, however applications built before X11 release 5 may become confused due to poor initial design of resource files.

To connect two screens, two command line options must be issued. Attaching two screens using only one argument produces a one-way mouse-travel path. You can create a wrap-around mouse path by attaching noncontiguous screen edges. The arguments are disabled on single screen systems.

Nonsensical screen connections are not allowed; the top edge of a particular screen must be connected with the bottom edge of another screen, and the right edge of a particular screen must be connected with the left edge of another screen. Left and right edges cannot be connected to top or bottom edges.

Examples

The following example specifies that screen has a resolution of 100x100 dots per inch and screen has a resolution of 75x70 dots per inch: Xdec -dpi0 100 -dpix1 75 -dpiy1 70

If no screen is specified, the value specified is used for all screens. If the screen resolution is not specified using command line options, a default value based on pixel dimensions and screen size is calculated for each screen.

The following example specifies that black pixels on screen have the hexadecimal value 3a009e005c0 prefixed with a number sign (#) and white pixels on screen are color "wheat" from the X rgb color database. Xdec -bp1 #3a009e005c0 -wp1 wheat For monochrome display devices, values of 0 and 1 are the only valid pixel colors.

To specify the default visual class of a root window on a particular screen, append the screen number (`0`, `1`, or `2`) to the command line option. Possible visual classes are:

StaticGray, StaticColor, PseudoColor, GrayScale, and TrueColor. The following example specifies that the screen root window is a TrueColor visual, and the screen root window is a PseudoColor visual. Xdec -class0 TrueColor -class1 PseudoColor

The following example attaches screen above screen and screen to the right of screen (an L-shaped configuration): Xdec -edge_top0 1 -edge_bottom1 0 -edge_right0 2 -edge_left2 0

The following example is identical to the default state (a horizontal line) with the addition of a wraparound from screen to screen Xdec -edge_left0 2 -edge_right0 1 -edge_left1 0 -edge_right1 2 \ -edge_left2 1 -edge_right2 0

Files

See Also

X(1X), xdm(1), Xserver(1)

X Window System: The Complete Reference to Xlib, X Protocol, ICCCM, XLFD, by Robert W. Scheifler and James Gettys, Second Edition, Digital Press, 1990

"X Window System Toolkit: The Complete Programmer's Guide and Specification, by Paul J. Asente and Ralph R. Swick, Digital Press, 1990

OSF/MOTIF Programmer's Guide and OSF/MOTIF Reference Guide, Open Software Foundation, Prentice-Hall, 1990

NAME

Xhp – cfb-based X window system server for Hewlett-Packard workstations

SYNOPSIS

This cfb-based X server implementation is contributed by Hewlett-Packard as a sample implementation for HP workstations. Its performance on HP workstations will be inferior to the product X servers available from Hewlett-Packard. Not all graphics display devices available from Hewlett-Packard are supported by this implementation.

SUPPORTED GRAPHICS DEVICES

Please refer to the HP catalog or to the apropriate data sheets for the displays. The data that follows relates the use of the product names with their official HP product numbers.

The following graphics display devices are supported by this implementation:

HPA4070A
This graphics device, known as "HCRX", is a 1280x1024 color device that has 8 planes.

HPA4071A
This graphics device, known as "HCRX24", is a 1280x1024 color device that has 24 planes. It is optionally available with a hardware accelerator, in which case the product number is HPA4071A_Z.

HPA1659A
This graphics device, known as "CRX", is a 1280x1024 color device that has 8 planes.

HPA1439A
This graphics device, known as "CRX24", is a 1280x1024 color device that has 24 planes. It is optionally available with a hardware accelerator.

HPA1924A
This graphics device, known as "GRX" is a 1280x1024 grayscale device that has 8 planes.

HPA2269A
This graphics device, known as "Dual CRX" is a 1280x1024 color device that has 8 planes. It implements support for two displays on a single graphics card.

HP710C
This graphics device is the internal graphics support optionally available on the HP9000s710 SPU. It supports 1280x1024 color displays and has 8 planes.

HP710G
This graphics device is the internal graphics support optionally available on the HP9000s710 SPU. It supports 1280x1024 grayscale displays and has 8 planes.

HP710L
This graphics device is the internal graphics support optionally available on the HP9000s710 SPU. It supports 1024x768 color displays and has 8 planes.

HP712
This graphics device is the internal graphics support available on the HP9000s712 SPU. It supports 640x480, 1024x768 or 1280x1024 color displays and has 8 planes.

MULTIPLE SCREEN SUPPORT

This Xhp X11 sample server supports multiple physical screens connected to a single X server. To use this feature, you must have an SPU that allows the installation of a second graphics display card. The file $(LIBDIR)/X*screens is read by the X11 server to determine information about the system screen configuration. You must modify this file to add information for the second graphics display.

$(LIBDIR) is /usr/X11R6/lib/X11 by default.

For a complete description of the X*screens file, refer to the HP-UX manuals, or view the sample file in $(LIBDIR).

24 PLANE SUPPORT FOR HCRX24 AND CRX24

This Xhp X11 sample server supports two modes for the HCRX24 and CRX24 display hardware: 8 plane and 24 plane, with 8 plane being the default. To run the server in 24 plane mode, you must add a depth parameter to the X*screens file. For example:

/dev/crt depth 24

In depth 24 mode, the default visual type is DirectColor.

KEYMAP FILE

This Xhp server loads a keymap that is appropriate for the attached keyboard from the XHPKeymaps file, which resides in $(LIBDIR). The XHPKeymaps file supplied with this Xhp server is a minimal file that supports US English, French, Spanish, German, and Japanese JIS keyboards. If you have some other keyboard, the appropriate keymap may not be contained in the XHPKeymaps file. In this case, if you have access to the Hewlett-Packard product X server, you can copy its keymap file (found in /usr/lib/X11/XHP-Keymaps) to $(LIBDIR).

FAST SCROLLING OPTION

Since scrolling speed is especially slow on this server compared to HP's product server, we have supplied fast scrolling support, using a .o to link into the server. Using HP's fast scrolling is optional. To enable the fast scrolling, set the token "HPFastScrolling" to TRUE in the config/hp.cf file. If you want to use the CFB scrolling module, simply remove the define in config/hp.cf, remake the Makefiles, and recompile.

TRADEMARKS

X Window System is a trademark of X Consortium, Inc.

NAME

Xsun, XsunMono, Xsun24 − Sun server for X Version 11

SYNOPSIS

Xsun [option] ...

DESCRIPTION

Xsun is the server for Version 11 of the X window system on Sun hardware. It will normally be started by the *xdm(1)* daemon or by a script that runs the program *xinit(1)*.

CONFIGURATIONS

XsunMono supports the BW2 monochrome frame buffer. *Xsun* supports the CG2, CG3, CG4, and CG6 8-bit color frame buffers in addition to the BW2 monochrome frame buffer. On Solaris 2.5 it also supports the TCX as an 8-bit color frame buffer. *Xsun24* supports the cgeight 24-bit color frame buffer in addition to the 8-bit color and monochrome frame buffers that *Xsun* supports.

If specific framebuffer device files aren't specified on the command line with the *−dev* switch or in the *XDEVICE* environment variable, the server will search for all installed frame buffers and will use all those that it finds.

Finally, if no specific framebuffers are found, the generic framebuffer interface */dev/fb* is used.

KEYBOARDS

Xsun, Xsun24, and XsunMono support the Type-2, Type-3, and many variations of the Type-4 and Type-5 keyboards.

Type-4 and Type-5 keyboards feature a key labeled *AltGraph* which is a mode-shift key. The mode-shift key is used to generate the symbols painted on the fronts of the keys. The mode-shift key works exactly like the *Shift*, *Control*, *Alt*, and *<Meta>* keys.

The ten function keys on the left side of the Type-5 keyboard may be considered as having L1..L10 painted on their fronts. Shift-AltGraph will cause different keysyms to be generated for some keys, e.g. the Type-5 *SysRq* key.

For compatibility with Sun's X11/NeWS server, the F11 and F12 keys may be made to generate the equivalent X11/NeWS keysyms by using mode-switch.

For backwards compatibility, the normal and mode-shifted keysyms for the ten function keys on the left side of Type-4 and Type-5 keyboards may be swapped via command line option. See *-swapLkeys*.

The X LEDs 1..4 correspond to the NumLock, ScrollLock, Compose, and CapsLock LEDs respectively. Pressing the key once turns the corresponding LED on. Pressing the key again turns the LED off. Turning an LED on or off with e.g. 'xset [-]led [1234]' is equivalent to pressing the corresponding key.

OPTIONS

In addition to the normal server options described in the *Xserver(1)* manual page, *Xsun* accepts the following command line switches:

−ar1 *milliseconds*
: This option specifies amount of time in milliseconds before which a pressed key should begin to autorepeat.

−ar2 *milliseconds*
: This option specifies the interval in milliseconds between autorepeats of pressed keys.

–swapLkeys
Swaps the normal keysyms for the function keys on the left side of Type-4 and Type-5 keyboards with the alternate keysyms, i.e. the keysyms painted on the front of the keys.

–flipPixels
The normal pixel values for white and black are 0 and 1 respectively. When -flipPixels is specified these values are reversed.

–mono When used with the **cgtwo**, this option indicates that the server should emulate a monochrome framebuffer instead of the normal color framebuffer. When used with the **cgfour**, this option indicates that the monochrome screen should be numbered 0 and the color screen numbered 1 (instead of the other way around).

–zaphod This option disables switching between screens by sliding the mouse off the left or right edges. With this disabled, a window manager function must be used to switch between screens.

–debug This option indicates that the server is being run from a debugger, and that it should **not** put its standard input, output and error files into non-blocking mode.

–dev *filename[:filename]...*
This option specifies the colon separated names of the framebuffer device files to be used.

–fbinfo This option indicates that the server should enumerate the available frame buffers that it will use.

ENVIRONMENT

XDEVICE
If present, and if no explicit **-dev** options are given, specifies the (colon separated) list of display devices to use.

SEE ALSO

X(1), Xserver(1), xdm(1), xinit(1)

BUGS

The auto-configuration depends on there being appropriate special files in the */dev* directory for the framebuffers which are to be used. Extra entries can confuse the server. For example, the X/160C in fact has the hardware for a monochrome **bwtwo0** on the CPU board. So if */dev* has a special file for */dev/bwtwo0*, the server will use it, even though there is no monitor attached to the monochrome framebuffer. The server will appear to start, but not to paint a cursor, because the cursor is on the monochrome frame buffer. The solution is to remove the */dev* entries for any device you don't have a monitor for.

There is a bug in pre-FCS operating systems for the Sun-4 which causes the server to crash driving a **cgtwo**.

AUTHORS

U. C. Berkeley
Adam de Boor.

Sun Microsystems
David Rosenthal, Stuart Marks, Robin Schaufler, Mike Schwartz, Frances Ho, Geoff Lee, and Mark Opperman.

MIT Laboratory for Computer Science
Bob Scheifler, Keith Packard, Kaleb Keithley

NAME

constype − print type of Sun console

SYNOPSIS

constype [*device_name*] [**−num**]

DESCRIPTION

The *constype* program writes on the standard output the Sun code for the type of display that the console is. The types output include these:

bw?	Black and White, where ? is 1-4. (eg) 3-50s are bw2
cg?	Colour Graphics display, where ? is 1-4
gp?	Optional Graphics Processor board, where ? is 1-2
ns?	Not Sun display — where ? is A-J

This is useful in determining startup values and defaults for window systems. The *device_name* argument, if given, is the device to examine. If not given, */dev/fb* is used. The **−num** option causes *constype* to follow the type keyword with the numeric value of that type, as returned by the FBIOGATTR or FBIOGTYPE ioctl and defined in fbio.h. This is useful if the type is not recognized by the program.

EXIT STATUS

The program exits with status 0 if it identified a known console type, 1 if the type was unknown, and 2 if the device could not be opened or another error occurred.

BUGS

Not tested on all monitor types

COPYRIGHT

AUTHOR

Doug Moran <moran@ai.sri.com>

NAME

kbd_mode – recover the Sun console keyboard

SYNOPSIS

kbd_mode [-a -e -n -u]

DESCRIPTION

Kbd_mode resets the Sun console keyboard to a rational state.

OPTIONS

The following options are supported, see *kb(4S)* for details:

–a Causes ASCII to be reported.

–e Causes *Firm_events* to be reported.

–n Causes up/down key codes to be reported.

–u Causes undecoded keyboard values to be reported.

SEE ALSO

kb(4S)

NAME

Xvfb − virtual framebuffer X server for X Version 11

SYNOPSIS

Xvfb [option] ...

DESCRIPTION

Xvfb is an X server that can run on machines with no display hardware and no physical input devices. It emulates a dumb framebuffer using virtual memory.

The primary use of this server was intended to be server testing. The mfb or cfb code for any depth can be exercised with this server without the need for real hardware that supports the desired depths. The X community has found many other novel uses for *Xvfb*, including testing clients against unusual depths and screen configurations, doing batch processing with *Xvfb* as a background rendering engine, load testing, as an aid to porting the X server to a new platform, and providing an unobtrusive way to run applications that don't really need an X server but insist on having one anyway.

BUILDING

To build *Xvfb*, put the following in your host.def and remake.

```
#define BuildServer YES /* if you aren't already building other servers */
#define XVirtualFramebufferServer YES
```

OPTIONS

In addition to the normal server options described in the *Xserver(1)* manual page, *Xvfb* accepts the following command line switches:

−screen *screennum WxHxD*
This option creates screen *screennum* and sets its width, height, and depth to W, H, and D respectively. By default, only screen 0 exists and has the dimensions 1280x1024x8.

−pixdepths *list-of-depths*
This option specifies a list of pixmap depths that the server should support in addition to the depths implied by the supported screens. *list-of-depths* is a space-separated list of integers that can have values from 1 to 32.

−fbdir *framebuffer-directory*
This option specifies the directory in which the memory mapped files containing the framebuffer memory should be created. See FILES. This option only exists on machines that have the mmap and msync system calls.

−shmem
This option specifies that the framebuffer should be put in shared memory. The shared memory ID for each screen will be printed by the server. The shared memory is in xwd format. This option only exists on machines that support the System V shared memory interface.

If neither **−shmem** nor **−fbdir** is specified, the framebuffer memory will be allocated with malloc().

−linebias *n*
This option specifies how to adjust the pixelization of thin lines. The value *n* is a bitmask of octants in which to prefer an axial step when the Bresenham error term is exactly zero. See the file Xserver/mi/miline.h for more information. This option is probably only useful to server developers to experiment with the range of line

pixelization possible with the cfb and mfb code.

−blackpixel *pixel-value*, **−whitepixel** *pixel-value*
These options specify the black and white pixel values the server should use.

FILES

The following files are created if the −fbdir option is given.

framebuffer-directory/Xvfb_screen<n>
Memory mapped file containing screen n's framebuffer memory, one file per screen. The file is in xwd format. Thus, taking a full-screen snapshot can be done with a file copy command, and the resulting snapshot will even contain the cursor image.

EXAMPLES

Xvfb :1 -screen 0 1600x1200x32
The server will listen for connections as server number 1, and screen 0 will be depth 32 1600x1200.

Xvfb :1 -screen 1 1600x1200x16
The server will listen for connections as server number 1, will have the default screen configuration (one screen, 1280x1024x8), and screen 1 will be depth 16 1600x1200.

Xvfb -pixdepths 3 27 -fbdir /usr/tmp
The server will listen for connections as server number 0, will have the default screen configuration (one screen, 1280x1024x8), will also support pixmap depths of 3 and 27, and will use memory mapped files in /usr/tmp for the framebuffer.

xwud -in /usr/tmp/Xvfb_screen0
Displays screen 0 of the server started by the preceding example.

SEE ALSO

X(1), Xserver(1), xwd(1), xwud(1), XWDFile.h

AUTHORS

David P. Wiggins, X Consortium, Inc.

NAME

XF68_FBDev - X Window System server for the Linux/m68k Frame Buffer Device

SYNOPSIS

XF68_FBDev [:displaynumber] [option] ...

DESCRIPTION

XF68_FBDev is a generic server for a Linux/m68k Frame Buffer Device. It contains all available drivers (mfb, ipl2p2, ipl2p4, ipl2p8, ilbm, afb and cfb).

The servers use the **/dev/fb*** special device nodes for connection to the hardware. The **/dev/fb*** node to use can be set via an environment variable or if this is absent, **/dev/fb0current** is used.

CONFIGURATIONS

The servers currently support the following pixel formats:

mfb:
Monochrome,

ipl2p{2|4|8}:
Interleaved planes with 2 bytes interleave and 2, 4 or 8 planes (Atari),

ilbm:
Interleaved bitplanes (Amiga),

afb: Normal bitplanes (Amiga),

cfb: Packed Pixels with 8 Bits per Pixel.

OPTIONS

In addition to the normal server options described in the *Xserver(1)* manual page, *XF68_FBDev* accept some more command line switches, as described in the *XFree86(1)* manpage.

SETUP

XFree86 uses a configuration file called **XF86Config** for its initial setup. See the *XF86Config(4/5)* manpage for general details. Here only the *XF68_FBDev* specific parts are explained.

Normally the X server will use videomodes as defined in the **Modes** section, unless you specify the special videomode '**default**'. In that case the X server will use the video mode that is associated with the used framebuffer device node, and mode switching will be disabled.

The environment variable **FRAMEBUFFER** holds the filename for the framebuffer to use. If it isn't set, /dev/fb0current is used as default. /dev/ should contain at least the following file: (major and minor number in parentheses)

fb0current (29,0): The current resolution

FILES

<XRoot>/bin/XF68_FBDev	The X server for a generic Linux/m68k Frame Buffer Device
/etc/XF86Config	Server configuration file
<XRoot>/lib/X11/XF86Config	Server configuration file (secondary location)
/dev/fb*	The files which represents framebuffers and their resolutions

<XRoot>/lib/X11/doc/README.fbdev
Extra documentation about the Frame Buffer Device Note: <XRoot> refers to the root of the X11 install tree.

SEE ALSO

X(1), Xserver(1), XFree86(1), XF86Config(4/5), xf86config(1), xdm(1), xinit(1), fbset(8) Note: <linux> refers to the location of the Linux source tree.

AUTHORS

In addition to the authors of *XFree86* the following people contributed major work to this server:

Martin Schaller
Developed the concept of the Linux/m68k Frame Buffer Device and wrote the support for Atari (Please no mail > 16000 bytes to this address).

Geert Uytterhoeven, *Geert.Uytterhoeven@cs.kuleuven.ac.be*
Wrote the support for Amiga and created the generic server.

Martin Schaller
Geert Uytterhoeven, *Geert.Uytterhoeven@cs.kuleuven.ac.be*
Andreas Schwab, *schwab@issan.informatik.uni-dortmund.de*
Guenther Kelleter, *guenther@Pool.Informatik.RWTH-Aachen.de*
Development and improvement of the frame buffer device specific code.

Gary Henderson *gary@daniver.demon.co.uk*
Wrote Xdaniver, on which afb is based and from which ilbm is derived.

See also the *XFree86(1)* manual page.

BUGS

The support for visuals other than monochrome and pseudo color is incomplete.

The drivers for interleaved and normal planes are slow.

The ilbm code has problems with chunky-to-planar conversions.

CONTACT INFO

XFree86 source is available from the FTP server *ftp.XFree86.org*. Send email to *XFree86@XFree86.org* for details.

NAME

SuperProbe - probe for and identify installed video hardware.

SYNOPSIS

SuperProbe [-verbose] [-no16] [-excl *list*] [-mask10] [-order *list*] [-noprobe *list*] [-bios *base*]
[-no_bios] [-no_dac] [-no_mem] [-info]

DESCRIPTION

SuperProbe is a a program that will attempt to determine the type of video hardware installed in an EISA/ISA/VLB-bus system by checking for known registers in various combinations at various locations (MicroChannel and PCI machines may not be fully supported; many work with the use of the **-no_bios** option). This is an error-prone process, especially on Unix (which usually has a lot more esoteric hardware installed than MS-DOS system do), so SuperProbe may likely need help from the user.

SuperProbe runs on SVR3, SVR4, Linux, 386BSD/FreeBSD/NetBSD, Minix-386, and Mach. It should be trivial to extend it to work on any other Unix-like operating system, and even non-Unix operating systems. All of the OS dependencies are isolated to a single file for each OS.

At this time, *SuperProbe* can identify MDA, Hercules, CGA, MCGA, EGA, VGA, and an entire horde of SVGA chipsets (see the *-info* option, below). It can also identify several HiColor/True-color RAMDACs in use on SVGA boards, and the amount of video memory installed (for many chipsets). It can identify 8514/A and some derivatives, but not XGA, or PGC (although the author intends to add those capabilities). Nor can it identify other esoteric video hardware (like Targa, TIGA, or Microfield boards).

OPTIONS

-verbose *SuperProbe* will be verbose and provide lots of information as it does its work.

-no16 *SuperProbe* will not attempt to use any ports that require 16-bit I/O address decoding. The original ISA bus only specified that I/O ports be decoded to 10 bits. Therefore some old cards (including many 8-bit cards) will mis-decode references to ports that use the upper 6 bits, and may get into funny states because they think that they are being addressed when they are not. It is recommended that this option be used initially if any 8-bit cards are present in the system.

-excl *list* *SuperProbe* will not attempt to access any I/O ports on the specified exclusion list. Some video cards use rather non-standard I/O ports that may conflict with other cards installed in your system. By specifying to *SuperProbe* a list of ports already in use, it will know that there cannot be any video cards that use those ports, and hence will not probe them (which could otherwise confuse your hardware). The exclusion list is specified as a comma-separated list of I/O ports or port ranges. A range is specified as "low-high", and is inclusive. The ports can be specified in decimal, in octal (numbers begin with '0'), or hexadecimal (numbers begin with '0x').

-mask10 This option is used in combination with *-excl.* It tells *SuperProbe* that when comparing an I/O port under test against the exclusion list, the port address should be masked to 10 bits. This is important with older 8-bit cards that only do 10 bit decoding, and for some cheap 16-bit cards as well. This option is simply a less-drastic form of the *-no16* option.

-order *list*
This option specifies which chipsets *SuperProbe* should test, and in which

order. The *list* parameter is a comma-separated list of chipset names. This list overrides the built-in default testing order. To find the list of acceptable names, use the *-info* option described below. Note that items displayed as "Standard video hardware" are not usable with the *-order* option.

-noprobe *list*
This options specifies which chipsets *SuperProbe* should **not** test. The order of testing will either be the default order, or that specified with the *-order* option described above. The *list* parameter is a comma-separated list of chipset names. To find the list of acceptable names, use the *-info* option described below. Note that items displayed as "Standard video hardware" are not usable with the *-noprobe* option.

-bios *base*
This option specifies the base address for the graphics-hardware BIOS. By default, *SuperProbe* will attempt to locate the BIOS base on its own (the normal address is 0xC0000). If it fails to correctly locate the BIOS (an error message will be printed if this occurs), the *-bios* option can be used to specify the base.

-no_bios Disallow reading of the video BIOS and assume that an EGA or later (VGA, SVGA) board is present as the primary video hardware.

-no_dac Skip probing for the RAMDAC type when an (S)VGA is identified.

-no_mem
Skip probing for the amount of installed video memory.

-info *SuperProbe* will print out a listing of all the video hardware that it knows how to identify.

EXAMPLES

To run *SuperProbe* in its most basic and automated form, simply enter:

SuperProbe

Note - you may want to redirect *stdout* to a file when you run *SuperProbe* (especially if your OS does not support Virtual Terminals on the console).

However, if you have any 8-bit cards installed, you should initially run *SuperProbe* as:

SuperProbe -verbose -no16

(the *-verbose* option is included so you can see what *SuperProbe* is skipping).

Finer granularity can be obtained with an exclusion list, for example:

SuperProbe -verbose -excl 0x200,0x220-0x230,0x250

which will not test for any device that use port 0x200, ports 0x220 through 0x230, inclusive, or port 0x250. If you have any 8-bit cards installed, you should add *-mask10* to the list of options.

To restrict the search to Western Digital, Tseng, and Cirrus chipset, run *SuperProbe* as follows:

SuperProbe -order WD,Tseng,Cirrus

BUGS

Probably a lot at this point. Please report any bugs or incorrect identifications to the author.

It is possible that SuperProbe can lock up your machine. Be sure to narrow the search by using the ***-no16*****,** ***-excl*****, and** ***-mask10*** **options provided to keep SuperProbe from conflicting with other installed hardware.**

SEE ALSO

The *vgadoc3.zip* documentation package by Finn Thoegersen, available in the MS-DOS archives of many FTP repositories.

Programmer's Guide to the EGA and VGA Cards, 2nd Ed, by Richard Ferraro.

AUTHOR

David E. Wexelblat <dwex@xfree86.org>
with help from David Dawes <dawes@xfree86.org> and the XFree86 development team.

NAME

XF86Config - Configuration File for XFree86

DESCRIPTION

XFree86 uses a configuration file called **XF86Config** for its initial setup. This configuration file is searched for in the following places:

/etc/XF86Config
<XRoot>/lib/X11/XF86Config.*hostname*
<XRoot>/lib/X11/XF86Config

where <XRoot> refers to the root of the X11 install tree. When an X server is started by a 'root' user, it will first search for an **XF86Config** file in that user's home directory.

This file is composed of a number of sections. Each section has the form:

```
Section "SectionName"
  SectionEntry
  ...
EndSection
```

The section names are:

Files (File pathnames)
Module (Dynamic module loading)
ServerFlags (Server flags)
Keyboard (Keyboard configuration)
Pointer (Pointer configuration)
Monitor (Monitor description)
Device (Graphics device description)
Screen (Screen configuration)
XInput (Extended Input devices configuration)

The **Files** section is used to specify the default font path and the path to the RGB database. These paths can also be set from the command line (see *Xserver(1)*). The entries available for this section are:

FontPath *"path"*
sets the search path for fonts. This path is a comma separated list of directories which the X server searches for font databases. Multiple **FontPath** entries may be specified, and they will be concatenated to build up the fontpath used by the server.

X11R6 allows the X server to request fonts from a font server. A font server is specified by placing a "<trans>/<hostname>:<port_number>" entry into the fontpath. For example, the fontpath "/usr/X11R6/lib/X11/fonts/misc/,tcp/zok:7100" tells the X server to first try to locate the font in the local directory /usr/X11R6/lib/X11/fonts/misc. If that fails, then request the font from the *font server* running on machine zok listening for connections on TCP port number 7100.

RGBPath *"path"*
sets the path name for the RGB color database.

ModulePath *"path"*
sets the search path for dynamic modules. This path is a comma separated list

fo directories which the X server searches for dynamic module loading in the order specified. Multiple **ModulePath** entries may be specified, and they will be concatenated to build the modulepath used by the server.

The **Module** section is used to specify which dynamic modules should be loaded. At present dynamic modules are only used for XInput devices, and are only supported on some systems (currently Linux ELF, FreeBSD 2.x and NetBSD 1.x). The entry available for this section is:

Load *"module"*
This instructs the server to load *"module"*. If the module is not specified with a full pathname, the directories specified in the **ModulePath** are searched. Modules are currently available to support extended input devices. The names of these are:

xf86Elo.so
xf86Jstk.so
xf86Wacom.so
xf86Summa.so

The PEX and XIE extension are also be available as modules on some systems. The names for these are:

pex5.so
xie.so

For an up-to-date listing, check in <XRoot>/lib/modules.

The **ServerFlags** section is used to specify some miscellaneous X server options. The entries available for this section are:

NoTrapSignals
This prevents the X server from trapping a range of unexpected fatal signals and exiting cleanly. Instead, the X server will die and drop core where the fault occurred. The default behaviour is for the X server exit cleanly, but still drop a core file. In general you never want to use this option unless you are debugging an X server problem.

DontZap
This disallows the use of the **Ctrl+Alt+Backspace** sequence. This sequence allows you to terminate the X server. Setting **DontZap** allows this key sequence to be passed to clients.

DontZoom
This disallows the use of the **Ctrl+Alt+Keypad-Plus** and **Ctrl+Alt+Keypad-Minus** sequences. These sequences allows you to switch between video modes. Setting **DontZoom** allows these key sequences to be passed to clients.

AllowNonLocalXvidtune
This allows the xvidtune client to connect from another host. By default non-local connections are not allowed.

DisableVidMode
This disables the parts of the VidMode extension used by the xvidtune client that can be used to change the video modes.

AllowNonLocalModInDev
This allows a client to connect from another host and change keyboard and mouse settings in the running server. By default non-local connections are not

allowed.

DisableModInDev
This disables the parts of the Misc extension that can be used to modify the input device settings dynamically.

AllowMouseOpenFail
This allows the server to start up even if the mouse device can't be opened/initialised.

The **Keyboard** section is used to specify the keyboard input device, parameters and some default keyboard mapping options. The entries available for this section are:

Protocol *"kbd-protocol"*
kbd-protocol may be either **Standard** or **Xqueue**. **Xqueue** is specified when using the event queue driver on SVR3 or SVR4.

AutoRepeat *delay rate*
changes the behavior of the autorepeat of the keyboard. This does not work on all platforms.

ServerNumLock
forces the X server to handle the numlock key internally. The X server sends a different set of keycodes for the numpad when the numlock key is active. This enables applications to make use of the numpad.

LeftAlt *mapping*
RightAlt *mapping*
AltGr *mapping*
ScrollLock *mapping*
RightCtl *mapping*
Allows a default mapping to be set for the above keys (note that **AltGr** is a synonym for **RightAlt**). The values that may be specified for *mapping* are:

Meta
Compose
ModeShift
ModeLock
ScrollLock
Control

The default mapping when none of these options are specified is:

LeftAlt	Meta
RightAlt	Meta
ScrollLock	Compose
RightCtl	Control

XLeds *led* **...**
makes *led* available for clients instead of using the traditional function (Scroll Lock, Caps Lock & Num Lock). *led* is a list of numbers in the range 1 to 3.

VTSysReq
enables the SYSV-style VT switch sequence for non-SYSV systems which support VT switching. This sequence is Alt-SysRq followed by a function key (Fn). This prevents the X server trapping the keys used for the default VT switch sequence.

VTInit *"command"*
Runs *command* after the VT used by the server has been opened. The command string is passed to "/bin/sh -c", and is run with the real user's id with stdin and stdout set to the VT. The purpose of this option is to allow system dependent VT initialisation commands to be run. One example is a command to disable the 2-key VT switching sequence which is the default on some systems.

XkbDisable
Turns the XKEYBOARD extension off, equivalent to using the -kb command line option.

XkbRules *"rulesfile"*
XkbModel *"model"*
XkbLayout *"layout"*
XkbVariant *"variant"*
XkbOptions *"optionlist"*
These specify the definitions which are used to determine which XKEYBOARD components to use. The optionlist, should be a comma separated list of options. The default mappings for each these are:

XkbRules	"xfree86"
XkbModel	"pc101"
XkbLayout	"us"
XkbVariant	""
XkbOptions	""

This is the preferred method of specifying the keyboard configuration, however, you can also specify the components directly with:

XkbKeymap *"keymap"*
XkbKeycodes *"keycodes"*
XkbTypes *"types"*
XkbCompat *"compat"*
XkbSymbols *"symbols"*
XkbGeometry *"geometry"*
If you specify only some of the components, the remaining components will use these default values:

XkbKeymap	none
XkbKeycodes	"xfree86"
XkbTypes	"default"
XkbCompat	"default"
XkbSymbols	"us(pc101)"
XkbGeometry	"pc"

The **Pointer** section is used to specify the pointer device and parameters. The entries available for this section are:

Protocol *"protocol-type"***"**
specifies the pointer device protocol type. The protocol types available are:

BusMouse
Logitech
Microsoft

MMSeries
Mouseman
MouseSystems
PS/2
MMHitTab
GlidePoint
IntelliMouse
Xqueue
OSMouse

One should specify **BusMouse** for the Logitech bus mouse. Also, many newer Logitech serial mice use either the **Microsoft** or **MouseMan** protocol. **Xqueue** should be specified here if it was used in the **Keyboard** section. **OSMouse** refers to the event-driver mouse interface available on SCO's SVR3, and the mouse interface provided for OS/2. This may optionally be followed by a number specifying the number of buttons the mouse has.

If you have a mouse connected to a PS/2 port, you should specify **PS/2**, regardless of the type of mouse you are using.

Device *"pointer-dev"*
specifies the device the server should open for pointer input (eg, **/dev/tty00** or **/dev/mouse**). A device should not be specified when using the **Xqueue** or **OSMouse** protocols.

Port *"pointer-dev"*
is an alternate form of the **Device** entry.

BaudRate *rate*
sets the baudrate of the serial mouse to *rate*. For mice that allow dynamic speed adjustments (like older Logitechs) the baudrate is changed in the mouse. Otherwise the rate is simply set on the computer's side to allow mice with non-standard rates (the standard rate is 1200). For 99% of mice you should not set this to anything other than the default (1200).

Emulate3Buttons
enables the emulation of the third mouse button for mice which only have two physical buttons. The third button is emulated by pressing both buttons simultaneously.

Emulate3Timeout *timeout*
sets the time (in milliseconds) that the server waits before deciding if two buttons were pressed "simultaneously" when 3 button emulation is enabled. The default timeout is 50ms.

ChordMiddle
handles mice which send left+right events when the middle button is used (like some Logitech Mouseman mice).

SampleRate *rate*
sets the number of motion/button-events the mouse sends per second. This is currently only supported for some Logitech mice.

ClearDTR
This option clears the DTR line on the serial port used by the mouse. This option is only valid for a mouse using the **MouseSystems** protocol. Some dual-protocol mice require DTR to be cleared to operate in MouseSystems

mode. Note, in versions of XFree86 prior to 2.1, this option also cleared the RTS line. A separate **ClearRTS** option has since been added for mice which require this.

ClearRTS
This option clears the RTS line on the serial port used by the mouse. This option is only valid for a mouse using the **MouseSystems** protocol. Some dual-protocol mice require both DTR and RTS to be cleared to operate in MouseSystems mode. Both the **ClearDTR** and **ClearRTS** options should be used for such mice.

The **Monitor** sections are used to define the specifications of a monitor and a list of video modes suitable for use with a monitor. More than one **Monitor** section may be present in an XF86Config file. The entries available for this section are:

Identifier *"ID string"*
This specifies a string by which the monitor can be referred to in a later **Screen** section. Each **Monitor** section should have a unique ID string.

VendorName *"vendor"*
This optional entry specifies the monitor's manufacturer.

ModelName *"model"*
This optional entry specifies the monitor's model.

HorizSync *horizsync-range*
gives the range(s) of horizontal sync frequencies supported by the monitor. *horizsync-range* may be a comma separated list of either discrete values or ranges of values. A range of values is two values separated by a dash. By default the values are in units of kHz. They may be specified in MHz or Hz if **MHz** or **Hz** is added to the end of the line. The data given here is used by the X server to determine if video modes are within the specifications of the monitor. This information should be available in the monitor's handbook.

VertRefresh *vertrefresh-range*
gives the range(s) of vertical refresh frequencies supported by the monitor. *vertrefresh-range* may be a comma separated list of either discrete values or ranges of values. A range of values is two values separated by a dash. By default the values are in units of Hz. They may be specified in MHz or kHz if **MHz** or **kHz** is added to the end of the line. The data given here is used by the X server to determine if video modes are within the specifications of the monitor. This information should be available in the monitor's handbook.

Gamma *gamma-value(s)*
This is an optional entry that can be used to specify the gamma correction for the monitor. It may be specified as either a single value or as three separate RGB values. Not all X servers are capable of using this information.

Mode *"name"*
indicates the start of a multi-line video mode description. The mode description is terminated with an **EndMode** line. The mode description consists of the following entries:

DotClock *clock*
is the dot clock rate to be used for the mode.

HTimings *hdisp hsyncstart hsyncend htotal*
specifies the horizontal timings for the mode.

VTimings *vdisp vsyncstart vsyncend vtotal*
specifies the vertical timings for the mode.

Flags *"flag" ...*
specifies an optional set of mode flags. **"Interlace"** indicates that the mode is interlaced. **"DoubleScan"** indicates a mode where each scanline is doubled. **"+HSync"** and **"-HSync"** can be used to select the polarity of the HSync signal. **"+VSync"** and **"-VSync"** can be used to select the polarity of the VSync signal. **"Composite"**, can be used to specify composite sync on hardware where this is supported. Additionally, on some hardware, **"+CSync"** and **"-CSync"** may be used to select the composite sync polarity.

HSkew *hskew*
specifies the number of pixels (towards the right edge of the screen) by which the display enable signal is to be skewed. Not all servers use this information. This option might become necessary to override the default value supplied by the server (if any). "Roving" horizontal lines indicate this value needs to be increased. If the last few pixels on a scan line appear on the left of the screen, this value should be decreased.

Modeline *"name" mode-description*
is a single line format for specifying video modes. The *mode-description* is in four sections, the first three of which are mandatory. The first is the pixel clock. This is a single number specifying the pixel clock rate for the mode. The second section is a list of four numbers specifying the horizontal timings. These numbers are the *hdisp*, *hsyncstart*, *hsyncend*, *htotal*. The third section is a list of four numbers specifying the vertical timings. These numbers are *vdisp*, *vsyncstart*, *vsyncend*, *vtotal*. The final section is a list of flags specifying other characteristics of the mode. **Interlace** indicates that the mode is interlaced. **DoubleScan** indicates a mode where each scanline is doubled. **+HSync** and **−HSync** can be used to select the polarity of the HSync signal. **+VSync** and **−VSync** can be used to select the polarity of the VSync signal. **Composite** can be used to specify composite sync on hardware where this is supported. Additionally, on some hardware, **+CSync** and **-CSync** may be used to select the composite sync polarity. The **HSkew** option mentioned above can also be used here.

The **Device** sections are used to define a graphics device (video board). More than one **Device** section may be present in an XF86Config file. The entries available for this section are:

Identifier *"ID string"*
This specifies a string by which the graphics device can be referred to in a later **Screen** section. Each **Device** section should have a unique ID string.

VendorName *"vendor"*
This optional entry specifies the graphics device's manufacturer.

BoardName *"model"*
This optional entry specifies the name of the graphics device.

Chipset *"chipset-type"*
This optional entry specifies the chipset used on the graphics board. In most

cases this entry is not required because the X servers will probe the hardware to determine the chipset type.

Ramdac *"ramdac-type"*
This optional entry specifies the type of RAMDAC used on the graphics board. This is only used by a few of the X servers, and in most cases it is not required because the X servers will probe the hardware to determine the RAMDAC type where possible.

DacSpeed *speed*
This optional entry specifies the RAMDAC speed rating (which is usually printed on the RAMDAC chip). The speed is in MHz. This is only used by a few of the X servers, and only needs to be specified when the speed rating of the RAMDAC is different from the default built in to the X server.

Clocks *clock ...*
specifies the dotclocks that are on your graphics board. The clocks are in MHz, and may be specified as a floating point number. The value is stored internally to the nearest kHz. The ordering of the clocks is important. It must match the order in which they are selected on the graphics board. Multiple **Clocks** lines may be specified. For boards with programmable clock chips, the **ClockChip** entry should be used instead of this. A **Clocks** entry is not mandatory for boards with non-programmable clock chips, but is highly recommended because it prevents the clock probing phase during server startup. This clock probing phase can cause problems for some monitors.

ClockChip *"clockchip-type"*
This optional entry is used to specify the clock chip type on graphics boards which have a programmable clock generator. Only a few X servers support programmable clock chips. For details, see the appropriate X server manual page.

ClockProg *"command"* **[***textclock***]**
This optional entry runs *command* to set the clock on the graphics board instead of using the internal code. The command string must consist of the full pathname (and no flags). When using this option, and no **Clocks** entry is specified, it is assumed that the card has a fully programmable clock generator; for a card with a set of preset clocks a **Clocks** entry is required to specify which clock values are to be made available to the server (up to 128 clocks may be specified). The optional *textclock* value is used to tell the server that *command* must be run to restore the textmode clock at server exit (or when VT switching). *textclock* must match one of the values in the **Clocks** entry. This parameter is required when the clock used for text mode is a programmable clock.

The command is run with the real user's id with stdin and stdout set to the graphics console device. Two arguments are passed to the command. The first is the clock frequency in MHz as a floating point number and the second is the index of the clock in the **Clocks** entry. The command should return an exit status of 0 when successful, and something in the range 1−254 otherwise.

The command is run when the initial graphics mode is set and when changing screen resolution with the hot-key sequences. If the program fails at initialisation the server exits. If it fails during a mode switch, the mode switch is aborted but the server keeps running. It is assumed that if the command fails the clock has not been changed.

Option *"optionstring"*
This optional entry allows the user to select certain options provided by the drivers. Multiple **Option** entries may be given. The supported values for *optionstring* are given in the appropriate X server manual pages and/or the chipset-specific README files.

VideoRam *mem*
This optional entry specifies the amount of videoram that is installed on the graphics board. This is measured in kBytes. In most cases this is not required because the X server probes the graphics board to determine this quantity.

BIOSBase *baseaddress*
This optional entry specifies the base address of the video BIOS for the VGA board. This address is normally 0xC0000, which is the default the X servers will use. Some systems, particularly those with on-board VGA hardware, have the BIOS located at an alternate address, usually 0xE0000. If your video BIOS is at an address other than 0xC0000, you must specify the base address in the XF86Config file. Note that some X servers don't access the BIOS at all, and those which do only use the BIOS when searching for information during the hardware probe phase.

MemBase *baseaddress*
This optional entry specifies the memory base address of a graphics board's linear frame buffer. This entry is only used by a few X servers, and the interpretation of this base address may be different for different X servers. Refer to the appropriate X server manual page for details.

IOBase *baseaddress*
This optional entry specifies the IO base address. This entry is only used for a few X servers. Refer to the appropriate X server manual page for details.

DACBase *baseaddress*
This optional entry specifies the DAC base address. This entry is only used for a few X servers. Refer to the appropriate X server manual page for details.

POSBase *baseaddress*
This optional entry specifies the POS base address. This entry is only used for a few X servers. Refer to the appropriate X server manual page for details.

COPBase *baseaddress*
This optional entry specifies the coprocessor base address. This entry is only used for a few X servers. Refer to the appropriate X server manual page for details.

VGABase *baseaddress*
This optional entry specifies the VGA memory base address. This entry is only used for a few X servers. Refer to the appropriate X server manual page for details.

Instance *number*
This optional entry specifies the instance (which indicates if the chip is integrated on the motherboard or on an expansion card). This entry is only used for a few X servers. Refer to the appropriate X server manual page for details.

Speedup *"selection"*
This optional entry specifies the selection of speedups to be enabled. This entry

is only used for a few X servers. Refer to the appropriate X server manual page for details.

S3MNAdjust *M N*
This optional entry is specific to the S3 X server. For details, refer to the *XF86_S3(1)* manual page.

S3MClk *clock*
This optional entry is specific to the S3 X server. For details, refer to the *XF86_S3(1)* manual page.

S3RefClock *clock*
This optional entry is specific to the S3 X server. For details, refer to the *XF86_S3(1)* manual page.

The **Screen** sections are used to specify which graphics boards and monitors will be used with a particular X server, and the configuration in which they are to be used. The entries available for this section are:

Driver *"driver-name"*
Each **Screen** section must begin with a **Driver** entry, and the *driver-name* given in each **Screen** section must be unique. The driver name determines which X server (or driver type within an X server when an X server supports more than one head) reads and uses a particular **Screen** section. The driver names available are:

Accel
Mono
SVGA
VGA2
VGA16

Accel is used by all the accelerated X servers (see *XF86_Accel(1)*). **Mono** is used by the non-VGA mono drivers in the 2-bit and 4-bit X servers (see *XF86_Mono(1)* and *XF86_VGA16(1)*). **VGA2** and **VGA16** are used by the VGA drivers in the 2-bit and 4-bit X servers respectively. **SVGA** is used by the XF86_SVGA X server.

Device *"device-id"*
specifies which graphics device description is to be used.

Monitor *"monitor-id"*
specifies which monitor description is to be used.

DefaultColorDepth *bpp-number*
specifies which color depth the server should use, when no -bpp command line parameter was given.

ScreenNo *scrnum*
This optional entry overrides the default screen numbering in a multi-headed configuration. The default numbering is determined by the ordering of the **Screen** sections in the *XF86Config* file. To override this, all relevant **Screen** sections must have this entry specified.

BlankTime *time*
sets the inactivity timeout for the blanking phase of the screensaver. *time* is in minutes, and the default is 10. This is equivalent to the Xserver's '-s' flag, and the value can be changed at run-time with *xset(1)*.

StandbyTime *time*
sets the inactivity timeout for the "standby" phase of DPMS mode. *time* is in minutes, the default is 20, and it can be changed at run-time with *xset(1)*. This is only suitable for VESA DPMS compatible monitors, and is only supported currently by some Xservers. The "power_saver" Option must be set for this to be enabled.

SuspendTime *time*
sets the inactivity timeout for the "suspend" phase of DPMS mode. *time* is in minutes, the default is 30, and it can be changed at run-time with *xset(1)*. This is only suitable for VESA DPMS compatible monitors, and is only supported currently by some Xservers. The "power_saver" Option must be set for this to be enabled.

OffTime *time*
sets the inactivity timeout for the "off" phase of DPMS mode. *time* is in minutes, the default is 40, and it can be changed at run-time with *xset(1)*. This is only suitable for VESA DPMS compatible monitors, and is only supported currently by some Xservers. The "power_saver" Option must be set for this to be enabled.

SubSection "Display"
This entry is a subsection which is used to specify some display specific parameters. This subsection is terminated by an **EndSubSection** entry. For some X servers and drivers (those requiring a list of video modes) this subsection is mandatory. For X servers which support multiple display depths, more than one **Display** subsection may be present. When multiple **Display** subsections are present, each must have a unique **Depth** entry. The entries available for the **Display** subsection are:

Depth *bpp*
This entry is mandatory when more than one **Display** subsection is present in a **Screen** section. When only one **Display** subsection is present, it specifies the default depth that the X server will run at. When more than one **Display** subsection is present, the depth determines which gets used by the X server. The subsection used is the one matching the depth at which the X server is run at. Not all X servers (or drivers) support more than one depth. Permitted values for *bpp* are 8, 15, 16, 24 and 32. Not all X servers (or drivers) support all of these values. *bpp* values of 24 and 32 are treated equivalently by those X servers which support them.

Weight *RGB*
This optional entry specifies the relative RGB weighting to be used for an X server running at 16bpp. This may also be specified from the command line (see *XFree86(1)*). Values supported by most 16bpp X servers are **555** and **565**. For further details, refer to the appropriate X server manual page.

Virtual *xdim ydim*
This optional entry specifies the virtual screen resolution to be used. *xdim* must be a multiple of either 8 or 16 for most colour X servers, and a multiple of 32 for the monochrome X server. The given value will be rounded down if this is not the case. For most X servers, video modes which are too large for the specified virtual size will be rejected. If this entry is not present, the virtual screen resolution will be set to accommodate all the

valid video modes given in the **Modes** entry. Some X servers do not support this entry. Refer to the appropriate X server manual pages for details.

ViewPort *x0 y0*
This optional entry sets the upper left corner of the initial display. This is only relevant when the virtual screen resolution is different from the resolution of the initial video mode. If this entry is not given, then the initial display will be centered in the virtual display area.

Modes *"modename" ...*
This entry is mandatory for most X servers, and it specifies the list of video modes to use. The video mode names must correspond to those specified in the appropriate **Monitor** section. Most X servers will delete modes from this list which don't satisfy various requirements. The first valid mode in this list will be the default display mode for startup. The list of valid modes is converted internally into a circular list. It is possible to switch to the next mode with **Ctrl+Alt+Keypad-Plus** and to the previous mode with **Ctrl+Alt+Keypad-Minus**.

InvertVCLK *"modename"* 0|1
This optional entry is specific to the S3 server only. It may be used to change the default VCLK invert/non-invert state for individual modes. If *"modename"* is "*" the setting applies to all modes unless unless overridden by later entries.

EarlySC *"modename"* 0|1
This optional entry is specific to the S3 server only. It may be used to change the default EarlySC setting for individual modes. This setting can affect screen wrapping. If *"modename"* is "*" the setting applies to all modes unless unless overridden by later entries.

BlankDelay *"modename" value1 value2*
This optional entry is specific to the S3 server only. It may be used to change the default blank delay settings for individual modes. This can affect screen wrapping. *value1* and *value2* must be integers in the range 0−7. If *"modename"* is "*" the setting applies to all modes unless unless overridden by later entries.

Visual *"visual-name"*
This optional entry sets the default root visual type. This may also be specified from the command line (see *Xserver(1)*). The visual types available for 8bpp X servers are (default is **PseudoColor**):

StaticGray
GrayScale
StaticColor
PseudoColor
TrueColor
DirectColor

The visual type available for the 16bpp and 32bpp X servers is **TrueColor**.

The visual type available for the 1bpp X server is **StaticGray**.

The visual types available for the 4bpp X server are (default is **StaticColor**):

StaticGray
GrayScale
StaticColor
PseudoColor

Option *"optionstring"*
This optional entry allows the user to select certain options provided by the drivers. Multiple **Option** entries may be given. The supported values for *optionstring* are given in the appropriate X server manual pages and/or the chipset-specific README files.

Black *red green blue*
This optional entry allows the ''black'' colour to be specified. This is only supported with the VGA2 driver in the XF86_Mono server (for details see *XF86_Mono(1)*).

White *red green blue*
This optional entry allows the ''white'' colour to be specified. This is only supported with the VGA2 driver in the XF86_Mono server (for details see *XF86_Mono(1)*).

The optional **XInput** section is used to specify configuration options for the extended input devices. For some OSs, the extended device support is dynamically loaded, and in this case you need to specify which Modules to load in the **Module** section (this is documented above). Each extended device has its own subsection. To enable an extended device the corresponding subsection must appear. The subsections names are:

Joystick (only on supported systems ie. Linux, FreeBSD and NetBSD)
WacomStylus (stylus of a Wacom tablet)
WacomEraser (eraser of a Wacom tablet)
WacomCursor (cursor of a Wacom tablet)
Elographics (Elographics touchscreen)
SummaSketch (SummaSketch tablet)
Mouse (Mouse)

The **Joystick** subsection supports the following entries:

Port *"path"*
sets the path to the special file which represents the device driver.

DeviceName *"name"*
sets the name of the X device.

TimeOut *timeout*
sets the time (in milliseconds) between two polls of the device driver. The value given here may be overriden by the Operating System's joystick driver.

MaximumXPosition *value*
sets the maximum X value reported by the device driver.

MininimumXPosition *value*
sets the minimum X value reported by the device driver.

MaximumYPosition *value*
sets the maximum Y value reported by the device driver.

MinimumYPosition *value*
sets the minimum Y value reported by the device driver.

CenterX *value*
sets the X center reported by the device driver when the joystick is idle. If this value is omitted, it is assumed that the joystick is centered when it is first enabled.

CenterY *value*
sets the Y center reported by the device driver when the joystick is idle. If this value is omitted, it is assumed that the joystick is centered when it is first enabled.

Delta *value*
sets the maximum value reported to the X server. i.e. coordinates will be incremented of *(+/−)value/2* at maximum deflection. This determines the sensitivity.

AlwaysCore
enables the sharing of the core pointer. When this feature is enabled you cannot put the device in extended mode (i.e. sending extended events). You can also use the latest integer feedback to control this feature. When the value of the feedback is zero, the feature is disabled. The feature is enabled for any other value.

Multiple instances of the Wacom devices can cohabit. It can be useful to define multiple devices with different active zones. The **WacomStylus**, **WacomEraser** and **WacomCursor** subsections support the following entries:

Port *"path"*
sets the path to the special file which represents serial line where the tablet is plugged. You have to specify it for each subsection with the same value if you want to have multiple devices with the same tablet.

DeviceName *"name"*
sets the name of the X device.

Suppress *number*
sets the position increment under which not to transmit coordinates. This entry must be specified only in the first Wacom subsection if you have multiple devices for one tablet.

Mode *Relative|Absolute*
sets the mode of the device.

TiltMode
enables tilt report if your tablet supports it (ROM version 1.4 and above). If this is enabled, multiple devices at the same time will not be reported.

HistorySize *number*
sets the motion history size. By default the value is zero.

AlwaysCore
enables the sharing of the core pointer. When this feature is enabled you cannot put the device in extended mode (i.e. sending extended events). You can also use the latest integer feedback to control this feature. When the value of the feedback is zero, the feature is disabled. The feature is enabled for any other value.

TopX *number*
X coordinate of the top corner of the active zone.

TopY *number*
Y coordinate of the top corner of the active zone.

BottomX *number*
X coordinate of the bottom corner of the active zone.

BottomY *number*
Y coordinate of the bottom corner of the active zone.

KeepShape
When this option is enabled, the active zone begins according to TopX and TopY. The bottom corner is calculated to keep shapes ie. the ratio width/heigth of the active zone is calculated to have the same ratio as the one of the screen.

The **Elographics** subsection support the following entries:

Port *"path"*
sets the path to the special file which represents the device driver.

DeviceName *"name"*
sets the name of the X device.

MaximumXPosition *position*
sets the maximum X position reported by the touchscreen.

MinimumXPosition *position*
sets the minimum X position reported by the touchscreen.

MaximumYPosition *position*
sets the maximum Y position reported by the touchscreen.

MinimumYPosition *position*
sets the minimum Y position reported by the touchscreen.

ScreenNo *number*
sets the screen number where the touchscreen is connected.

UntouchDelay *value*
sets the delay (in tens of milliseconds) after which the device considers that an untouch occurs.

ReportDelay *value*
sets the delay (in ten of milliseconds) between two reports of positions.

AlwaysCore
enables the sharing of the core pointer. When this feature is enabled you cannot put the device in extended mode (i.e. sending extended events). You can also use the latest integer feedback to control this feature. When the value of the feedback is zero, the feature is disabled. The feature is enabled for any other value.

The **SummaSketch** subsection support the following entries:

Port *"path"*
sets the path to the special file which represents the device driver.

DeviceName *"name"*
sets the name of the X device.

Mode *Relative|Absolute*
sets the mode of the device.

Cursor *Stylus|Puck*
sets the cursor type, stylus or 4 button puck.

Increment *value*
sets the maximum change in coordinates before new report.

HistorySize *number*
sets the motion history size. By default the value is zero.

AlwaysCore
enables the sharing of the core pointer. When this feature is enabled you cannot put the device in extended mode (i.e. sending extended events). You can also use the latest integer feedback to control this feature. When the value of the feedback is zero, the feature is disabled. The feature is enabled for any other value.

The **Mouse** subsection support the same entries as the standard **Pointer** section, plus the following:

DeviceName *"name"*
sets the name of the X device.

AlwaysCore
enables the sharing of the core pointer. When this feature is enabled you cannot put the device in extended mode (i.e. sending extended events). You can also use the latest integer feedback to control this feature. When the value of the feedback is zero, the feature is disabled. The feature is enabled for any other value.

For an example of an XF86Config file, see the file installed as <XRoot>/lib/X11/XF86Config.eg.

FILES

/etc/XF86Config
<XRoot>/lib/X11/XF86Config.*hostname*
<XRoot>/lib/X11/XF86Config

Note: <XRoot> refers to the root of the X11 install tree.

SEE ALSO

X(1), Xserver(1), XFree86(1), XF86_SVGA(1), XF86_VGA16(1), XF86_Mono(1), XF86_S3(1), XF86_8514(1), XF86_Mach8(1), XF86_Mach32(1), XF86_P9000(1), XF86_AGX(1), XF86_W32(1).

AUTHORS

Refer to the *XFree86(1)* manual page.

NAME

XF86Setup - Graphical configuration utility for XFree86

SYNOPSIS

XF86Setup [-sync] [-name *appname*] [-nodialog] [-- *arg* ...] **XF86Setup** [-sync] [-name *appname*] [-script] [-display *display*] [-geometry *geometry*] *filename* [[--] *arg* ...]

DESCRIPTION

XF86Setup is normally used to either perform the initial setup of the XFree86 X servers or to make adjustments to the existing configuration.

When used to initially configure the XFree86 servers, it will start the VGA 16 server and then allow configuration settings to be entered. Once that is completed the appropriate server for the hardware will be started and the configuration settings will be saved.

For a more complete description see the *Quick-Start Guide to XFree86 Setup*.

When run from within an already running server environment (e.g. from an xterm window), *XF86Setup* can be used to make adjustments to the current server configuration.

Note that when the existing XF86Config file is read, it is used to set the default values for various settings, but the complete contents of the existing file are not preserved. For example, when the new XF86Config file is written, it will *not* contain any additional comment lines from the original file.

When a filename is specified, it is interpreted as a script file containing Tcl/Tk commands to run. If the filename contains a slash, it is assumed that it specifies the path to the file to execute, otherwise the file is searched for in the user's PATH, unless the **-script** option is given, in which case, it is expected to exist in the *scripts* subdirectory of the *XF86Setup* library directory.

The result is entirely dependant on the contents of the file.

ARGUMENTS

-sync Turn on synchronization for all communication with an X server.

-name *appname* Use *appname* as the window name.

-display *display* Specify the display to talk to.

-nodialog Do not use the Dialog program for text mode user interaction. Normally *XF86Setup* will use the Dialog program, if it is found in the user's PATH, otherwise a simple text interface is used.

-geometry *geomspec*
Specify the initial geometry for the window.

-script Look for the specified filename in the *scripts* directory, instead of searching the user's PATH (if the filename doesn't specify a path).

ENVIRONMENT

PATH Used to search for the location of the Dialog program

XWINHOME If set, indicates the parent directory of X bin and lib directories.

XF86SETUPLIB
If set, overrides the default location of the library directory (normally <XRoot>/lib/X11/XF86Setup).

TMPDIR Directory in which to store temporary files (defaults to /tmp)

FILES

<XRoot>/lib/X11/XF86Config
or /etc/XF86Config
Can optionally be read to set default values for configuration settings

SEE ALSO

Quick-Start Guide to XFree86 Setup
Xserver(1), XFree86(1), XF86Config(4/5), xvidtune(1), xdm(1), xf86config(1), xinit(1), XF86Misc(3), XF86VidMode(3), xmseconfig(1), dialog(1)

AUTHOR

Joe Moss, *joe@XFree86.org*

BUGS

Does not support all possible configurations, for example, graphics tablets must be configured by editing the XF86Config file manually.

NAME

xmseconfig - Graphical mouse configuration utility

SYNOPSIS

xmseconfig [-sync] [-display *display*] [-geometry *geometry*]

DESCRIPTION

xmseconfig enables the user to make changes to the server's mouse configuration. Changes apply only to the currently running server and are not saved to a configuration file as they are with **XF86Setup**.

ARGUMENTS

-sync Turn on synchronization for all communication with an X server.

-display *display* Specify the display to talk to.

-geometry *geomspec*
Specify the initial geometry for the window.

ENVIRONMENT

XWINHOME If set, indicates the parent directory of X bin and lib directories.

XF86SETUPLIB
If set, overrides the default location of the library directory (normally <XRoot>/lib/X11/XF86Setup).

FILES

<XRoot>/lib/X11/XF86Config
or /etc/XF86Config
Used to determine the link to the mouse device

SEE ALSO

Quick-Start Guide to XFree86 Setup
XF86Setup(1), XFree86(1), XF86Config(4/5), xf86config(1), XF86Misc(3)

AUTHOR

Joe Moss, *joe@XFree86.org*

NOTES

If the program is unable to read the mouse configuration from the existing XF86Config file or if the user is not the superuser, changes to the mouse device are disallowed.

NAME

XF86_Accel - accelerated X Window System servers for UNIX on x86 and Alpha platforms with an S3, Mach8, Mach32, Mach64, P9000, AGX, ET4000/W32, ET6000, I128, TGA, or 8514/A accelerator board

SYNOPSIS

XF86_S3 [:displaynumber] [option] ... **XF86_S3V** [:displaynumber] [option] ... **XF86_Mach8** [:displaynumber] [option] ... **XF86_Mach32** [:displaynumber] [option] ... **XF86_Mach64** [:displaynumber] [option] ... **XF86_P9000** [:displaynumber] [option] ... **XF86_AGX** [:displaynumber] [option] ... **XF86_W32** [:displaynumber] [option] ... **XF86_I128** [:displaynumber] [option] ... **XF86_TGA** [:displaynumber] [option] ... **XF86_8514** [:displaynumber] [option] ...

DESCRIPTION

XF86_S3 is an 8-bit PseudoColor, 16-bit TrueColor and 32-bit TrueColor server for S3 graphic accelerator boards. Note, 16-bit and 32-bit operation is not supported on all S3 accelerator boards. Refer to *README.S3* for details of which boards are supported at which depths.

XF86_S3V is an 8-bit PseudoColor and 16-bit TrueColor server for S3 ViRGE and ViRGE/VX graphic accelerator boards.

XF86_Mach8 is an 8-bit PseudoColor server for ATI Mach8 graphic accelerator boards.

XF86_Mach32 is an 8-bit PseudoColor and 16-bit TrueColor server for ATI Mach32 graphic accelerator boards. Note, 16-bit operation is not supported on all Mach32 accelerator boards. Refer to *README.Mach32* for details of which RAMDACs are supported at which depths.

XF86_Mach64 is an 8-bit PseudoColor, 16-bit TrueColor, and 32-bit TrueColor server for ATI Mach64 graphic accelerator boards. Note, 16-bit and 32-bit operation is not supported for all RAMDACs. Refer to *README.Mach64* for details of which RAMDACs are supported at which depths.

XF86_P9000 is an 8-bit PseudoColor, 16-bit TrueColor, and 32-bit TrueColor server for Weitek Power 9000 (P9000) graphic accelerator boards.

XF86_AGX is an 8-bit PseudoColor and 16-bit TrueColor server for AGX/XGA graphic accelerator boards.

XF86_W32 is an 8-bit PseudoColor server for ET4000/W32, ET4000/W32i, ET4000/W32p and ET6000 graphic accelerator boards. For 16, 24 and 32 bits per pixel support on the ET6000, try the XF86_SVGA server.

XF86_I128 is an server for Number Nine Imagine 128 graphic accelerator boards.

XF86_TGA is an server for Digital's 21030 (TGA) chipset. This server is available only for Alpha platforms.

XF86_8514 is an 8-bit PseudoColor server for 8514/A graphic accelerator boards.

These are derived from the *X386* server provided with X11R5, and from the *X8514* server developed by Kevin Martin <martin@cs.unc.edu>.

CONFIGURATIONS

The servers support the following chipsets:

XF86_S3: 86C911, 86C924, 86C801, 86C805, 86C805i, 86C928, 86C928-P, 86C732 (Trio32), 86C764 (Trio64), 86C765 (Trio64V+),

	86C864, 86C868, 86C964, 86C968
XF86_S3V:	86C325 (ViRGE), 86C375 (ViRGE/DX), 86C385 (ViRGE/GX), 86C988 (ViRGE/VX)
XF86_Mach8:	ATI Mach8, ATI Mach32
XF86_Mach32:	ATI Mach32
XF86_Mach64:	ATI Mach64 (GX, CT, ET, VT, GT)
XF86_P9000:	Diamond Viper VLB, Diamond Viper PCI, Orchid P9000, and some clones (Weitek P9000)
XF86_AGX:	AGX-010, AGX-014, AGX-015, AGX-016, XGA-1, XGA-2
XF86_W32:	ET4000/W32, ET4000/W32i, ET3000/W32p, ET6000
XF86_I128:	Number Nine Imagine 128 series 1 and 2
XF86_TGA:	DEC 21030
XF86_8514:	IBM 8514/A and true clones

For S3 virtual resolutions up to (approximately) 1152x800 are supported, using (up to) 1Mb of display memory (the S3 uses an internal width of 1280 for some older chips, hence 1Mb can't support 1152x900). Physical resolutions up to 1280x1024 (1600x1280 on some cards) are possible using 2Mb or more of display memory (virtual resolution is dependent solely on the amount of memory installed, with the maximum virtual width being 2048, and max virtual height is 4096).

Similar resolutions are supported on the Mach64. Refer to *README.Mach64* for configuration details.

Similar resolutions are supported on the Mach32. For the Mach32, the maximum virtual width is 1536, and the maximum virtual height is 1280.

For Mach8, the maximum virtual width is 1024.

For 8514 the maximum resolution is 1024x768.

For the AGX chips, maximum resolution depends upon the chip revision and amount of available display memory. Refer to *README.agx* for configuration details.

For the P9000, the virtual and physical resolutions must be the same. With sufficient memory, resolutions up to 1280x1024 are supported.

All the servers which support 24 bit visuals currently do so using a 32 bit per pixel configuration where 8 bits in every 32 bits is unused. This needs to be taken into account when calculating the maximum virtual display size that can be supported at this depth.

OPTIONS

In addition to the normal server options described in the *Xserver(1)* manual page, these servers accept some more command line switches, as described in the *XFree86(1)* manpage.

The Mach64, Mach32, S3, S3V, P9000 and AGX servers support more than 8 bit color. The Mach32, S3V and AGX servers support 16 bit TrueColor and the Mach64, S3, I128, and P9000 servers support 16 and 32 bit TrueColor. The 32 bit TrueColor mode only uses 24 bits per pixel for color information (giving you 16 million colors). These modes may be used by specifying the **−bpp** option as specified in the *XFree86(1)* manpage.

SETUP

XFree86 uses a configuration file called **XF86Config** for its initial setup. See the *XF86Config(4/5)* manpage for general details. Here only the parts specific to the *XF86_S3, XF86_S3V, XF86_Mach8, XF86_Mach32, XF86_Mach64, XF86_P9000, XF86_AGX, XF86_W32, XF86_TGA, XF86_I128,* and *XF86_8514* servers are explained.

Entries for the **Device** section in the XF86Config file include:

Chipset *"name"*

specifies a chipset so the correct driver can be used. Possible chipsets are:

XF86_S3:

- s3_generic (for a standard IO driven server)
- mmio_928 (for a memory mapped IO driven server on 86C928, 86C732, 86C764, 86C864, 86C868, 86C964 and 86C968 boards)
- newmmio (for a newer memory mapped IO driven server on 86C765, 86C868, and 86C968 boards)

XF86_S3V:

- s3_virge

XF86_Mach8:

- mach8 (to force the Mach8 server to run on Mach32 boards)

XF86_Mach32:

- mach32

XF86_Mach64:

- mach64

XF86_P9000:

- vipervlb (for the Diamond Viper VLB)
- viperpci (for the Diamond Viper PCI)
- orchidp9000 (for the Orchid P9000 and many generic P9000-based boards)

XF86_AGX:

- agx-016
- agx-015
- agx-014
- agx-010
- xga-2
- xga-1 (note: Only the agx-016, agx-015, agx-014 and XGA-2 have been tested. Refer to *README.agx* before attempting to use.)

XF86_W32:

et4000w32

et4000w32i

et4000w32i_rev_b

et4000w32i_rev_c

et4000w32p_rev_a

et4000w32p_rev_b

et4000w32p_rev_c

et4000w32p_rev_d

et6000

XF86_I128:

i128

XF86_TGA:

tga

XF86_8514:

ibm8514

Clocks *clock ...*

For boards with non-programmable clock chips, the clocks can be specified here (see *XF86Config(4/5)*). The P9000 server no longer requires a **Clocks** line. It will work the same way as other servers with a programmable clock chip (i.e., use the clocks as specified in the Modes). Note, clocks over 110 Mhz are not recommended or supported by the P9000 server. The Mach64 server also does not require a **Clocks** line since the clocks are normally read directly from the video card's BIOS or programmed as required. For the Mach64 server, the clocks given in the XF86Config file are ignored unless the "no_bios_clocks" option is given (see below).

ClockChip *"clockchip-type"*

For boards with programmable clock chips (except with the P9000 and AGX servers), the name of the clock chip is given. The only supported values for the W32 server are **"icd2061a"**, **"dcs2834"**, **"ics5341"** and **"stg1703"**. Possible values for the S3 server include **"icd2061a"**, **"ics9161a"**, **"dcs2834"**, **"sc11412"**, **"s3gendac"**, **"s3_sdac"**, **"ti3025"**, **"ti3026"**, **"ti3030"**, **"ics2595"**, **"ics5300"**, **"ics5342"**, **"ch8391"**, **"stg1703"**, **"att20c408"**, **"att20c409"**, **"att20c499"**, and **"ibm_rgb5xx"**. Possible values for the Mach64 server include **"ati18818"**, **"ics2595"**, **"stg1703"**, **"ch8398"**, **"ibm_rgb514"** and **"att20c408"**. The only possible clockchip value for the ET6000 is **"et6000"**.

Ramdac *"ramdac-type"*

This specifies the type of RAMDAC used on the board. Only the S3, AGX, Mach32, Mach64, and W32 servers use this.

normal - (S3, AGX, W32) Card does not have one of the other RAMDACs mentioned here. This option is only required for the S3 server if the server incorrectly detects one of those other RAMDACs. The AGX server does not yet auto-detect RAMDACs, this is the default if no RAMDAC is specified. Use this option for W32 server if it incorrectly identifies your RAMDAC or if RAMDAC detection fails entirely.

att20c490 - (S3, AGX, Mach32) Card has an AT&T 20C490 or AT&T 20C491 RAMDAC. When the **dac_8_bit** option is specified, these RAMDACs may be operated in 8 bit per RGB mode. It also allows 16bpp operation with 801/805/928 boards. True AT&T 20C490 RAMDACs should be auto-detected by the S3 server. This RAMDAC must be specified explicitly in other cases. Note that 8 bit per RGB mode does not appear to work with the Winbond 82C490 RAMDACs (which SuperProbe identifies as AT&T 20C492). 16bpp works fine with the Winbond 82C490. The Diamond SS2410 RAMDAC is reported to be compatible when operating in 15bpp mode (not 16bpp). The Chrontel 8391 appears to be compatible in all modes.

sc15025 - (S3, AGX) Card has a Sierra SC15025 or SC15026 RAMDAC. The S3 server has code to auto-detect this RAMDAC.

sc11482 - (S3) Card has a Sierra SC11482, SC11483 or SC11484 RAMDAC. The S3 server has code to auto-detect this RAMDAC.

sc11485 - (S3) Card has a Sierra SC11485, SC11487 or SC11489 RAMDAC. The S3 server will detect these RAMDACs as a **sc11482**, so this option must be specified to take advantage of extra features (they support 16bpp, 15bpp and 8bpp while the others only support 15bpp and 8bpp).

bt485 - (S3) Card has a BrookTree Bt485 or Bt9485 RAMDAC. This must be specified if the server fails to detect it.

att20c505 - (S3) Card has an AT&T 20C505 RAMDAC. This must be specified either if the server fails to detect the 20C505, or if the card has a Bt485 RAMDAC and there are problems using clocks higher than 67.5Mhz.

att20c498 - (S3) Card has an AT&T 20C498 or 21C498 RAMDAC. This must be specified if the server fails to detect it.

att22c498 - (S3) Card has an AT&T 22C498 RAMDAC. This must be specified if the server fails to detect it.

ibm_rgb514 - (S3) Card has an IBM RGB514 RAMDAC. This must be specified if the server fails to detect it.

ibm_rgb524 - (S3) Card has an IBM RGB524 RAMDAC. This must be specified if the server fails to detect it.

ibm_rgb525 - (S3) Card has an IBM RGB525 RAMDAC. This must be specified if the server fails to detect it.

ibm_rgb526 - (S3) Card has an IBM RGB526 RAMDAC. This must be specified if the server fails to detect it.

ibm_rgb528 - (S3) Card has an IBM RGB528 RAMDAC. This must be specified if the server fails to detect it.

stg1700 - (S3) Card has an STG1700 RAMDAC. This must be specified if the server fails to detect it.

stg1703 - (S3,W32) Card has an STG1703 RAMDAC. This must be specified if the server fails to detect it. Using the W32 server you MUST explicitly set the STG1703 as ClockChip to be able to use the programming capabilities.

s3gendac - (S3) Card has an S3 86C708 GENDAC. This RAMDAC does not support 8 bit per RGB mode (don't specify the **dac_8_bit** option). It allows 16bpp operation with 801/805 boards. There is currently no auto-detection for this RAMDAC.

s3_sdac - (S3) Card has an S3 86C716 SDAC RAMDAC. This must be specified if the server fails to detect it.

ics5300 - (S3) Card has an ICS5300 RAMDAC. This must be specified if the server fails to detect it (the server will recognise this as an S3 GENDAC which is OK).

ics5341 - (W32) Card has an ICS5341 RAMDAC. This must be specified if the server fails to detect it. For pixel clocks higher than 86MHz the server uses pixel multiplexing which seems to fail in a small band around 90MHz on most boards. While the ICS5341 RAMDAC is usually recognized as RAMDAC you MUST set it as ClockChip to be able to use the programming capabilities.

ics5342 - (S3) Card has an ICS5342 RAMDAC. This must be specified if the server fails to detect it (the server will recognise this as an S3 SDAC which is OK).

ti3020 - (S3) Card has a TI ViewPoint Ti3020 RAMDAC. This must be specified if the server fails to detect the Ti3020. Note that pixel multiplexing will be used for this RAMDAC if any mode requires a dot clock higher than 70MHz.

ti3025 - (S3) Card has a TI ViewPoint Ti3025 RAMDAC. This must be specified if the server fails to detect the Ti3025.

ti3026 - (S3) Card has a TI ViewPoint Ti3026 RAMDAC. This must be specified if the server fails to detect the Ti3026.

ti3030 - (S3) Card has a TI ViewPoint Ti3030 RAMDAC. This must be specified if the server fails to detect the Ti3030.

bt481 - (AGX, Mach32) Card has a BrookTree Bt481 RAMDAC.

bt482 - (AGX) Card has a BrookTree Bt482 RAMDAC.

herc_dual_dac - (AGX) Card (Hercules Graphite Pro) has both the 84-pin (Bt485 or AT&T20C505) and 44-pin (Bt481 or Bt482) RAMDACs installed.

herc_small_dac - (AGX) Card (Hercules Graphite Pro) has only the 44-pin (Bt481 or Bt482) RAMDAC installed.

ati68875 - (Mach64, Mach32) Card has an ATI 68875 RAMDAC. This must be specified if the server fails to detect it.

tlc34075 - (Mach64, Mach32) Card has a TI 34075 RAMDAC. This must be specified if the server fails to detect it.

ati68860 - (Mach64) Card has an ATI 68860 RAMDAC. This must be specified if the server fails to detect it.

ati68880 - (Mach64) Card has an ATI 68860 RAMDAC. This must be specified if the server fails to detect it.

stg1702 - (Mach64) Card has an STG1702 RAMDAC. This must be specified if the server fails to detect it.

ch8398 - (Mach64) Card has an Chrontel 8398 RAMDAC. This must be specified if the server fails to detect it.

att20c408 - (Mach64) Card has an AT&T 20C408 RAMDAC. This must be specified if the server fails to detect it.

IOBase *ioaddress*
specified the base address for extended IO registers. This is only used by the AGX server, and by the P9000 server for the Viper PCI. For details of how to use it, refer to *README.agx* and *README.P9000*.

MemBase *memaddress*
specifies the hard-wired part of the linear framebuffer base address. This option is only used by the P9000, S3, Mach64, Mach32, and TGA servers (and only when using a linear framebuffer). For the S3 server, the hard-wired part is the high 10 bits of the 32-bit address (ie *memaddress* is masked with *0xFFC00000*). Note: this should not be required for the 864 and later chips where the entire framebuffer address is software-selectable, or for PCI cards. Also, note that the in versions prior to 3.1.1, the S3 server used only the top 6 bits of *memaddress*, and ored it with 0x3C00000. To get the same behaviour, or 0x3C00000 with the value given previously. For the Mach32 server, the mask is *0xF8000000* (except for PCI cards, where the membase setting is ignored).

This option must be specified with the P9000 server. With local bus Diamond Vipers the value of *memaddress* can be either *0x80000000*, *0x20000000*, or *0xA0000000*. The default is *0x80000000*. Any value should work as long as it does not conflict with another device already at that address. For the Viper PCI, refer to *README.P9000*. For the Orchid P9000, the base address may be *0xC0000000*, *0xD0000000* or *0xE0000000*, and must correspond the the board's jumper setting.

Note: The S3 server will normally probe for this address automatically. Setting this option overrides that probe. This is not normally recommended because the failure of the server's probe usually indicates problems in using the linear framebuffer.

Note: The Mach64 server requires the memory aperture. For ISA bus video cards, this means that the aperture must be enabled and the aperture address must be set to a value less than 16Mb (which means that, on ISA systems only, to use the Mach64 server you must have 12Mb of main memory or less). Normally the Mach64 server will use pre-defined values for this address, but setting this option will override the pre-defined address.

The Mach32 server should not require the use of this option under normal circumstances.

COPBase *baseaddress*
This sets the coprocessor base address for the AGX server. Refer to *README.agx* for details.

Instance *instance*
This sets the XGA instance number for the AGX server. Refer to *README.agx* for details.

S3MClk *memclk*
This allows the video card's memory clock value to be specified. This is only used for 805i, 864 and Trio32/64 cards, and the value should not normally be given here for cards with an S3 Gendac or Trio64). This entry doesn't change the card's memory clock, but it is used to calculate the DRAM timing parameters. For further details refer to *README.S3*.

S3MNAdjust *M N*
This allows some memory timing parameters to be adjusted for DRAM cards. This entry is not normally required.

S3RefClk *refclk*
This allows the PLL reference clock to be specified. This may be required for some cards that use the IBM RGB5xx RAMDACs. The value is in MHz. For further details refer to *README.S3*.

Option flags may be specified in either the **Device** section or the **Display** subsection of the XF86Config file.

Option *"optionstring"*
allows the user to select certain options provided by the drivers. Currently the following strings are recognized:

nomemaccess - (S3) disable direct access to video memory. This option is ignored for the 864 and 964 chips.

noaccel - (AGX, P9000) disable hardware acceleration for the P9000, and disables the font cache with the AGX.

vram_128 - (AGX, P9000) when memory probe fails, use if you have 128Kx8 VRAMs.

vram_256 - (AGX, P9000) when memory probe fails, use if you don't have 128Kx8 VRAMs.

nolinear - (S3 and Mach32) disable use of a linear-mapped framebuffer.

ti3020_curs - (S3) Enables the Ti3020's internal HW cursor. (Default)

no_ti3020_curs - (S3) Disables the Ti3020's internal HW cursor.

sw_cursor - (S3, Mach32, Mach64, P9000, AGX) Disable the hardware cursor.

dac_8_bit - (S3, Mach32, Mach64, AGX) Enables 8-bit per RGB. Currently only supported with the Ti3020/5/6, Bt485, AT&T 20C505, AT&T 20C490/1, Sierra SC15025/6, AT&T 20C498 and STG1700/3, IBM RGB5xx (S3 server), Bt481 and Bt482 (AGX server), ATI68875/TLC34075/Bt885 (Mach32 server), ATI68875, TLC34075, ATI68860, ATI68880, STG1702, and STG1703 (Mach64 server) RAMDACs. This is now set by default in the S3 server when one of the above RAMDACs other than the AT&T 20C490/1 is used. Is also the default for the AGX server, except for the default VGA "normal" RAMDAC.

dac_6_bit - (S3, AGX) Force 6-bit per RGB in cases where 8-bit mode would automatically be enabled.

sync_on_green - (S3, P9000) Enables generation of sync on the green signal on cards with Bt485, AT&T 20C505, Ti3020/5/6 or IBM RGB5xx RAMDACs. **Note:** Although these RAMDACs support sync on green, it won't work on many cards because of the way they are designed.

power_saver - (S3, Mach64) This option enables the server to use the power saving features of VESA DPMS compatible monitors. The suspend level is currently only supported for the Mach64 and for the 732, 764, 864, 868, 964, 968 S3 chips. Refer to the *XF86Config(4/5)* manual page for details of how to set the timeouts for the different levels of operation. This option is experimental.

intel_gx - (Mach32) Sets the hard-wired offset for the linear framebuffer correctly for the Intel GX Pro cards. This option is equivalent to setting the **membase** to *0x78000000*.

ast_mach32 - (Mach32) Sets special handling for the Mach32 cards sold soldered-in on some AST motherboards.

spea_mercury - (S3) Enables pixel multiplex support for SPEA Mercury cards (928 + Bt485 RAMDAC). For these cards, pixel multiplexing is required in order to use dot clocks higher than 67.5 MHz and to access more than 1MB of video memory. Pixel multiplexing is currently supported only for non-interlaced modes, and modes with a physical width no smaller than 1024.

stb_pegasus - (S3) Enables pixel multiplex support for STB Pegasus cards (928 + Bt485 RAMDAC). For these cards, pixel multiplexing is required in order to use dot clocks higher than 67.5 MHz. Pixel multiplexing is currently supported only for non-interlaced modes, and modes with a physical width no smaller than 1024.

number_nine - (S3) Enables pixel multiplex support for Number Nine GXe level 10, 11, 12 cards (928 + Bt485 RAMDAC). For these cards, pixel multiplexing is required in order to use dot clocks higher than 85 MHz. Pixel multiplexing is currently supported only for non-interlaced modes, and modes with a physical width no smaller than 800. This option is also required for some other Number Nine cards (eg, GXE64 and GXE64pro).

diamond - (S3) This option may be required for some Diamond cards (in particular, the 964/968 VRAM cards).

elsa_w1000pro - (S3) Enables support for the ELSA Winner 1000 PRO. This option is not usually required because the board can be auto-detected.

elsa_w1000isa - (S3) Enables support for the ELSA Winner 1000 ISA. This option is not usually required because the board can be auto-detected.

elsa_w2000pro - (S3) Enables support for the ELSA Winner 2000 PRO. This option is not usually required because the board can be auto-detected.

pci_hack - (S3) Enables a workaround for problems seen with some PCI 928 cards on machines with a buggy SMC UART.

s3_964_bt485_vclk - (S3) Enables a workaround for possible problems on cards using the 964 and Bt485.

genoa, **stb**, **hercules** or **number_nine**, - (S3) These options may used to select different defaults for the blank delay settings for untested cards with IBM RGB5xx RAMDACs to avoid pixel wrapping problems.

slow_vram - (S3) Adjusts the VRAM timings for cards using slow VRAM. This is required for some Diamond Stealth 64 VRAM and Hercules Terminator 64 cards.

fast_vram - (S3) Adjusts the VRAM timings for faster VRAM access. There will be display errors and pixel garbage if your card can't support it.

slow_dram - (W32) Adjusts the DRAM refresh for cards with slow DRAM. Try this if your monitor goes into power save mode with the W32 server and older W32 cards.

slow_dram_refresh - (S3) Adjusts the DRAM refresh for cards with slow DRAM to avoid lines of corrupted pixels when switching modes.

pci_burst_on - (W32) Turns on the PCI burst for the W32p chipset. Use this if your picture looks distorted and your mouse leaves trails behind with burst disabled.

pci_burst_off - (W32) Turns off the PCI burst for the W32p chipset. Use this if your picture looks distorted and your mouse leaves trails behind with burst enabled.

w32_interleave_on - (W32) Turns on the memory interleave for the W32i and W32p chipset. Try this if your server runs stable with it.

w32_interleave_off - (W32) Turns off the memory interleave for the W32i and W32p chipset. Try this if your picture looks distorted or you don't get a picture at all.

no_block_write - (Mach64) Disables the block write mode on certain types of VRAM Mach64 cards. If noise or shadows appear on the screen, this option should remove them.

block_write - (Mach64) Enables the block write mode on certain types of VRAM Mach64 cards. Normally the Mach64 server will automatically determine if the card can handle block write mode, but this option will override the probe result.

no_bios_clocks - (Mach64) The Mach64 server normally reads the clocks from the BIOS. This option overrides the BIOS clocks and forces the server to use the clocks given in the XF86Config file.

no_program_clocks - (Mach64) The Mach64 server will automatically detect the clock chip and programs it directly from the video modes given. This option disables the clock chip programming and forces the use of the pre-programmed clocks either read from the BIOS or given on the Clocks line in the XF86Config file.

clkdiv2 - for all accelerated chipsets using a set of discrete clocks (i.e. not using a programmable ClockChip or a ClockProg, like older S3 cards and most ET4000W32 cards). With this option enabled, the accelerated driver can also use all the clocks mentionned in the clocks line divided by 2, presenting the server with twice as many clocks to choose from, especially in the low- and mid-range. This is useful for creating very low resolution modes like 320x200, because the lowest available clock on many cards (25.175 MHz) is too high to create a standard 320x200 mode. A few SVGA chips don't support this option, causing a distorted screen (S3-805 rev C and P are known to have this problem).

There are also numerous tuning options for the AGX server. Refer to *README.agx* for details.

Note that *XFree86* has some internal capabilities to determine what hardware it is running on. Thus normally the keywords *chipset*, *clocks*, and *videoram* don't have to be specified. But there may be occasions when this autodetection mechanism fails, (for example, too high of load on the machine when you start the server). For cases like this, one should first run the server on an unloaded machine, look at the results of the autodetection (that are printed out during server startup) and then explicitly specify these parameters in the configuration file. **It is recommended that all parameters, especially Clock values, be specified in the XF86Config file.**

FILES

<XRoot>/bin/XF86_S3	The 8, 16, and 32-bit color X server for S3

<XRoot>/bin/XF86_S3V	The 8, and 16-bit color X server for S3 ViRGE
<XRoot>/bin/XF86_Mach8	The 8-bit color X server for Mach8
<XRoot>/bin/XF86_Mach32	The 8, and 16-bit color X server for Mach32
<XRoot>/bin/XF86_Mach64	The 8, 16, and 32-bit color X server for Mach64
<XRoot>/bin/XF86_P9000	The 8, 16, and 32-bit color X server for the P9000
<XRoot>/bin/XF86_AGX	The 8, and 16-bit color X server for AGX and XGA
<XRoot>/bin/XF86_W32	The 8-bit color X server for ET4000/W32 and ET6000
<XRoot>/bin/XF86_I128	The 8, 16 and 32-bit color X server for Imagine 128
<XRoot>/bin/XF86_TGA	The 8-bit color X server for DEC TGA
<XRoot>/bin/XF86_8514	The 8-bit color X server for IBM 8514 and true compatibles
/etc/XF86Config	Server configuration file
<XRoot>/lib/X11/XF86Config	Server configuration file (secondary location)
<XRoot>/lib/X11/doc/README.agx	Extra documentation for the AGX server
<XRoot>/lib/X11/doc/README.P9000	Extra documentation for the P9000 server
<XRoot>/lib/X11/doc/README.S3	Extra documentation for the S3 server
<XRoot>/lib/X11/doc/README.W32	Extra documentation for the W32 server Note: <XRoot> refers to the root of the X11 install tree.

SEE ALSO

X(1), Xserver(1), XFree86(1), XF86Config(4/5), xvidtune(1), xdm(1), xf86config(1), xinit(1)

AUTHORS

In addition to the authors of *XFree86* the following people contributed major work to this server:

Kevin Martin, *martin@cs.unc.edu*
Jon Tombs, *tombs@XFree86.org*
Rik Faith, *faith@cs.unc.edu*
Did the overall work on the base accelerated servers.

David Dawes, *dawes@XFree86.org*
Dirk Hohndel, *hohndel@XFree86.org*
David Wexelblat, *dwex@XFree86.org*
Merged their work into XFree86.

Jon Tombs, *tombs@XFree86.org*
David Wexelblat, *dwex@XFree86.org*
David Dawes, *dawes@XFree86.org*
Amancio Hasty, *hasty@netcom.com*
Robin Cutshaw, *robin@XFree86.org*
Norbert Distler, *Norbert.Distler@physik.tu-muenchen.de*
Leonard N. Zubkoff, *lnz@dandelion.com*

Harald Koenig, *koenig@tat.physik.uni-tuebingen.de*
Bernhard Bender, *br@elsa.mhs.compuserve.com*
Hans Nasten, *nasten@everyware.se*
Dirk Hohndel, *hohndel@XFree86.org*
Joe Moss, *joe@morton.rain.com*
Development and improvement of the S3 specific code.

Kevin Martin, *martin@cs.unc.edu*
Rik Faith, *faith@cs.unc.edu*
Tiago Gons, *tiago@comosjn.hobby.nl*
Hans Nasten, *nasten@everyware.se*
Scott Laird, *scott@laird.com*
Development and improvement of the Mach8 and 8514/A specific code.

Kevin Martin, *martin@cs.unc.edu*
Rik Faith, *faith@cs.unc.edu*
Mike Bernson, *mike@mbsun.mlb.org*
Mark Weaver, *Mark_Weaver@brown.edu*
Craig Groeschel, *craig@metrolink.com*
Bryan Feir, *jenora@istar.ca*
Development and improvement of the Mach32 specific code.

Kevin Martin, *martin@cs.unc.edu*
Development of the Mach64 specific code.

Erik Nygren, *nygren@mit.edu*
Harry Langenbacher, *harry@brain.jpl.nasa.gov*
Chris Mason, *clmtch@osfmail.isc.rit.edu*
Henrik Harmsen, *harmsen@eritel.se*
Development and improvement of the P9000 specific code.

Henry Worth, *henry.worth@amail.amdahl.com*
Development of the AGX specific code.

Glenn Lai, *glenn@cs.utexas.edu*
Dirk Hohndel, *hohndel@XFree86.org*
Koen Gadeyne, *koen.gadeyne@barco.com*
Development of the ET4000/W32 and ET6000 specific code.

See also the *XFree86(1)* manual page.

BUGS

Some S3 cards with Bt485 RAMDACs are currently restricted to dot-clocks less than 85MHz.

The P9000 server may still have problems with cards other than the Diamond Viper VLB. There may still be problems with VGA mode restoration, but these should almost never occur. Using physical resolutions different from the virtual resolution is not supported and is not possible with the P9000. Use at dot-clocks greater than 110 MHz is not recommended and not supported. Diamond claims that 135 MHz is the maximum clock speed, but some of their bt485's are not rated that high. If you do not have a 135 MHz bt485 on your Viper, contact Diamond tech support and they will send you an RMA number to replace the board. Acceleration is being added in slowly. At the present, only CopyArea and MoveWindow and DrawLine are implemented. Other accelerated features are being tested and may be available in the next release. There seems to be a problem with olvwm when used with xdm and VT switching. The cursor will be messed up when you return to a VT if the cursor changed while you were in the VT.

The ET6000 server is quite new, and therefor not very thoroughly tested. Accelerated support is present, but doesn't use the full potential of the ET6000 chip (yet).

CONTACT INFO

XFree86 source is available from the FTP server *ftp.XFree86.Org* and mirrors. Send email to *XFree86@XFree86.Org* for details.

NAME

XF86_Mono - 1 bit non-accelerated X Window System servers for UNIX on x86 platforms

SYNOPSIS

XF86_Mono [:displaynumber] [option] ...

DESCRIPTION

XF86_Mono is a 1-bit StaticGrey server for VGA and Super-VGA cards and for some other monochrome cards.

CONFIGURATIONS

The *XF86_Mono* server supports the following popular SuperVGA chipsets in monochrome mode:

ATI:
: 18800, 18800-1, 28800-2, 28800-4, 28800-5, 28800-6, 68800-3, 68800-6, 68800AX, 68800LX, 88800GX-C, 88800GX-D, 88800GX-E, 88800GX-F, 88800CX, 264CT, 264ET, 264VT, 264VT2, 264GT

Tseng:
: ET3000, ET4000, ET4000/W32

Western Digital:
: PVGA1, WD90C00, WD90C10, WD90C11, WD90C30, WD90C31, WD90C33

Genoa:
: GVGA

Trident:
: TVGA8800CS, TVGA8900B, TVGA8900C, TVGA8900CL, TVGA9000

NCR:
: 77C22, 77C22E

Compaq:
: AVGA

Oak:
: OTI067, OTI077, OTI087

Cirrus:
: CLGD5420, CLGD5422, CLGD5424, CLGD5426, CLGD5428, CLGD5429, CLGD5430, CLGD5434, CLGD5436, CLGD6205, CLGD6215, CLGD6225, CLGD6235, CL6410, CL6412, CL6420, CL6440

The *XF86_Mono* server supports the following monochrome cards and resolutions:

Sigma:
: L-View, LaserView PLUS (in 1bpp mode): 1664x1280

Hyundai:
: HGC-1280: 1280[1472]x1024

Apollo:
: Monochrome card (with ID 9) from Apollo workstations: 1280x1024

Hercules and compatibles cards: 720x348

Additionally it supports generic VGA cards with a maximum virtual resolution of (approximately) 800x650.

On supported SVGA chipsets, *XF86_Mono* will use up to 1/4 of display memory, which yields a maximum virtual resolution of (approximately) 1664x1260 with 1MB of display memory. *XF86_Mono* does not support the accelerated functions of the supported chipsets.

OPTIONS

In addition to the normal server options described in the *Xserver(1)* manual page, *XF86_Mono* accepts some more command line switches, as described in the *XFree86(1)* manpage.

SETUP

XFree86 uses a configuration file called **XF86Config** for its initial setup. See the *XF86Config(4/5)* manpage for general details. Here only the *XF86_Mono* specific parts are explained.

The **Driver** entry in **Screen** section of the XF86Config file should be set to **vga2** for VGA and SVGA boards, and **mono** for non-vga mono boards. If **Screen** sections are present for both of these, the server will start in a dual-headed configuration.

Entries for the **Device** section in the XF86Config file include:

chipset *"name"*

specifies a chipset so the correct driver can be used. Possible chipsets are:

VGA2:

ATI:
ati (see README.ati for other recognized names)

Tseng:
et3000, et4000, et4000w32, et4000w32i, et4000w32p

Western Digital:
pvga1, wd90c00, wd90c10, wd90c30, wd90c31, wd90c33

Genoa:
gvga

Trident:
tvga8800cs, tvga8900b, tvga8900c, tvga8900cl, tvga9000

NCR:
ncr77c22, ncr77c22e

Compaq:
cpq_avga

OAK:
oti067, oti077, oti087

Cirrus:
clgd5420, clgd5422, clgd5424, clgd5426, clgd5428, clgd5429, clgd5430, clgd5434, clgd5436, clgd6205, clgd6215, clgd6225, clgd6235, cl6410, cl6412, cl6420, cl6440

Generic VGA:
generic

MONO:

Hyundai:
hgc1280

Sigma:
sigmalview

Apollo:
apollo9

Hercules:
hercules

MemBase *memaddress*
specifies the base address of the video memory. This option is only used for the Sigma LaserView cards. Valid addresses for these cards are *0xA0000*, *0xB0000*, *0xC0000*, *0xD0000*, *0xE0000*. The default is *0xE0000*.

Black *red green blue*
sets the ‘‘black’’ colour to the rgb values specified. These values must be given as integers in the range 0–63. The default is *0 0 0*. This option is only valid for the **vga2** screen type.

White *red green blue*
sets the ‘‘white’’ colour to the rgb values specified. These values must be given as integers in the range 0–63. The default is *63 63 63*. This option is only valid for the **vga2** screen type.

Option *"optionstring"*
allows the user to select certain options provided by the drivers. Currently the following strings are recognized:

legend - for Sigma Legend ET4000-based boards. This option enables a special clock-selection algorithm used on Legend boards, and MUST be specified for these boards to function correctly.

swap_hibit - for Western Digital/PVGA1 chipsets. Some Western Digital based boards require the high-order clock-select lead to be inverted. It is not possible for the server to determine this information at run-time. If the 9th clock in the list of clocks detected by the server is less than 30Mhz, this option likely needs to be set.

hibit_low, **hibit_high** - for Tseng ET4000 chipsets. With some ET4000 cards, the server has difficulty getting the state of the high-order clocks select bit right when started from a high-resolution text mode. These options allow the correct initial state of that bit to be specified. To find out what the correct initial state is, start the server from an 80x25 text mode. This option is only needed if the clocks reported by the server when started from a high-resolution text mode differ from those reported when it is started from an 80x25 text mode.

8clocks - for the PVGA1 chipset the default is 4 clocks. Some cards with this chipset may support 8 clocks. Specifying this option will allow the driver to detect and use the extra clocks.

16clocks - for Trident TVGA8900B and 8900C chipsets. Some newer boards using 8900B and 8900C chipsets actually support 16 clocks rather than the standard 8 clocks. Such boards will have a "TCK9002" or "TCK9004" chip on them. Specifying this option will allow the driver to detect and use the extra 8 clocks.

power_saver - This option enables the server to use the power saving features of VESA DPMS compatible monitors. The suspend level is currently not supported. Refer to the *XF86Config(4/5)* manual page for details of how to set the timeouts for the different levels of operation. This option is experimental.

secondary - for the hgc1280 and apollo9 chipsets. This option allows to use these cards jumpered to the secondary I/O / memory address. These addresses are:

hgc1280:
 I/O 0x3B0-0x3BF, mem 0xB0000-0xBFFFF (prim.)
 I/O 0x390-0x39F, mem 0xC8000-0xCFFFF (sec.)

apollo9:
 I/O 0x3B0-0x3BF, mem 0xFA0000-0xFDFFFF (prim.)
 I/O 0x3D0-0x3DF, mem 0xA0000-0xDFFFF (sec.)

XFree86 can detect the HGC-1280 on both primary and secondary address; for the apollo card the primary address is used by default.

Note that *XFree86* has some internal capabilities to determine what hardware it is running on. Thus normally the keywords *chipset*, *clocks*, and *videoram* don't have to be specified. But there may be occasions when this autodetection mechanism fails, (for example, too high of load on the machine when you start the server). For cases like this, one should first run *XF86_Mono* on an unloaded machine, look at the results of the autodetection (that are printed out during server startup) and then explicitly specify these parameters in the configuration file. **It is recommended that all parameters, especially Clock values, be specified in the XF86Config file.**

FILES

<XRoot>/bin/XF86_Mono	The monochrome X server for VGA, SVGA and other monochrome cards
/etc/XF86Config	Server configuration file
<XRoot>/lib/X11/XF86Config	Server configuration file Note: <XRoot> refers to the root of the X11 install tree.

SEE ALSO

X(1), Xserver(1), XFree86(1), XF86Config(4/5), xf86config(1), xvidtune(1), xdm(1), xinit(1)

BUGS

There are no known bugs at this time, although we welcome reports emailed to the address listed below.

CONTACT INFO

XFree86 source is available from the FTP server *ftp.XFree86.org*. Send email to *XFree86@XFree86.org* for details.

AUTHORS

Refer to the *XFree86(1)* manual page.

NAME

XF86_SVGA - SVGA X Window System servers for UNIX on x86 platforms

SYNOPSIS

XF86_SVGA [:displaynumber] [option] ...

DESCRIPTION

XF86_SVGA is an 8-bit PseudoColor, 16-bit TrueColor and 24-bit TrueColor server for Super-VGA cards. It is derived from the *X386* server provided with X11R5. Note: 16-bit and 24-bit TrueColor are currently only supported for some Cirrus, ARK, Chips & Technologies, and Trident chips, the ET6000 and some ET4000W32 chips.

CONFIGURATIONS

The *XF86_SVGA* server supports the following popular SuperVGA chipsets in 256 color mode. Virtual resolutions up to (approximately) 1152x900 are supported, using (up to) 1Mb of display memory. The Western Digital WD90C33, some of the Cirrus chipsets, ARK, the Oak OTI087, the Chips & Technologies 65550 or 65554 and the ET6000 support up to 2Mb of display memory and virtual resolutions of 1280x1024 and higher. Some of the Cirrus and ARK chipsets also support 16bpp, 24bpp and 32bpp (truecolor) modes on certain configurations. Some Trident chips are supported at 16bpp. Some Chips & Technologies chipset also support 16bpp and 24bpp (truecolor) modes. The ET6000 and some ET4000W32i or ET4000W32p chips supports all color depths. Generic VGA cards are also supported at 8bpp 320x200 only. The Matrox Millennium supports all color depths.

ATI:
: 18800, 18800-1, 28800-2, 28800-4, 28800-5, 28800-6, 68800-3, 68800-6, 68800AX, 68800LX, 88800GX-C, 88800GX_D, 88800GX-E, 88800GX-F, 88800CX, 264CT, 264ET, 264VT, 264VT2, 264GT

Tseng:
: ET3000, ET4000, ET4000/W32, ET6000

Western Digital:
: PVGA1, WD90C00, WD90C10, WD90C11, WD90C24A, WD90C30, WD90C31, WD90C33

Genoa:
: GVGA

Trident:
: TVGA8800CS, TVGA8900B, TVGA8900C, TVGA8900CL, TVGA9000 TVGA9000i, TVGA9100B, TVGA9200CXr, TGUI9320LCD, TGUI9400CXi, TGUI9420, TGUI9420DGi, TGUI9430DGi, TGUI9440AGi, TGUI9660XGi, TGUI9680

NCR:
: 77C22, 77C22E

Cirrus Logic:
: CLGD5420, CLGD5422, CLGD5424, CLGD5426, CLGD5428, CLGD5429, CLGD5430, CLGD5434, CLGD5436, CLGD5440, CLGD5446, CLGD5462, CLGD5464, CLGD6205, CLGD6215, CLGD6225, CLGD6235, CL6410, CL6412, CL6420, CL6440

ARK:
: ARK1000PV, ARK1000VL, ARK2000PV, ARK2000MT

RealTek:
RTG3106

Compaq:
AVGA

Oak:
OTI067, OTI077, OTI087

Avance Logic:
AL2101, ALI2301, ALI2302, ALI2308, ALI2401

Chips & Technologies:
65520, 65530, 65540, 65545, 65546, 65548, 65550, 65554

SiS: SIS 86C201, 86C202, 86C205

MX:
MX68000, MX68010

Video7:
HT216-32

Alliance:
AP6422

Matrox:
MGA2064W

NVidia:
NV1, STG2000

Accelerated support is included for most of the Cirrus chipsets, the Western Digital WD90C31 and WD90C33 chipsets, the Oak OTI087 chipset, Chips & Technologies, ARK Logic chipsets, the ET6000 and the ET4000/W32p. Accelerated 8bpp support for the ET4000/W32 (i and p) and ET6000 is implemented in a separate server (see *XF86_W32(1)*). Users of boards based on ATI's Mach8, Mach32 and Mach64 chipsets should refer to the *XF86_Mach8(1)*, *XF86_Mach32(1)* and *XF86_Mach64(1)* manual pages, respectively.

OPTIONS

In addition to the normal server options described in the *Xserver(1)* manual page, *XF86_SVGA* accepts some more command line switches, as described in the *XFree86(1)* manpage.

SETUP

XFree86 uses a configuration file called **XF86Config** for its initial setup. See the *XF86Config(4/5)* manpage for general details. Here only the *XF86_SVGA* specific parts are explained.

This server requires a **Screen** section in the XF86Config file with the **Driver** entry set to **svga**.

Entries for the **Device** section in the XF86Config file include:

chipset *"name"*
specifies a chipset so the correct driver can be used. Possible chipsets are:

ATI:
ati (See README.ati for other recognized names)

Tseng:
et3000, et4000W32, et4000W32i, et4000W32i_rev_b, et4000W32i_rev_c, et4000W32p, et4000W32p_rev_a, et4000W32p_rev_b, et4000W32p_rev_c, et4000W32p_rev_d, et6000

Western Digital:
pvga1, wd90c00, wd90c10, wd90c24, wd90c30, wd90c31, wd90c33

Genoa:
gvga

Trident:
tvga8800cs, tvga8900b, tvga8900c, tvga8900cl, tvga9000, tvga9000i, tvga9100b, tvga9200cxr, tgui9320lcd, tgui9400cxi, tgui9420, tgui9420dgi, tgui9430dgi, tgui9440agi, tgui9660xgi, tgui9680

NCR:
ncr77c22, ncr77c22e

Cirrus Logic:
clgd5420, clgd5422, clgd5424, clgd5426, clgd5428, clgd5429, clgd5430, clgd5434, clgd5436, clgd5446, clgd5462, clgd5464, clgd6205, clgd6215, clgd6225, clgd6235, cl6410, cl6412, cl6420, cl6440

RealTek:
realtek

ARK:
ark1000pv, ark1000vl, ark2000pv, ark2000mt

OAK:
oti067, oti077, oti087

Avance Logic:
al2101, ali2301, ali2302, ali2308, ali2401

Chips & Technologies:
ct65520, ct65530, ct65540, ct65545, ct65546, ct65548, ct65550, ct65554

SiS: sis86c201, sis86c202, sis86c205

Alliance:
AP6422

MX:
mx

Video7:
video7

Matrox: (you should not need to specify this)
mga2064w

NVidia:
nv1, stg2000

Generic:
generic

Option *"optionstring"*
allows the user to select certain options provided by the drivers. Currently the following strings are recognized:

legend - for Sigma Legend ET4000-based boards. This option enables a special clock-selection algorithm used on Legend boards, and MUST be specified for these boards to function correctly.

swap_hibit - for Western Digital/PVGA1 chipsets. Some Western Digital based boards require the high-order clock-select lead to be inverted. It is not possible for the server to determine this information at run-time. If the 9th clock in the list of clocks detected by the server is less than 30Mhz, this option likely needs to be set.

hibit_low, **hibit_high** - for Tseng ET4000 chipsets. With some ET4000 cards, the server has difficulty getting the state of the high-order clocks select bit right when started from a high-resolution text mode. These options allow the correct initial state of that bit to be specified. To find out what the correct initial state is, start the server from an 80x25 text mode. This option is only needed if the clocks reported by the server when started from a high-resolution text mode differ from those reported when it is started from an 80x25 text mode.

8clocks - for the PVGA1 chipset the default is 4 clocks. Some cards with this chipset may support 8 clocks. Specifying this option will allow the driver to detect and use the extra clocks.

16clocks - for Trident TVGA8900B and 8900C chipsets. Some newer boards using 8900B and 8900C chipsets actually support 16 clocks rather than the standard 8 clocks. Such boards will have a "TCK9002" or "TCK9004" chip on them. Specifying this option will allow the driver to detect and use the extra 8 clocks.

probe_clocks - for Cirrus chipsets. The Cirrus driver has a fixed set of clocks that are normally used. Specifying this option will force the driver to probe for clocks instead of reporting the built-in defaults. This option is for debugging purposes only.

hw_clocks - for C&T chipsets. The Chips & Technologies driver will by default attempt to use programmable clocks for chips that support them. Specifying this option will force the server to use the limited number of fixed clocks supported by the hardware.

clkdiv2 - for all chipsets using a set of discrete clocks (i.e. not using a programmable ClockChip or a ClockProg). With this option enabled, the SVGA driver can also use all the clocks mentioned in the clocks line divided by 2, presenting the server with twice as many clocks to choose from, especially in the low- and mid-range. This is useful for creating very low resolution modes like 320x200, because the lowest available clock on many cards (25.175 MHz) is too high to create a standard 320x200 mode. A few SVGA chips (S3, WD90cxx) don't support this option, causing a distorted screen.

power_saver - This option enables the server to use the power saving features of VESA DPMS compatible monitors. Refer to the *XF86Config(4/5)* manual page for details of how to set the timeouts for the different levels of operation.

noaccel - for Cirrus, WD, ARK, C&T, MGA and Tseng chipsets. This option disables the accelerated features for the clgd542X, clgd543x, clgd5446, clgd546X, wd90c24, wd90c31 and wd90c33, C&T, ARK, MGA and Tseng chipsets.

no_pixmap_cache - for any chip that uses XAA for acceleration, which includes ARK, Cirrus, C&T, MGA, SiS, Trident 9440/9680, and Tseng chipsets. This disables the use of a pixmap cache by XAA.

xaa_no_color_exp - for any chip that uses XAA for acceleration. This option disables the use by XAA of hardware color expansion features. If you see text rendering problems, try this.

xaa_benchmarks - most useful for a chip that uses XAA for acceleration. This option makes the server perform and report the results of benchmarks performed at start-up of XAA acceleration primitives.

fifo_conservative - for Cirrus chipsets 542x/3x/46/6x, oti087, and ARK. This option sets the CRT FIFO threshold to a conservative value for higher dot clocks. This slightly reduces performance, but may help in eliminating problems with "streaks" on the screen caused by video memory bandwidth contention during BitBLT operations

fifo_aggressive - for Cirrus chipsets 542x/3x/46/6x, and oti087, and ARK. This option sets the CRT FIFO threshold to an aggressive value.

slow_dram - for Cirrus chipsets 542x/3x/46, and Trident. This option sets the DRAM timings for slow DRAM chips.

med_dram - for Cirrus chipsets 542x/3x/46 and Trident. This option sets the DRAM timings for medium-speed DRAM chips.

fast_dram - for ET4000, ET6000 and Cirrus chipsets 542x/3x/46, and Trident. This option sets the DRAM timings for fast DRAM chips. Normally not recommended.

no_2mb_banksel - for Cirrus chipsets. This option is required for Cirrus cards with 2MB of videoram which is in the form of 512kx8 DRAMs (4 chips) rather than 256kx4 DRAMs (16 chips).

no_bitblt - for Cirrus and C&T chipsets. This option disables use of hardware BitBLT.

no_imageblt - for Cirrus and C&T chipsets. This option disables use of CPU-assisted BitBLT functions.

mmio - for Cirrus and C&T chipsets. This option enables memory-mapped I/O for BitBLT communication.

linear - Attempt a linear mapping of the framebuffer into high memory. Currently only supported for some Cirrus and oti087 configurations, the ET6000, the ET4000W32p rev C and D and the ET4000W32i. Automatically selected (and hence the option is ignored) for the clgd5462 and clgd5464, C&T, NVidia, and the ARK chipsets. This is also true for Tseng chips, except for 8bpp modes, where this flag allows linear mapping to be forced (it defaults to banked mode at 8bpp).

nolinear - Disable linear mapping. Useful for C&T, ARK and some Cirrus chipsets.

med_dram, favour_bitblt, clgd6225_lcd, no_stretch, no_mmio - more Cirrus-specific options. Refer to /usr/X11R6/lib/X11/doc/README.cirrus for a detailed description of Cirrus options.

ext_fram_buf, use_modeline, fix_panel_size, no_stretch, lcd_center, suspend_hack, use_18bit_bus - more Chips & Technologies specific options for use with LCD screens. Refer to <XRoot>/lib/X11/doc/README.chips for a detailed description of these options.

hw_cursor - for ARK or ET6000 chipsets. This option enables the hardware cursor.

sw_cursor - for Cirrus and C&T chipsets. This option disables the hardware cursor.

sync_on_green - for Chips & Technologies 65550's and 65554's and Matrox MGA2064W's. This option enables the generation of a sync on green signal.

pci_retry - for all accelerated Tseng chipsets. This option enables a "bus-hold" feature on the Tseng chip that will increase accelerator performance, but at the cost of long system bus blocking periods. This may cause bus timeouts for some ISA DMA hardware (soundcards, floppy tape drives, etc). Despite the name, this is not limited to PCI bus cards only.

speedup *"selection"*
sets the selection of SpeedUps to use. The optional selection string can take the following values:

none
all

If the selection string is omitted, or if the **speedup** option is omitted, the selection defaults to "all". Some of the SpeedUps can only be used with the ET4000, WD90C31 and WD90C33 chipsets and others requires a virtual resolution with a xdim of 1024. SpeedUps that won't work with a given configuration are automatically disabled.

nospeedup
disables the SpeedUp code. This is equivalent to **speedup "none"**.

Ramdac *"ramdac-type"*
This specifies the type of RAMDAC used on the board. The ARK and ET4000W32i/p drivers use this. Refer to the chips-specific documentation for details.

TextClockFreq *txtclk*
For some Chips & Technologies chipsets it is impossible to determine the clock used at the text console. The server therefore assumes that a 25.175MHz clock is being used. This can easily be wrong and corrupt the restored text console. This option allows the user to specify a clock that will be restored.

Note that *XFree86* has some internal capabilities to determine what hardware it is running on. Thus normally the keywords *chipset*, *clocks*, and *videoram* don't have to be specified. But there may be occasions when this autodetection mechanism fails, (for example, too high of load on the machine when you start the server). For cases like this, one should first run *XF86_SVGA* on an unloaded machine, look at the results of the autodetection (that are printed out during server startup) and then explicitly specify these parameters in the configuration file. **It is recommended that all parameters, especially Clock values, be specified in the XF86Config file.**

FILES

<XRoot>/bin/XF86_SVGA
The SVGA color X server

/etc/XF86Config
Server configuration file

<XRoot>/lib/X11/XF86Config
Server configuration file

<XRoot>/lib/X11/doc/README.ark
Extra documentation for the ARK driver

<XRoot>/lib/X11/doc/README.ati
Extra documentation for the ATI vgawonder driver

<XRoot>/lib/X11/doc/README.chips
Extra documentation for the Chips & Technologies driver

<XRoot>/lib/X11/doc/README.cirrus
Extra documentation for the Cirrus driver

<XRoot>/lib/X11/doc/README.MGA
Extra documentation for the Matrox MGA driver

<XRoot>/lib/X11/doc/README.trident
Extra documentation for the Trident driver

<XRoot>/lib/X11/doc/README.tseng
Extra documentation for the ET4000 and ET3000 drivers

<XRoot>/lib/X11/doc/README.Oak
Extra documentation for the Oak driver

<XRoot>/lib/X11/doc/README.Video7
Extra documentation for the Video7 driver

<XRoot>/lib/X11/doc/README.WstDig
Extra documentation for the WD/PVGA driver Note: <XRoot> refers to the root of the X11 install tree.

SEE ALSO

X(1), Xserver(1), XFree86(1), XF86Config(4/5), xf86config(1), XF86Setup(1), xvidtune(1), xdm(1), xinit(1)

BUGS

Bug reports are welcome, and should be emailed to the address listed below.

CONTACT INFO

XFree86 source is available from the FTP server *ftp.XFree86.org*. Send email to *XFree86@XFree86.org* for details.

AUTHORS

Refer to the *XFree86(1)* manual page.

NAME

XF86_VGA16 - 4 bit non-accelerated X Window System server for UNIX on x86 platforms

SYNOPSIS

XF86_VGA16 [:displaynumber] [option] ...

DESCRIPTION

XF86_VGA16 is a 4-bit color server for VGA cards. The default root visual for this server is StaticColor. It also includes support for the non-VGA monochrome cards described in the *XF86_Mono(1)* manual page. It may be run in a dual-headed configuration.

CONFIGURATIONS

The *XF86_VGA16* server supports the following popular SVGA chipsets in 16-colour mode.

ATI:
: 18800, 18800-1, 28800-2, 28800-4, 28800-5, 28800-6, 68800-3, 68800-6, 68800AX, 68800LX, 88800GX-C, 88800GX-D, 88800GX-E, 88800GX-F, 88800CX, 264CT, 264ET, 264VT, 264VT2, 264GT

Tseng:
: ET4000

Trident:
: TVGA8800CS, TVGA8900B, TVGA8900C, TVGA8900CL, TVGA9000

Cirrus:
: CL6410, CL6412, CL6420, CL6440

Oak:
: OTI067, OTI077, OTI087

Additionally it supports generic VGA cards.

XF86_VGA16 does not support the accelerated functions of the supported chipsets.

OPTIONS

In addition to the normal server options described in the *Xserver(1)* manual page, *XF86_VGA16* accepts some more command line switches, as described in the *XFree86(1)* manpage.

SETUP

XFree86 uses a configuration file called **XF86Config** for its initial setup. See the *XF86Config(4/5)* manpage for general details. Here only the *XF86_VGA16* specific parts are explained.

The **Driver** entry in the **Screen** section of the XF86Config file should be set to **vga16**. To run in dual-headed configuration, there should also be a **Screen** section with the **Driver** entry set to **mono**.

Entries for the **Device** section in the XF86Config file include:

chipset *"name"*
specifies a chipset so the correct driver can be used. Possible chipsets are:

ATI:
ati (see README.ati for additional recoginzed names)

Tseng:
et4000, et4000w32, et4000w32i, et4000w32p

Trident:
tvga8800cs, tvga8900b, tvga8900c, tvga8900cl, tvga9000

Cirrus:
cl6410, cl6412, cl6420, cl6440

Oak:
oti067, oti077, oti087

Generic VGA:
generic

Option *"optionstring"*
allows the user to select certain options provided by the drivers. Currently the following strings are recognized:

legend - for Sigma Legend ET4000-based boards. This option enables a special clock-selection algorithm used on Legend boards, and MUST be specified for these boards to function correctly.

hibit_low, **hibit_high** - for Tseng ET4000 chipsets. With some ET4000 cards, the server has difficulty getting the state of the high-order clocks select bit right when started from a high-resolution text mode. These options allow the correct initial state of that bit to be specified. To find out what the correct initial state is, start the server from an 80x25 text mode. This option is only needed if the clocks reported by the server when started from a high-resolution text mode differ from those reported when it is started from an 80x25 text mode.

clkdiv2 - for all chipsets using a set of discrete clocks (i.e. not using a programmable ClockChip or a ClockProg). With this option enabled, the VGA driver can also use all the clocks mentionned in the clocks line divided by 2, presenting the server with twice as many clocks to choose from, especially in the low- and mid-range. This is useful for creating very low resolution modes like 320x200, because the lowest available clock on many cards (25.175 MHz) is too high to create a standard 320x200 mode. A few VGA chips don't support this option, causing a distorted screen.

power_saver - This option enables the server to use the power saving features of VESA DPMS compatible monitors. The suspend level is currently not supported. Refer to the *XF86Config(4/5)* manual page for details of how to set the timeouts for the different levels of operation. This option is experimental.

Note that *XFree86* has some internal capabilities to determine what hardware it is running on. Thus normally the keywords *chipset*, *clocks*, and *videoram* don't have to be specified. But there may be occasions when this autodetection mechanism fails, (for example, too high of load on the machine when you start the server). For cases like this, one should first run *XF86_VGA16* on an unloaded machine, look at the results of the autodetection (that are printed out during server startup) and then explicitly specify these parameters in the configuration file. **It is recommended that all parameters, especially Clock values, be specified in the XF86Config file.**

FILES

<XRoot>/bin/XF86_VGA16	The 16 colour X server
/etc/XF86Config	Server configuration file
<XRoot>/lib/X11/XF86Config	Server configuration file Note: <XRoot> refers to the root of the X11 install tree.

SEE ALSO

X(1), Xserver(1), XFree86(1), XF86Config(4/5), XF86_Mono(1), xf86config(1), xvidtune(1), xdm(1), xinit(1)

CONTACT INFO

XFree86 source is available from the FTP server *ftp.XFree86.org*. Send email to *XFree86@XFree86.org* for details.

AUTHORS

The primary developer of this server is

Gertjan Akkerman, *akkerman@dutiba.twi.tudelft.nl*

See also the *XFree86(1)* manual page.

NAME

XFree86 - X11R6 for UNIX on x86 platforms

DESCRIPTION

XFree86 is a collection of X servers for UNIX-like OSs on Intel x86 platforms. This work is derived from *X386 1.2* which was contributed to X11R5 by Snitily Graphics Consulting Service.

CONFIGURATIONS

XFree86 operates under the following operating systems:

-- SVR3.2: SCO 3.2.2, 3.2.4, ISC 3.x, 4.x
-- SVR4.0: ESIX, Microport, Dell, UHC, Consensys, MST, ISC, AT&T, NCR
-- SVR4.2: Consensys, Univel (UnixWare)
-- Solaris (x86) 2.1, 2.4
-- FreeBSD 1.1.5, 2.0, 2.0.5, NetBSD 1.0 (i386 port only)
-- BSD/386 version 1.1 and BSD/OS 2.0
-- Mach (from CMU)
-- Linux
-- Amoeba version 5.1
-- Minix-386vm version 1.6.25.1
-- LynxOS AT versions 2.2.1, 2.3.0 and 2.4.0, LynxOS microSPARC 2.4.0

NETWORK CONNECTIONS

XFree86 supports connections made using the following reliable byte-streams:

Local

XFree86 supports local connections via Streams pipe via various mechanisms, using the following paths (*n* represents the display number):

/dev/X/server.**n** (SVR3 and SVR4)
/dev/X/Nserver.**n** (SVR4)
/dev/X**n**S and /dev/X**n**R (SCO SVR3)

On SVR4.0.4, if the *Advanced Compatibility Package* is installed, and in SVR4.2, *XFree86* supports local connections from clients for SCO XSight/ODT, and (with modifications to the binary) clients for ISC SVR3.

Unix Domain

XFree86 uses */tmp/.X11-unix/X***n** as the filename for the socket, where *n* is the display number.

TCPIP

XFree86 listens on port htons(6000+*n*), where *n* is the display number.

Amoeba RPC

This is the default communication medium used under native Amoeba. Note that under Amoeba, the server should be started with a "*hostname*:*displaynumber*" argument.

ENVIRONMENT VARIABLES

For operating systems that support local connections other than Unix Domain sockets (SVR3 and SVR4), there is a compiled-in list specifying the order in which local connections should be attempted. This list can be overridden by the *XLOCAL* environment variable described below. If the display name indicates a best-choice connection should be made (e.g. *:0.0*), each connection mechanism is tried until a connection succeeds or no more mechanisms are available. Note: for these OSs, the Unix Domain socket connection is treated differently from the other local connection types. To use it the connection must be made to *unix:0.0*.

The *XLOCAL* environment variable should contain a list of one more more of the following:

NAMED
PTS
SCO
ISC

which represent SVR4 Named Streams pipe, Old-style USL Streams pipe, SCO XSight Streams pipe, and ISC Streams pipe, respectively. You can select a single mechanism (e.g. *XLOCAL=NAMED*), or an ordered list (e.g. *XLOCAL="NAMED:PTS:SCO"*). This variable overrides the compiled-in defaults. For SVR4 it is recommended that *NAMED* be the first preference connection. The default setting is *PTS:NAMED:ISC:SCO*.

To globally override the compiled-in defaults, you should define (and export if using *sh* or *ksh*) *XLOCAL* globally. If you use *startx/xinit*, the definition should be at the top of your *.xinitrc* file. If you use *xdm*, the definitions should be early on in the *<XRoot>/lib/X11/xdm/Xsession* script.

OPTIONS

In addition to the normal server options described in the *Xserver(1)* manual page, *XFree86* accepts the following command line switches:

vt*XX* *XX* specifies the Virtual Terminal device number which *XFree86* will use. Without this option, *XFree86* will pick the first available Virtual Terminal that it can locate. This option applies only to SVR3, SVR4, Linux, and BSD OSs with the ‘syscons’ or ‘pcvt’ driver.

-crt /dev/tty*XX*
SCO only. This is the same as the **vt** option, and is provided for compatibility with the native SCO X server.

−probeonly
Causes the server to exit after the device probing stage. The XF86Config file is still used when this option is given, so information that can be auto-detected should be commented out.

−quiet Suppress most informational messages at startup.

−bpp *n* Set number of bits per pixel. The default is 8. Legal values are 8, 15, 16, 24, 32. Not all servers support all values.

−weight *nnn*
Set RGB weighting at 16 bpp. The default is 565. This applies only to those servers which support 16 bpp.

−flipPixels
Swap the default values for the black and white pixels.

−disableVidMode
Disable the the parts of the VidMode extension used by the xvidtune client that can be used to change the video modes.

−allowNonLocalXvidtune
Allow the xvidtune client to connect from another host. By default non-local connections are not allowed.

−disableModInDev
Disable dynamic modification of input device settings.

−allowNonLocalModInDev
Allow changes to keyboard and mouse settings from non-local clients. By default, connections from non-local clients are not allowed to do this.

−allowMouseOpenFail
Allow the server to start up even if the mouse device can't be opened or initialised.

−gamma *value*
Set the gamma correction. *value* must be between 0.1 and 10. The default is 1.0 This value is applied equally to the R, G and B values. Not all servers support this.

−rgamma *value*
Set the red gamma correction. *value* must be between 0.1 and 10. The default is 1.0 Not all servers support this.

−ggamma *value*
Set the green gamma correction. *value* must be between 0.1 and 10. The default is 1.0 Not all servers support this.

−bgamma *value*
Set the blue gamma correction. *value* must be between 0.1 and 10. The default is 1.0 Not all servers support this.

−showconfig
Print out the server version, patchlevel, and a list of screen drivers configured in the server.

−verbose
Multiple occurrences of this flag increase the amount of information printed on stderr (more than the default).

−version Same as **−showconfig**.

−xf86config *file*
Read the server configuration from *file*. This option is only available when the server is run as root (i.e, with real-uid 0).

−keeptty Prevent the server from detaching its initial controlling terminal. This option is only useful when debugging the server.

KEYBOARD

Multiple key presses recognized directly by *XFree86* are:

Ctrl+Alt+Backspace
Immediately kills the server -- no questions asked. (Can be disabled by specifying "DontZap" in the **ServerFlags** section of the XF86Config file.)

Ctrl+Alt+Keypad-Plus
Change video mode to next one specified in the configuration file, (increasing video resolution order).

Ctrl+Alt+Keypad-Minus
Change video mode to previous one specified in the configuration file, (decreasing video resolution order).

Ctrl+Alt+F1...F12
For BSD systems using the syscons driver and Linux, these keystroke combinations are used to switch to Virtual Console 1 through 12.

SETUP

XFree86 uses a configuration file called **XF86Config** for its initial setup. Refer to the *XF86Config(4/5)* manual page for more information.

FILES

<XRoot>/bin/XF86_SVGA	The color SVGA X server
<XRoot>/bin/XF86_Mono	The monochrome X server for VGA and other mono cards
<XRoot>/bin/XF86_S3	The accelerated S3 X server
<XRoot>/bin/XF86_Mach8	The accelerated Mach8 X server
<XRoot>/bin/XF86_Mach32	The accelerated Mach32 X server
<XRoot>/bin/XF86_Mach64	The accelerated Mach64 X server
<XRoot>/bin/XF86_P9000	The accelerated P9000 X server
<XRoot>/bin/XF86_AGX	The accelerated AGX X server
<XRoot>/bin/XF86_W32	The accelerated ET4000/W32 and ET6000 X server
<XRoot>/bin/XF86_8514	The accelerated 8514/A X server
/etc/XF86Config	Server configuration file
<XRoot>/lib/X11/XF86Config.*hostname*	Server configuration file
<XRoot>/lib/X11/XF86Config	Server configuration file
<XRoot>/bin/*	Client binaries
<XRoot>/include/*	Header files
<XRoot>/lib/*	Libraries
<XRoot>/lib/X11/fonts/*	Fonts
<XRoot>/lib/X11/rgb.txt	Color names to RGB mapping
<XRoot>/lib/X11/XErrorDB	Client error message database
<XRoot>/lib/X11/app-defaults/*	Client resource specifications
<XRoot>/man/man?/*	Manual pages
/etc/X*n*.hosts	Initial access control list for display *n* Note: <XRoot> refers to the root of the X11 install tree.

SEE ALSO

X(1), Xserver(1), xdm(1), xinit(1), XF86Config(4/5), xf86config(1), XF86_SVGA(1), XF86_VGA16(1), XF86_Mono(1), XF86_Accel(1), xvidtune(1)

AUTHORS

For X11R5, *XF86 1.2* was provided by:

Thomas Roell, *roell@informatik.tu-muenchen.de*
TU-Muenchen: Server and SVR4 stuff

Mark W. Snitily, *mark@sgcs.com*
SGCS: SVR3 support, X Consortium Sponsor

... and many more people out there on the net who helped with ideas and bug-fixes.

XFree86 was integrated into X11R6 by the following team:

Stuart Anderson *anderson@metrolink.com*
Doug Anson *danson@lgc.com*
Gertjan Akkerman *akkerman@dutiba.twi.tudelft.nl*
Mike Bernson *mike@mbsun.mlb.org*
Robin Cutshaw *robin@XFree86.org*
David Dawes *dawes@XFree86.org*
Marc Evans *marc@XFree86.org*
Pascal Haible *haible@izfm.uni-stuttgart.de*
Matthieu Herrb *Matthieu.Herrb@laas.fr*
Dirk Hohndel *hohndel@XFree86.org*
David Holland *davidh@use.com*
Alan Hourihane *alanh@fairlite.demon.co.uk*
Jeffrey Hsu *hsu@soda.berkeley.edu*
Glenn Lai *glenn@cs.utexas.edu*
Ted Lemon *mellon@ncd.com*
Rich Murphey *rich@XFree86.org*
Hans Nasten *nasten@everyware.se*
Mark Snitily *mark@sgcs.com*
Randy Terbush *randyt@cse.unl.edu*
Jon Tombs *tombs@XFree86.org*
Kees Verstoep *versto@cs.vu.nl*
Paul Vixie *paul@vix.com*
Mark Weaver *Mark_Weaver@brown.edu*
David Wexelblat *dwex@XFree86.org*
Philip Wheatley *Philip.Wheatley@ColumbiaSC.NCR.COM*
Thomas Wolfram *wolf@prz.tu-berlin.de*
Orest Zborowski *orestz@eskimo.com*

The *XFree86* enhancement package was provided by:

David Dawes, *dawes@XFree86.org*
Release coordination, administration of FTP repository and mailing lists. Source tree management and integration, accelerated server integration, fixing, and coding.

Glenn Lai, *glenn@cs.utexas.edu*
The SpeedUp code for ET4000 based SVGA cards, and ET4000/W32 accelerated server.

Jim Tsillas, *jtsilla@ccs.neu.edu*
Many server speedups from the fX386 series of enhancements.

David Wexelblat, *dwex@XFree86.org*
Integration of the fX386 code into the default server, many driver fixes, and driver documentation, assembly of the VGA card/monitor database, development of the generic video mode listing. Accelerated server integration, fixing, and coding.

Dirk Hohndel, *hohndel@XFree86.org*
Linux shared libraries and release coordination. Accelerated server integration

and fixing. Generic administrivia and documentation.

Amancio Hasty Jr., *hasty@netcom.com*
Porting to **386BSD** version 0.1 and XS3 development.

Rich Murphey, *rich@XFree86.org*
Ported to **386BSD** version 0.1 based on the original port by Pace Willison. Support for **386BSD**, **FreeBSD**, and **NetBSD**.

Robert Baron, *Robert.Baron@ernst.mach.cs.cmu.edu*
Ported to **Mach**.

Orest Zborowski, *orestz@eskimo.com*
Ported to **Linux**.

Doug Anson, *danson@lgc.com*
Ported to **Solaris x86**.

David Holland, *davidh@use.com*
Ported to **Solaris x86**.

David McCullough, *davidm@stallion.oz.au*
Ported to **SCO SVR3**.

Michael Rohleder, *michael.rohleder@stadt-frankfurt.de*
Ported to **ISC SVR3**.

Kees Verstoep, *versto@cs.vu.nl*
Ported to **Amoeba** based on Leendert van Doorn's original Amoeba port of X11R5.

Marc Evans, *Marc@XFree86.org*
Ported to **OSF/1**.

Philip Homburg, *philip@cs.vu.nl*
Ported to **Minix-386vm**.

Thomas Mueller, *tm@systrix.de*
Ported to **LynxOS**.

Jon Tombs, *tombs@XFree86.org*
S3 server and accelerated server coordination.

Harald Koenig, *koenig@tat.physik.uni-tuebingen.de*
S3 server development.

Bernhard Bender, *br@elsa.mhs.compuserve.com*
S3 server development.

Kevin Martin, *martin@cs.unc.edu*
Overall work on the base accelerated servers (ATI and 8514/A), and Mach64 server.

Rik Faith, *faith@cs.unc.edu*
Overall work on the base accelerated servers (ATI and 8514/A).

Tiago Gons, *tiago@comosjn.hobby.nl*
Mach8 and 8514/A server development

Hans Nasten, *nasten@everyware.se*
Mach8, 8514/A, and S3 server development and BSD/386 support

Mike Bernson, *mike@mbsun.mlb.org*
Mach32 server development.

Mark Weaver, *Mark_Weaver@brown.edu*
Mach32 server development.

Craig Groeschel, *craig@metrolink.com*
Mach32 server development.

Henry Worth, *Henry.Worth@amail.amdahl.com*
AGX server.

Erik Nygren, *nygren@mit.edu*
P9000 server.

Harry Langenbacher *harry@brain.jpl.nasa.gov*
P9000 server.

Chris Mason, *mason@mail.csh.rit.edu*
P9000 server.

Henrik Harmsen *harmsen@eritel.se*
P9000 server.

Simon Cooper, *scooper@vizlab.rutgers.edu*
Cirrus accelerated code (based on work by Bill Reynolds).

Harm Hanemaayer, *hhanemaa@cs.ruu.nl*
Cirrus accelerated code, and ARK driver.

Thomas Zerucha, *zerucha@shell.portal.com*
Support for Cirrus CL-GD7543.

Leon Bottou, *bottou@laforia.ibp.fr*
ARK driver.

Mike Tierney, *floyd@eng.umd.edu*
WD accelerated code.

Bill Conn, *conn@bnr.ca*
WD accelerated code.

Brad Bosch, *brad@lachman.com*
WD 90C24A support.

Alan Hourihane, *alanh@fairlite.demon.co.uk*
Trident SVGA driver, SiS SVGA driver and DEC 21030 server.

Marc La France, *Marc.La-France@ualberta.ca*
ATI vgawonder SVGA driver

Steve Goldman, *sgoldman@encore.com*
Oak 067/077 SVGA driver.

Jorge Delgado, *ernar@dit.upm.es*
Oak SVGA driver, and 087 accelerated code.

Bill Conn, *conn@bnr.ca*
WD accelerated code.

Paolo Severini, *lendl@dist.dist.unige.it*
AL2101 SVGA driver

Ching-Tai Chiu, *cchiu@netcom.com*
Avance Logic ALI SVGA driver

Manfred Brands, *mb@oceonics.nl*
Cirrus 64xx SVGA driver

Randy Hendry, *randy@sgi.com*
Cirrus 6440 support in the cl64xx SVGA driver

Frank Dikker, *dikker@cs.utwente.nl*
MX SVGA driver

Regis Cridlig, *cridlig@dmi.ens.fr*
Chips & Technologies driver

Jon Block, *block@frc.com*
Chips & Technologies driver

Mike Hollick, *hollick@graphics.cis.upenn.edu*
Chips & Technologies driver

Nozomi Ytow
Chips & Technologies driver

Egbert Eich, *Egbert.Eich@Physik.TH-Darmstadt.DE*
Chips & Technologies driver

David Bateman, *dbateman@ee.uts.edu.au*
Chips & Technologies driver

Xavier Ducoin, *xavier@rd.lectra.fr*
Chips & Technologies driver

Peter Trattler, *peter@sbox.tu-graz.ac.at*
RealTek SVGA driver

Craig Struble, *cstruble@acm.vt.edu*
Video7 SVGA driver

Gertjan Akkerman, *akkerman@dutiba.twi.tudelft.nl*
16 colour VGA server, and XF86Config parser.

Davor Matic, *dmatic@Athena.MIT.EDU*
Hercules driver.

Pascal Haible, *haible@izfm.uni-stuttgart.de*
Banked monochrome VGA support, Hercules support, and mono frame buffer support for dumb monochrome devices

Martin Schaller,

Geert Uytterhoeven,*Geert.Uytterhoeven@cs.kuleuven.ac.be*
Linux/m68k Frame Buffer Device driver

Andreas Schwab, *schwab@issan.informatik.uni-dortmund.de*
Linux/m68k Frame Buffer Device driver

Guenther Kelleter, *guenther@Pool.Informatik.RWTH-Aachen.de*
Linux/m68k Frame Buffer Device driver

Frederic Lepied, epied@XFree86.Org
XInput extension integration. Wacom, joystick and extended mouse drivers.

Patrick Lecoanet, ecoanet@cena.dgac.fr
Elographics touchscreen driver.

Steven Lang, iger@tyger.org
SummaSketch tablet driver.

... and many more people out there on the net who helped with beta-testing this enhancement.

XFree86 source is available from the FTP server *ftp.XFree86.org*, among others. Send email to *XFree86@XFree86.org* for details.

Ì¾Á°

XF86Config - XFree86 "ÍÑ¹½À®¾ðÊó¥Õ¥¡¥¤¥ë"

ÀâÌÀ

XFree86 ¤Ï½é´üÀßÄê¤Î¤¿¤á¤Î **XF86Config** ¤È¤¤¤¦¹½À®¾ðÊó¥Õ¥¡¥¤¥ë¤ò»ÈÍÑ ¤·¤Þ¤¹¡£¤³¤Î¹½À®¾ðÊó¥Õ¥¡¥¤¥ë¤Ï¼¡¤Î¾ì½ê¤«¤éÃµ¤·¤Þ¤¹¡£

/etc/XF86Config
<XRoot>/lib/X11/XF86Config.*hostname*
<XRoot>/lib/X11/XF86Config

¤³¤³¤Ç <XRoot> ¤Ï X11 ¤òÆ³Æþ¤·¤¿¥Ç¥£¥ì¥¯¥È¥ê¤Ç¤¹¡£ X ¥µ¡¼¥Ð¤ò 'root' ¥æ¡¼¥¶¤Ç³«»Ï¤·¤¿¤È¤-¡¢ **XF86Config** ¥Õ¥¡¥¤¥ë¤ò 'root' ¥æ¡¼¥¶¤Î¥Û¡¼¥à¥Ç¥£¥ì¥¯¥È¥ê¤«¤éÃµ¤·¤Þ¤¹¡£

¤³¤Î¥Õ¥¡¥¤¥ë¤Ï¤¤¤¯¤Ä¤«¤ÎÀá¤Ç¹½À®¤·¤Æ¤¤¤Þ¤¹¡£¤½¤ì¤¾¤ì¤ÎÀá¤Ï¼¡¤Î·Á¼°¤Ë ¤Ê¤Ã¤Æ¤¤¤Þ¤¹:

Section "*ÀáÌ¾¾Î*"
  Àá¹àÌÜ
  ...
EndSection

ÀáÌ¾¾Î¤Ï¼¡¤Î¤È¤ª¤ê¤Ç¤¹:

Files (¥Õ¥¡¥¤¥ë¤Î¥Ñ¥¹Ì¾)
Module (Æ°Åª¤Ê¥â¥¸¥å¡¼¥ë¤Î¥í¡¼¥É)
ServerFlags (¥µ¡¼¥Ð¥ª¥×¥·¥ç¥ó)
Keyboard (¥-¡¼¥Ü¡¼¥É¹½À®¾ðÊó)
Pointer (¥Ý¥¤¥ó¥¿¹½À®¾ðÊó)
Monitor (¥â¥Ë¥¿¡¼¤ÎÀâÌÀ)
Device (¥Ó¥Ç¥ª¥«¡¼¥ÉÅù¤ÎÀâÌÀ)
Screen (²èÌÌ¤Î¹½À®¾ðÊó)
XInput (³ÈÄ¥ÆþÎÏµ¡´ï¤Î¹½À®¾ðÊó)

Files Àá¤ÏÉ¸½à¤Î¥Õ¥©¥ó¥È¤Î¥Ñ¥¹¤È RGB ¥Ç¡¼¥¿¥Ù¡¼¥¹¤Ø¤Î¥Ñ¥¹¤ò »ØÄê¤·¤Þ¤¹¡£¤³¤ì¤é¤Î¥Ñ¥¹¤Ï¥³¥Þ¥ó¥É¥é¥¤¥ó¤«¤é¤âÀßÄê½ÐÍè¤Þ¤¹¡£¤³¤ÎÀá¤Ë µ-½Ò¤¹¤ëÆâÍÆ¤Ï¼¡¤Î¤è¤¦¤Ë½ñ¤-¤Þ¤¹:

FontPath *"path"*

¥Õ¥©¥ó¥È¤Î¸¡º÷·ÐÏ©(¥Ñ¥¹)¤òÀßÄê¤·¤Þ¤¹¡£¤³¤Î¥Ñ¥¹¤Ï X ¥µ¡¼¥Ð¤¬¥Õ¥©¥ó¥È ¥Ç¡¼¥¿¥Ù¡¼¥¹¤ò¸¡º÷¤¹¤ë¥Ç¥£¥ì¥¯¥È¥ê·²¤ò¥«¥ó¥Þ¤Ç¶èÀÚ¤Ã¤¿¥ê¥¹¥È¤Ç¤¹¡£ Ê£¿ô¤Î **FontPath** ¤¬µ-°Ü¤µ¤ì¤Æ¤¤¤ë¤È»×¤¤¤Þ¤¹¤¬¡¢¤³¤ì¤é¤ò·ë¹ç ¤·¤Æ¥µ¡¼¥Ð¤¬»ÈÍÑ¤¹¤ë¥Õ¥©¥ó¥È¥Ñ¥¹ (fontpath) ¤òºîÀ®¤·¤Þ¤¹¡£

X11R6 ¤Ï X ¥µ¡¼¥Ð¤¬¥Õ¥©¥ó¥È¥µ¡¼¥Ð¤Ë¥Õ¥©¥ó¥È¤òÍ×µá¤¹¤ë¤³¤È¤¬²ÄÇ½ ¤Ë¤Ê¤ê¤Þ¤·¤¿¡£¥Õ¥©¥ó¥È¥µ¡¼¥Ð¤Ï¼¡¤Î¾ì½ê¤ò "<trans>/<host-name>:<port_number>" ¤È¥Õ¥©¥ó¥È¥Ñ¥¹¤Ë»ØÄê¤·¤Þ¤¹¡£Îã¤¨¤Ð¡¢¥Õ¥©¥ó¥È¥Ñ¥¹ "/usr/X11R6/lib/X11/fonts/misc/,tcp/zok:7100" ¤Ï¤Þ¤ºÇ½é¤Ë¥í¡¼¥«¥ë ¥Ç¥£¥ì¥¯¥È¥ê /usr/X11R6/lib/X11/fonts/misc ¤ò X ¥µ¡¼¥Ð¤ËÅÁ¤¨¤Þ¤¹¡£ ¤³¤ì¤¬¼ºÇÔ¤·¤¿¤È¤-¤Ë¡¢¥Ý¡¼¥ÈÈÖ¹æ 7100 ¤«¤é TCP ¤ÇÀÜÂ³¤·¤ÆÊÌ¤¤¤Æ (listening) ¤¤¤ë zok ¤È¤¤¤¦¥Û¥¹¥È¤ÇÆ°ºî¤·¤Æ¤¤¤ë

¥Õ¥©¥ó¥È¥µ¡¼¥Ð ¤Ë¥Õ¥©¥ó¥È¤òÍ×µá¤·¤Þ¤¹¡£

RGBPath *"path"*

¤Ï RGB ¥«¥é¡¼¥Ç¡¼¥¿¥Ù¡¼¥¹ÍÑ¤Î¥Ñ¥¹Ì¾¾Î¤òÀßÄê¤·¤Þ¤¹¡£

ModulePath *"path"*

¤ÏÆ°Åª¥â¥¸¥å¡¼¥ë¤òÃµ°÷¤¹¤ë·ÐÏ©(¥Ñ¥¹)¤òÀßÄê¤·¤Þ¤¹¡£¤³¤Î¥Ñ¥¹¤Ï X ¥µ¡¼¥Ð¤¬ »ØÄê¤·¤¿½ç¤Ë¥í¡¼¥É¤¹¤ëÆ°Åª¥â¥¸¥å¡¼¥ë¤ò¸¡°÷¤¹¤ë¥Ç¥£¥ì¥¯¥È¥ê·²¤ò¥«¥ó¥Þ¤Ç¶èÀÚ¤Ã¤¿ ¥ê¥¹¥È¤Ç¤¹¡£ Ê£¿ô¤Î **ModulePath** ¤¬µº¤µ¤ì¤Æ¤¤¤ë¤È»×¤¤¤Þ¤¹¤¬¡¢¤³¤ì¤é¤ò·ë¹ç ¤·¤Æ¥µ¡¼¥Ð¤¬»ÈÍÑ¤¹¤ë¥â¥¸¥å¡¼¥ë¥Ñ¥¹ (modulepath) ¤ò°À®¤·¤Þ¤¹¡£ **Module** Àá¤Ï¥í¡¼¥É¤¹¤ëÆ°Åª¥â¥¸¥å¡¼¥ë¤ò»ØÄê¤¹¤ë¤È¤³¤í¤Ç¤¹¡£ ¸½¹Ô¤ÎÆ°Åª¥â¥¸¥å¡¼¥ë¤Ï XInput µ¡´ï¤Ç¤·¤«»ÈÍÑ¤·¤Æ¤¤¤Ê¤¤¤Æ¡¢¤¤¤¤¤Ä¤«¤Î ¥·¥¹¥Æ¥à(º£¤Î¤È¤³¤í Linux ELF, FreeBSD 2.x ¤È NetBSD 1.x) ¤Î¤ß¥µ¥Ý¡¼ ¥È¤·¤Æ¤¤¤Þ¤¹¡£

Load *"module"*

¤³¤ì¤Ï¥µ¡¼¥Ð¤Ë¥í¡¼¥É¤¹¤ë *"module"* ¤Î»ØÄê¤Ç¤¹¡£¥â¥¸¥å¡¼¥ë¤Ï ´°Á´¥Ñ¥¹Ì¾¤Ç»ØÄê¤·¤Ê¤¤¾ì¹ç¡¢ **ModulePath** ¤Ç»ØÄê¤·¤¿ ¥Ç¥£¥ì¥¯¥È¥ê¤«¤éÃµ¤·¤Þ¤¹¡£¥â¥¸¥å¡¼¥ë¤Ï°£¤Î¤È¤³¤í³ÈÄ¥¥ÆþÎÏµ¡´ï¤ò ¥µ¥Ý¡¼¥È¤·¤Æ¤¤¤Þ¤¹¡£¤½¤Î¾¤Ï¼¡¤Î¤È¤ª¤ê¤Ç¤¹:

xf86Elo.so
xf86Jstk.so
xf86Wacom.so
xf86Summa.so

PEX ¤È XIE µ¡Ç½³ÈÄ¥¤â¤¤¤¯¤Ä¤«¤Î¥·¥¹¥Æ¥à¤Ç¥â¥¸¥å¡¼¥ë¤È¤·¤ÆÍøÍÑ²ÄÇ½¤Ç¤¹¡£ ¤½¤Î¾¤Ï¼¡¤Î¤È¤ª¤ê¤Ç¤¹:

pex5.so
xie.so

ºÇ¿·¤Î°ì÷¤Ï <XRoot>/lib/modules ¤ò³Îç§¤·¤Æ¤¯¤À¤µ¤¤¡£

ServerFlags ¤ÎÀá¤Ï¤¤¤¯¤Ä¤«¤Î¿§¡¹¤Ê X ¥µ¡¼¥Ð¥ª¥×¥·¥ç¥ó¤ò »ØÄê¤¹¤ë¤È¤³¤í¤Ç¤¹¡£¤³¤ÎÀá¤ÇÍøÍÑ²ÄÇ½¤Ê¹àÌÜ¤Ï¼¡¤Î¤È¤ª¤ê¤Ç¤¹:

NoTrapSignals

¤³¤Î¹àÌÜ¤Ï X ¥µ¡¼¥Ð¤¬ÆÍÈ¯Åª¤ÊÃ×Ì¿Åª¤ÊÈÏ°Ï¤Î¥·¥°¥Ê¥ë¤òÊáÂª¤¹¤ë¤Î¤òËÉ»ß ¤·¡¢Àµ³ï¤Ë½ªÎ»¤µ¤»¤ë¤â¤Î¤Ç¤¹¡£¤³¤Î»ØÄê¤¬¤Ê¤¤¤È¡¢X ¥µ¡¼¥Ð¤ÏÄä»ß¤·¤Æ ÉÔÀµ¤¬¤µ¤¿¤Ê¤³¤È¤Ç¥³¥¢¥À¥ó¥×¤ò°À®¤·¤Þ¤¹¡£É¸½à¤ÎÆ°ºî¤Ï X ¥µ¡¼¥Ð¤¬ Àµ¾ï½ªÎ»¤·¤Þ¤¹¤¬¡¢¥³¥¢¥À¥ó¥×¤ò°À®¤·¤Þ¤¹¡£°ìÈÌ¤Ë X ¥µ¡¼¥Ð¤Î¥Ç¥Ð¥Ã¥° ¤ò¹Ô¤¦»þ¤ò½ü¤¤¤Æ¤³¤Î¥ª¥×¥·¥ç¥ó¤òÉ¬Í×¤È¤·¤Ê¤¤¤Ç¤·¤ç¤¦¡£

DontZap

¤³¤Î¹àÌÜ¤Ï **Ctrl+Alt+Backspace** ¤ÎÆþÎÏ¤òÌµ¸ú¤Ë¤·¤Þ¤¹¡£ ¤³¤Î¥¡¼ÆþÎÏ¤Ï X ¥µ¡¼¥Ð¤ò½ªÎ»¤µ¤»¤ë¤â¤Î¤Ç¤¹¡£ **DontZap** ¤òÀßÄê¤¹¤ë¤È ¤³¤Î¥¡¼ÆþÎÏ¤ò¥¯¥é¥¤¥¢¥ó¥È¤Ë¤½¤Î¤Þ¤ÞÅÏ¤·¤Þ¤¹¡£

DontZoom

¤³¤Î¹àÌÜ¤Ï **Ctrl+Alt+Keypad-Plus** ¤È**Ctrl+Alt+Keypad-Minus** ¤Î¥¡¼ÆþÎÏ¤ò»È¤¨¤Ê¤¤¤è¤¦¤Ë¤·¤Þ¤¹¡£¤³¤Î¥¡¼ÆþÎÏ¤Ï¥Ó¥Ç¥ª¥â¡¼¥É¤ÎÀÚ¤êÂØ¤¨¤ò ¹Ô¤¦¤â¤Î¤Ç¤¹¡£**DontZoom** ¤ÎÀßÄê¤ò¤¹¤ë¤È¡¢¤³¤Î¥¡¼ÆþÎÏ¤ò¥¯¥é¥¤¥¢¥ó¥È¤Ë ¤½¤Î¤Þ¤ÞÅÏ¤·¤Þ¤¹¡£

AllowNonLocalXvidtune
¤³¤Î¹àÌÜ¤Ï xvidtune ¥¯¥é¥¤¥¢¥ó¥È¤¬Â¾¤Î¥Û¥¹¥È¤«¤éÀÜÂ³¤¹¤ë¤³¤È¤òÍ¸ú¤Ë¤·¤Þ¤¹¡£ É¸½à¤Ç¤Ï¥í¡¼¥«¥ëÀÜÂ³°Ê³°¤Ï½ÐÍè¤Þ¤»¤ó¡£

DisableVidMode
¤³¤Î¹àÌÜ¤Ï ¥Ó¥Ç¥ª¥â¡¼¥É¤òÊÑ¹¹¤¹¤ë xvidtune ¥¯¥é¥¤¥¢¥ó¥È ¤¬»ÈÍÑ¤¹¤ë VidMode μ¡Ç½³ÈÄ¥¤òÌμ¸ú¤Ë¤·¤Þ¤¹¡£

AllowNonLocalModInDev
¤³¤Î¹àÌÜ¤Ï¥¯¥é¥¤¥¢¥ó¥È¤¬Â¾¤Î¥Û¥¹¥È¤«¤éÀÜÂ³¤·¤Æ¥¡¼¥Ü¡¼¥É¤È¥Þ¥¦¥¹¤ÎÀßÄê¤ò ²ÔÆ¯Ãæ¤Î¥μ¡¼¥Ð¤ÇÊÑ¹¹¤¹¤ë»ö¤òÍ¸ú¤Ë¤·¤Þ¤¹¡£É¸½à¤Ç¤Ï¥í¡¼¥«¥ëÀÜÂ³°Ê³°¤Ï½ÐÍè ¤Þ¤»¤ó¡£

DisableModInDev
¤³¤Î¹àÌÜ¤ÏÆþÎÏμ¡´ï¤ÎÀßÄê¤òÆ°Åª¤ËÊÑ¹¹¤Ç¤¤ë Misc μ¡Ç½³ÈÄ¥¤Î°ìÉô¤òÍ¸ú¤Ë ¤·¤Þ¤¹¡£

Keyboard Àá¤Ï¥¡¼¥Ü¡¼¥ÉÆþÎÏμ¡´ï¡¢¥Ñ¥é¥á¡¼¥¿¤ÈÈÉ¸½à¤Î¥¡¼³ä¤êÅö¤ÆÍ¤ò»ØÄê¤¹¤ë¤â¤Î¤Ç¤¹¡£¤³¤ÎÀá¤ËμÆþ½ÐÍè¤ëÆâÍÆ¤Ï¼¡¤Î¤È¤ª¤ê¤Ç¤¹:

Protocol *"kbd-protocol"*
¤Î *kbd-protocol* ¤Ï **Standard** ¤« **Xqueue** ¤Î¤É¤Á¤é¤«¤Ç¤¹¡£ **Xqueue** ¤Ï SVR3 ¤« SVR4 ¤Î¥¤¥Ù¥ó¥È¥¥å¡¼¥É¥é¥¤¥Ð¤ò»ÈÍÑ¤¹¤ë»þ¤Ë »ØÄê¤¹¤ë¤â¤Î¤Ç¤¹¡£

AutoRepeat *delay rate*
¤Ï¥¡¼¥Ü¡¼¥É¤Î¥ª¡¼¥È¥ê¥Ô¡¼¥È¤ÎÀßÄê¤òÊÑ¹¹¤¹¤ë¤â¤Î¤Ç¤¹¡£¤³¤Î¥ª¥×¥·¥ç¥ó¤Ï Á´¤Æ¤Îμ¡´ï¤ÇÆ°ºî¤¹¤ë¤â¤Î¤Ç¤Ï¤¢¤ê¤Þ¤»¤ó¡£

ServerNumLock
¤Ï X ¥μ¡¼¥Ð¤ÎÆâÉô¤Ç¥Ê¥à¥í¥Ã¥¯¥¡¼(num-lock)¤ò°·¤¦¤è¤¦¤Ë¤¹¤ë¤â¤Î¤Ç¤¹¡£ X ¥μ¡¼¥Ð¤Ï¥Ê¥à¥í¥Ã¥¯¥¡¼¤¬¥ª¥ó¤Ë¤Ê¤Ã¤Æ¤¤¤ë¤È¤¤Ë¤Ï°Û¤Ê¤ë¥³¡¼¥É ¤ÎÀÈ¤ß¹ç¤ï¤»¤òÁ÷½Ð¤·¤Þ¤¹¤³¤Î¥ª¥×¥·¥ç¥ó¤¬Í¸ú¤Ë¤Ê¤Ã¤Æ¤¤¤ë¤È¥¢¥×¥ê ¥±¡¼¥·¥ç¥ó¤Ç¥Æ¥ó¥¡¼¤¬»È¤¨¤ë¤è¤¦¤Ë¤Ê¤ê¤Þ¤¹¡£

LeftAlt *mapping*
RightAlt *mapping*
AltGr *mapping*
ScrollLock *mapping*
RightCtl *mapping*
¤³¤ì¤é¤Ï¥¡¼¤¬É¸½à¤Î¥¡¼³ä¤êÅö¤Æ¤Ç¤¹(**AltGr** ¤Ï **RightAlt** ¤ÈÆ±°Õ¸ì)¡£*mapping* ¤Ë¼¡¤Î¥¡¼³¡¼¥É¤ò³ä¤êÅö¤Æ¤·¤Þ¤¹:

Meta
Compose
ModeShift
ModeLock
ScrollLock
Control

¤É¤Î¥ª¥×¥·¥ç¥ó¤â»ØÄê¤μ¤ì¤Ê¤«¤Ã¤¿¾ì¹ç¤ÎÉ¸½àÃÍ¤Ï¼¡¤Î¤È¤ª¤ê¤Ç¤¹:

LeftAlt Meta
RightAlt Meta
ScrollLock Compose

RightCtl Control

XLeds *led* **...**
¤Ï¥¯¥é¥¤¥¢¥ó¥È¤Ë$µ$ì¼°¤Î$µ$¡Ç½(¥¹¥¯¥í¡¼¥ë¥í¥Ã¥¯¡¢¥¥ã¥Ã¥×¥¹¥í¥Ã¥¯¤È¥Ê¥à¥í¥Ã¥¯)¤Î
Âå¤ï¤ê¤Ë *led* ¤òÍøÍÑ²ÄÇ½¤Ë¤·¤Þ¤¹¡£ *led* ¤Ï 1 ¤«¤é 3 ¤ÎÈÏ°Ï
¤Î¿ôÃÍ¤Î¥ê¥¹¥È¤Ç¤¹¡£

VTSysReq
¤Ï VT ÀÚ¤êÂØ¤¨¤ò»ý¤ÄÈó SYSV ¥·¥¹¥Æ¥à¤Ë SYSV »ÅÍÍ¤Î VT
ÀÚ¤êÂØ¤¨
¼ê½ç¤ò²ÄÇ½¤Ë¤·¤Þ¤¹¡£¤³¤Î¼ê½ç¤Ï¥Õ¥¡¥ó¥¯¥·¥ç¥ó¥¡¼(Fn)¤Î¸å¤Ë Alt-
SysRq ¤¬Á÷½Ð¤$µ$ì¤¤Þ¤¹¡£¤³¤Î¹àÌÜ¤Ï X ¥$µ$¡¼¥Ð¤¬É¸½à¤Î VT
ÀÚ¤êÂØ¤¨¼ê½ç ¤òÊáÂª¤·¤Ê¤¤¤è¤¦¤Ë¤·¤Þ¤¹¡£

VTInit *"command"*
¤Ï *command* ¤ò¥$µ$¡¼¥Ð¤¬»È¤Ã¤Æ¤¤¤ë VT ¤ò³«¤¤¤¿¸å¤Ç¼Â¹Ô¤·¤Þ¤¹¡£
¥³¥Þ¥ó¥ÉÊ¸»úÎó¤Ï "/bin/sh -c" ¤ËÅÏ¤$µ$ì¡¢Æâ¤É¥æ¡¼¥¶¡¼ID ¤Ç¼Â¹Ô¤$µ$ì
É¸½àÆþ½ÐÎÏ¤Ï Åö³º VT ¤Ë³ä¤êÅö¤¿¤ê¤Þ¤¹¡£¤³¤Î¹àÌÜ¤ÎÌÜÅª¤Ï
¥·¥¹¥Æ¥à¤Ë°Í¸¤·¤¿ VT
¤Î½é´ü²½¥³¥Þ¥ó¥É¤ò¼Â¹Ô¤¹¤ë¤³¤È¤ò²ÄÇ½¤Ë¤¹¤ë¤³¤È
¤Ç¤¹¡£¤³¤Î¤è¤¦¤Ê¥·¥¹¥Æ¥à¤Ç¤ÏÉ¸½à¤Ç£²¤Ä¤Î¥¡¼Æþ¤ÎÏ¤Ç¤Î VT
ÀÚ¤êÂØ¤¨¤ò Ì$µ$¸ú¤Ë¤¹¤ë¥³¥Þ¥ó¥É¤¬¤½¤Î¤ã¤Ç¤¹¡£

XkbDisable
¤Ï XKEYBOARD $µ$¡Ç½³ÈÄ¥¤òÌ$µ$¸ú¤Ë¤·¤Þ¤¹¡£ -kb
¥³¥Þ¥ó¥É¥ª¥×¥·¥ç¥ó¤ÈÆ±Åù¤Ç¤¹¡£

XkbRules *"rulesfile"*
XkbModel *"model"*
XkbLayout *"layout"*
XkbVariant *"variant"*
XkbOptions *"optionlist"*
¤³¤ì¤é¤Ï XKEYBOARD
¹½À®Í×ÁÇ¤¬Ç§¼±¤¹¤ë¹àÌÜ¤òÆÃÄê¤¹¤ë¤¿¤á¤Ë»ØÄê¤·¤Þ¤¹¡£ optionlist
¤Ï¥ª¥×¥·¥ç¥ó¤ò¥«¥ó¥Þ¤Ç¶èÀÚ¤Ã¤¿¥ê¥¹¥È¤Ç¤¹¡£

XkbRules "xfree86"
XkbModel "pc101"
XkbLayout "us"
XkbVariant ""
XkbOptions ""

¤³¤Î¹àÌÜ¤Ï¥¡¼¥Ü¡¼¥É¹½À®¤ò»ØÄê¤¹¤ë¹¥¤Þ¤·¤¤ÊýË¡¤Ç¤¹¤¬¡¢¤Þ¤¿¼¡¤Î¤è¤¦¤ËÄ¾ÀÜ¡¢
¹½À®Í×ÁÇ¤ò»ØÄê¤Ç¤¤Þ¤¹:

XkbKeymap *"keymap"*
XkbKeycodes *"keycodes"*
XkbTypes *"types"*
XkbCompat *"compat"*
XkbSymbols *"symbols"*
XkbGeometry *"geometry"*
¤¤¤¯¤Ä¤«¤Î¹½À®Í×ÁÇ¤À¤±¤ò»ØÄê¤·¤¿¾ì¹ç¡¢»Ä¤ê¤Î¹½À®Í×ÁÇ¤Ï¼¡¤Î¤è¤¦¤ËÉ¸½àÃÍ¤ò
»ÈÍÑ¤·¤Þ¤¹:

XkbKeymap none

```
XkbKeycodes  "xfree86"
XkbTypes     "default"
XkbCompat    "default"
XkbSymbols   "us(pc101)"
XkbGeometry  "pc"
```

Pointer Àá¤Ï¥Þ¥¦¥¹Åù¤È¤½¤Î¥Ñ¥é¥á¥¿¤ò»ØÄê¤·¤Þ¤¹¡£
¤³¤ÎÀá¤Ç»ØÄê²ÄÇ½¤Ê¹àÌÜ¤Ï¼¡¤Î¤È¤ª¤ê¤Ç¤¹:

Protocol *"protocol-type"*
¤Ï¥Þ¥¦¥¹Åù¤Î¥×¥í¥È¥³¥ë¤Î¼ïÎà¤ò»ØÄê¤·¤Þ¤¹¡£»ØÄê²ÄÇ½¤Ê¥×¥í¥È¥³¥ë¤Î¼ïÎà¤Ï¼¡¤Î¤È¤ª¤ê¤Ç¤¹:

BusMouse
Logitech
Microsoft
MMSeries
Mouseman
MouseSystems
PS/2
MMHitTab
GlidePoint
Xqueue
OSMouse

Logitech ¥Ð¥¹¥Þ¥¦¥¹¤Ë¤Ï **BusMouse** ¤ò»ØÄê¤·¤Þ¤·¤ç¤¦¡£¤Þ¤¿¡¢Â¿¤¯¤Î¿·¤·¤¤ Logitech ¥·¥ê¥¢¥ë¥Þ¥¦¥¹¤Ï**Microsoft** ¤Þ¤¿¤Ï **MouseMan** ¥×¥í¥È¥³¥ë¤ò»ÈÍÑ¤·¤Þ¤¹¡£
Keyboard Àá¤Ç **Xqueue** ¤ò»ØÄê¤·¤¿¾ì¹ç¤Ï¤³¤ÎÀá¤Ë **Xqueue** ¤ò»ØÄê¤·¤Þ¤·¤ç¤¦¡£**OSMouse** ¤ÏSCO ¤Î SVR3 ¤Ç Í¸ú¤Ê¥¤¥Ù¥ó¥È¶îÆ°·¿¤Î¥Þ¥¦¥¹¥¤¥ó¥¿¥Õ¥§¡¼¥¹¤ËÅö¤Æ¤Ï¤Þ¤ê¤Þ¤¹¡£¤½¤Î¸å¤Ç¡¢¥Þ¥¦¥¹¤Î¥Ü¥¿¥ó¤Î¿ô¤ò»ØÄê¤·¤Þ¤·¤ç¤¦¡£

PS/2 ¥Ý¡¼¥È¤ËÀÜÂ³¤µ¤ì¤¿¥Þ¥¦¥¹¤ò»ÈÍÑ¤¹¤ë¾ì¹ç¤Ï¡¢»ÈÍÑ¤¹¤ë¥Þ¥¦¥¹¤Î·¿¤Ë¤«¤«¤ï¤é¤º **PS/2**¤ò»ØÄê¤·¤Æ¤¯¤À¤µ¤¤¡£

Device *"pointer-dev"*
¤Ï¥µ¡¼¥Ð¤¬¥Æ¥þÎÏ¤µ¡´ïÍÑ¤Ë¥ª¡¼¥×¥ó¤¹¤ë¥Ç¥Ð¥¤¥¹¥Õ¥¡¥¤¥ë (Îã¤¨¤Ð, **/dev/tty00** ¤Þ¤¿¤Ï **/dev/mouse**) ¤ò»ØÄê¤·¤Þ¤¹¡£
Xqueue ¤Þ¤¿¤Ï **OSMouse** ¥×¥í¥È¥³¥ë¤ò»ÈÍÑ¤¹¤ë¤È¤¤Ï ¥Ç¥Ð¥¤¥¹¥Õ¥¡¥¤¥ë¤Ï»ØÄê¤·¤Ê¤¤¤Ç¤¯¤À¤µ¤¤¡£

Port *"pointer-dev"*
¤Ï **Device** ¹àÌÜ¤ÈÆ±¤¸¤â¤Î¤Ç¤¹¡£

BaudRate *rate*
¤Ï¥·¥ê¥¢¥ë¥Þ¥¦¥¹¤Î¥Ü¡¼¥ì¡¼¥È(ÄÌ¿®Â®ÅÙ)¤ò *rate* ¤ËÀßÄê¤·¤Þ¤¹¡£Æ°Åª¤ËÄÌ¿®Â®ÅÙ¤òÄ´Àá¤¹¤ë (Logitech ¤Î¤è¤¦¤Ê) ¥Þ¥¦¥¹¤Ç¤ÏÄÌ¿®Â®ÅÙ¤Ï ¥Þ¥¦¥¹¤ÎÊý¤ÇÊÑ¹¹¤·¤Þ¤¹¡£¤µ¤â¤Ê¤±¤ì¤Ð¡¢Ã±½ã¤ËÊ³¥ó¥Ó¥Ô¥å¡¼¥¿¡Â¦¤Ç ¥Þ¥¦¥¹¤¬¬ÃÐ±þ¤·¤Æ¤¤¤ëÉ¸½à°Ê³°¤ÎÄÌ¿®Â®ÅÙ¤ÎÀßÄê¤·¤Þ¤·¤ç¤¦ (É¸½à¤ÎÄÌ¿®Â®ÅÙ¤Ï 1200 ¤Ç¤¹)¡£

Emulate3Buttons
¤ÏÊªÍýÅª¤Ë£²¤ÄÂ¤·¤«¥Ü¥¿¥ó¤¬Ì¤¤¤¥Þ¥¦¥¹¤Ç£³¤Ä¤á¤Î¥Ü¥¿¥ó¤Î¥¨¥ß¥å¥ì¡¼¥·¥ç¥ó¤ò

²ÄÇ½¤Ë¤·¤Þ¤¹¡££³¤Ä¤á¤Î¥Ü¥¿¥ó¤ÏÆ±»þ¤ËÎ¾Êý¤Î¥Ü¥¿¥ó¤ò²¡¤¹¤³¤È¤Ç¥¨¥ß¥å¥ì¡¼¥·¥ç¥ó¤·¤Þ¤¹¡£

Emulate3Timeout *timeout*

¤Ï£³¥Ü¥¿¥ó¥¨¥ß¥å¥ì¡¼¥·¥ç¥ó¤¬Í¸ú¤Ë¤Ê¤Ã¤Æ¤¤¤ë¤È¤¡¢"Æ±»þ"¤Ë£²¤Ä¤Î¥Ü¥¿¥ó¤¬²¡¤µ¤ì¤¿¤«¤É¤¦¤«·èÄê¤¹¤ëÁ°¤ÎÂÔ¤Á»þ´Ö¤ò (¥ß¥êÉÃÃ±°Ì¤Ç) ÀßÄê¤·¤Þ¤¹¡£ É¸½à¤ÎÂÔ¤Á»þ´Ö(¥¿¥¤¥à¥¢¥¦¥È)¤Ï 50ms ¤Ç¤¹¡£

ChordMiddle

¤Ï(¤¤¤¯¤Ä¤«¤Î Logitech Mouseman ¥Þ¥¦¥¹¤Î¤è¤¦¤Ë) Ãæ±û¤Î¥Ü¥¿¥ó¤¬²¡¤µ¤ì¤¿¤È¤¡¢¢±¦¤Ë°¸¤Î¥Ü¥¿¥ó¤¬²¡¤µ¤ì¤¿¤è¤¦¤Ê¿®¹æ¤òÁ÷¿®¤¹¤ë¤è¤¦¤Ë¥Þ¥¦¥¹¤ò¼è¤ê°·¤¤¤Þ¤¹¡£

SampleRate *rate*

¤ÏËèÉÃ¤Î¥Þ¥¦¥¹°ÜÆ°ÎÌ/¥Ü¥¿¥ó¤Î¥¤¥Ù¥ó¥ÈÈ¯À¸ÎÌ¤òÀßÄê¤·¤Þ¤¹¡£°£¤Î¤È¤³¤í¤³¤Î¥ª¥×¥·¥ç¥ó¤Ï¤¤¤¯¤Ä¤«¤Î Logitech ¥Þ¥¦¥¹¤Î¤ß¤ÇÍøÍÑ²ÄÇ½¤Ç¤¹¡£

ClearDTR

¤³¤Î¥ª¥×¥·¥ç¥ó¤Ï¥Þ¥¦¥¹¤Ç»ÈÍÑ¤¹¤ë¥·¥ê¥¢¥ë¥Ý¡¼¥È¤Î DTR ¿®¹æ¤ò¥ê¥»¥Ã¥È ¤·¤Þ¤¹¡£¤³¤Î¥ª¥×¥·¥ç¥ó¤Ï **MouseSystems** ¥×¥í¥È¥³¥ë¤ò»ÈÍÑ¤¹¤ë¥Þ¥¦¥¹¤Î¾ì¹ç¤ËÍ¸ú¤Ç¤¹¡££²¤Ä¤Î¥×¥í¥È¥³¥ë¤¬»È¤¨¤ë¥Þ¥¦¥¹¤Ç MouseSystems ¥â¡¼¥É¤ò»È¤¦¤È¤¤Ï DTR ¿®¹æ¤ò¥ê¥»¥Ã¥È¤¹¤ëÉ¬Í×¤¬¤¢¤ê¤Þ¤¹¡£Äí°Õ»ö¹à ¤È¤·¤ÆÉ¡¢2.1 ¤è¤êÁ°¤Î¥Ð¡¼¥¸¥ç¥ó¤Î XFree86 ¤Ç¤Ï¤³¤Î¥ª¥×¥·¥ç¥ó¤Ï RTS ¿®¹æ¤â¥ê¥»¥Ã¥È¤·¤Þ¤¹¡£Ê¬¤¥¤µ¤ì¤¿ **ClearRTS** ¥ª¥×¥·¥ç¥ó¤Ï RTS ¤Î¥ê¥»¥Ã¥È¤¬É¬Í×¤Ê¥Þ¥¦¥¹¤Î°Ù¤ËÀß¤±¤é¤ì¤Þ¤·¤¿¡£

ClearRTS

This option clears the RTS line on the serial port used by the mouse. This option is only valid for a mouse using the **MouseSystems** protocol. Some dual-protocol mice require both DTR and RTS to be cleared to operate in MouseSystems mode. Both the **ClearDTR** and **ClearRTS** options should be used for such mice. ¤³¤Î¥ª¥×¥·¥ç¥ó¤Ï¥Þ¥¦¥¹¤Ç»ÈÍÑ¤¹¤ë¥·¥ê¥¢¥ë¥Ý¡¼¥È¤Î RTS ¿®¹æ¤ò¥ê¥»¥Ã¥È ¤·¤Þ¤¹¡£¤³¤Î¥ª¥×¥·¥ç¥ó¤Ï **MouseSystems** ¥×¥í¥È¥³¥ë¤ò»ÈÍÑ¤¹¤ë¥Þ¥¦¥¹¤Î¾ì¹ç¤ËÍ¸ú¤Ç¤¹¡££²¤Ä¤Î¥×¥í¥È¥³¥ë¤¬»È¤¨¤ë¥Þ¥¦¥¹¤Ç MouseSystems ¥â¡¼¥É¤ò»È¤¦¤È¤¤Ï DTR ¿®¹æ¤È RTS ¿®¹æ¤ò¥ê¥»¥Ã¥È¤¹¤ëÉ¬Í×¤¬¤¢¤ê¤Þ¤¹¡£ **ClearDTR** ¤È **ClearRTS** ¤ÎÎ¾Êý¤Î¥ª¥×¥·¥ç¥ó¤Ï£²¤Ä¤Î¥×¥í¥È¥³¥ë ¤¬»È¤¨¤ë¥Þ¥¦¥¹¤Ç¤ÏÍ¸ú¤Ç¤¹¡£

Monitor Àá¤Ï¥â¥Ë¥¿¡¼¤Î»ÅÍÍ¤È¥â¥Ë¥¿¡¼¤Ç»ÈÍÑ¤¹¤ë¤Î¤Ë¹çÃ×¤¹¤ë¥Ó¥Ç¥ª¥â¡¼¥É¤Î°ìÍ÷¤òÄêµÁ¤·¤Þ¤¹¡£XF86Config ¥Õ¥¡¥¤¥ë¤Î¤Ê¤«¤Ë°ì¤Ä°Ê¾å ¤Î **Monitor** Àá¤¬¢¤¤ë¤È»×¤¤¤Þ¤¹¡£¤³¤ÎÀá¤ÎÆâÍÆ¤Ï¼¡¤Î¤È¤ª¤ê¤Ç¤¹:

Identifier *"ID string"*

¤³¤ì¤Ï°Ê¹ß¤Î **Screen** Àá¤¬»²¾È¤¹¤ë¤«¤â¤·¤ì¤Ê¤¤¥â¥Ë¥¿¡¼Ì¾¤ò Ê¸»úÎó¤Ç»ØÄê¤·¤Þ¤¹¡£¤½¤ì¤¾¤ì¤Î **Monitor** Àá¤ÏÌ¾¾Î¤Ï½Å¤Ê¤é¤Ê¤¤¤ÍÍ¤Ê Ê¸»úÎó¤òÀßÄê¤·¤Æ¤¯¤À¤µ¤¤¡£

VendorName *"vendor"*

¤³¤ÎÄÉ²Ã¹àÌÜ¤Ï¥â¥Ë¥¿¡¼¤ÎÀ½Â¤²ñ¼Ò¤ò»ØÄê¤·¤Æ¤¯¤À¤µ¤¤

ModelName *"model"*

¤³¤ÎÄÉ²Ã¹àÌÜ¤Ï¥â¥Ë¥¿¡¼¤Î¥â¥Ç¥ëÌ¾¾Î(·¿Ì¾)¤ò»ØÄê¤·¤Æ¤¯¤À¤µ¤¤¡£

HorizSync *horizsync-range*
¤Ï¥â¥Ë¥¿¡¼¤Î»ÅÍÍÆâ¤Î¿åÊ¿Æ±´ü¼þÇÈ¿ô¤ÎÂÓ°èÉý¤òÀßÄê¤·¤Þ¤¹¡£ *horizsync-range* ¤Ï¥«¥ó¥Þ¤Ç¶èÀÚ¤é¤ì¤¿Î¥»¶¤·¤¿¿ôÃÍ¤«Éý¤Î¤¢¤ë¿ôÃÍ¤Î¥ê¥¹¥È¤Ç¤¹¡£Éý¤Î¤¢¤ë¿ôÃÍ¤ÏÆó¤Ä¤Î¿ô»ú¤ò¥Ï¥¤¥Õ¥ó¤Ç¶èÀÚ¤Ã¤¿¤â¤Î¤Ç¤¹¡£É¸½à¤Ç¤Ï¤³¤ì¤é¤Î¿ôÃÍ¤ÎÃ±°Ì¤Ï kHz ¤Ç¤¹¡£**MHz** ¤ä **Hz** ¤ò¹Ô¤Î¸å¤í¤ËÉÕ¤±¤ë¤È MHz ¤ä Hz ¤Ç»ØÄê¤¹¤ë¤³¤È¤Ë¤Ê¤ê¤Þ¤¹¡£¤³¤³¤ÇÍ¿¤¨¤ë¿ôÃÍ¤¬¥Ó¥Ç¥ª¥â¡¼¥É¤¬¥â¥Ë¥¿¡¼¤Î»ÅÍÍ¤Ë¼ý¤Þ¤Ã¤Æ¤¤¤ë¾ì¹ç¤Ï X ¥µ¡¼¥Ð
¤ËÇ§¼±¤µ¤ì¤Þ¤¹¡£¤³¤Î»ÅÍÍ¤Ï¥â¥Ë¥¿¡¼¤Î¼è°·ÀâÌÀ½ñ¤Ë½ñ¤¤¤Æ¤¢¤ë¤Ç¤·¤ç¤¦¡£

VertRefresh *vertrefresh-range*
¤Ï¥â¥Ë¥¿¡¼¤Î»ÅÍÍÆâ¤Î¿âÄ¾Æ±´ü¼þÇÈ¿ô¤ÎÂÓ°èÉý¤òÀßÄê¤·¤Þ¤¹¡£ *vertrefresh-range* ¤Ï¥«¥ó¥Þ¤Ç¶èÀÚ¤é¤ì¤¿Î¥»¶¤·¤¿¿ôÃÍ¤«Éý¤Î¤¢¤ë¿ôÃÍ¤Î¥ê¥¹¥È¤Ç¤¹¡£Éý¤Î¤¢¤ë¿ôÃÍ¤ÏÆó¤Ä¤Î¿ô»ú¤ò¥Ï¥¤¥Õ¥ó¤Ç¶èÀÚ¤Ã¤¿¤â¤Î¤Ç¤¹¡£É¸½à¤Ç¤Ï¤³¤ì¤é¤Î¿ôÃÍ¤ÎÃ±°Ì¤Ï kHz ¤Ç¤¹¡£**MHz** ¤ä **Hz** ¤ò¹Ô¤Î¸å¤í¤ËÉÕ¤±¤ë¤È MHz ¤ä Hz ¤Ç»ØÄê¤¹¤ë¤³¤È¤Ë¤Ê¤ê¤Þ¤¹¡£¤³¤³¤ÇÍ¿¤¨¤ë¿ôÃÍ¤¬¥Ó¥Ç¥ª¥â¡¼¥É¤¬¥â¥Ë¥¿¡¼¤Î»ÅÍÍ¤Ë¼ý¤Þ¤Ã¤Æ¤¤¤ë¾ì¹ç¤Ï X ¥µ¡¼¥Ð
¤ËÇ§¼±¤µ¤ì¤Þ¤¹¡£¤³¤Î¾ðÊó¤Ï¥â¥Ë¥¿¡¼¤Î¼è°·ÀâÌÀ½ñ¤Ë½ñ¤¤¤Æ¤¢¤ë¤Ç¤·¤ç¤¦¡£

Gamma *gamma-value(s)*
¥â¥Ë¥¿¡¼¤Î¥¬¥ó¥ÞÊäÀµ¤ÎÀßÄê¤ò¹Ô¤¦¤¿¤á¤ÎÄÉ²Ã¹àÌÜ¤Ç¤¹¡££±¤Ä¤Î¿ôÃÍ¤« RGB ¤½¤ì¤¾¤ìÂÐ±þ¤Î£³¤Ä¤Î¿ôÃÍ¤òÀßÄê¤·¤Þ¤¹¡£¤³¤Î¾ðÊó¤ÏÁ´¤Æ¤Î X ¥µ¡¼¥Ð ¤Ç»È¤¨¤ëÌõ¤Ç¤Ï¤¢¤ê¤Þ¤»¤ó¡£

Mode *"name"*
¤ÏÊ£¿ô¹Ô¤Ç»ØÄê¤¹¤ë¥Ó¥Ç¥ª¥â¡¼¥É¤Îµ½Ò¤Î»Ï¤Þ¤ê¤òÉ½¤·¤Þ¤¹¡£¥Ó¥Ç¥ª¥â¡¼¥É¤Îµ½Ò¤Î½ª¤ï¤ê¤Ï **EndMode** ¤Ç¤¹¡£¥Ó¥Ç¥ª¥â¡¼¥É¤Îµ½Ò¤ÎÆâÍÆ¤Ï¼¡¤Î¤È¤ª¤ê¤Ç¤¹:

DotClock *clock*
¤Ï¤½¤Î¥â¡¼¥É¤Ç»ÈÍÑ¤¹¤ë¥É¥Ã¥È¥¯¥í¥Ã¥¯¤Î¼þÇÈ¿ô¤Ç¤¹¡£

HTimings *hdisp hsyncstart hsyncend htotal*
¤Ï¿åÊ¿Æ±´ü¤ÎÄ´À°ÃÍ¤ò»ØÄê¤·¤Þ¤¹¡£

VTimings *vdisp vsyncstart vsyncend vtotal*
¤Ï¿âÄ¾Æ±´ü¤ÎÄ´À°ÃÍ¤ò»ØÄê¤·¤Þ¤¹¡£

Flags *"flag" ...*
¤Ï¥â¡¼¥É¥Õ¥é¥°¤ÎÄÉ²ÃÀßÄê¤ò»ØÄê¤·¤Þ¤¹¡£ **"Interlace"** ¤Ï¥¤¥ó¥¿¡¼¥ì¡¼¥¹¥â¡¼¥É¤Ç¤¢¤ë¤³¤È¤ò»Ø¤·¤Þ¤¹¡£ **"DoubleScan"** ¤Ï¤½¤ì¤¾¤ì¤ÎÁöººÀþ¤òÆó½Å¤Ë¤¹¤ë¥â¡¼¥É¤Ç¤¹¡£ **"+HSync"** ¤È **"-HSync"** ¤Ï¿åÊ¿Æ±´ü¿®¹æ¤Î¶ËÀ¤òÁªÂò¤·¤Þ¤¹¡£ **"+VSync"** ¤È **"-VSync"** ¤Ï¿âÄ¾Æ±´ü¿®¹æ¤Î¶ËÀ¤òÁªÂò¤·¤Þ¤¹¡£ **"Composite"** ¤Ï¥°¥é¥Õ¥£¥Ã¥¯¥Ü¡¼¥É¤¬ÂÐ±þ¤·¤Æ¤¤¤ì¤Ð¡¢¥³¥ó¥Ý¥¸¥Ã¥ÈÆ±´ü¤ò»ØÄê¤·¤Þ¤¹¡£ÉÕ¤±²Ã¤¨¤ì¤Ð¡¢¤¤¤¯¤Ä¤«¤Î¥°¥é¥Õ¥£¥Ã¥¯¥Ü¡¼¥É¤Ç¤Ï **"+CSync"** ¤È **"-CSync"** ¤Ç¥³¥ó¥Ý¥¸¥Ã¥ÈÆ±´ü¤Î¶ËÀ¤¬ÁªÂò½ÐÍè¤ë¤Ç¤·¤ç¤¦¡£

HSkew *hskew*
²èÌÌ¤Î±¦Ã¼¤ËÂÐ¤·¤ÆÉ½¼¨²ÄÇ½¿®¹æ¤ò·¹¤±¤ë¥Ô¥¯¥»¥ë¿ô¤ò»ØÄê¤·¤Þ¤¹¡£Á´¤Æ¤Î¥µ¡¼¥Ð¤Ç
¤³¤Î¾ðÊó¤ò»ÈÍÑ¤·¤Þ¤»¤ó¡£¤³¤Î¥ª¥×¥·¥ç¥ó¤Ï¥µ¡¼¥Ð¤¬Äó¶¡¤·¤Æ¤¤¤ë
(¤â¤·¤¢¤ì¤Ð)
É¸½à¤ÎÃÍ¤òÃÖ¤´¹¤¨¤ë¤¿¤á¤ËÉ¬Í×¤¬¤¢¤ì¤Ð»ØÄê¤·¤Þ¤·¤ç¤¦¡£¿åÊ¿Àþ¤¬"¤Õ¤é¤Ä¤¤¤Æ¤¤¤ë"¾ì¹ç¤Ï

¤³¤Î¿ôÃÍ¤òÁý²Ã¤·¤Æ¤¯¤À¤µ¤¤¡£Áö°ºÀþ¤Î°Ç¸å¤Î¿ô¥Ô¥¯¥»¥ë¤¬²èÌÌ¤Î°¸¤Ë¤«¤«¤Ã¤Æ¤¤¤ë
¾ì¹ç¤Ï¤³¤Î¿ôÃÍ¤ò¸º¤é¤·¤Æ¤¯¤À¤µ¤¤¡£

Modeline *"name" mode-description*

¤Ï¥Ó¥Ç¥ª¥â¡¼¥É¤ÎÀßÄê¤ò£±¹Ô¤Ç¹Ô¤¦¤â¤Î¤Ç¤¹¡£*mode-description* ¤Ï £´¤Ä¤ÎÉôÊ¬¤ËÊ¬¤«¤ì¤Æ¤¤¤Æ¡¢°Ç½é¤Î£³¤Ä¤ÏÉ¬¿Ü¤Ç¤¹¡£°Ç½é¤Ï¥Ô¥¯¥»¥ë ¥¯¥í¥Ã¥¯¤Ç¤¹¡£ÃÐ±þ¤¹¤ë¥â¡¼¥É¤Î¥Ô¥¯¥»¥ë¥¯¥í¥Ã¥¯¤Î¼þÇÈ¿ô¤ò£±¤Ä¤Î¿ôÃÍ¤Ç »ØÄê¤·¤Þ¤¹¡£¼¡¤ÎÉôÊ¬¤Ï¿åÊ¿Êý¸þ¤ÎÄ´À°ÃÍ¤Ç¡¢£´¤Ä¤Î¿ôÃÍ¤ò¥ê¥¹¥È¤Ç»ØÄê ¤·¤Þ¤¹¡£¤½¤ì¤¾¤ì¤Î¿ôÃÍ¤Ï *hdisp*, *hsyncstart*, *hsyncend*, *htotal* ¤Ç¤¹¡££³ÈÖÌÜ¤ÎÉôÊ¬¤Ï¿âÄ¾Êý¸þ¤ÎÄ´À°ÃÍ¤Ç¡¢£´¤Ä¤Î¿ôÃÍ¤ò ¥ê¥¹¥È¤Ç»ØÄê¤·¤Þ¤¹¡£¤½¤ì¤¾¤ì¤Î¿ôÃÍ¤Ï *vdisp*, *vsyncstart*, *vsyncend*, *vtotal* ¤Ç¤¹¡£°Ç¸å¤ÎÉôÊ¬¤Ï¤½¤Î¥â¡¼¥É¤ÎÆÃÄ§ ¤ò»ØÄê¤¹¤ë¥Õ¥é¥°¤Î¥ê¥¹¥È¤Ç¤¹¡£**Interlace** ¤Ï¥¤¥ó¥¿¡¼¥ì¡¼¥¹¥â¡¼¥É ¤ò»Ø¤·¤Þ¤¹¡£**"DoubleScan"** ¤ÏÁö°ºÀþ¤òÆó½Å¤Ë¤¹¤ë¥â¡¼¥É¤Ç¤¹¡£ **"+HSync"** ¤È **"-HSync"** ¤Ï¿åÊ¿Æ±´ü¿®¹æ¤Î¶ËÀ¤òÁªÂò¤·¤Þ¤¹¡£ **"+VSync"** ¤È **"-VSync"** ¤Ï¿âÄ¾Æ±´ü¿®¹æ¤Î¶ËÀ¤òÁªÂò¤·¤Þ¤¹¡£ **"Composite"** ¤Ï¥°¥é¥Õ¥£¥Ã¥¯¥Ü¡¼¥É¤¬ÂÐ±þ¤·¤Æ¤¤¤ì¤Ð¡¢¥³¥ó¥Ý¥¸¥Ã¥ÈÆ±´ü¤ò»ØÄê¤·¤Þ¤¹¡£É¬¤º±²Ã¤¨¤ì¤Ð¡¢¤¤¤¯¤Ä¤«¤Î¥°¥é¥Õ¥£¥Ã¥¯¥Ü¡¼¥É¤Ç¤Ï **"+CSync"** ¤È **"-CSync"** ¤Ç¥³¥ó¥Ý¥¸¥Ã¥ÈÆ±´ü¤Î¶ËÀ¤¬Áª¤Ö½ÐÍè¤ë¤Ç¤·¤ç¤¦¡£¾å¤µ¤Î **HSkew** ¤â¤³¤³¤Ç¤â»ÈÍÑ½ÐÍè¤Þ¤¹¡£

Device Àá¤Ï²èÁü¤µ¡´ï¡Ê¥Ó¥Ç¥ª¥Ü¡¼¥É¡Ë¤ÎÄê¤µÁ¤Ç¤¹¡££±¤Ä°Ê³å¤Î **Device** Àá¤¬ XF86Config ¥Õ¥¡¥¤¥ë¤Ë¤¢¤ë¤È»×¤¤¤Þ¤¹¡£¤³¤ÎÀá¤Ç »ÈÍÑ½ÐÍè¤ë¹àÌÜ¤Ï¼¡¤Î¤È¤ª¤ê¤Ç¤¹:

Identifier *"ID string"*

¤³¤ì¤Ï¼¡¤Î **Screen** Àá¤ÎÃæ¤Î¥Ó¥Ç¥ª¥«¡¼¥É¤Ç»²¾È¤µ¤ì¡¢Ê¸»úÎó¤Ç »ØÄê¤·¤Þ¤¹¡£¤½¤ì¤¾¤ì¤Î **Device** Àá¤Ï½ÅÊ£¤·¤ÊÌ¾Á°¾Î¤Ç¤Ê¤±¤ì¤Ð ¤¤¤±¤Þ¤»¤ó¡£

VendorName *"vendor"*

¤³¤ÎÄÉ²Ã¹àÌÜ¤Ï²èÁü¤µ¡´ï¤ÎÀ½Â¤²ñ¼Ò¤ò»ØÄê¤·¤Þ¤¹¡£

BoardName *"model"*

¤³¤ÎÄÉ²Ã¹àÌÜ¤Ï²èÁü¤µ¡´ï¤ÎÌ¾Á°¤ò»ØÄê¤·¤Þ¤¹¡£

Chipset *"chipset-type"*

¤³¤ÎÄÉ²Ã¹àÌÜ¤Ï¥Ó¥Ç¥ª¥Ü¡¼¥É¤Ë»È¤ï¤ì¤Æ¤¤¤ë¥Á¥Ã¥×¥»¥Ã¥È¤ò»ØÄê¤·¤Þ¤¹¡£ X ¥µ¡¼¥Ð¤¬¥Á¥Ã¥×¥»¥Ã¥È¤Î·¿¤òÇ§¼±¤¹¤ë¤¿¤á¤Ë¥Ó¥Ç¥ª¥Ü¡¼¥É¤òÃµ°º¤¹ ¤ë¤Î¤Ç¡¢ÄÌ¾ï¤Ï¾ì¹çÉ¬¿Ü¤Ç¤Ï¤¢¤ê¤Þ¤»¤ó¡£

Ramdac *"ramdac-type"*

¤³¤ÎÄÉ²Ã¹àÌÜ¤Ï¥Ó¥Ç¥ª¥Ü¡¼¥É¤Ë»È¤ï¤ì¤Æ¤¤¤ë RAMDAC ¤Î·¿¤ò»ØÄê¤·¤Þ¤¹¡£ ¾¯¿ô¤Î X ¥µ¡¼¥Ð¤Ç¤Îß»ÈÍÑ¤·¤Æ¤¤¤Þ¤¹¤¬¡¢X ¥µ¡¼¥Ð¤¬ RAMDAC ¤Î ·¿¤òÇ§¼±¤¹¤ë¤¿¤á¤Ë¥Ó¥Ç¥ª¥Ü¡¼¥É¤òÃµ°º¤¹¤ë¤Î¤Ç¡¢ÄÌ¾ï¤Ï¾ì¹ç É¬¿Ü¤Ç¤Ï¤¢¤ê¤Þ¤»¤ó¡£

DacSpeed *speed*

¤³¤ÎÄÉ²Ã¹àÌÜ¤Ï RAMDAC ¤Î¼þÇÈ¿ô(ÉáÄÌ RAMDAC ¥Á¥Ã¥×¤Î¾å¤Ë°õºþ¤µ¤ì¤Æ ¤¤¤Þ¤¹)¤ò»ØÄê¤·¤Þ¤¹¡£¼þÇÈ¿ô¤Ï MHz ¤Ç»ØÄê¤·¤Þ¤¹¡£¾¯¿ô¤Î X ¥µ¡¼¥Ð ¤Ç¤Îß»ÈÍÑ¤·¤Æ¤¤¤Þ¤¹¤¬¡¢ RAMDAC ¤Î¼þÇÈ¿ô¤¬ X ¥µ¡¼¥Ð É¸½à¤ÎÆâÂ¢¼þÇÈ¿ô¤È°Û¤Ê¤Ã¤Æ¤¤¤ë¾ì¹ç¤ÎßÉ¬Í×¤Ë¤Ê¤ê¤Þ¤¹¡£

Clocks *clock ...*

¤Î¥Ó¥Ç¥ª¥Ü¡¼¥É¤Î¥É¥Ã¥È¥¯¥í¥Ã¥¯¤ò»ØÄê¤·¤Þ¤¹¡£¥É¥Ã¥È¥¯¥í¥Ã¥¯¤Î¼þÇÈ¿ô¤Ï MHz ¤Ç¡¢ÉâÆ°¾®¿ôÅÀ¤Ç»ØÄê¤¹¤ë«¤â¤·¤ì¤Þ¤»¤ó¡£¿ôÃÍ¤ÏÆâÉôÅª¤Ë¤Ï °ìÈÖ¶á¤¤ kHz ¤ÇÃß¤¨¤é¤ì¤Þ¤¹¡£¥¯¥í¥Ã¥¯¤Î»ØÄê½çÈÖ¤¬½ÅÍ×¤Ç¤¹¡£Á°Âò¤·¤¿ ¥Ó¥Ç¥ª¥Ü¡¼¥É¤ËÂÐ±þ¤·¤¿½çÈÖ¤ËÊ¤Ã¤Æ¤¤¤¤ë¤³¤È¤¬É¬¿Ü¤Ç¤¹¡£Ê£¿ô¤Î **Clocks** ¹Ô¤¬»ØÄê¤µ¤ì¤Æ¤¤¤ë¤Ç¤·¤ç¤¦¡£¥×¥í¥°¥é¥Þ¥Ö¥ë¥¯¥í¥Ã¥¯¤Î ¥Á¥Ã¥×Åë°Ü¤Î¥Ó¥Ç¥ª¥Ü¡¼¥É¤Ç¤Ï¡¢**ClockChip** ¤Î¹àÌÜ¤Ï Clocks ¤ÎÂå¤ê¤Ë¤Ê¤ê¤Þ¤¹¡£**Clocks** ¤Î¹àÌÜ¤Ï¥×¥í¥°¥é¥Þ¥Ö¥ë¥¯¥í¥Ã¥¯¤Ç¤Ï ¤Ê¤¤¥Ü¡¼¥É¤Ç¤ÏÉ¬¿Ü¤Ç¤Ï¤¢¤ê¤Þ¤»¤ó¤¬¡¢¥µ¡¼¥Ð¤Î¯Æ°»þ¤Ë¥¯¥í¥Ã¥¯¤ÎÃµº ¤¬À¤µ¤¯¹Ô¤¨¤Ê¤¾¹ç¤¬¤¢¤ê¤Þ¤¹¤Î¤Ç»ØÄê¤¹¤ë¤³¤È¤ò¶¯¤¯¤ª´«¤á¤·¤Þ¤¹¡£ ¤¤¤¤Ä¤«¤Î¥â¥Ë¥¿¡¼¤Ç¤Ï¥¯¥í¥Ã¥¯¤ÎÃµº¤ËÌäÂê¤¬À¸¤¸¤Þ¤¹¡£

ClockChip *"clockchip-type"*

¤³¤ÎÄÉ²Ã¹àÌÜ¤Ï¥×¥í¥°¥é¥Þ¥Ö¥ë¥¯¥í¥Ã¥¯¥¸¥§¥Í¥ì¡¼¥¿Åë°Ü¤Î¥¯¥í¥Ã¥¯ ¥Á¥Ã¥×¤Î·¿¤ò»ØÄê¤·¤Þ¤¹¡£¾¯¿ô¤Î¥×¥í¥°¥é¥Þ¥Ö¥ë¥¯¥í¥Ã¥¯¥Á¥Ã¥×¤ÂÐ±þ¤Î X ¥µ¡¼¥Ð¤Ç¤Î¤ß»ÈÍÑ¤·¤Æ¤¤¤Þ¤¹¡£¾Ü¤·¤¯¤Ï¡¢³°Åö¤¹¤ë X ¥µ¡¼¥Ð¤Î ¥ª¥ó¥é¥¤¥ó¥Þ¥Ë¥å¥¢¥ë¤ò»²¾È¤·¤Æ¤¯¤À¤µ¤¤¡£

ClockProg *"command"* [*textclock*]

¤³¤ÎÄÉ²Ã¹àÌÜ¤ÏÆâÉô¥×¥í¥°¥é¥à¤ÎÂå¤ê¤Ë *command* ¤ò¼Â¹Ô¤·¤Æ ¥Ó¥Ç¥ª¥Ü¡¼¥É¤Î¥¯¥í¥Ã¥¯¤òÀßÄê¤·¤Þ¤¹¡£¥³¥Þ¥ó¥É¤Î¥Ñ¥¹¤Ï¥Õ¥ë¥Ñ¥¹¤Ç Í¿¤¨¤Ê¤±¤ì¤Ð¤¤¤±¤Þ¤»¤ó(¥³¥Þ¥ó¥É¤Î¥ª¥×¥·¥ç¥ó¤Ï»ØÄê¤Ç¤¤Þ¤»¤ó)¡£ ¤³¤Î¹àÌÜ¤ò»ÈÍÑ¤¹¤ë¤È¤¤Ï¡¢**Clocks** ¹Ô¤Î¹àÌÜ¤Ï¥µ¡¼¥Ð¤ÇÍøÍÑ ²ÄÇ½¤Ê¿ôÃÍ¤ò»ØÄê¤¹¤ëÉ¬Í×¤¬¤¢¤ê¤Þ¤¹(128 °Ê¾å¤Î¼þÇÈ¿ô¤ò»ØÄê¤¹¤ë É¬Í×¤¬¤¢¤ë«¤â¤·¤ì¤Þ¤»¤ó)¡£ *textclock* ¤È¤¤¤¦ÄÉ²Ã¹àÌÜ¤Ï¡¢ ¥µ¡¼¥Ð¤¬½ª»·¤¹¤¿¤È¤¤Ë(Èô¤Ï VT ÀÚ¤êÂØ¤¨»þ¤Î)¸µ¤Î¥Æ¥¥¹¥È¥â¡¼¥É ¤Î¥¯¥í¥Ã¥¯¤òÀßÄê¤¹¤ë *command* ¤ò¼Â¹Ô¤¹¤Ù¤¥µ¡¼¥Ð¤ËÅÁ¤¨¤ë ¤Î¤Ë»È¤¤¤Þ¤¹¡£ *textclock* ¤Ï **Clocks** ¹àÌÜ¤Î¤É¤ì«¤Ë°ìÃ×¤·¤Ê¤±¤ì¤Ð¤¤¤± ¤Þ¤»¤ó¡£¤³¤ÎÆÃÀÃÍ¤Ï¥Æ¥¥¹¥È¥â¡¼¥É¤¬¥×¥í¥°¥é¥Þ¥Ö¥ë¥¯¥í¥Ã¥¯¤Î¾ì¹ç¡¢ É¬Í×¤Ë¤Ê¤ê¤Þ¤¹¡£

¥³¥Þ¥ó¥É¤ÏÉ¸½àÆþ½ÐÎÏ¤ò¥°¥é¥Õ¥£¥Ã¥¯¥³¥ó¥½¡¼¥ë¤Ë³äÅö¤ÆÅ¤Æ¤é¤ì¤¿ÆâÉô ¥æ¡¼¥¶ id ¤Ç¼Â¹Ô¤µ¤ì¤Þ¤¹¡£Æó¤Ä¤Î°ú¿ô¤ò¤¬¥³¥Þ¥ó¥É¤ËÅÏ¤µ¤ì¤Þ¤¹¡£ °ì½é¤Î°ú¿ô¤ÏÉâÆ°¾®¿ô¤Î MHz Ã±°Ì¤ÇÍ¿¤¨¤ë¼þÇÈ¿ô¤Ç¡¢Æó¤ÄÌÜ¤Î°ú¿ô¤ô ¤Ï **Clocks** ¹àÌÜ¤Î°÷°ú¤Ç¤¹¡£¥³¥Þ¥ó¥É¤Î½ª»·¾õÂÖ¤¬ 0 ¤Î¾ì¹ç¤Ï À®¾ï½ª»·¤Ç¡¢¤½¤¦¤Ç¤Ê¤¤¾ì¹ç¤Ï 1–254 ¤Î´Ö¤Î¿ôÃÍ¤¬É½¼¨¤µ¤ì¤Þ¤¹¡£

¥³¥Þ¥ó¥É¤Ï½é´ü¾õÂÖ¤Î¥°¥é¥Õ¥£¥Ã¥¯¥â¡¼¥É¤òÀßÄê¤·¡¢¥Û¥Ã¥È¥¡¼¤¬²¡¤µ ¤ì¤¿»þ¤Î²èÌÌ¤Î²òÁüÅÙ¤òÊÑ¹¹¤·¤Þ¤¹¡£¥³¥Þ¥ó¥É¤¬½é´ü²½¤Ë¼ºÇÔ¤¹¤ë¤È ¥µ¡¼¥Ð¤ò½ª»·¤µ¤»¤Þ¤¹¡£¥â¡¼¥ÉÀÚ¤êÂØ¤¨¤ÎÅÓÃæ¤Ç¼ºÇÔ¤¹¤ë¤È¡¢¥â¡¼¥É ÀÚ¤êÂØ¤¨¤ÏÃæÃÇ¤·¥µ¡¼¥Ð¤ÏÂ³¹Ô¤·¤Þ¤¹¡£¥³¥Þ¥ó¥É¤¬¼ºÇÔ¤·¤¿¤È¤¤Ï ¥¯¥í¥Ã¥¯¤ÎÊÑ¹¹¤Ï¹Ô¤ï¤ì¤Þ¤»¤ó¡£

Option *"optionstring"*

¤³¤ÎÄÉ²Ã¹àÌÜ¤Ï¥µ¡¼¥Ð¤¬Äó¶¡¤¹¤ë¥É¥é¥¤¥Ð¡¼¤ÎÃæ¤«¤éÅ¬ÀÚ¤Ê¤â¤Î¤ò ¥æ¡¼¥¶¤¬Áª¤Ö¤Ç¤¤Þ¤¹¡£Ê£¿ô¤Î **Option** ¹àÌÜ¤¬Äê¤µÁ¤·¤Æ¤¢¤ë¤È »×¤¤¤Þ¤¹¡£ *optionstring* ¤ÇÍ¿¤¨¤ë¤³¤È¤¬½ÐÍè¤ë¥É¥é¥¤¥Ð¡¼¤Ï ³°Åö¤¹¤ë X ¥µ¡¼¥Ð¤Î¥ª¥ó¥é¥¤¥ó¥Þ¥Ë¥å¥¢¥ë¤Ëµ½Ò¤·¤Æ¤¤¤Þ¤¹¡£

VideoRam *mem*

¤³¤ÎÄÉ²Ã¹àÌÜ¤Ï¥°¥é¥Õ¥£¥Ã¥¯¥Ü¡¼¥É¤ËÅëºÜ¤·¤Æ¤¤¤ë¥Ó¥Ç¥ª¥á¥â¥ê¤Î Âç¤¤µ¤ò»ØÄê¤·¤Þ¤¹¡£¤³¤ì¤Ï ¥¥í¥Ð¥¤¥È Ã±°Ì¤Ç»ØÄê¤·¤Æ¤¯¤À¤µ¤¤¡£

¤Û¤È¤ó¤É¤Î¾ì¹ç X ¥µ¡¼¥Ð¤¬¥°¥é¥Õ¥£¥Ã¥¯¥Ü¡¼¥É¤òÄ´¤Ù¤Æ¥á¥â¥êÎÌ¤ò Ç§¼±¤·¤Þ¤¹¤Î¤ÇÉ¬Í×¤¢¤ê¤Þ¤»¤ó¡£

BIOSBase *baseaddress*
¤³¤ÎÄÉ²Ã¹àÌÜ¤Ï VGA ¥Ü¡¼¥É¤Î¥Ó¥Ç¥ª BIOS ¤Î´ðÄì¥¢¥É¥ì¥¹¤ò»ØÄê¤·¤Þ ¤¹¡£¤³¤Î X ¥µ¡¼¥Ð¤ÇÉ¸½à¤Ç»ÈÍÑ¤¹¤ë¥¢¥É¥ì¥¹¤ÏÉáÄÌ 0xC0000 ¤Ç¤¹¡£ ÆÃ¤Ë¥Þ¥¶¡¼¥Ü¡¼¥ÉÆâÂ¢¤Î VGA ¥Ó¥Ç¥ªÁõÃÖ¤Î¤¤¤¯¤Ä¤«¤Î¥·¥¹¥Æ¥à¤Ç¤Ï¡¢ 0xE0000 ¤È¤¤¤¦°Û¤Ê¤ë¥¢¥É¥ì¥¹¤Ë BIOS ¤¬Åë¹Ü¤µ¤ì¤Æ¤¤¤Þ¤¹¡£¥Ó¥Ç¥ª BIOS ¤¬ 0xC0000 °Ê³°¤Ë¤¢¤ë¾ì¹ç¤Ï¡¢XF86Config ¥Õ¥¡¥¤¥ë¤Ë¤³¤Î´ðÄì ¥¢¥É¥ì¥¹¤ò»ØÄê¤¹¤ëÉ¬Í×¤¬¤¢¤ê¤Þ¤¹¡£¤¤¤¯¤Ä¤«¤Î X ¥µ¡¼¥Ð¤Ç¤Ï¤Þ¤Ã ¤¿¤¯ BIOS ¤ò¥¢¥¯¥»¥¹¤·¤Ê¤¤¤¿¤á¤Ë¤â¤¢¤ë»ö¤Ë¡¢µ¡´ï¾ÊÊó¤ÎÃµºÃÊ³¬¤Ç BIOS ¤ò»È¤¦¤À¤±¤Î¤â¤Î¤¬¤¢¤ë»ö¤ËÃí°Õ¤·¤Æ¤¯¤À¤µ¤¤¡£

MemBase *baseaddress*
¤³¤ÎÄÉ²Ã¹àÌÜ¤Ï¥Ó¥Ç¥ª¥Ü¡¼¥É¤Î¥é¥Ë¥¢¥Õ¥ì¡¼¥à¥Ð¥Ã¥Õ¥¡¤Î¥á¥â¥ê¤Î´ðÄì ¥¢¥É¥ì¥¹¤ò»ØÄê¤·¤Þ¤¹¡£¤³¤Î¹àÌÜ¤Ï¤ì¤º¤«¤Î X ¥µ¡¼¥Ð¤Ç»ÈÍÑ¤µ¤ì¡¢ ¿§¡¹¤Ê X ¥µ¡¼¥Ð¤ÎÁê°ãÅÀ¤ò¤³¤Î´ðÄì¥¢¥É¥ì¥¹¤ÇÈ½Ã¢¤·¤Þ¤¹¡£

IOBase *baseaddress*
¤³¤ÎÄÉ²Ã¹àÌÜ¤ÏÆþ½ÐÎÏ¤Î´ðÄì¥¢¥É¥ì¥¹¤ò»ØÄê¤·¤Þ¤¹¡£¤³¤Î¹àÌÜ¤Ï¤ì¤º¤«¤Î X ¥µ¡¼¥Ð¤À¤±¤Ç»ÈÍÑ¤µ¤ì¤Þ¤¹¡£¾ÜºÙ¤Ï³ºÅö¤¹¤ë X ¥µ¡¼¥Ð¤Î¥ª¥ó¥é¥¤¥ó ¥Þ¥Ë¥å¥¢¥ë¤ò»²¾È¤·¤Æ¤¯¤À¤µ¤¤¡£

DACBase *baseaddress*
¤³¤ÎÄÉ²Ã¹àÌÜ¤Ï DAC ¤Î´ðÄì¥¢¥É¥ì¥¹¤ò»ØÄê¤·¤Þ¤¹¡£¤³¤Î¹àÌÜ¤Ï¤ì¤º¤«¤Î X ¥µ¡¼¥Ð¤À¤±¤Ç»ÈÍÑ¤µ¤ì¤Þ¤¹¡£¾ÜºÙ¤Ï³ºÅö¤¹¤ë X ¥µ¡¼¥Ð¤Î¥ª¥ó¥é¥¤¥ó ¥Þ¥Ë¥å¥¢¥ë¤ò»²¾È¤·¤Æ¤¯¤À¤µ¤¤¡£

POSBase *baseaddress*
¤³¤ÎÄÉ²Ã¹àÌÜ¤Ï POS ¤Î´ðÄì¥¢¥É¥ì¥¹¤ò»ØÄê¤·¤Þ¤¹¡£¤³¤Î¹àÌÜ¤Ï¤ì¤º¤«¤Î X ¥µ¡¼¥Ð¤À¤±¤Ç»ÈÍÑ¤µ¤ì¤Þ¤¹¡£¾ÜºÙ¤Ï³ºÅö¤¹¤ë X ¥µ¡¼¥Ð¤Î¥ª¥ó¥é¥¤¥ó ¥Þ¥Ë¥å¥¢¥ë¤ò»²¾È¤·¤Æ¤¯¤À¤µ¤¤¡£

COPBase *baseaddress*
¤³¤ÎÄÉ²Ã¹àÌÜ¤Ï ¿ôÃÍ±é»»¥³¥×¥í¥»¥Ã¥µ ¤Î´ðÄì¥¢¥É¥ì¥¹¤ò»ØÄê¤·¤Þ¤¹¡£¤³¤Î¹àÌÜ¤Ï¤ì¤º¤«¤Î X ¥µ¡¼¥Ð¤À¤±¤Ç»ÈÍÑ¤µ¤ì¤Þ¤¹¡£¾ÜºÙ¤Ï³ºÅö¤¹¤ë X ¥µ¡¼¥Ð ¤Î¥ª¥ó¥é¥¤¥ó¥Þ¥Ë¥å¥¢¥ë¤ò»²¾È¤·¤Æ¤¯¤À¤µ¤¤¡£

VGABase *baseaddress*
¤³¤ÎÄÉ²Ã¹àÌÜ¤Ï VGA ¥á¥â¥ê ¤Î´ðÄì¥¢¥É¥ì¥¹¤ò»ØÄê¤·¤Þ¤¹¡£¤³¤Î¹àÌÜ¤Ï¤ì¤º¤«¤Î X ¥µ¡¼¥Ð¤À¤±¤Ç»ÈÍÑ¤µ¤ì¤Þ¤¹¡£¾ÜºÙ¤Ï³ºÅö¤¹¤ë X ¥µ¡¼¥Ð¤Î¥ª¥ó¥é¥¤¥ó ¥Þ¥Ë¥å¥¢¥ë¤ò»²¾È¤·¤Æ¤¯¤À¤µ¤¤¡£

Instance *number*
¤³¤ÎÄÉ²Ã¹àÌÜ¤Ï (¥Þ¥¶¡¼¥Ü¡¼¥É¤ËÁÈ¤ß¹þ¤Þ¤ì¤Æ¤¤¤ë¤«¡¢³ÈÄ¥¥«¡¼¥É¤ËÅë¹Ü ¤µ¤ì¤Æ¤¤¤ë) ¼ÂÂÎ ¤Î´ðÄì¥¢¥É¥ì¥¹¤ò»ØÄê¤·¤Þ¤¹¡£¤³¤Î¹àÌÜ¤Ï¤ì¤º¤«¤Î X ¥µ¡¼¥Ð¤À¤±¤Ç»ÈÍÑ¤µ¤ì¤Þ¤¹¡£¾ÜºÙ¤Ï³ºÅö¤¹¤ë X ¥µ¡¼¥Ð¤Î¥ª¥ó¥é¥¤¥ó ¥Þ¥Ë¥å¥¢¥ë¤ò»²¾È¤·¤Æ¤¯¤À¤µ¤¤¡£

Speedup *"selection"*
¤³¤ÎÄÉ²Ã¹àÌÜ¤Ï²ÄÇ½¤ÊÈ¤·¤¿¤¤¹âÂ®²½¤ÎÁªÂò¤ò»ØÄê¤·¤Þ¤¹¡£¤³¤Î¹àÌÜ¤Ï¤ì¤º¤«¤Î X ¥µ¡¼¥Ð¤À¤±¤Ç»ÈÍÑ¤µ¤ì¤Þ¤¹¡£¾ÜºÙ¤Ï³ºÅö¤¹¤ë X ¥µ¡¼¥Ð¤Î¥ª¥ó¥é¥¤¥ó ¥Þ¥Ë¥å¥¢¥ë¤ò»²¾È¤·¤Æ¤¯¤À¤µ¤¤¡£

S3MNAdjust *M N*
¤³¤ÎÃÉ²Ã¹àÌÜ¤Ï S3 X ¥µ¡¼¥Ð¤Ë»ØÄê¤·¤Þ¤¹¡£¾Ü ºÙ¤Ï *XF86_S3(1)* ¤Î ¥ª¥ó¥é¥¤¥ó¥Þ¥Ë¥å¥¢¥ë¤ò»²¾È¤·¤Æ¤¯¤À¤µ¤¤¡£

S3MClk *clock*
¤³¤ÎÃÉ²Ã¹àÌÜ¤Ï S3 X ¥µ¡¼¥Ð¤Ë»ØÄê¤·¤Þ¤¹¡£¾Ü ºÙ¤Ï *XF86_S3(1)* ¤Î ¥ª¥ó¥é¥¤¥ó¥Þ¥Ë¥å¥¢¥ë¤ò»²¾È¤·¤Æ¤¯¤À¤µ¤¤¡£

S3RefClock *clock*
¤³¤ÎÃÉ²Ã¹àÌÜ¤Ï S3 X ¥µ¡¼¥Ð¤Ë»ØÄê¤·¤Þ¤¹¡£¾Ü ºÙ¤Ï *XF86_S3(1)* ¤Î ¥ª¥ó¥é¥¤¥ó¥Þ¥Ë¥å¥¢¥ë¤ò»²¾È¤·¤Æ¤¯¤À¤µ¤¤¡£

Screen Àá¤ÏÆÃÄê¤Î X ¥µ¡¼¥Ð¤ÇÍÑ¤¤¤é¤ì¤ë¥Ó¥Ç¥ª¥Ü¡¼¥É¤È¥â¥Ë¥¿¡¼ ¤Î»ØÄê¤È¡¢¤½¤³¤Ç»ÈÍÑ¤¹¤ë¹½À®¾ðÊó¤ò»ØÄê¤·¤Þ¤¹¡£¤³¤ÎÀá¤ÇÍ-¸ú¤Ê¹àÌÜ¤Ï ¼¡¤Î¤È¤ª¤ê¤Ç¤¹:

Driver *"driver-name"*
¤½¤ì¤¾¤ì¤Î **Screen** Àá¤Ï **Driver** ¹àÌÜ¤Ç½ñ¤-¤Ï¤¸¤á¤ë¤É¬Í×¤¬¤¢¤ê ¤Þ¤¹¡£*driver-name* ¤Ï¤½¤ì¤¾¤ì¤Î **Screen** Àá¤Ç½Å¤Ê¤é¤Ê¤¤¤è¤¦¤Ë ¤·¤Ê¤±¤ì¤Ð¤¤¤±¤Þ¤»¤ó¡£¥É¥é¥¤¥Ð¤Î¾¾Î¤Ï X ¥µ¡¼¥Ð(¼ã¤¯¤Ï X ¥µ¡¼¥Ð ¤¬ÆâÂ¢¤·¤Æ¤¤¤ë°ì¤Ä°Ê¾å¤Î¥É¥é¥¤¥Ð¤Î¼ïÎà)¤òÆÉ¤ß¹þ¤à¤«·èÄê¤·¡¢¸Ä¡¹¤Î **Screen** Àá¤Ç»ÈÍÑ¤·¤Þ¤¹¡£»ÈÍÑ²ÄÇ½¤Ê¥É¥é¥¤¥Ð¤Î¾¾Î¤Ï¼¡¤Î¤È¤ª¤ê¤Ç¤¹:

Accel
Mono
SVGA
VGA2
VGA16

Accel ¤ÏÁ´¤Æ¤Î¥¢¥¯¥»¥é¥ì¡¼¥¿ X ¥µ¡¼¥Ð¤ÇÍÑ¤¤¤Þ¤¹ (*XF86_Mono(1)* ¤È *XF86_VGA16(1)* ¤ò»²¾È¤·¤Æ¤¯¤À¤µ¤¤)¡£**VGA2** ¤È **VGA16** ¤Ï 2-¥Ó¥Ã¥È ¤È 4-¥Ó¥Ã¥È X ¥µ¡¼¥Ð ¤½¤ì¤¾¤ì¤Î VGA ¥É¥é¥¤¥Ð¡¼¤ÇÍÑ¤¤¤Þ¤¹¡£**SVGA** ¤Ï XF86_SVGA X ¥µ¡¼¥Ð¡¼¤ÇÍÑ¤¤¤Þ¤¹¡£

Device *"device-id"*
¤Ï»ÈÍÑ¤·¤¿¤¤¥Ó¥Ç¥ª¥É¥é¥¤¥Ðµ-½Ò»Ò¤ò»ØÄê¤·¤Þ¤¹¡£

Monitor *"monitor-id"*
¤Ï¥â¥Ë¥¿¡¼µ-½Ò»Ò¤ò»ØÄê¤·¤Þ¤¹¡£

DefaultColorDepth *bpp-number*
¤Ï¥µ¡¼¥Ð¤Ç»ÈÍÑ¤¹¤ë¥«¥é¡¼¤Î¿¼¤µ¤ò¥³¥Þ¥ó¥É¹Ô¥Ñ¥é¥á¥¿¤Î -bpp ¤Î¿ôÃÍ¤ò»ØÄê¤·¤Þ¤¹¡£

ScreenNo *scrnum*
¤³¤ÎÃÉ²Ã¹àÌÜ¤ÏÊ£¿ô²èÌÌ¹½À®¤Î¾ì¹ç¤ËÈÖ¹æÉÕ¤±¤¹¤ëÉ¸½à¤Î¥¹¥¯¥ê¡¼¥ó¤ò ÃÖ¤-´¹¤¨¤Þ¤¹¡£*XF86Config* ¥Õ¥¡¥¤¥ë¤Î **Screen** Àá¤Î½çÈÖ¤Ç É¸½à¤ÎÈÖ¹æÉÕ¤±¤¬·èÄê¤·¤Þ¤¹¡£¤³¤ì¤òÃÖ¤-´¹¤¨¤ë¤¿¤á¤ËÁ´¤Æ¤Î´ØÏ¢¤¹¤ë **Screen** Àá¤Ë¤³¤Î¹àÌÜ¤ò»ØÄê¤¹¤ëÉ¬Í×¤¬¤¢¤ê¤Þ¤¹¡£

BlankTime *time*
¤Ï¥¹¥¯¥ê¡¼¥ó¥»¡¼¥Ð¡¼¤Ë¤è¤Ã¤Æ°Å¤¯¤¾õÂÖ¤Ë¤Ê¤ë¤Þ¤Ç¤Î»þ´Ö¤òÀßÄê¤·¤Þ¤¹¡£ *time* ¤ÏÊ¬Ã±°Ì¤Ç»ØÄê¤·¡¢É¸½àÃÍ¤Ï 10 ¤Ç¤¹¡£ ¤³¤ì¤Ï X ¥µ¡¼¥Ð¤Ç '-s' ¥ª¥×¥·¥ç¥ó¤ò»ØÄê¤·¤¿¤Î¤ÈÅù²Á¤Ç¡¢¼Â¹Ô»þ¤Ë ¿ôÃÍ¤Ï *xset(1)* ¤ÇÊÑ¹¹½ÐÍè¤Þ¤¹¡£

SuspendTime *time*

¤Ï¥¹¥¯¥ê¡¼¥ó¥»¡¼¥Ð¡¼¤Ë¤è¤Ã¤Æ "suspend" ¾õÂÖ¤Ë¤Ê¤ë¤Þ¤Ç¤Î»þ´Ö¤òÀßÄê¤·¤Þ¤¹¡£ *time* ¤ÏÊ¬Ã±°Ì¤Ç»ØÄê¤·¡¢É¸½àÃÍ¤Ï 15 ¤Ç¡¢¼Â¹Ô»þ¤Ë *xvidtune(1)* ¤ÇÊÑ¹¹½ÐÍè¤Þ¤¹¡£ VESA DPMS ¸ß´¹¥â¥Ë¥¿¡¼¤ËÂÐ¤ß¤Å¬¤·¡¢¤¤¤¯¤Ä¤«¤Î X ¥µ¡¼¥Ð ¤Ç¤Ï¤ß¼Â¸½¤·¤Æ¤¤¤Þ¤¹¡£ "power_saver" ¥ª¥×¥·¥ç¥ó¤Ï¥µ¥¹¥Ú¥ó¥É¤¬Í¸ú¤Ç¤¢¤ë É¬Í×¤¬¤¢¤ê¤Þ¤¹¡£

OffTime *time*

¤Ï¥¹¥¯¥ê¡¼¥ó¥»¡¼¥Ð¡¼¤Ë¤è¤Ã¤Æ "off" ¾õÂÖ¤Ë¤Ê¤ë¤Þ¤Ç¤Î»þ´Ö¤òÀßÄê¤·¤Þ¤¹¡£ *time* ¤ÏÊ¬Ã±°Ì¤Ç»ØÄê¤·¡¢É¸½àÃÍ¤Ï 30 ¤Ç¡¢¼Â¹Ô»þ¤Ë *xvidtune(1)* ¤ÇÊÑ¹¹½ÐÍè¤Þ¤¹¡£ VESA DPMS ¸ß´¹¥â¥Ë¥¿¡¼¤ËÂÐ¤ß¤Å¬¤·¡¢¤¤¤¯¤Ä¤«¤Î X ¥µ¡¼¥Ð ¤Ç¤Ï¤ß¼Â¸½¤·¤Æ¤¤¤Þ¤¹¡£ "power_saver" ¥ª¥×¥·¥ç¥ó¤Ï¥µ¥¹¥Ú¥ó¥É¤¬Í¸ú¤Ç¤¢¤ë É¬Í×¤¬¤¢¤ê¤Þ¤¹¡£

SubSection "Display"

¤³¤Î¹àÌÜ¤Ï¤¤¤¯¤Ä¤«¤Î¥Ç¥£¥¹¥×¥ì¥¤¤ÎÆÃÀ¤ò»ØÄê¤¹¤ëÉûÀá(¥µ¥Ö¥»¥¯¥·¥ç¥ó)¤Ç¤¹¡£¤³¤ÎÉûÀá¤Ï **EndSubSection** ¹àÌÜ¤Ç½ª¤ï¤ê¤Þ¤¹¡£¤¤¤¯¤Ä¤«¤Î X ¥µ¡¼¥Ð¤È¥É¥é¥¤¥Ð(¤³¤ì¤é¤Ï¥Ó¥Ç¥ª¥â¡¼¥É¤Î°ìÍ÷¤ÇÉ¬Í×¤Ë¤Ê¤ê¤Þ¤¹) ÃÐ±þ¤Ë É¬¿Ü¤Ç¤¹¡£Ê£¿ô¤Î¥Ç¥£¥¹¥×¥ì¥¤¤Î¿¼¤µ¤ò»È¤¦ X ¥µ¡¼¥Ð¤Ç¤Ï °ì¤Ä°Ê¾å¤Î **Display** ÉûÀá¤¬¤Ä¤ê¤Þ¤¹¡£¤½¤ì¤¾¤Ì **Display** ÉûÀá ¤¬¤Ä¤ê¤Þ¤¹¤¬¡¢¤½¤ì¤¾¤ì¤Î **Depth** ¹àÌÜ¤Ï½ÅÊ£¤·¤Ê¤¤¤è¤¦¤Ë¤¹¤ë É¬Í×¤¬¤¢¤ê¤Þ¤¹¡£**Display** ÉûÀá¤Ç»ØÄê¤Ç¤¤ë¹àÌÜ¤Ï¼¡¤Î¤È¤ª¤ê¤Ç¤¹:

Depth *bpp*

¤³¤Î¹àÌÜ¤Ï£±¤Ä¤è¤ê¤âÂ¿¤¯¤Î **Display** ÉûÀá¤¬ **Screen** Àá¤ËÄêµÁ¤µ¤ì¤Æ¤¤¤ë¾ì¹ç¤ËÉ¬¿Ü¤Ç¤¹¡£**Display** ÉûÀá¤¬£±¤Ä¤À¤±¤Î¾ì¹ç¤Ï¡¢X ¥µ¡¼¥Ð¼Â¹Ô»þ¤Î¿¼¤µ¤ÎÉ¸½àÃÍ¤ò»ØÄê¤·¤Þ¤¹¡£ £±¤Ä¤è¤ê¤âÂ¿¤¯¤Î **Display** ÉûÀá¤¬ ÄêµÁ¤µ¤ì¤Æ¤¤¤ë¾ì¹ç¡¢X ¥µ¡¼¥Ð ¤¬»ÈÍÑ¤¹¤ë¿¼¤µ¤ò·èÄê¤·¤Þ¤¹¡£X ¥µ¡¼¥Ð¼Â¹Ô»þ°ìÃ×¤¹¤ë¿¼¤µ¤òÉûÀá¤Ë ÄêµÁ¤·¤Þ¤¹¡£ËØ¤ÉÎ¤Î X ¥µ¡¼¥Ð(¼ã¤¤¤Ï¥É¥é¥¤¥Ð) ¤ÏÊ£¿ô¤Î¿¼¤µ¤ò¥µ¥Ý¡¼¥È¤·¤Æ¤¤¤Þ¤¹¡£*bpp* ¤Ç»ØÄê¤Ç¤¤ë¿ôÃÍ¤Ï 8, 15, 16, 24 ¤È 32 ¤Ç¤¹¡£Á´¤Æ¤Î¥µ¡¼¥Ð(¼ã¤¤¤Ï¥É¥é¥¤¥Ð) ¤¬Á´¤Æ¤Î¿ôÃÍ¤ò¥µ¥Ý¡¼¥È¤·¤Æ¤Ï ¤¤¤Þ¤»¤ó¡£*bpp* ¤Î¿ôÃÍ¤¬ 24 ¤È 32 ¤Î¾ì¹ç¤¢ X ¥µ¡¼¥Ð¤¬¤½¤ì¤ò¥µ¥Ý¡¼¥È¤·¤Æ¤¤¤ë¾ì¹ç¤ÏÆ±Åù¤Ë¼è¤ê°·¤ï¤ì¤Þ¤¹¡£

Weight *RGB*

¤³¤ÎÄÉ²Ã¹àÌÜ¤Ï X ¥µ¡¼¥Ð¤¬ 16bpp ¤Ç¼Â¹Ô¤·¤Æ¤¤¤ë¾ì¹ç¤ËÍÁª¤Ê RGB ¤Î½Å¤ßÉÕ¤±¤ò»ØÄê¤·¤Þ¤¹¡£¤³¤Î½Å¤ßÉÕ¤±¤Ï¥³¥Þ¥ó¥É¹Ô¤«¤é¤â»ØÄê¤Ç¤¤Þ¤¹(*XFree86(1)* ¤ò»²¾È¤Î¤³¤È)¡£ËØ¤ÉÎ¤Î 16bpp X ¥µ¡¼¥Ð¤Ï **555** ¤È **565** ¤ò¥µ¥Ý¡¼¥È¤·¤Æ¤¤¤Þ¤¹¡£¤µ¤é¤Ê¤ë¾Ü°Ü¤Ë¤Ä¤¤¤Æ¤Ï¡¢³°Åö¤¹¤ë X ¥µ¡¼¥Ð¤Î¥ª¥ó¥é¥¤¥ó¥Þ¥Ë¥å¥¢¥ë¤ò»²¾È¤·¤Æ¤¯¤À¤µ¤¤¡£

Virtual *xdim ydim*

¤³¤ÎÄÉ²Ã¹àÌÜ¤Ï»ÈÍÑ¤¹¤ë²¾ÁÛ²èÌÌ¤Î²òÁüÅÙ¤ò»ØÄê¤·¤Þ¤¹¡£ËØ¤ÉÎ¤Î¥«¥é¡¼¤Î X ¥µ¡¼¥Ð¤Ç¤Ï *xdim* ¤Ï 8 ¤« 16 ¤ÎÇÜ¿ô¤Î¿§¿ô¤Ç¡¢¥â¥Î¥¯¥í¤Î X ¥µ¡¼¥Ð¤Ç¤Ï 32 ¤ÎÇÜ¿ô¤Ç¤¢¤ëÉ¬Í×¤¬¤¢¤ê¤Þ¤¹¡£¤³¤ì¤è°Û¤Ê¤ë¾ì¹ç¤Ï¡¢Í¿¤¨¤é¤ì¿¿¿ôÃÍ¤ò¾®¤µ¤¤Û¤¦¤Î¿ôÃÍ¤ËË´Ý¤á¤¤Þ¤¹¡£ËØ¤ÉÎ¤Î X

¥µ¡¼¥Ð¤Ç¤Ï¡¢
»ØÄê¤µ¤ì¤¿²¾ÁÛ²èÌÌ¤ÎÂç¤¤µ¤¬Âç¤¤¹¤®¤ë¥Ó¥Ç¥ª¥â¡¼¥É¤ÏµñÈÝ¤µ¤ì¤Þ¤¹¡£
²¾ÁÛ²èÌÌ¤ÎÀßÄê¤¬¤Ê¤¤¾ì¹ç¡¢²¾ÁÛ²èÌÌ¤Î²òÁüÅÙ¤Ï **Modes**
¹àÌÜ¤ÇÍ¿¤¨
¤é¤ì¤¿Å¬Àµ¤ÊÁ´¤Æ¤Î¥Ó¥Ç¥ª¥â¡¼¥É¤«¤éÄ´°¤·¤ÆÀßÄê¤·¤Þ¤¹¡£¤¤¤¯¤Ä¤«¤Î
X
¥µ¡¼¥Ð¤Ç¤Ï¤³¤Î¹àÌÜ¤Ï¥µ¥Ý¡¼¥È¤·¤Æ¤¤¤Þ¤»¤ó¡£¾Ü°Ù¤Ë¤Ä¤¤¤Æ¤Ï¡¢³°Åö¤¹¤ë
X ¥µ¡¼¥Ð¤Î¥ª¥ó¥é¥¤¥ó¥Þ¥Ë¥å¥¢¥ë¤ò»²¾È¤·¤Æ¤¯¤À¤µ¤¤¡£

ViewPort *x0 y0*
¤³¤ÎÄÉ²Ã¹àÌÜ¤Ï½é´ü²èÌÌ¤Î¸¾å¶ù¤òÀßÄê¤·¤Þ¤¹¡£²¾ÁÛ²èÌÌ¤Î²òÁüÅÙ¤¬ºÇ½é
¤Î¥Ó¥Ç¥ª¥â¡¼¥É¤Î²òÁüÅÙ¤È°Û¤Ê¤ë¾ì¹ç¤À¤±´Ø·¸¤¬¤¢¤ê¤Þ¤¹¡£¤³¤Î¹àÌÜ¤¬
¤Ê¤¤¾ì¹ç¡¢½é´ü²èÌÌ¤Ï²¾ÁÛ²èÌÌÎ°è¤ÎÃæ±û¤Ë¤Ê¤ê¤Þ¤¹¡£

Modes *"modename" ...*
¤³¤Î¹àÌÜ¤ÏËØ¤É¤Î X
¥µ¡¼¥Ð¤ËÉ¬¿Ü¤Ç¡¢»ÈÍÑ¤¹¤ë¥Ó¥Ç¥ª¥â¡¼¥É¤Î¥ê¥¹¥È
¤ò»ØÄê¤·¤Þ¤¹¡£¥Ó¥Ç¥ª¥â¡¼¥ÉÌ¾¾Î¤ÏÅ¬ÀÚ¤Ê **Monitor**
ÀáÆâ¤Ç»ØÄê¤·¤¿ Ì¾¾Î¤È°ìÃ×¤·¤Ê¤±¤ì¤Ð¤¤¤±¤Þ¤»¤ó¡£ËØ¤É¤Î X
¥µ¡¼¥Ð¤Ç¤Ï¼¡¤ÎÍ×µá¤òËþ
¤¿¤¹¤³¤È¤¬½ÐÍè¤Ê¤¤¤è¤¦¤Ê¥â¡¼¥É¤Ï¤³¤Î¥ê¥¹¥È¤«¤é¾Ãµî¤¹¤ë¤Ç¤·¤ç¤¦¡£
¤³¤Î¥ê¥¹¥È¤Î¤Ê¤«¤ÇºÇ½é¤ËÍ¸ú¤Ë¤Ê¤ë¥â¡¼¥É¤Ï¥µ¡¯Æ°»þ¤ÎÉ¸½à¤ÎÉ½¼¨¥â¡¼¥É
¤Ë¤Ê¤ë¤Ç¤·¤ç¤¦¡£Í¸ú¤Ê¥â¡¼¥É¤Î¥ê¥¹¥È¤Ï½Û´Ä¥ê¥¹¥È¤ËÆâÉôÅª¤ËÊÑ´¹¤µ¤ì
¤Þ¤¹¡£½Û´Ä¥ê¥¹¥ÈÆâ¤Î¥â¡¼¥É¤Ï **Ctrl+Alt+Keypad-Plus** ¤Ç¼¡¤Î¥â¡¼
¥É¡¢**Ctrl+Alt+Keypad-Minus**
¤ÇÁ°¤Î¥â¡¼¥É¤ËÀÚ¤êÂØ¤¨¤ë¤³¤È¤¬²ÄÇ½ ¤Ç¤¹¡£[ÌõÃí¡§Keypad-Plus
¤Ï¥Æ¥ó¥¡¼¤Î¡Ü¥¡¼¡¢Keypad-Minus ¤Ï¥Æ¥ó
¥¡¼¤Î¡Ý¥¡¼¤ò°ÕÌ£¤·¤Þ¤¹¡£]

InvertVCLK *"modename"* 0|1
¤³¤ÎÄÉ²Ã¹àÌÜ¤Ï S3
¥µ¡¼¥Ð¤Î¤ß¤Ç»ØÄê¤·¤Þ¤¹¡£¤³¤Î¹àÌÜ¤ÏÉ¸½à¤Î¸Ä¡¹¤Î¥â¡¼¥É Ëè¤Î
VCLK ¤ÎÈ¿Å¾¥¿¥¤¥ß¥ó¥°¤ÎÀßÄê¤òÊÑ¹¹¤·¤Þ¤¹¡£ *"modename"* ¤¬ "*"
¤Ç¤Ï¤¸¤Þ¤ë¾ì¹ç¤Ï¡¢¤³¤Î¸å¤Ë»ØÄê¤¹¤ëÀßÄê¹àÌÜ¤Ç
¾å½ñ¤¤µ¤ì¤Ê¤¤¸Â¤ê¡¢Á´¤Æ¤Î¥â¡¼¥É¤ËÂ¤·¤Æ¤Î¥â¡¼¥É¤ÎÀßÄêÃÍ¤¬Å¬±þ¤µ¤ì¤Þ¤¹¡£

EarlySC *"modename"* 0|1
¤³¤ÎÄÉ²Ã¹àÌÜ¤Ï S3
¥µ¡¼¥Ð¤Î¤ß¤Ç»ØÄê¤·¤Þ¤¹¡£¤³¤Î¹àÌÜ¤ÏÉ¸½à¤Î½¤¤ì¤¾¤ì¤Î¥â¡¼¥É Ëè¤Î
EarlySC ¤ÎÀßÄê¤òÊÑ¹¹¤·¤Þ¤¹¡£ *"modename"* ¤¬ "*"
¤Ç¤Ï¤¸¤Þ¤ë¾ì¹ç¤Ï¡¢¤³¤Î¸å¤Ë»ØÄê¤¹¤ëÀßÄê¹àÌÜ¤Ç
¾å½ñ¤¤µ¤ì¤Ê¤¤¸Â¤ê¡¢Á´¤Æ¤Î¥â¡¼¥É¤ËÂ¤·¤Æ¤Î¥â¡¼¥É¤ÎÀßÄêÃÍ¤¬Å¬±þ¤µ¤ì¤Þ¤¹¡£

BlankDelay *"modename" value1 value2*
¤³¤ÎÄÉ²Ã¹àÌÜ¤Ï S3
¥µ¡¼¥Ð¤Î¤ß¤Ç»ØÄê¤·¤Þ¤¹¡£¤³¤Î¹àÌÜ¤ÏÉ¸½à¤Î½¤¤ì¤¾¤ì¤Î¥â¡¼¥É
Ëè¤Î¶õÇòÃÙ±ä»þ´Ö¤ÎÀßÄê¤òÊÑ¹¹¤·¤Þ¤¹¡£¤½¤ÎÀßÄê¤Ï²èÌÌ¤Î²ö¤ê¤Þ¤ß¸½¾Ý¤ËºîÍÑ¤·
¤Þ¤¹¡£ *value1* ¤È *value2* ¤Î¿ôÃÍ¤Ï 0–7 ¤ÎÈÏ°Ï¤ÎÀ°¿ô¤ÇÍ¿¤¨¤ë
É¬Í×¤¬¤¢¤ê¤Þ¤¹¡£ *"modename"* ¤¬ "*"
¤Ç¤Ï¤¸¤Þ¤ë¾ì¹ç¤Ï¡¢¤³¤Î¸å¤Ë»ØÄê¤¹¤ëÀßÄê¹àÌÜ¤Ç
¾å½ñ¤¤µ¤ì¤Ê¤¤¸Â¤ê¡¢Á´¤Æ¤Î¥â¡¼¥É¤ËÂ¤·¤Æ¤Î¥â¡¼¥É¤ÎÀßÄêÃÍ¤¬Å¬±þ¤µ¤ì¤Þ¤¹¡£

Visual *"visual-name"*
¤³¤ÎÄÉ²Ã¹àÌÜ¤ÏÉ¸½à¤Î¥ë¡¼¥È¥Ï¥£¥ó¥É¥¦¤ÎÉ½¼¨·Á¼°¤òÀßÄê¤·¤Þ¤¹¡£¤³¤Î¹àÌÜ¤Ï
¥³¥Þ¥ó¥É¥é¥¤¥ó¤«¤é¤â»ØÄê¤Ç¤¤Þ¤¹(*Xserver(1)*

¤ò»²¾È¤·¤Æ¤¯¤À¤µ¤¤)¡£ 8bpp ¤Î X ¥µ¡¼¥Ð¤Ç»ÈÍÑ²ÄÇ½¤ÊÉ½¼¨·Á¼°¤Ï¼¡¤Î¤È¤ª¤ê¤Ç¤¹(É¸½à¤Ç¤Ï **PseudoColor** ¤Ç¤¹):

StaticGray
GrayScale
StaticColor
PseudoColor
TrueColor
DirectColor

16bpp ¤È 32bpp ¤Î X ¥µ¡¼¥Ð¤Ç»ÈÍÑ²ÄÇ½¤ÊÉ½¼¨·Á¼°¤Ï **TrueColor** ¤Ç¤¹¡£

1bpp ¤Î X ¥µ¡¼¥Ð¤Ç»ÈÍÑ²ÄÇ½¤ÊÉ½¼¨·Á¼°¤Ï **StaticGray** ¤Ç¤¹¡£

4bpp ¤Î X ¥µ¡¼¥Ð¤Ç»ÈÍÑ²ÄÇ½¤ÊÉ½¼¨·Á¼°¤Ï¼¡¤Î¤È¤ª¤ê¤Ç¤¹(É¸½à¤Ç¤Ï **PseudoColor** ¤Ç¤¹):

StaticGray
GrayScale
StaticColor
PseudoColor

Option *"optionstring"*
¤³¤ÎÄÉ²Ã¹àÌÜ¤Ï¥É¥é¥¤¥Ð¤ÎÄó¶¡¤¹¤ë°ìÄê¤ÎÁªÂò¹àÌÜ¤ò»ØÄê¤·¤Þ¤¹¡£ Ê£¿ô¤Î **Option** ¹àÌÜ¤¬»ØÄê¤Ç¤¤Þ¤¹¡£ *optionstring* ¤Ç»ÈÍÑ¤Ç¤¤ëÊ¸»úÎó¤Ï³ºÅö¤¹¤ë X ¥µ¡¼¥Ð¤Î¥ª¥ó¥é¥¤¥ó¥Þ¥Ë¥å¥¢¥ë¤ò»²¾È ¤·¤Æ¤¯¤À¤µ¤¤¡£

Black *red green blue*
¤³¤ÎÄÉ²Ã¹àÌÜ¤Ï "¹õ" ¿§¤òÄêµÁ¤·¤Þ¤¹¡£¤³¤ÎÄêµÁ¤Ï XF86_Mono ¥µ¡¼¥Ð ¤Î VGA2 ¥É¥é¥¤¥Ð¤Î¤ß¤Ç»ØÄê¤Ç¤¤Þ¤¹(¾ÜºÙ¤Ï *XF86_Mono(1)* ¤ò»²¾È ¤·¤Æ¤¯¤À¤µ¤¤)¡£

White *red green blue*
¤³¤ÎÄÉ²Ã¹àÌÜ¤Ï "Çò" ¿§¤òÄêµÁ¤·¤Þ¤¹¡£¤³¤ÎÄêµÁ¤Ï XF86_Mono ¥µ¡¼¥Ð ¤Î VGA2 ¥É¥é¥¤¥Ð¤Î¤ß¤Ç»ØÄê¤Ç¤¤Þ¤¹(¾ÜºÙ¤Ï *XF86_Mono(1)* ¤ò»²¾È ¤·¤Æ¤¯¤À¤µ¤¤)¡£

Ç¤°Õ¤Î **XInput** Àá¤Ï³ÈÄ¥ÆþÎÏµ¡´ï¤Î¹½À®¥ª¥×¥·¥ç¥ó¤ò»ØÄê¤·¤Þ¤¹¡£¤¤¤¯¤Ä¤«¤Î OS ¤Ç¤Ï **Module** Àá¤Ë¤¢¤ë³ÈÄ¥¥Ç¥Ð¥¤¥¹¥É¥é¥¤¥Ð¤ÎÆ°Åª¥í¡¼¥É¤ò¥µ¥Ý¡¼¥È¤·¤Æ¤¤¤Þ¤¹(¤³¤ì¤Ï¾å¤µ¤¤Ç½Ò¤Ù¤Þ¤·¤¿)¡£

Joystick (¥µ¥Ý¡¼¥È¤·¤Æ¤¤¤ë¥·¥¹¥Æ¥à¤ÏÎã¤¨¤Ð Linux, FreeBSD ¤È NetBSD ¤Î¤ß)
WacomStylus (Wacom ¥¿¥Ö¥ì¥Ã¥È¤Î¥¹¥¿¥¤¥é¥¹¥Ú¥ó)
WacomEraser (Wacom ¥¿¥Ö¥ì¥Ã¥È¤Î¥¤¥ì¡¼¥µ)
WacomCursor (Wacom ¥¿¥Ö¥ì¥Ã¥È¤Î¥«¡¼¥½¥ë)
Elographics (Elographics ¤Î¥¿¥Ã¥Á¥¹¥¯¥ê¡¼¥ó)
SummaSketch (SummaSketch ¥¿¥Ö¥ì¥Ã¥È)
Mouse (Mouse)

Joystick ÉûÀá ¤Ç¤Ï¼¡¤Î¹àÌÜ¤¬»ØÄê¤Ç¤¤Þ¤¹:

Port *"path"*
¤Ï¥Ç¥Ð¥¤¥¹¥É¥é¥¤¥Ð¤ò°ÕÌ£¤¹¤ë¥¹¥Ú¥·¥ã¥ë¥Õ¥¡¥¤¥ë¤Î¥Ñ¥¹¤òÀßÄê¤·¤Þ¤¹¡£

DeviceName *"name"*
¥Ç¥Ð¥¤¥¹¤Î̾¾Î(name)¤òÀßÄê¤·¤Þ¤¹¡£

TimeOut *timeout*
¤Ï¥Ç¥Ð¥¤¥¹¥É¥é¥¤¥Ð¤«¤é¤Î 2
¤Ä¤Î¥Ñ¥ë¥¹¤Î´Ö³Ö¤ò(¥ß¥êÉÃ¤Ç)»þ´Ö¤òÀßÄê¤·¤Þ¤¹¡£
¤³¤³¤ÇÍ¿¤¨¤¿¿ôÃÍ¤Ï¥ª¥Ú¥ì¡¼¥Æ¥£¥ó°¥·¥¹¥Æ¥à¤Î¥¸¥ç¥¤¥¹¥Æ¥£¥Ã¥¯¤Î¥Ç¥Ð¥¤¥¹
¥É¥é¥¤¥Ð¤Ç¾å½ñ¤¤·¤Æ¤·¤Þ¤¦¤«¤â¤·¤ì¤Þ¤»¤ó¡£

MaximumXPosition *value*
¤Ï¥Ç¥Ð¥¤¥¹¥É¥é¥¤¥Ð¤¬Êó¹ð¤¹¤ë X °ÂÉ¸¤Î°ÇÂçÃÍ¤òÀßÄê¤·¤Þ¤¹¡£

MininimumXPosition *value*
¤Ï¥Ç¥Ð¥¤¥¹¥É¥é¥¤¥Ð¤¬Êó¹ð¤¹¤ë X °ÂÉ¸¤Î°Ç¾®ÃÍ¤òÀßÄê¤·¤Þ¤¹¡£

MaximumYPosition *value*
¤Ï¥Ç¥Ð¥¤¥¹¥É¥é¥¤¥Ð¤¬Êó¹ð¤¹¤ë Y °ÂÉ¸¤Î°ÇÂçÃÍ¤òÀßÄê¤·¤Þ¤¹¡£

MinimumYPosition *value*
¤Ï¥Ç¥Ð¥¤¥¹¥É¥é¥¤¥Ð¤¬Êó¹ð¤¹¤ë Y °ÂÉ¸¤Î°Ç¾®ÃÍ¤òÀßÄê¤·¤Þ¤¹¡£

CenterX *value*
¥¸¥ç¥¤¥¹¥Æ¥£¥Ã¥¯¤¬¥¢¥¤¥É¥ë¤Î°ÌÃÖ¤Ë¤¢¤ë¤È¤¤Ë¥Ç¥Ð¥¤¥¹¥É¥é¥¤¥Ð¤¬Êó¹ð¤¹¤ë
X °ÂÉ¸¤ÎÃæ¿´¤òÀßÄê¤·¤Þ¤¹¡£

CenterY *value*
¥¸¥ç¥¤¥¹¥Æ¥£¥Ã¥¯¤¬¥¢¥¤¥É¥ë¤Î°ÌÃÖ¤Ë¤¢¤ë¤È¤¤Ë¥Ç¥Ð¥¤¥¹¥É¥é¥¤¥Ð¤¬Êó¹ð¤¹¤ë
Y
°ÂÉ¸¤ÎÃæ¿´¤òÀßÄê¤·¤Þ¤¹¡£¤³¤Î¿ôÃÍ¤¬¾ÊÎ¬¤µ¤ì¤Æ¤¤¤ë¤È°Ç½é¤ËÍ¸ú¤Ë¤Ê¤Ã¤¿
»þ¤Î¥¸¥ç¥¤¥¹¥Æ¥£¥Ã¥¯¤Î°ÌÃÖ¤òÃæ¿´¤È¤·¤Þ¤¹¡£

Delta *value*
¤Ï X
¥µ¡¼¥Ð¤¬Êó¹ð¤¹¤ë°ÇÂç¤Î¿ôÃÍ¤òÀßÄê¤·¤Þ¤¹¡£Îã¤¨¤Ð°ÂÉ¸¤Ç¤Ï¤¤¤¤Ã¤Ñ¤¤¤Ë¶Ê¤²¤¿
»þ¤Î¿ôÃÍ¤ò *(+/−)value/2*
¤È¤·¤ÆÄýÊ¬ÃÍ¤ÇÀßÄê¤·¤Þ¤¹¡£´¶³Ð¤Ç·èÄê¤·¤Æ ¤¯¤À¤µ¤¤¡£

WacomStylus, **WacomEraser** ¤È **WacomCursor** ÉûÀá¤Ç¤Ï
¤Ç¤Ï¼¡¤Î¹àÌÜ¤¬»ØÄê¤Ç¤¤Þ¤¹:

Port *"path"*
¤Ï¥¿¥Ö¥ì¥Ã¥È¤¬ÀÜÂ³¤µ¤ì¤Æ¤¤¤ë¥·¥ê¥¢¥ë¥Ý¡¼¥È¤Î¥¹¥Ú¥·¥ã¥ë¥Õ¥¡¥¤¥ë¤Î¥Ñ¥¹¤ò
ÀßÄê¤·¤Þ¤¹¡£Æ±¤¸¥¿¥Ö¥ì¥Ã¥È¤ÇÊ£¿ô¤Î¥Ç¥Ð¥¤¥¹¤ò»È¤¦¤¿¤á¤Ë¤Ï
¤½¤ì¤¾¤ì¤ÎÉûÀá¤Ë¤³¤Î¥¹¥Ú¥·¥ã¥ë¥Õ¥¡¥¤¥ë¤òÀßÄê¤¹¤ëÉ¬Í×¤¬¤¢¤ê¤Þ¤¹¡£

DeviceName *"name"*
¤Ï¥Ç¥Ð¥¤¥¹¤Î̾¾Î(name)¤òÀßÄê¤·¤Þ¤¹¡£

Suppress *number*
¤Ï¥¿¥Ö¥ì¥Ã¥È¤¬°ÂÉ¸¤òÅ¾Á÷¤·¤Ê¤¤¤è¤¦¤Ë¤¹¤ë°ÌÃÖ¤ÎÁýÊ¬¤ÎÎÌ¤òÀßÄê¤·¤Þ¤¹¡£
¤³¤Î¹àÌÜ¤Ï°ì¤Ä¤Î¥¿¥Ö¥ì¥Ã¥È¤ÇÊ£¿ô¤Î¥Ç¥Ð¥¤¥¹¤¬É¬Í×¤Ê¤È¤¤ä¾ì¹ç¡¢
°Ç½é¤Î Wacom ÉûÀá¤Ë¤Î¤ßÀßÄê¤·¤Ê¤±¤ì¤Ð¤¤¤±¤Þ¤»¤ó¡£

Mode *Relative|Absolute*
¤Ï¥Ç¥Ð¥¤¥¹¤Î¥â¡¼¥É(ÁêÂÐ°ÌÃÖ¥â¡¼¥É|ÀäÂÐ°ÌÃÖ¥â¡¼¥É)¤òÀßÄê¤·¤Þ¤¹¡£

TiltMode
¤Ï¤â¤·¥Á¥ë¥È¾ðÊó¤ò¥¿¥Ö¥ì¥Ã¥È¤¬¥µ¥Ý¡¼¥È¤·¤Æ¤¤¤ë¤È¤(ROM
¤Î¥Ð¡¼¥¸¥ç¥ó¤¬ 1.4 °Ê¾å¤Î¾ì¹ç) ¥Á¥ë¥È¾ðÊó¤òÍ¸ú¤Ë¤·¤Þ¤¹¡£

HistorySize *number*
¿åÊ¿Êý¸þ¤ÎÆ°¤¤ÎÂç¤¤µ¤òÀßÄê¤·¤Þ¤¹¡£É¸½àÃÍ¤Ï 0 ¤Ç¤¹¡£

Elographics ÉûÀá¤Ç¤Ï¼¡¤Î¹àÌÜ¤¬»ØÄê¤Ç¤¤Þ¤¹:

Port *"path"*
¥Ç¥Ð¥¤¥¹É¥é¥¤¥Ð¤ò°Õ¸£¤¹¤ë¥¹¥Ú¥·¥ã¥ë¥Õ¥¡¥¤¥ë¤Î¥Ñ¥¹¤òÀßÄê¤·¤Þ¤¹¡£

DeviceName *"name"*
¤Ï¥Ç¥Ð¥¤¥¹¤ÎÌ¾¾Î(name)¤òÀßÄê¤·¤Þ¤¹¡£

MaximumXPosition *position*
¤Ï¥¿¥Ã¥Á¥¹¥¯¥ê¡¼¥ó¤ÎÊó¹ð¤¹¤ë X °ÂÉ¸¤Îº÷ÂçÃÍ¤òÀßÄê¤·¤Þ¤¹¡£

MinimumXPosition *position*
¤Ï¥¿¥Ã¥Á¥¹¥¯¥ê¡¼¥ó¤ÎÊó¹ð¤¹¤ë X °ÂÉ¸¤ÎºÇ¾®ÃÍ¤òÀßÄê¤·¤Þ¤¹¡£

MaximumYPosition *position*
¤Ï¥¿¥Ã¥Á¥¹¥¯¥ê¡¼¥ó¤ÎÊó¹ð¤¹¤ë Y °ÂÉ¸¤Îº÷ÂçÃÍ¤òÀßÄê¤·¤Þ¤¹¡£

MinimumYPosition *position*
¤Ï¥¿¥Ã¥Á¥¹¥¯¥ê¡¼¥ó¤ÎÊó¹ð¤¹¤ë Y °ÂÉ¸¤ÎºÇ¾®ÃÍ¤òÀßÄê¤·¤Þ¤¹¡£

ScreenNo *number*
¤Ï¥¿¥Ã¥Á¥¹¥¯¥ê¡¼¥ó¤òÀÜÂ³¤·¤¿¤È¤¤Î²èÌÌÈÖ¹æ¤òÀßÄê¤·¤Þ¤¹¡£

UntouchDelay *value*
¤Ï²èÌÌ¤«¤éÊü¤¹¤¿(untouch) ¤È¥Ç¥Ð¥¤¥¹¤¬¤ß¤Ê¤¹¤Þ¤Ç¤ÎÃÙ¤ì»þ´Ö¤ò
(10 ¤ÎÇÜ¿ô¤Î¥ß¥êÉÃÃ±°Ì¤Ç)ÀßÄê¤·¤Þ¤¹¡£

ReportDelay *value*
2 ²Ó½ê¤Î°ÌÃÖ¤Î´Ö¤ÎÃÙ¤ì»þ´Ö¤ò(¥ß¥êÉÃÃ±°Ì¤Ç)ÀßÄê¤·¤Þ¤¹¡£

SummaSketch ÉûÀá¤Ç¤Ï¼¡¤Î¹àÌÜ¤¬»ØÄê¤Ç¤¤Þ¤¹:

Port *"path"*
¥Ç¥Ð¥¤¥¹É¥é¥¤¥Ð¤ò°Õ¸£¤¹¤ë¥¹¥Ú¥·¥ã¥ë¥Õ¥¡¥¤¥ë¤Î¥Ñ¥¹¤òÀßÄê¤·¤Þ¤¹¡£

DeviceName *"name"*
¤Ï¥Ç¥Ð¥¤¥¹¤ÎÌ¾¾Î(name)¤òÀßÄê¤·¤Þ¤¹¡£

Mode *Relative|Absolute*
¤Ï¥Ç¥Ð¥¤¥¹¤Î¥â¡¼¥É(ÁêÂÐ°ÌÃÖ¥â¡¼¥É|ÀäÂÐ°ÌÃÖ¥â¡¼¥É)¤òÀßÄê¤·¤Þ¤¹¡£

Cursor *Stylus|Puck*
¤Ï¥¹¥¿¥¤¥é¥¹¥Ú¥ó¤« 4
¥Ü¥¿¥ó¥Ñ¥Ã¥¯¤Ê¤Î¤«¡¢¥«¡¼¥½¥ë¤Î·Á¼°¤òÀßÄê¤·¤Þ¤¹¡£

Increment *value*
¤Ï¿·¤·¤¤Êó¹ð¤ò¼õ¤±¤ëÁ°¤Ë°ÂÉ¸¤ÎºÇÂçÊÑ²½ÎÌ¤òÀßÄê¤·¤Þ¤¹¡£

HistorySize *number*
¤ÏÁà°ÎúÎò¤ÎÂç¤¤µ¤òÀßÄê¤·¤Þ¤¹¡£É¸½àÃÍ¤Ï 0 ¤Ç¤¹¡£

Mouse ÉûÀá¤ÏÉ¸½à¤Î **Pointer** Àá¤ÈÆ±¤¸¤Ç¤¹¡£
¤½¤ì¤ËÉÕ¤±²Ã¤¨¤Æ¼¡¤Î¹àÌÜ¤¬»ØÄê¤Ç¤¤Þ¤¹:

DeviceName *"name"*
¤Ï¥Ç¥Ð¥¤¥¹¤ÎÌ¾¾Î(name)¤òÀßÄê¤·¤Þ¤¹¡£

XF86Config ¥Õ¥¡¥¤¥ë¤ÎÎãÂê¤Ï<XRoot>/lib/X11/XF86Config.eg ¤Ë¤¢¤ê¤Þ¤¹¡£

¥Õ¥¡¥¤¥ë

/etc/XF86Config
<XRoot>/lib/X11/XF86Config.*hostname*
<XRoot>/lib/X11/XF86Config

Ãí°Õ: <XRoot> ¤Ï X11 ¤òÆ³Æþ¤·¤¿¥Ç¥£¥ì¥¯¥È¥ê¤Ç¤¹¡£

´ØÏ¢¹àÌÜ

X(1), Xserver(1), XFree86(1), XF86_SVGA(1), XF86_VGA16(1), XF86_Mono(1), XF86_S3(1), XF86_8514(1), XF86_Mach8(1), XF86_Mach32(1), XF86_P9000(1), XF86_AGX(1), XF86_W32(1).

Ãø¼Ô

XFree86(1) ¤Î¥ª¥ó¥é¥¤¥ó¥Þ¥Ë¥å¥¢¥ë¤ò»²¾È¤·¤Æ¤¯¤À¤µ¤¤¡£

Ì¾Á°

XFree86 - x86 ¥Þ¥·¥ó¤ÇÆ°ºî¤¹¤ë UNIX ÍÑ¤Î X11R6

ÀâÌÀ

XFree86 ¤Ï¥¤¥ó¥Æ¥ë x86 ¥Þ¥·¥ó¤ÇÆ°ºî¤¹¤ë UNIX ¤Ë»÷¤¿ OSÍÑ¤Î X ¥µ¡¼¥Ð¤ò¼ý½¸¤·¤¿¤â¤Î¤Ç¤¹¡£ XFree86 ¤Ï *X386 1.2* ¤«¤éÇÉÀ¸¤·¡¢ Snitily Graphics Consulting Service ¤Ë¤è¤Ã¤Æ X11R5 ¤Ë´óÂ£¤µ¤ì¤Þ¤·¤¿¡£

¹½À®

XFree86 ¤Ï¼¡¤Î¥ª¥Ú¥ì¡¼¥Æ¥£¥ó¥°¥·¥¹¥Æ¥à¤Ç»ÈÍÑ²ÄÇ½¤Ç¤¹¡£:

-- SVR3.2: SCO 3.2.2, 3.2.4, ISC 3.x, 4.x
-- SVR4.0: ESIX, Microport, Dell, UHC, Consensys, MST, ISC, AT&T, NCR
-- SVR4.2: Consensys, Univel (UnixWare)
-- Solaris (x86) 2.1, 2.4
-- FreeBSD 1.1.5, 2.0, 2.0.5, NetBSD 1.0 (i386 ¤Ë¤Î¤ß°Ü¿¢¤µ¤ì¤Æ¤¤¤Þ¤¹)
-- BSD/386 version 1.1 ¤È BSD/OS 2.0
-- Mach (CMU ¤«¤é)
-- Linux
-- Amoeba version 5.1
-- Minix-386vm version 1.6.25.1
-- LynxOS AT versions 2.2.1, 2.3.0 and 2.4.0, LynxOS microSPARC 2.4.0

¥Í¥Ã¥È¥ï¡¼¥¯ ÀÜÂ³

XFree86 ¤Ï°Ê²¼¤Î³Î¼Â¤ÊÊÝ¸¤¥È-¥¹¥È¥ê¡¼¥à¤Ë¤è¤ëÀÜÂ³¤ò¥µ¥Ý¡¼¥È¤·¤Æ¤¤¤Þ¤¹¡£:

Local

XFree86 ¤Ï°Ê²¼¤Î¥Ñ¥¹¤Î¼ï¡¹¤Î»Å³Ý¤±¤Î Streams ¥Ñ¥¤¥×·Ð³¤Î local ÀÜÂ³¤ò¥µ¥Ý¡¼¥È¤·¤Æ¤¤¤Þ¤¹¡£(¤³¤³¤Ç *n* ¤Ï¥Ç¥£¥¹¥×¥ì¥¤ÈÖ¹æ¤Ç¤¹):

/dev/X/server.**n** (SVR3 ¤È SVR4)
/dev/X/Nserver.**n** (SVR4)
/dev/X**n**S and /dev/X**n**R (SCO SVR3)

SVR4.0.4 ¤Ç¤Ï¡¢*Advanced Compatibility Package* ¤¬¥¤¥ó¥¹¥È¡¼¥ë¤µ¤ì¤Æ¤¤¤ë¾ì¹ç¤È SVR4.2 ¤Î¾ì¹ç¤Ï¡¢ *XFree86* ¤Ï SCO XSight/ODT ÍÑ¤Î¥¯¥é¥¤¥¢¥ó¥È¤«¤é¤Î¡Ê¥Ð¥¤¥Ê¥ê¥Ñ¥Ã¥±¡¼¥¸¤ËÊÑ¹¹¤ò²Ã¤¨¤¿¡ËISC SVR3 ÍÑ¤Î¥¯¥é¥¤¥¢¥ó¥È ¤«¤é¤Î local ÀÜÂ³¤ò¥µ¥Ý¡¼¥È¤·¤Æ¤¤¤Þ¤¹¡£

Unix Domain

XFree86 ¤Ï */tmp/.X11-unix/X***n** ¤È¤¤¤¦¥Õ¥¡¥¤¥ë¤ò¥½¥±¥Ã¥ÈÍÑ¤Ë»ÈÍÑ¤·¤Þ¤¹¡£¤³¤³¤Ç *n* ¤Ï¥Ç¥£¥¹¥×¥ì¥¤ÈÖ¹æ¤Ç¤¹¡£

TCPIP

XFree86 ¤Ï htons(6000+*n*) ¥Ý¡¼¥È¤«¤éÊ¹¤- (listen) ¤Þ¤¹¡£¤³¤³¤Ç *n* ¤Ï¥Ç¥£¥¹¥×¥ì¥¤ÈÖ¹æ¤Ç¤¹¡£

Amoeba RPC

Amoeba ¤Ç¤ÏÉ¸½à¤ÎÄÌ¿®¼êÃÊ¤Ç¤¹¡£Amoeba ¤Ç¤ÎÃí°Õ»ö¹à¤È¤·¤Æ¡¢¥µ¡¼¥Ð¤ò "*¥Û¥¹¥È¾:¥Ç¥£¥¹¥×¥ì¥¤ÈÖ¹æ*" ¤ò°ú¤-¿ô¤Ë²Ã¤¨¤Æµ¯Æ°¤·¤Ê¤±¤ì¤Ð ¤¤¤±¤Þ¤»¤ó¡£

´Ä¶-ÀßÄê

Unix Domain ¥½¥±¥Ã¥È (SVR3 ¤È SVR4) °Ê³°¤Î¥í¡¼¥«¥ëÀÜÂ³¤ò¥µ¥Ý¡¼¥È¤·¤Æ¤¤¤ë ¥ª¥Ú¥ì¡¼¥Æ¥£¥ó¥°¥·¥¹¥Æ¥à¤Ï¤¿¤á¤Ë¡¢ÀÜÂ³¤ò»î¤ß¤ë½çÈÖ¤¬ÁÈ¤ß¹þ¤Þ¤ì¤¿¥ê¥¹¥È¤¬¤¢¤ê¤Þ¤¹ ¤³¤Î¥ê¥¹¥È¤Ï¸å½Ò¤Î *XLOCAL*

´Ä¶-ÊÑ¿ô¤Ë¤è¤Ã¤Æ¾å½ñ¤-²ÄÇ½¤Ç¤¹¡£¥Ç¥£¥¹¥×¥ì¥¤Ì¾¾Î ¤¬ºÇÅ¬¤ÊÀÜÂ³¤¬½ÐÍè¤ë¤è¤¦¤Ë (Îã¤¨¤Ð *:0.0*) ¤Ê¤Ã¤Æ¤¤¤ë¾ì¹ç¤Ï¡¢¤½¤ì¤¾¤ì¤Î ÀÜÂ³μ¡¹½¤ÏÀÜÂ³¤¬´°Î»¤¹¤ë¤Þ¤Ç¤«¡¢Ëô¤Ï»ÈÍÑ½ÐÍè¤ëμ¡¹½¤¬Ìμ¤¯¤Ê¤ë¤Þ¤Ç»î¹Ô¤·¤Þ¤¹¡£ Ãí°Õ»ö¹à: ¤³¤ì¤é¤Î OS ¤Ç¤Ï Unix Domain ¥½¥±¥Ã¥ÈÀÜÂ³¤Ï¤½¤ÎÂ¾¤Î¥í¡¼¥«¥ëÀÜÂ³¤Î·¿¤È¤Ï °Û¤Ê¤ë¼è¤ê°·¤¤¤Ç¤¹¡£Unix Domain ¥½¥±¥Ã¥ÈÀÜÂ³¤ò»ÈÍÑ¤¹¤ë¤Ë¤Ï¥Ç¥£¥¹¥×¥ì¥¤Ì¾¾Î¤Ë *unix:0.0* ¤È»ØÄê¤·¤Ê¤±¤ì¤Ð¤¤¤±¤Þ¤»¤ó¡£

XLOCAL ´Ä¶-ÊÑ¿ô¤Ë¤Ï°Ê²¼¤Î¥ê¥¹¥È¤¬´Þ¤Þ¤ì¤Æ¤¤¤Þ¤¹:

NAMED
PTS
SCO
ISC

¤½¤ì¤¾¤ì¤Ï¡¢SVR4 Named Streams pipe, Old-style USL Streams pipe, SCO XSight Streams pipe ¤È ISC Streams pipe ¤òÉ½¤·¤Æ¤¤¤Þ¤¹¡£ ¤³¤ÎÆâ¡¢°ì¤Ä¤Îμ¡¹½¤òÁªÂò¤·¤Þ¤¹ (Îã¤¨¤Ð *XLOCAL=NAMED*)¡¢ ¼ã¤·¤¯¤Ï ½ç½øÉÕ¤±¤·¤¿¥ê¥¹¥È¤òÁªÂò¤·¤Þ¤¹(Îã¤¨¤Ð *XLOCAL="NAMED:PTS:SCO"*)¡£ ¤³¤Î´Ä¶-ÊÑ¿ô¤ÏÁÈ¤ß¹þ¤Þ¤ì¤Æ¤¤¤ëÉ¸½àÃÍ¤ò¾å½ñ¤-¤·¤Þ¤¹¡£SVR4 ¤Ç¤Ïº¹½é¤ËÍ¥Àè¤ª¤Ê ÀÜÂ³¤Ë *NAMED* ¤ò¿ä¾©¤·¤Æ¤¤¤Þ¤¹¡£É¸½àÃÍ¤Ï *PTS:NAMED:ISC:SCO* ¤Ç¤¹¡£

ÁÈ¹þ¤ß¤ÎÉ¸½àÃÍ¤òÁ´ÂÎÅª¤ËÃÖ¤-´¹¤¨¤ë¤Ê¤é¤Ð¡¢ *XLOCAL* ¤òÁ´ÂÎÅª¤ËÄêμÁ(define) (¤½¤·¤Æ¡¢*csh* ¤Þ¤¿¤Ï *ksh* ¤ò»È¤Ã¤Æ¤¤¤ë¤Ê¤é¤Ð export) ¤·¤Þ¤·¤ç¤¦¡£ *startx/xinit* ¤ò»È¤¦¤Ê¤é¤Ð¡¢*.xinitrc* ¥Õ¥¡¥¤¥ë¤ÎÀèÆ¬¤Ë ÄêμÁ¤·¤Þ¤·¤ç¤¦¡£*xdm* ¤ò»È¤¦¤Ê¤é¤Ð *<XRoot>/lib/X11/xdm/Xsession* ¥¹¥¯¥ê¥×¥È¤ÎÁ°¤ÎÊý¤ËÄêμÁ¤·¤Þ¤·¤ç¤¦¡£

¥ª¥×¥·¥ç¥ó

Xserver(1) ¤È¤¤¤¦¥ª¥ó¥é¥¤¥ó¥Þ¥Ë¥å¥¢¥ë¤ËÄÌ¾ï¤Î¥μ¡¼¥Ð¤ØØ¤Î¥ª¥×¥·¥ç¥ó ¤Ë¤Ä¤¤¤Æ¤ÀâÌÀ¤¬¤¢¤ë¤³¤È¡¢*XFree86* ¤Ï°Ê²¼¤Î¥³¥Þ¥ó¥É¥é¥¤¥ó¥ª¥×¥·¥ç¥ó¤ò ¼õ¤±ÉÕ¤±¤ë»ö¤òÄÉμ-¤·¤Þ¤¹:

vt*XX* *XX* ¤Ï *XFree86* ¤¬»ÈÍÑ¤¹¤ë²¾ÁÛÃ¼Ëö(Virtual Terminal) ¤Î¥Ç¥Ð¥¤¥¹ÈÖ¹æ ¤Ç¤¹¡£¤³¤Î¥ª¥×¥·¥ç¥ó¤òÉÕ¤±¤Ê¤¤¤È¡¢ *XFree86* ¤Ï°Ç½é¤Ë»ÈÍÑ²ÄÇ½¤Ê²¾ÁÛÃ¼Ëö¤ò ³äÅö¤Æ¤Þ¤¹¡£¤³¤Î¥ª¥×¥·¥ç¥ó¤Ï 'syscons' ¤Þ¤¿¤Ï 'pcvt' ¥É¥é¥¤¥Ð¤¬¤¢¤ë SVR3, SVR4, Linux ¤È BSD OS ¤À¤±¤ËÍ-¸ú¤Ç¤¹¡£

-crt /dev/tty*XX*
SCO ¤Î¤ß¥μ¥Ý¡¼¥È¤·¤Æ¤¤¤Þ¤¹¡£ **vt** ¥ª¥×¥·¥ç¥ó¤ÈÆ±Åù¤Ç¡¢SCO ¾å¤Î X ¥μ¡¼¥Ð¤È¤Î¸ß´¹À-¤Î°Ù¤ËÄó¶¡¤·¤Þ¤¹¡£

−probeonly
μ¡¯ï¾∂Êó¤ÎÃμºº¸å¤Ë¥μ¡¼¥Ð¤ò½ªÎ»¤μ¤»¤Þ¤¹¡£XF86Config ¥Õ¥¡¥¤¥ë¤Ï¤³¤Î¥ª¥×¥·¥ç¥ó ¤òÉÕ¤±¤¿¤È¤-¤Ç¤â»ÈÍÑ¤μ¤ì¤ë¤Î¤Ç¡¢¼«Æ°Ãμºº¤·¤Æ¤â¤è¤¤¾∂Êó¤Ï¥³¥á¥ó¥È¤Ë¤·¤Æ¤ª¤-¤Þ ¤·¤ç¤¦¡£

−quiet μ¯Æ°»þ¤Î¤Û¤È¤ó¤É¤Î¥á¥Ã¥»¡¼¥¸¤òÍÞ»ß¤·¤Þ¤¹¡£

−bpp *n* ¥Ô¥¯¥»¥ë¤è¤ê¤Î¥Ó¥Ã¥È¿ô¤òÀßÄê¤·¤Þ¤¹¡£É¸½àÃÍ¤Ï 8 ¤Ç¤¹¡£»ÈÍÑ²ÄÇ½¤ÊÃÍ¤Ï 8, 15, 16, 24, 32 ¤Ç¤¹¡£¤¹¤Ù¤Æ¤ÎÃÍ¤¬¤¹¤Ù¤Æ¤Î¥μ¡¼¥Ð¤Ç»È¤¨¤ë¤ï¤±¤Ç¤Ï ¤¢¤ê¤Þ¤»¤ó¡£

−weight *nnn*
16 bpp ¤Ç¤Î RGB ½Å¤ßÉÕ¤±¤òÀßÄê¤·¤Þ¤¹¡£É¸½àÃÍ¤Ï 565 ¤Ç¤¹¡£¤³¤ì¤Ï 16

bpp ¤ò¥µ¥Ý¡¼¥È¤·¤Æ¤¤¤ë¥µ¡¼¥Ð¤Ç¤·¤«Å¬±þ½ÐÍè¤Þ¤»¤ó¡£

–flipPixels

Çò¤È¹õ¤Î¥Ô¥¯¥»¥ë¤ÎÉ¸½àÃÍ¤òÆþ¤ìÂØ¤¨¤Þ¤¹¡£

–disableVidMode

¥Ó¥Ç¥ª¥â¡¼¥É¤òÊÑ¹¹¤¹¤ë xvidtune ¥¯¥é¥¤¥¢¥ó¥È¤Ç»ÈÍÑ¤·¤Æ¤¤¤ë Vid-Mode µ¡Ç½³ÈÄ¥ ¤Î°ìÉô¤òÌµ¸ú¤Ë¤·¤Þ¤¹¡£

–allowNonLocalXvidtune

Â¾¤Î¥Û¥¹¥È¤«¤éÀÜÂ³¤·¤Æ xvidtune ¥¯¥é¥¤¥¢¥ó¥È¤Î¼Â¹Ô¤òµö²Ä¤¹¤ë¡£É¸½à¤Ç¤Ï ¥í¡¼¥«¥ëÀÜÂ³°Ê³°¤Ï¶Ø»ß¤·¤Æ¤¤¤Þ¤¹¡£

–gamma *value*

¥¬¥ó¥ÞÊäÀµ¤òÀßÄê¤·¤Þ¤¹¡£*value* ¤Ï 0.1 ¤È 10 ¤Î¤¢¤¤¤À¤ÎÃÍ¤Ç¤Ê¤±¤ì¤Ð ¤¤¤±¤Þ¤»¤ó¡£É¸½àÃÍ¤Ï 1.0 ¤Ç¤¹¡£¤³¤ÎÃÍ¤Ï R, G ¤È B ¤Ë¸øÊ¿¤ËÅ¬±þ¤µ¤ì¤Þ¤¹¡£ ¤³¤ì¤ÏÁ´¤Æ¤Î¥µ¡¼¥Ð¤Ç»È¤¨¤ë¤ï¤±¤Ç¤Ï¤¢¤ê¤Þ¤»¤ó¡£

–rgamma *value*

ÀÖ¤Î¥¬¥ó¥ÞÊäÀµ¤òÀßÄê¤·¤Þ¤¹¡£*value* ¤Ï 0.1 ¤È 10 ¤Î¤¢¤¤¤À¤ÎÃÍ¤Ç¤Ê¤±¤ì¤Ð ¤¤¤±¤Þ¤»¤ó¡£É¸½àÃÍ¤Ï 1.0 ¤Ç¤¹¡£ ¤³¤ì¤ÏÁ´¤Æ¤Î¥µ¡¼¥Ð¤Ç»È¤¨¤ë¤ï¤±¤Ç¤Ï¤¢¤ê¤Þ¤»¤ó¡£

–ggamma *value*

ÎÐ¤Î¥¬¥ó¥ÞÊäÀµ¤òÀßÄê¤·¤Þ¤¹¡£*value* ¤Ï 0.1 ¤È 10 ¤Î¤¢¤¤¤À¤ÎÃÍ¤Ç¤Ê¤±¤ì¤Ð ¤¤¤±¤Þ¤»¤ó¡£É¸½àÃÍ¤Ï 1.0 ¤Ç¤¹¡£ ¤³¤ì¤ÏÁ´¤Æ¤Î¥µ¡¼¥Ð¤Ç»È¤¨¤ë¤ï¤±¤Ç¤Ï¤¢¤ê¤Þ¤»¤ó¡£

–bgamma *value*

ÀÄ¤Î¥¬¥ó¥ÞÊäÀµ¤òÀßÄê¤·¤Þ¤¹¡£*value* ¤Ï 0.1 ¤È 10 ¤Î¤¢¤¤¤À¤ÎÃÍ¤Ç¤Ê¤±¤ì¤Ð ¤¤¤±¤Þ¤»¤ó¡£É¸½àÃÍ¤Ï 1.0 ¤Ç¤¹¡£ ¤³¤ì¤ÏÁ´¤Æ¤Î¥µ¡¼¥Ð¤Ç»È¤¨¤ë¤ï¤±¤Ç¤Ï¤¢¤ê¤Þ¤»¤ó¡£

–showconfig

¥µ¡¼¥Ð¤ËÀßÄê¤µ¤ì¤Æ¤¤¤ë¥¹¥¯¥ê¡¼¥ó¥É¥é¥¤¥Ð¤Î°ìÍ÷¤òÉ½¼¨¤·¤Þ¤¹¡£

–verbose

µ¯Æ°»þ¤ËÉ½¼¨¤µ¤ì¤ë¾ðÊó¤òºÇÂç²½¤·¤Þ¤¹(É¸½à¤è¤êÁý¤¨¤Þ¤¹)¡£

–version –showconfig ¤ÈÆ±¤¸¤Ç¤¹¡£

–xf86config *file*

file ¤«¤é¥µ¡¼¥Ð¤Î¹½À®¾ðÊó¤òÆÉ¤ß¹þ¤ß¤Þ¤¹¡£¤³¤Î¥ª¥×¥·¥ç¥ó¤Ï root (Îã¤¨¤Ð¡¢ÆâÉô uid ÈÖ¹æ ¤¬ 0) ¤Çµ¯Æ°¤·¤¿¥µ¡¼¥Ð¤Î¤È¤¤Ë¤Î¤ßÍ¸ú¤Ç¤¹¡£

–keeptty ºÇ½é¤Ëµ¯Æ°¤·¤¿À©¸æÍÑ¤ÎÃ¼Ëö¤ò¥µ¡¼¥Ð¤¬ÀÚ¤êÎ¥¤µ¤Ê¤¤¤è¤¦¤Ë¤·¤Þ¤¹¡£ ¤³¤Î¥ª¥×¥·¥ç¥ó¤Ï¥µ¡¼¥Ð¤Î¥Ç¥Ð¥Ã¥°»þ¤Ë¤Î¤ßÍÍÑ¤Ç¤¹¡£

¥¡¼¥Ü¡¼¥É

XFree86 ¤¬Ä¾ÀÜÇ§¼±¤¹¤ëÆ±»þ¤ËÊ£¿ô¤Î¥¡¼ÆþÎÏ¤Ï°Ê²¼¤Î¤È¤ª¤ê¤Ç¤¹:

Ctrl+Alt+Backspace

Â®¤ä¤«¤Ë¥µ¡¼¥Ð¤òÄä»ß¤·¤Þ¤¹ -- ²¿¤â³þÅú¤Ï¤¢¤ê¤Þ¤»¤ó¡£(XF86Config ¥Õ¥¡¥¤¥ë¤Î **ServerFlags** Àá¤Ë "DontZap" ¤È»ØÄê¤¹¤ë»ö¤Ë¤è¤êÌµ¸ú¤Ë½ÐÍè¤Þ¤¹¡£)

Ctrl+Alt+Keypad-Plus

¹½À®¾ðÊó¥Õ¥¡¥¤¥ë¤Ë»ØÄê¤·¤Æ¤¢¤ë¼¡¹Ô¤Î¥Ó¥Ç¥ª¥â¡¼¥É(¥Ó¥Ç¥ª¤Î²òÁüÅÙ¤¬Áý¤¹¤ëÊý¸þ¤

¤ËÊÑ¹¹¤·¤Þ¤¹¡£

Ctrl+Alt+Keypad-Minus

¹½À®¾ðÊó¥Õ¥¡¥¤¥ë¤Ë»ØÄê¤·¤Æ¤¢¤ëÁ°¹Ô¤Î¥Ó¥Ç¥ª¥â¡¼¥É(¥Ó¥Ç¥ª¤Î²òÁüÅÙ¤¬¸º¾¯¤¹¤ëÊý¸þ¤Ë
¤ËÊÑ¹¹¤·¤Þ¤¹¡£

Ctrl+Alt+F1...F12

syscons ¥É¥é¥¤¥Ð¤ò»ÈÍÑ¤·¤Æ¤¤¤ë BSD ¥·¥¹¥Æ¥à¤È Linux ¤Ç¤Ï¡¢¤³¤Î¤Æþ¤Ï¤ÎÁÈ¹ç¤ï¤»¤Ï ²¾ÁÛ¥³¥ó¥½¡¼¥ë£±ÈÖ¤«¤é£±£²ÈÖ¤Þ¤ÇËÀÚ¤êÂØ¤¨¤ë¤Î¤Ë»È¤Ã¤Æ¤¤¤Þ¤¹¡£

ÀßÄê

XFree86 ¤Ï½é´ü¾õÂÖ¤Ç¤Ï **XF86Config** ¤È¸Æ¤Ð¤ì¤ë¹½À®¾ðÊó¥Õ¥¡¥¤¥ë¤ò»ÈÍÑ¤·¤Æ¤¤¤Þ¤¹¡£ ¤è¤ê¾Ü¤·¤¯¤Ï *XF86Config(4/5)* ¤Î¥ª¥ó¥é¥¤¥ó¥Þ¥Ë¥å¥¢¥ë¤ò»²¾È¤·¤Æ¤¯¤À¤µ¤¤¡£

¥Õ¥¡¥¤¥ë

<XRoot>/bin/XF86_SVGA	¥«¥é¡¼¤Î SVGA X ¥µ¡¼¥Ð¤Ç¤¹¡£
<XRoot>/bin/XF86_Mono	VGA ¤ÈÂ¾¤ÎÇò¹õ¤Î¥Ó¥Ç¥ª¥«¡¼¥ÉÍÑ¤ÎÇò¹õ¤Î X ¥µ¡¼¥Ð¤Ç¤¹¡£
<XRoot>/bin/XF86_S3	S3 ÍÑ¤Î¥¢¥¯¥»¥é¥ì¡¼¥¿ X ¥µ¡¼¥Ð¤Ç¤¹¡£
<XRoot>/bin/XF86_Mach8	Mach8 ÍÑ¤Î¥¢¥¯¥»¥é¥ì¡¼¥¿ X ¥µ¡¼¥Ð¤Ç¤¹¡£
<XRoot>/bin/XF86_Mach32	Mach32 ÍÑ¤Î¥¢¥¯¥»¥é¥ì¡¼¥¿ X ¥µ¡¼¥Ð¤Ç¤¹¡£
<XRoot>/bin/XF86_Mach64	Mach64 ÍÑ¤Î¥¢¥¯¥»¥é¥ì¡¼¥¿ X ¥µ¡¼¥Ð¤Ç¤¹¡£
<XRoot>/bin/XF86_P9000	P9000 ÍÑ¤Î¥¢¥¯¥»¥é¥ì¡¼¥¿ X ¥µ¡¼¥Ð¤Ç¤¹¡£
<XRoot>/bin/XF86_AGX	AGX ÍÑ¤Î¥¢¥¯¥»¥é¥ì¡¼¥¿ X ¥µ¡¼¥Ð¤Ç¤¹¡£
<XRoot>/bin/XF86_W32	ET4000/W32 ÍÑ¤Î¥¢¥¯¥»¥é¥ì¡¼¥¿ X ¥µ¡¼¥Ð¤Ç¤¹¡£
<XRoot>/bin/XF86_8514	8514/A ÍÑ¤Î¥¢¥¯¥»¥é¥ì¡¼¥¿ X ¥µ¡¼¥Ð¤Ç¤¹¡£
/etc/XF86Config	¥µ¡¼¥Ð¹½À®¾ðÊó¥Õ¥¡¥¤¥ë¤Ç¤¹¡£
<XRoot>/lib/X11/XF86Config.*hostname*	¥µ¡¼¥Ð¹½À®¾ðÊó¥Õ¥¡¥¤¥ë¤Ç¤¹¡£
<XRoot>/lib/X11/XF86Config	¥µ¡¼¥Ð¹½À®¾ðÊó¥Õ¥¡¥¤¥ë¤Ç¤¹¡£
<XRoot>/bin/*	¥¯¥é¥¤¥¢¥ó¥ÈÍÑ¥¢¥×¥ê¥±¡¼¥·¥ç¥ó¤Ç¤¹¡£
<XRoot>/include/*	¥Ø¥Ã¥À¡¼¥Õ¥¡¥¤¥ë¤Ç¤¹¡£
<XRoot>/lib/*	¥é¥¤¥Ö¥é¥ê¤Ç¤¹¡£
<XRoot>/lib/X11/fonts/*	¥Õ¥©¥ó¥È¤Ç¤¹¡£
<XRoot>/lib/X11/rgb.txt	¥«¥é¡¼Ì¾¾Î¤Î RGB ¤Ø¤Î³ä¤êÅö¤Æ¤Ç¤¹¡£
<XRoot>/lib/X11/XErrorDB	¥¯¥é¥¤¥¢¥ó¥È¤Î¥¨¥é¡¼¥á¥Ã¥»¡¼¥¸¥Ç¡¼¥¿¥Ù¡¼¥¹¤Ç¤¹¡£
<XRoot>/lib/X11/app-defaults/*	¥¯¥é¥¤¥¢¥ó¥È¤Î¥ê¥½¡¼¥¹¤Î»ØÄê¤Ç¤¹¡£
<XRoot>/man/man?/*	¥ª¥ó¥é¥¤¥ó¥Þ¥Ë¥å¥¢¥ë¤Ç¤¹¡£
/etc/X*n*.hosts	¥Ç£¥¹¥×¥ì¥¤ÈÖ¹æ *n* ¤ÈÂÐ¤¹¤ë¥¢¥¯¥»¥¹À©¸æ¤Î½é´ü¾õÂÖ¤Ç¤¹¡£ Ãí°Õ: <XRoot> ¤Ï X11 ¤ÎÆ³Æþ¥Ç£¥ì¥¯¥È¥ê¤Î¥ë¡¼¥È¤ò»²¾È¤·¤Æ¤¯¤À¤µ¤¤¡£

´ØÏ¢¹àÌÜ

X(1), Xserver(1), xdm(1), xinit(1), XF86Config(4/5), xf86config(1), XF86_SVGA(1), XF86_VGA16(1), XF86_Mono(1), XF86_Accel(1), xvidtune(1)

Ãø¼Ô

X11R5 ÂÐ±þ¤Î *XF86 1.2* ¤Ç¤Ê¤µ¤ìÄó¶¡¤Ï:

Thomas Roell, *roell@informatik.tu-muenchen.de*
TU-Muenchen: ¥µ¡¼¥Ð¤È SVR4 ¥¹¥¿¥Ã¥Õ

Mark W. Snitily, *mark@sgcs.com*
SGCS: SVR3 ¥µ¥Ý¡¼¥È, X Consortium ¥¹¥Ý¥ó¥µ¡¼

... ¤½¤·¤Æ¹½ÁÛ¤È½¤Àµ¤ò½õ¤±¤Æ¤¯¤ì¤¿¥¤¥ó¥¿¡¼¥Í¥Ã¥È¾å¤ÎÂ¿¤¯¤Î³§ÍÍ¡£

°Ê²¼¤Î¥Á¡¼¥à¤Ç XFree86 ¤ò X11R6 ¤ËÅý¹ç¤·¤Þ¤·¤¿:

Stuart Anderson *anderson@metrolink.com*
Doug Anson *danson@lgc.com*
Gertjan Akkerman *akkerman@dutiba.twi.tudelft.nl*
Mike Bernson *mike@mbsun.mlb.org*
Robin Cutshaw *robin@XFree86.org*
David Dawes *dawes@XFree86.org*
Marc Evans *marc@XFree86.org*
Pascal Haible *haible@izfm.uni-stuttgart.de*
Matthieu Herrb *Matthieu.Herrb@laas.fr*
Dirk Hohndel *hohndel@XFree86.org*
David Holland *davidh@use.com*
Alan Hourihane *alanh@fairlite.demon.co.uk*
Jeffrey Hsu *hsu@soda.berkeley.edu*
Glenn Lai *glenn@cs.utexas.edu*
Ted Lemon *mellon@ncd.com*
Rich Murphey *rich@XFree86.org*
Hans Nasten *nasten@everyware.se*
Mark Snitily *mark@sgcs.com*
Randy Terbush *randyt@cse.unl.edu*
Jon Tombs *tombs@XFree86.org*
Kees Verstoep *versto@cs.vu.nl*
Paul Vixie *paul@vix.com*
Mark Weaver *Mark_Weaver@brown.edu*
David Wexelblat *dwex@XFree86.org*
Philip Wheatley *Philip.Wheatley@ColumbiaSC.NCR.COM*
Thomas Wolfram *wolf@prz.tu-berlin.de*
Orest Zborowski *orestz@eskimo.com*

XFree86 µ¡Ç½³ÈÄ¥¥Ñ¥Ã¥±¡¼¥¸¤ÎÄó¶¡¤Ï°Ê²¼¤Î¤È¤ª¤ê:

David Dawes, *dawes@XFree86.org*
¸ø³«ºî¶È¤Î¼è¤êÅ»¤á¡¢FTP ¥µ¡¼¥Ð¤È¥á¡¼¥ê¥ó¥°¥ê¥¹¥È¤Î´ÉÍý¡£ ¥½¡¼¥¹¥Õ¥¡¥¤¥ë´ÉÍý¤ÈÅý¹ç¡¢¥¢¥¯¥»¥é¥ì¡¼¥¿¥µ¡¼¥Ð¤ÎÅý¹ç¡¢½¤Àµ¤È ¥×¥í¥°¥é¥ß¥ó¥°¡£

Glenn Lai, *glenn@cs.utexas.edu*
ET4000 ¤ò´∂¤Ë¤·¤¿ SVGA ¥«¡¼¥ÉÂÐ±þ¤Î¥×¥í¥°¥é¥à¤ÎÀÇ½¸þ¾å¤È ET4000/W32 ¤Î ¥¢¥¯¥»¥é¥ì¡¼¥¿¥µ¡¼¥Ð¡£

Jim Tsillas, *jtsilla@ccs.neu.edu*
°ìÏ¢¤Î fX386 ¤Î²þÁ±¤Ë¤è¤ëÂ¿¤¯¤Î¥µ¡¼¥Ð¤ÎÀÇ½¸þ¾å¡£

David Wexelblat, *dwex@XFree86.org*
É¸½à¥µ¡¼¥Ð¤Ø¤Î fX386 ¤ÎÅý¹ç¡¢Â¿¤¯¤Î¥É¥é¥¤¥Ð¤Î½¤Àµ¤È¥É¥é¥¤¥Ð¤Î»ñÎÁºîÀ®¡¢ VGA ¥«¡¼¥É¤È¥â¥Ë¥¿¡¡¼¤Î¥Ç¡¼¥¿¥Ù¡¼¥¹¤Î¼ý½¸¡¢ÈÆÍÑ¥Ó¥Ç¥ª¥â¡¼¥É°ìÍ÷¤ÎºîÀ®¡£ ¥¢¥¯¥»¥é¥ì¡¼¥¿¥µ¡¼¥Ð¤ÎÅý¹ç¡¢½¤Àµ¤È¥×¥í¥°¥é¥ß¥ó¥°¡£

Dirk Hohndel, *hohndel@XFree86.org*
Linux ¶¦Í¥é¥¤¥Ö¥é¥ê¤È¸ø³«º¡¶È¤Î¼è¤êÅ»¤á¡£¥¢¥¯¥»¥é¥ì¡¼¥¿¥µ¡¼¥Ð¤ÎÅý¹ç¤È½¤Àµ¡£ °ìÈÌ´ÉÍý¶ÈÌ³¤È»ñÎÁºîÀ®¡£

Amancio Hasty Jr., *hasty@netcom.com*
386BSD version 0.1 ¤Ø¤Î°Ü¿¢¤È XS3 ¤Î³«È¯¡£

Rich Murphey, *rich@XFree86.org*
Pace Willison »á ¤¬»Ï¤á¤¿°Ü¿¢¤ÈÇ¤ò´∂¤Ë¤·¤¿ **386BSD** version 0.1 ¤Ø¤Î°Ü¿¢¡£ **386BSD**, **FreeBSD** ¤È **NetBSD** ¤ÎÊÝ¼é¡£

Robert Baron, *Robert.Baron@ernst.mach.cs.cmu.edu*
Mach ¤Ø¤Î°Ü¿¢¡£

Orest Zborowski, *orestz@eskimo.com*
Linux ¤Ø¤Î°Ü¿¢¡£

Doug Anson, *danson@lgc.com*
Solaris x86 ¤Ø¤Î°Ü¿¢¡£

David Holland, *davidh@use.com*
Solaris x86 ¤Ø¤Î°Ü¿¢¡£

David McCullough, *davidm@stallion.oz.au*
SCO SVR3 ¤Ø¤Î°Ü¿¢¡£

Michael Rohleder, *michael.rohleder@stadt-frankfurt.de*
ISC SVR3 ¤Ø¤Î°Ü¿¢¡£

Kees Verstoep, *versto@cs.vu.nl*
Leendert van Doorn ¤¬»Ï¤á¤¿ X11R5 ¤Î Amoeba ¤Ø¤Î°Ü¿¢¤ÈÇ¤ò´∂¤Ë¤·¤¿ **Amoeba** ¤Ø¤Î°Ü¿¢¡£

Marc Evans, *Marc@XFree86.org*
Port. **OSF/1** ¤Ø¤Î°Ü¿¢¡£

Philip Homburg, *philip@cs.vu.nl*
Minix-386vm ¤Ø¤Î°Ü¿¢¡£

Thomas Mueller, *tm@systrix.de*
LynxOS ¤Ø¤Î°Ü¿¢¡£

Jon Tombs, *tombs@XFree86.org*
S3 ¥µ¡¼¥Ð¤È¥¢¥¯¥»¥é¥ì¡¼¥¿¥µ¡¼¥Ð¤Î¼è¤êÅ»¤á¡£

Harald Koenig, *koenig@tat.physik.uni-tuebingen.de*
S3 ¥µ¡¼¥Ð¤Î³«È¯¡£

Bernhard Bender, *br@elsa.mhs.compuserve.com*
S3 ¥µ¡¼¥Ð¤Î³«È¯¡£

Kevin Martin, *martin@cs.unc.edu*
´ðÁÃ¤Ë¤Ê¤Ã¤¿¥¢¥¯¥»¥é¥ì¡¼¥¿¥µ¡¼¥Ð·² (ATI ¤È 8514/A) ¤ÎÁ´°î¶È¤È Mach64 ¥µ¡¼¥Ð¡£

Rik Faith, *faith@cs.unc.edu*
´ðÁÃ¤Ë¤Ê¤Ã¤¿¥¢¥¯¥»¥é¥ì¡¼¥¿¥µ¡¼¥Ð·² (ATI ¤È 8514/A) ¤ÎÁ´°î¶È¡£

Tiago Gons, *tiago@comosjn.hobby.nl*
Mach8 ¤È 8514/A ¥µ¡¼¥Ð¤Î³«È¯¡£

Hans Nasten, *nasten@everyware.se*
Mach8, 8514/A ¤È S3 ¥µ¡¼¥Ð¤Î³«È¯¤È BSD/386 ÈÇ¤ÎÊÝ¼é¡£

Mike Bernson, *mike@mbsun.mlb.org*
Mach32 ¥µ¡¼¥Ð¤Î³«È¯¡£

Mark Weaver, *Mark_Weaver@brown.edu*
Mach32 ¥µ¡¼¥Ð¤Î³«È¯¡£

Craig Groeschel, *craig@metrolink.com*
Mach32 ¥µ¡¼¥Ð¤Î³«È¯¡£

Henry Worth, *Henry.Worth@amail.amdahl.com*
AGX ¥µ¡¼¥Ð¡£

Erik Nygren, *nygren@mit.edu*
P9000 ¥µ¡¼¥Ð¡£

Harry Langenbacher *harry@brain.jpl.nasa.gov*
P9000 ¥µ¡¼¥Ð¡£

Chris Mason, *mason@mail.csh.rit.edu*
P9000 ¥µ¡¼¥Ð¡£

Henrik Harmsen *harmsen@eritel.se*
P9000 ¥µ¡¼¥Ð¡£

Simon Cooper, *scooper@vizlab.rutgers.edu*
Cirrus ¥¢¥¯¥»¥é¥ì¡¼¥¿¤Î¥×¥í¥°¥é¥ß¥ó¥°(Bill Reynolds ¤ÎÀ®²Ì¤ò´ð¤Ë¤·¤Æ¤¤¤ë¤Þ¤¹)¡£

Harm Hanemaayer, *hhanemaa@cs.ruu.nl*
Cirrus ¥¢¥¯¥»¥é¥ì¡¼¥¿¤Î¥×¥í¥°¥é¥ß¥ó¥°¤È ARK ¥É¥é¥¤¥Ð¡£

Thomas Zerucha, *zerucha@shell.portal.com*
Cirrus CL-GD7543 ¤ÎÊÝ¼é¡£

Leon Bottou, *bottou@laforia.ibp.fr*
ARK ¥É¥é¥¤¥Ð¡£

Mike Tierney, *floyd@eng.umd.edu*
WD ¥¢¥¯¥»¥é¥ì¡¼¥¿¤Î¥×¥í¥°¥é¥ß¥ó¥°¡£

Bill Conn, *conn@bnr.ca*
WD ¥¢¥¯¥»¥é¥ì¡¼¥¿¤Î¥×¥í¥°¥é¥ß¥ó¥°¡£

Brad Bosch, *brad@lachman.com*
WD 90C24A ¤ÎÊÝ¼é¡£

Alan Hourihane, *alanh@fairlite.demon.co.uk*
Trident SVGA ¥É¥é¥¤¥Ð¡¢SiS SVGA ¥É¥é¥¤¥Ð¤È DEC 21030 ¥µ¡¼¥Ð¡£

Marc La France, *Marc.La-France@ualberta.ca*
ATI vgawonder SVGA ¥É¥é¥¤¥Ð¡£

Steve Goldman, *sgoldman@encore.com*
Oak 067/077 SVGA ¥É¥é¥¤¥Ð¡£

Jorge Delgado, *ernar@dit.upm.es*
Oak SVGA ¥É¥é¥¤¥Ð¤È 087 ¥¢¥¯¥»¥é¥ì¡¼¥¿¤Î¥×¥í¥°¥é¥ß¥ó¥°¡£

Bill Conn, *conn@bnr.ca*
WD ¥¢¥¯¥»¥é¥ì¡¼¥¿¤Î¥×¥í¥°¥é¥ß¥ó¥°¡£

Paolo Severini, *lendl@dist.dist.unige.it*
AL2101 SVGA ¥É¥é¥¤¥Ð¡£

Ching-Tai Chiu, *cchiu@netcom.com*
Avance Logic ALI SVGA ¥É¥é¥¤¥Ð¡£

Manfred Brands, *mb@oceonics.nl*
Cirrus 64xx SVGA ¥É¥é¥¤¥Ð¡£

Randy Hendry, *randy@sgi.com*
cl64xx SVGA ¥É¥é¥¤¥ÐÆâ¤Î Cirrus 6440 ¤ÎÊÝ¼é¡£

Frank Dikker, *dikker@cs.utwente.nl*
MX SVGA ¥É¥é¥¤¥Ð¡£

Regis Cridlig, *cridlig@dmi.ens.fr*
Chips & Technology ¥É¥é¥¤¥Ð¡£

Jon Block, *block@frc.com*
Chips & Technology ¥É¥é¥¤¥Ð¡£

Mike Hollick, *hollick@graphics.cis.upenn.edu*
Chips & Technology ¥É¥é¥¤¥Ð¡£

Nozomi Ytow
Chips & Technology ¥É¥é¥¤¥Ð¡£

Egbert Eich, *Egbert.Eich@Physik.TH-Darmstadt.DE*
Chips & Technology ¥É¥é¥¤¥Ð¡£

David Bateman, *dbateman@ee.uts.edu.au*
Chips & Technology ¥É¥é¥¤¥Ð¡£

Xavier Ducoin, *xavier@rd.lectra.fr*
Chips & Technology ¥É¥é¥¤¥Ð¡£

Peter Trattler, *peter@sbox.tu-graz.ac.at*
RealTek SVGA ¥É¥é¥¤¥Ð¡£

Craig Struble, *cstruble@acm.vt.edu*
Video7 SVGA ¥É¥é¥¤¥Ð¡£

Gertjan Akkerman, *akkerman@dutiba.twi.tudelft.nl*
16 ¿§ VGA ¥µ¡¼¥Ð¤È XF86Config ¹½Ê¸²òÀÏ¥×¥í¥°¥é¥à¡£

Davor Matic, *dmatic@Athena.MIT.EDU*
Hercules ¥É¥é¥¤¥Ð¡£

Pascal Haible, *haible@izfm.uni-stuttgart.de*
¥Ð¥ó¥¯ÀÚ¤êÂØ¤¨ÉÕ¤-Çò¹õ VGA ¤ÎÊÝ¼é¡¢Hercules ¤ÎÊÝ¼é¡¢ÉáÄÌ¤ÎÇò¹õ¥Ç¥Ð¥¤¥¹ÍÑ¤Î Çò¹õ¥Õ¥ì¡¼¥à¥Ð¥Ã¥Õ¥¡¤ÎÊÝ¼é¡£

...
¤½¤·¤Æ¤³¤Îµ¡Ç½³ÈÄ¥¤Î¥Ù¡¼¥¿¥Æ¥¹¥È¤Ç½õ¤±¤Æ¤¤¤¿¤À¤¤¤¿¥¤¥ó¥¿¡¼¥Í¥Ã¥È¾å¤Î ¤è¤êÂ¿¤¯¤Î³§ÍÍ¡£

XFree86 ¤Î¥½¡¼¥¹¤Ï *ftp.XFree86.org* ¤Î FTP ¥µ¡¼¥Ð¡¢¤½¤ÎÂ¾¤Ë ¤¢¤ê¤Þ¤¹¡£¾Ü°Ù¤Ë¤Ä¤¤¤Æ¤Ï *XFree86@XFree86.org* ¤ØÅÅ»Ò¥á¡¼¥ë¤ò²¼¤µ¤¤¡£

NAME

XF86DGAQueryExtension, XF86DGAQueryVersion, XF86DGAQueryDirectVideo, XF86DGAGetVideo, XF86DGADirectVideo, XF86DGASetVidPage, XF86DGASetViewPort, XF86DGAViewPortChanged, XF86DGAGetViewPortSize, XF86DGAInstallColormap, XF86DGAForkApp – XFree86-DGA extension interface functions

SYNTAX

```
#include <X11/extensions/xf86dga.h>
Bool XF86DGAQueryExtension(
   Display *display,
   int *event_base_return,
   int *error_base_return);
Bool XF86DGAQueryVersion(
   Display *display,
   int *major_version_return,
   int *minor_version_return);
Bool XF86DGAQueryDirectVideo(
   Display *display,
   int screen,
   int *flags_return);
Bool XF86DGAGetVideo(
   Display *display,
   int screen,
   char **addr_return,
   int *width_return,
   int *banksize_return,
   int *memsize_return);
Bool XF86DGADirectVideo(
   Display *display,
   int screen,
   int flags);
Bool XF86DGASetVidPage(
   Display *display,
   int screen,
   int page);
Bool XF86DGASetViewPort(
   Display *display,
   int screen,
   int x,
   int y);
Bool XF86DGAViewPortChanged(
   Display *display,
   int screen,
   int num_pages);
Bool XF86DGAGetViewPortSize(
   Display *display,
   int screen,
   int *viewport_width_return,
   int *viewport_height_return);
```

```
Bool XF86DGAInstallColormap(
  Display *display,
  int screen,
  Colormap cmap);
int XF86DGAForkApp(
  int screen);
```

ARGUMENTS

display	Specifies the connection to the X server.
screen	Specifies which screen number the setting apply to.
event_base_return	Returns the base event number for the extension.
error_base_return	Returns the base error number for the extension.
major_version_return	Returns the major version number of the extension.
minor_version_return	Returns the minor version number of the extension.
addr_return	Returns a pointer to the start of the video framebuffer.
width_return	Returns the framebuffer line width.
banksize_return	Returns the framebuffer bank size.
memsize_return	Returns the size of the framebuffer memory.
flags	Sets the DirectVideo access features. When zero, DirectVideo mode is disabled. *flags* may be a a bit-wise combination of the following values: XF86DGADirectGraphics enable Direct Video mode XF86DGADirectMouse enable reporting of pointer movement as relative motion XF86DGADirectKeyb enable direct reporting of keyboard events
flags_return	Reports the DirectVideo features supported by the hardware. When zero, the hardware does not support direct video at all. *flags* may be a a bit-wise combination of the following values: XF86DGADirectPresent DirectVideo support is present
page	Indicates the framebuffer page (bank) to activate for read/write access.
x	Incidates the x coordinate for the upper-left corner of the view port.
y	Incidates the y coordinate for the upper-left corner of the view port.
viewport_width_return	Reports the width of the view port.
viewport_width_return	Reports the height of the view port.
num_pages	Indicates the number of pages when doing hardware multi-buffering.
cmap	Indicates the colormap to install.

DESCRIPTION

These functions provide an interface to the server extension *XFree86-DGA* which allows a local client direct access to the video framebuffer. Applications that use these functions must be linked with **-lXxf86dga**

DGA FUNCTIONS

The function **XF86DGAQueryDirectVideo** returns the DirectVideo capabilities supported by the graphics device.

The **XF86DGAGetVideo** function is used to get the parameters for the framebuffer. The parameters returned are a pointer to the start of the framebuffer, the framebuffer line width, framebuffer bank size, and framebuffer memory size.

XF86DGADirectVideo is used to enable or disable DirectVideo mode.

The **XF86DGASetVidPage** function sets the currently active framebuffer page (bank). This is only required for hardware which has a banked memory layout (banksize < memsize).

The **XF86DGASetViewPort** function sets the framebuffer coordinates to use for the upper-left corner of the view port.

The **XF86DGAViewPortChanged** function checks whether a previous XF86DGASetViewPort command has been completed by the hardware, that is, whether a vertical retrace has occurred since a previous XF86DGASetViewPort. This can (must, in fact) be used with page-flipping; you can start writing to the next page only when this function returns TRUE. For some devices this will be the case immediately after XF86DGASetViewPort, however this may be changed in the future. The number of pages used is specified with *num_pages*; it should be 2 for simple page flipping (double buffering). If n is greater than two (triple or multi-buffering), the function checks whether the (n − 2)-before-last SetViewPort has been completed.

The **XF86DGAGetViewPortSize** function returns the size of the view port, which is the part of the framebuffer that is visible on the screen.

The **XF86DGAInstallColormap** function is used to install a colormap. This must be called after DirectVideo mode has been enabled.

The **XF86DGAForkApp** function causes the client application to fork, leaving the parent process to hang around and return to non-DGA mode should the client exit for any reason. This function returns 0 for success, or the error condition returned by fork().

OTHER FUNCTIONS

The **XF86DGAQueryVersion** function can be used to determine the version of the extension built into the server.

The function **XF86DGAQueryExtension** returns the lowest numbered error and event values assigned to the extension.

SEE ALSO

XFree86(1), XF86Config(4/5)

AUTHORS

Jon Tombs, Harm Hanemaayer, Mark Vojkovich.

NAME

XF86MiscQueryExtension, XF86MiscQueryVersion, XF86MiscGetMouseSettings, XF86MiscSetMouseSettings, XF86MiscGetKbdSettings, XF86MiscSetKbdSettings – XFree86-Misc extension interface functions

SYNTAX

```
#include <X11/extensions/xf86misc.h>
Bool XF86MiscQueryExtension(
   Display *display,
   int *event_base_return,
   int *error_base_return);
Bool XF86MiscQueryVersion(
   Display *display,
   int *major_version_return,
   int *minor_version_return);
Status XF86MiscGetMouseSettings(
   Display *display,
   XF86MiscMouseSettings *mseinfo);
Status XF86MiscSetMouseSettings(
   Display *display,
   XF86MiscMouseSettings *mseinfo);
Status XF86MiscGetKbdSettings(
   Display *display,
   XF86MiscKbdSettings *kbdinfo);
Status XF86MiscSetKbdSettings(
   Display *display,
   XF86MiscKbdSettings *kbdinfo);
```

ARGUMENTS

display Specifies the connection to the X server.

screen Specifies which screen number the setting apply to.

event_base_return Returns the base event number for the extension.

error_base_return Returns the base error number for the extension.

major_version_return Returns the major version number of the extension.

minor_version_return Returns the minor version number of the extension.

mseinfo Specifies a structure which contains the mouse parameters.

kbdinfo Specifies a structure which contains the keyboard parameters.

STRUCTURES

Mouse:

```
typedef struct {
   char *device;              /* returned path to device */
   int type;                  /* mouse protocol */
   int baudrate;              /* 1200, 2400, 4800, or 9600 */
   int samplerate;            /* samples per second */
   Bool emulate3buttons;      /* Button1+Button3 -> Button2 ? */
   int emulate3timeout;       /* in milliseconds */
   Bool chordmiddle;          /* Button1+Button3 == Button2 ? */
```

```
   int flags;                          /* Device open flags */
} XF86MiscMouseSettings;
Keyboard:
typedef struct {
   int type;                           /* of keyboard: 84-key, 101-key, Xqueue  */
   int rate;                           /* repeat rate */
   int delay;                          /* delay until repeat starts */
   Bool servnumlock;                   /* Server handles NumLock ? */
} XF86MiscKbdSettings;
```

DESCRIPTION

These functions provide an interface to the *XFree86-Misc* extension which allows various server settings to be queried and changed dynamically. Applications that use these functions must be linked with **-lXxf86misc**

POWER-SAVER FUNCTIONS

The **XF86MiscGetSaver** and **XF86MiscSetSaver** functions have been removed. This functionality is now provided by the DPMS extension.

MOUSE FUNCTIONS

Mouse parameters can be queried using the function **XF86MiscGetMouseSettings**. The structure pointed to by its second argument is filled in with the current mouse settings.

Not all fields are valid in all cases. For example, when the protocol indicates a bus mouse (i.e. the type field has value **MTYPE_BUSMOUSE** as defined in **xf86misc.h**), then the value in the **baudrate** field should be ignored as it does not apply to bus mice.

The **samplerate** field contains the resolution in lines per inch when using the Hitachi tablet protocol.

The device field of the structure points to dynamically allocated storage which should be freed by the caller.

Any of the fields of the structure can be altered and then passed to the **XF86MiscSetMouseSettings** function to change their value in the server, with the following restrictions:

1) The device can not be changed

2) The protocol can not be changed to or from Xqueue or OsMouse

3) Invalid combinations of parameters are not allowed

The server will generate an error if any of the above is attempted, except the first – the contents of the device field are simply ignored.

A change of the protocol causes the device to be closed and reopened. Changes to the baud rate, sample rate, or flags, when applicable to the selected protocol, also cause a reopen of the device. A reopen can be forced by using the MF_REOPEN flag, except in the case of the OsMouse and Xqueue protocols which ignore all attempts to reopen the device.

KEYBOARD FUNCTIONS

The **XF86MiscGetKbdSettings** function allows you to retrieve the current keyboard-related settings from the server.

Using the **XF86MiscSetKbdSettings** function, the keyboard autorepeat delay and rate can be set. Requests to change the **type** and **servnumlock** fields are ignored (except for checking for an invalid keyboard type). This is expected to change in a future release.

OTHER FUNCTIONS

Two functions, **XF86MiscQueryExtension** and **XF86MiscQueryVersion**, are provided which allow the client to query some information regarding the extension itself.

PREDEFINED VALUES

The header file **X11/extensions/xf86misc.h** contains definitions for

MTYPE_*	Mouse protocols
KTYPE_*	Keyboard types
MF_*	Mouse flags

SEE ALSO

xset(1)

AUTHORS

Joe Moss and David Dawes, The XFree86 Project, Inc.

NAME

XF86VidModeQueryExtension, XF86VidModeQueryVersion, XF86VidModeGetModeLine, XF86VidModeGetAllModeLines, XF86VidModeDeleteModeLine, XF86VidModeModModeLine, XF86VidModeValidateModeLine, XF86VidModeSwitchMode, XF86VidModeSwitchToMode, XF86VidModeLockModeSwitch, XF86VidModeGetMonitor, XF86VidModeGetViewPort, XF86VidModeSetViewPort − XFree86-VidMode extension interface functions

SYNTAX

```
#include <X11/extensions/xf86vmode.h>
Bool XF86VidModeQueryExtension(
   Display *display,
   int *event_base_return,
   int *error_base_return);
Bool XF86VidModeQueryVersion(
   Display *display,
   int *major_version_return,
   int *minor_version_return);
Bool XF86VidModeGetModeLine(
   Display *display,
   int screen,
   int *dotclock_return,
   XF86VidModeModeLine *modeline);
Bool XF86VidModeGetAllModeLines(
   Display *display,
   int screen,
   int *modecount_return,
   XF86VidModeModeInfo **modesinfo);
Bool XF86VidModeDeleteModeLine(
   Display *display,
   int screen,
   XF86VidModeModeInfo *modeline);
Bool XF86VidModeModModeLine(
   Display *display,
   int screen,
   XF86VidModeModeLine *modeline);
Status XF86VidModeValidateModeLine(
   Display *display,
   int screen,
   XF86VidModeModeLine *modeline);
Bool XF86VidModeSwitchMode(
   Display *display,
   int screen,
   int zoom);
Bool XF86VidModeSwitchToMode(
   Display *display,
   int screen,
   XF86VidModeModeInfo *modeline);
Bool XF86VidModeLockModeSwitch(
   Display *display,
```

```
      int screen,
      int lock);
   Bool XF86VidModeGetMonitor(
      Display *display,
      int screen,
      XF86VidModeMonitor *monitor);
   Bool XF86VidModeGetViewPort(
      Display *display,
      int screen,
      int *x_return,
      int *y_return);
   Bool XF86VidModeSetViewPort(
      Display *display,
      int screen,
      int x,
      int y);
```

ARGUMENTS

display	Specifies the connection to the X server.
screen	Specifies which screen number the setting apply to.
event_base_return	Returns the base event number for the extension.
error_base_return	Returns the base error number for the extension.
major_version_return	Returns the major version number of the extension.
minor_version_return	Returns the minor version number of the extension.
dotclock_return	Returns the clock for the mode line.
modecount_return	Returns the number of video modes available in the server.
zoom	If greater than zero, indicates that the server should switch to the next mode, otherwise switch to the previous mode.
lock	Indicates that mode switching should be locked, if non-zero.
modeline	Specifies or returns the timing values for a video mode.
modesinfo	Returns the timing values and dotclocks for all of the available video modes.
monitor	Returns information about the monitor.
x	Specifies the desired X location for the viewport.
x_return	Returns the current X location of the viewport.
y	Specifies the desired Y location for the viewport.
y_return	Returns the current Y location of the viewport.

STRUCTURES

Video Mode Settings:

```
typedef struct {
   unsigned short      hdisplay;      /* Number of display pixels horizontally */
   unsigned short      hsyncstart;    /* Horizontal sync start */
   unsigned short      hsyncend;      /* Horizontal sync end */
```

```
   unsigned short          htotal;         /* Total horizontal pixels */
   unsigned short          vdisplay;       /* Number of display pixels vertically */
   unsigned short          vsyncstart;     /* Vertical sync start */
   unsigned short          vsyncend;       /* Vertical sync start */
   unsigned short          vtotal;         /* Total vertical pixels */
   unsigned int            flags;          /* Mode flags */
   int                     privsize;       /* Size of private */
   INT32                   *private;       /* Server privates */
} XF86VidModeModeLine;

typedef struct {
   unsigned int            dotclock;       /* Pixel clock */
   unsigned short          hdisplay;       /* Number of display pixels horizontally */
   unsigned short          hsyncstart;     /* Horizontal sync start */
   unsigned short          hsyncend;       /* Horizontal sync end */
   unsigned short          htotal;         /* Total horizontal pixels */
   unsigned short          vdisplay;       /* Number of display pixels vertically */
   unsigned short          vsyncstart;     /* Vertical sync start */
   unsigned short          vsyncend;       /* Vertical sync start */
   unsigned short          vtotal;         /* Total vertical pixels */
   unsigned int            flags;          /* Mode flags */
   int                     privsize;       /* Size of private */
   INT32                   *private;       /* Server privates */
} XF86VidModeModeInfo;
```

Monitor information:

```
typedef struct {
   char*                   vendor;         /* Name of manufacturer */
   char*                   model;          /* Model name */
   float                   bandwidth;      /* Monitor bandwidth */
   unsigned char           nhsync;         /* Number of horiz sync ranges */
   XF86VidModeSyncRange*   hsync;          /* Horizontal sync ranges */
   unsigned char           nvsync;         /* Number of vert sync ranges */
   XF86VidModeSyncRange*   vsync;          /* Vertical sync ranges */
} XF86VidModeMonitor;

typedef struct {
   float                   hi;             /* Top of range */
   float                   lo;             /* Bottom of range */
} XF86VidModeSyncRange;
```

DESCRIPTION

These functions provide an interface to the server extension *XFree86-VidModeExtension* which allows the video modes to be queried and adjusted dynamically and mode switching to be controlled. Applications that use these functions must be linked with **-lXxf86vm**

MODELINE FUNCTIONS

The **XF86VidModeGetModeLine** function is used to query the settings for the currently selected video mode. The calling program should pass a pointer to a **XF86VidModeModeLine** structure that it has already allocated. The function fills in the fields of the structure.

If there are any server private values (currently only applicable to the S3 server) the function will allocate storage for them. Therefore, if the **privsize** field is non-zero, the calling program should call **Xfree(private)** to free the storage.

XF86VidModeGetAllModeLines returns the settings for all video modes. The calling program supplies the address of a pointer which will be set by the function to point to an array of **XF86VidModeModeInfo** structures. The memory occupied by the array is dynamically allocated by the **XF86VidModeGetAllModeLines** function and should be freed by the caller. The first element of the array corresponds to the current video mode.

The **XF86VidModeModModeLine** function can be used to change the settings of the current video mode provided the requested settings are valid (e.g. they don't exceed the capabilities of the monitor).

Modes can be deleted with the **XF86VidModeDeleteModeLine** function. The specified mode must match an existing mode. To be considered a match, all of the fields of the given **XF86VidModeModeInfo** structure must match, except the **privsize** and **private** fields. If the mode to be deleted is the current mode, a mode switch to the next mode will occur first. The last remaining mode can not be deleted.

The validity of a mode can be checked with the **XF86VidModeValidateModeLine** function. If the specified mode can be used by the server (i.e. meets all the constraints placed upon a mode by the combination of the server, card, and monitor) the function returns **MODE_OK**, otherwise it returns a value indicating the reason why the mode is invalid (as defined in *xf86.h*)

MODE SWITCH FUNCTIONS

When the function **XF86VidModeSwitchMode** is called, the server will change the video mode to next (or previous) video mode. The **XF86VidModeSwitchToMode** function can be used to switch directly to the specified mode. Matching is as specified in the description of the **XF86VidModeDeleteModeLine** function above. The **XF86VidModeLockModeSwitch** function can be used to allow or disallow mode switching whether the request to switch modes comes from a call to the **XF86VidModeSwitchMode** or **XF86VidModeSwitchToMode** functions or from one of the mode switch key sequences.

MONITOR FUNCTIONS

Information known to the server about the monitor is returned by the **XF86VidModeGetMonitor** function. The **hsync** and **vsync** fields each point to an array of **XF86VidModeSyncRange** structures. The arrays contain **nhsync** and **nvsync** elements, respectively. The **hi** and **low** values will be equal if a discreate value was given in the **XF86Config** file.

The **vendor**, **model**, **hsync**, and **vsync** fields point to dynamically allocated storage that should be freed by the caller.

VIEWPORT FUNCTIONS

The **XF86VidModeGetViewPort** and **XF86VidModeSetViewPort** functions can be used to, respectively, query and change the location of the upper left corner of the viewport into the virtual screen.

OTHER FUNCTIONS

The **XF86VidModeQueryVersion** function can be used to determine the version of the extension built into the server.

The function **XF86VidModeQueryExtension** returns the lowest numbered error and event values assigned to the extension.

SEE ALSO

XFree86(1), XF86Config(4/5), xvidtune(1)

AUTHORS

Kaleb Keithley, Jon Tombs, David Dawes, and Joe Moss

NAME

kbd_mode – recover the PC console keyboard

SYNOPSIS

kbd_mode [-a -u]

DESCRIPTION

Kbd_mode resets the PC console keyboard to a rational state.

OPTIONS

The following options are supported:

–a Set the keyboard so that ASCII characters are read from the console.

–u Set the keyboard so that undecoded keyboard values are read from the console.

EXAMPLES

If the server crashes or otherwise fails to put the keyboard back in ascii mode when it exits, it can leave your keyboard dead. If you are able to login remotely, you can reset it typing:

```
kbd_mode -a
```

Conversely, changing the keyboard to ascii mode while the server is running will make the keyboard appear to be dead while the the mouse continues to work. Again, if you are able to login remotely, you can reset it typing:

```
kbd_mode -u
```

NAME

reconfig – convert old Xconfig to new XF86Config

SYNOPSIS

reconfig < *Xconfig* > *XF86Config*

DESCRIPTION

The *reconfig* program converts the Xconfig file format used in XFree86 versions prior to 3.1 into the XF86Config format currently used. The XF86Config format contains more information than the Xconfig format, so manual editing is required after converting.

SEE ALSO

XFree86(1), XF86Config(4/5), xf86config(1)

AUTHOR

Gertjan Akkerman.

BUGS

Comment lines are stripped out when converting.

NAME

xf86config − generate an XF86Config file

SYNOPSIS

xf86config

DESCRIPTION

xf86config is an interactive program for generating an XF86Config file for use with XFree86 X servers.

FILES

<xroot>/lib/X11/Cards Video cards database

SEE ALSO

XFree86(1), XF86Config(4/5), reconfig(1)

AUTHOR

Harm Hanemaayer.

NAME

Xnest – a nested X server

SYNOPSIS

Xnest [-options]

DESCRIPTION

Xnest is a client and a server. *Xnest* is a client of the real server which manages windows and graphics requests on its behalf. *Xnest* is a server to its own clients. *Xnest* manages windows and graphics requests on their behalf. To these clients *Xnest* appears to be a conventional server.

OPTIONS

Xnest supports all standard options of the sample server implementation. For more details, please see the manual page on your system for *Xserver*. The following additional arguments are supported as well.

–display *string*
This option specifies the display name of the real server that *Xnest* should try to connect with. If it is not provided on the command line *Xnest* will read the *DISPLAY* environment variable in order to find out the same information.

–sync
This option tells *Xnest* to synchronize its window and graphics operations with the real server. This is a useful option for debugging, but it will slow down the performance considerably. It should not be used unless absolutely necessary.

–full
This option tells *Xnest* to utilize full regeneration of real server objects and reopen a new connection to the real server each time the nested server regenerates. The sample server implementation regenerates all objects in the server when the last client of this server terminates. When this happens, *Xnest* by default maintains the same top level window and the same real server connection in each new generation. If the user selects full regeneration, even the top level window and the connection to the real server will be regenerated for each server generation.

–class *string*
This option specifies the default visual class of the nested server. It is similar to the *-cc* option from the set of standard options except that it will accept a string rather than a number for the visual class specification. The string must be one of the following six values: *StaticGray*, *GrayScale*, *StaticColor*, *PseudoColor*, *TrueColor*, or *DirectColor*. If both, *-class* and *-cc* options are specified, the last instance of either option assumes precedence. The class of the default visual of the nested server need not be the same as the class of the default visual of the real server; although, it has to be supported by the real server. See *xdpyinfo* for a list of supported visual classes on the real server before starting *Xnest*. If the user chooses a static class, all the colors in the default colormap will be preallocated. If the user chooses a dynamic class, colors in the default colormap will be available to individual clients for allocation.

–depth *int*
This option specifies the default visual depth of the nested server. The depth of the default visual of the nested server need not be the same as the depth of the default visual of the real server; although, it has to be supported by the real server. See *xdpyinfo* for a list of supported visual depths on the real server before starting *Xnest*.

–sss This option tells *Xnest* to use the software screen saver. By default *Xnest* will use the screen saver that corresponds to the hardware screen saver in the real server. Of course, even this screen saver is software generated since *Xnest* does not control any actual hardware. However, it is treated as a hardware screen saver within the sample server code.

–geometry *W+H+X+Y*
This option specifies geometry parameters for the top level *Xnest* windows. These windows corresponds to the root windows of the nested server. The width and height specified with this option will be the maximum width and height of each top level *Xnest* window. *Xnest* will allow the user to make any top level window smaller, but it will not actually change the size of the nested server root window. As of yet, there is no mechanism within the sample server implementation to change the size of the root window after screen initialization. In order to do so, one would probably need to extend the X protocol. Therefore, it is not likely that this will be available any time soon. If this option is not specified *Xnest* will choose width and height to be 3/4 of the dimensions of the root window of the real server.

–bw *int*
This option specifies the border width of the top level *Xnest* window. The integer parameter must be a positive number. The default border width is 1.

–name *string*
This option specifies the name of the top level *Xnest* window. The default value is the program name.

–scrns *int*
This option specifies the number of screens to create in the nested server. For each screen, *Xnest* will create a separate top level window. Each screen is referenced by the number after the dot in the client display name specification. For example, *xterm -display :1.1* will open an *xterm* client in the nested server with the display number *:1* on the second screen. The number of screens is limited by the hard coded constant in the server sample code which is usually 3.

–install
This option tells *Xnest* to do its own colormap installation by bypassing the real window manager. For it to work properly the user will probably have to temporarily quit the real window manager. By default *Xnest* will keep the nested client window whose colormap should be installed in the real server in the *WM_COLORMAP_WINDOWS* property of the top level *Xnest* window. If this colormap is of the same visual type as the root window of the nested server, *Xnest* will associate this colormap with the top level *Xnest* window as well. Since this does not have to be the case, window managers should look primarily at the *WM_COLORMAP_WINDOWS* property rather than the colormap associated with the top level *Xnest* window. Unfortunately, window managers are not very good at doing that yet so this option might come in handy.

–parent *window_id*
This option tells *Xnest* to use the *window_id* as the root window instead of creating a window. This option is used by the xrx xnestplugin.

USAGE

Starting up *Xnest* is as simple as starting up *xclock* from a terminal emulator. If a user wishes to run *Xnest* on the same workstation as the real server, it is important that the nested server is given its own listening socket address. Therefore, if there is a server already running on the user's workstation, *Xnest* will have to be started up with a new display number. Since there is usually no more than one server running on a workstation, specifying *Xnest :1* on the command line will be sufficient for most users. For each server running on the workstation the display number needs to be incremented by one. Thus, if you wish to start another *Xnest*, you will need to type *Xnest :2* on the command line.

To run clients in the nested server each client needs to be given the same display number as the nested server. For example, *xterm -display :1* will start up an *xterm* in the first nested server and *xterm -display :2* will start an *xterm* in the second nested server from the example above. Additional clients can be started from these *xterm*s in each nested server.

XNEST AS A CLIENT

Xnest behaves and looks to the real server and other real clients as another real client. It is a rather demanding client, however, since almost any window or graphics request from a nested client will result in a window or graphics request from *Xnest* to the real server. Therefore, it is desirable that *Xnest* and the real server are on a local network, or even better, on the same machine. As of now, *Xnest* assumes that the real server supports the shape extension. There is no way to turn off this assumption dynamically. *Xnest* can be compiled without the shape extension built in, and in that case the real server need not support it. The dynamic shape extension selection support should be considered in further development of *Xnest*.

Since *Xnest* need not use the same default visual as the the real server, the top level window of the *Xnest* client always has its own colormap. This implies that other windows' colors will not be displayed properly while the keyboard or pointer focus is in the *Xnest* window, unless the real server has support for more than one installed colormap at any time. The colormap associated with the top window of the *Xnest* client need not be the appropriate colormap that the nested server wants installed in the real server. In the case that a nested client attempts to install a colormap of a different visual from the default visual of the nested server, *Xnest* will put the top window of this nested client and all other top windows of the nested clients that use the same colormap into the *WM_COLORMAP_WINDOWS* property of the top level *Xnest* window on the real server. Thus, it is important that the real window manager that manages the *Xnest* top level window looks at the *WM_COLORMAP_WINDOWS* property rather than the colormap associated with the top level *Xnest* window. Since most window managers appear to not implement this convention properly as of yet, *Xnest* can optionally do direct installation of colormaps into the real server bypassing the real window manager. If the user chooses this option, it is usually necessary to temporarily disable the real window manager since it will interfere with the *Xnest* scheme of colormap installation.

Keyboard and pointer control procedures of the nested server change the keyboard and pointer control parameters of the real server. Therefore, after *Xnest* is started up, it will change the keyboard and pointer controls of the real server to its own internal defaults. Perhaps there should be a command line option to tell *Xnest* to inherit the keyboard and pointer control parameters from the real server rather than imposing its own. This is a future consideration.

XNEST AS A SERVER

Xnest as a server looks exactly like a real server to its own clients. For the clients there is no way of telling if they are running on a real or a nested server.

As already mentioned, *Xnest* is a very user friendly server when it comes to customization. *Xnest* will pick up a number of command line arguments that can configure its default visual class and depth, number of screens, etc. In the future, *Xnest* should read a customization input file to provide even greater freedom and simplicity in selecting the desired layout. Unfortunately, there is no support for backing store and save under as of yet, but this should also be considered in the future development of *Xnest*.

The only apparent intricacy from the users' perspective about using *Xnest* as a server is the selection of fonts. *Xnest* manages fonts by loading them locally and then passing the font name to the real server and asking it to load that font remotely. This approach avoids the overload of sending the glyph bits across the network for every text operation, although it is really a bug. The proper implementation of fonts should be moved into the *os* layer. The consequence of this approach is that the user will have to worry about two different font paths - a local one for the nested server and a remote one for the real server - since *Xnest* does not propagate its font path to the real server. The reason for this is because real and nested servers need not run on the same file system which makes the two font paths mutually incompatible. Thus, if there is a font in the local font path of the nested server, there is no guarantee that this font exists in the remote font path of the real server. *Xlsfonts* client, if run on the nested server will list fonts in the local font path and if run on the real server will list fonts in the remote font path. Before a font can be successfully opened by the nested server it has to exist in local and remote font paths. It is the users' responsibility to make sure that this is the case.

BUGS

Won't run well on servers supporting different visual depths. Still crashes randomly. Probably has some memory leaks.

AUTHOR

Davor Matic, MIT X Consortium

NAME

appres − list X application resource database

SYNOPSIS

appres [[class [instance]] [−1] [toolkitoptions]

DESCRIPTION

The *appres* program prints the resources seen by an application (or subhierarchy of an application) with the specified *class* and *instance* names. It can be used to determine which resources a particular program will load. For example,

```
% appres  XTerm
```

will list the resources that any *xterm* program will load. If no application class is specified, the class *-AppResTest-* is used.

To match a particular instance name, specify an instance name explicitly after the class name, or use the normal Xt toolkit option. For example,

```
% appres  XTerm  myxterm
```

or

```
% appres  XTerm  −name  myxterm
```

To list resources that match a subhierarchy of an application, specify hierarchical class and instance names. The number of class and instance components must be equal, and the instance name should not be specified with a toolkit option. For example,

```
% appres  Xman.TopLevelShell.Form  xman.topBox.form
```

will list the resources of widgets of *xman* topBox hierarchy. To list just the resources matching a specific level in the hierarcy, use the −1 option. For example,

```
% appres  XTerm.VT100  xterm.vt100  −1
```

will list the resources matching the *xterm* vt100 widget.

SEE ALSO

X(1), xrdb(1), listres(1)

AUTHOR

Jim Fulton, MIT X Consortium

NAME

bdftopcf − convert X font from Bitmap Distribution Format to Portable Compiled Format

SYNOPSIS

bdftopcf [**−p***n*] [**−u***n*] [**−m**] [**−l**] [**−M**] [**−L**] [**−t**] [**−i**] [**−o** *outputfile*] fontfile.bdf

DESCRIPTION

Bdftopcf is a font compiler for the X server and font server. Fonts in Portable Compiled Format can be read by any architecture, although the file is structured to allow one particular architecture to read them directly without reformatting. This allows fast reading on the appropriate machine, but the files are still portable (but read more slowly) on other machines.

OPTIONS

−p*n*
Sets the font glyph padding. Each glyph in the font will have each scanline padded in to a multiple of *n* bytes, where *n* is 1, 2, 4 or 8.

−u*n*
Sets the font scanline unit. When the font bit order is different from the font byte order, the scanline unit *n* describes what unit of data (in bytes) are to be swapped; the unit *i* can be 1, 2 or 4 bytes.

−m
Sets the font bit order to MSB (most significant bit) first. Bits for each glyph will be placed in this order; i.e., the left most bit on the screen will be in the highest valued bit in each unit.

−l
Sets the font bit order to LSB (least significant bit) first. The left most bit on the screen will be in the lowest valued bit in each unit.

−M
Sets the font byte order to MSB first. All multi-byte data in the file (metrics, bitmaps and everything else) will be written most significant byte first.

−L
Sets the font byte order to LSB first. All multi-byte data in the file (metrics, bitmaps and everything else) will be written least significant byte first.

−t
When this option is specified, *bdftopcf* will convert fonts into "terminal" fonts when possible. A terminal font has each glyph image padded to the same size; the X server can usually render these types of fonts more quickly.

−i
This option inhibits the normal computation of ink metrics. When a font has glyph images which do not fill the bitmap image (i.e., the "on" pixels don't extend to the edges of the metrics) *bdftopcf* computes the actual ink metrics and places them in the .pcf file; the −t option inhibits this behaviour.

−o *output-file-name*
By default *bdftopcf* writes the pcf file to standard output; this option gives the name of a file to be used instead.

SEE ALSO

X(1)

AUTHOR

Keith Packard, MIT X Consortium

NAME

bitmap, bmtoa, atobm − bitmap editor and converter utilities for the X Window System

SYNOPSIS

bitmap [−*options* ...] [*filename*] [*basename*]

bmtoa [**−chars** ...] [*filename*]

atobm [**−chars** *cc*] [**−name** *variable*] [**−xhot** *number*] [**−yhot** *number*] [*filename*]

DESCRIPTION

The *bitmap* program is a rudimentary tool for creating or editing rectangular images made up of 1's and 0's. Bitmaps are used in X for defining clipping regions, cursor shapes, icon shapes, and tile and stipple patterns.

The *bmtoa* and *atobm* filters convert *bitmap* files (FILE FORMAT) to and from ASCII strings. They are most commonly used to quickly print out bitmaps and to generate versions for including in text.

COMMAND LINE OPTIONS

Bitmap supports the standard X Toolkit command line arguments (see *X*(1)). The following additional arguments are supported as well.

−size *WIDTHxHEIGHT*
Specifies size of the grid in squares.

−sw *dimension*
Specifies the width of squares in pixels.

−sh *dimension*
Specifies the height of squares in pixels.

−gt *dimension*
Grid tolerance. If the square dimensions fall below the specified value, grid will be automatically turned off.

−grid, +grid
Turns on or off the grid lines.

−axes, +axes
Turns on or off the major axes.

−dashed, +dashed
Turns on or off dashing for the frame and grid lines.

−stippled, +stippled
Turns on or off stippling of highlighted squares.

−proportional, +proportional
Turns proportional mode on or off. If proportional mode is on, square width is equal to square height. If proportional mode is off, *bitmap* will use the smaller square dimension, if they were initially different.

−dashes *filename*
Specifies the bitmap to be used as a stipple for dashing.

−stipple *filename*
Specifies the bitmap to be used as a stipple for highlighting.

−hl *color*
Specifies the color used for highlighting.

−fr *color*
Specifies the color used for the frame and grid lines.

filename
Specifies the bitmap to be initially loaded into the program. If the file does not exist, *bitmap* will assume it is a new file.

basename
Specifies the basename to be used in the C code output file. If it is different than the basename in the working file, *bitmap* will change it when saving the file.

Bmtoa accepts the following option:

−chars *cc*
This option specifies the pair of characters to use in the string version of the bitmap. The first character is used for 0 bits and the second character is used for 1 bits. The default is to use dashes (−) for 0's and sharp signs (#) for 1's.

Atobm accepts the following options:

−chars *cc*
This option specifies the pair of characters to use when converting string bitmaps into arrays of numbers. The first character represents a 0 bit and the second character represents a 1 bit. The default is to use dashes (−) for 0's and sharp signs (#) for 1's.

−name *variable*
This option specifies the variable name to be used when writing out the bitmap file. The default is to use the basename of the *filename* command line argument or leave it blank if the standard input is read.

−xhot *number*
This option specifies the X coordinate of the hotspot. Only positive values are allowed. By default, no hotspot information is included.

−yhot *number*
This option specifies the Y coordinate of the hotspot. Only positive values are allowed. By default, no hotspot information is included.

USAGE

Bitmap displays grid in which each square represents a single bit in the picture being edited. Actual size of the bitmap image, as it would appear normaly and inverted, can be obtained by pressing **Meta-I** key. You are free to move the image popup out of the way to continue editing. Pressing the left mouse button in the popup window or **Meta-I** again will remove the real size bitmap image.

If the bitmap is to be used for defining a cursor, one of the squares in the images may be designated as the hot spot. This determines where the cursor is actually pointing. For cursors with sharp tips (such as arrows or fingers), this is usually at the end of the tip; for symmetric cursors (such as crosses or bullseyes), this is usually at the center.

Bitmaps are stored as small C code fragments suitable for including in applications. They provide an array of bits as well as symbolic constants giving the width, height, and hot spot (if specified) that may be used in creating cursors, icons, and tiles.

EDITING

To edit a bitmap image simply click on one of the buttons with drawing commands

(**Point, Curve, Line, Rectangle,** etc.) and move the pointer into the bitmap grid window. Press one of the buttons on your mouse and the appropriate action will take place. You can either set, clear or invert the gird squares. Setting a grid square corresponds to setting a bit in the bitmap image to 1. Clearing a grid square corresponds to setting a bit in the bitmap image to 0. Inverting a grid square corresponds to changing a bit in the bitmap image from 0 to 1 or 1 to 0, depending what its previous state was. The default behavior of mouse buttons is as specified below.

MouseButton1	Set
MouseButton2	Invert
MouseButton3	Clear
MouseButton4	Clear
MouseButton5	Clear

This default behavior can be changed by setting the button function resources. An example is provided below.

```
bitmap*button1Function: Set
bitmap*button2Function: Clear
bitmap*button3Function: Invert
etc.
```

The button function applies to all drawing commands, including copying, moving and pasting, flood filling and setting the hot spot.

DRAWING COMMANDS

Here is the list of drawing commands accessible through the buttons at the left side of the application's window. Some commands can be aborted by pressing A inside the bitmap window, allowing the user to select different guiding points where applicable.

Clear
This command clears all bits in the bitmap image. The grid squares will be set to the background color. Pressing C inside the bitmap window has the same effect.

Set
This command sets all bits in the bitmap image. The grid squares will be set to the foreground color. Pressing S inside the bitmap window has the same effect.

Invert
This command inverts all bits in the bitmap image. The grid squares will be inverted appropriately. Pressing I inside the bitmap window has the same effect.

Mark
This command is used to mark an area of the grid by dragging out a rectangular shape in the highlighting color. Once the area is marked, it can be operated on by a number of commands (see **Up, Down, Left, Right, Rotate, Flip, Cut,** etc.) Only one marked area can be present at any time. If you attempt to mark another area, the old mark will vanish. The same effect can be achieved by pressing **Shift-MouseButton1** and dragging out a rectangle in the grid window. Pressing **Shift-MouseButton2** will mark the entire grid area.

Unmark
This command will cause the marked area to vanish. The same effect can be achieved by pressing **Shift-MouseButton3**.

Copy
This command is used to copy an area of the grid from one location to another. If there is no marked grid area displayed, **Copy** behaves just like **Mark** described above. Once there is a marked grid area displayed in the highlighting color, this command has two alternative behaviors. If you click a mouse button inside the marked area, you will be able to drag the rectangle that represents the marked area to the desired location. After you release the mouse button, the area will be copied. If you click outside the marked area, **Copy** will assume that you wish to mark a different region of the bitmap image, thus it will behave like **Mark** again.

Move
This command is used to move an area of the grid from one location to another. Its behavior resembles the behavior of **Copy** command, except that the marked area will be moved instead of copied.

Flip Horizontally
This command will flip the bitmap image with respect to the horizontal axes. If a marked area of the grid is highlighted, it will operate only inside the marked area. Pressing H inside the bitmap window has the same effect.

Up This command moves the bitmap image one pixel up. If a marked area of the grid is highlighted, it will operate only inside the marked area. Pressing UpArrow inside the bitmap window has the same effect.

Flip Vertically
This command will flip the bitmap image with respect to the vertical axes. If a marked area of the grid is highlighted, it will operate only inside the marked area. Pressing V inside the bitmap window has the same effect.

Left
This command moves the bitmap image one pixel to the left. If a marked area of the grid is highlighted, it will operate only inside the marked area. Pressing LeftArrow inside the bitmap window has the same effect.

Fold
This command will fold the bitmap image so that the opposite corners become adjacent. This is useful when creating bitmap images for tiling. Pressing F inside the bitmap window has the same effect.

Right
This command moves the bitmap image one pixel to the right. If a marked area of the grid is highlighted, it will operate only inside the marked area. Pressing RightArrow inside the bitmap window has the same effect.

Rotate Left
This command rotates the bitmap image 90 degrees to the left (counter clockwise.) If a marked area of the grid is highlighted, it will operate only inside the marked area. Pressing L inside the bitmap window has the same effect.

Down
This command moves the bitmap image one pixel down. If a marked area of the grid is highlighted, it will operate only inside the marked area. Pressing DownArrow inside the bitmap window has the same effect.

Rotate Right
This command rotates the bitmap image 90 degrees to the right (clockwise.) If a marked area of the grid is highlighted, it will operate only inside the marked area.

Pressing R inside the bitmap window has the same effect.

Point
This command will change the grid squares underneath the mouse pointer if a mouse button is being pressed down. If you drag the mouse button continuously, the line may not be continuous, depending on the speed of your system and frequency of mouse motion events.

Curve
This command will change the grid squares underneath the mouse pointer if a mouse button is being pressed down. If you drag the mouse button continuously, it will make sure that the line is continuous. If your system is slow or *bitmap* receives very few mouse motion events, it might behave quite strangely.

Line
This command will change the gird squares in a line between two squares. Once you press a mouse button in the grid window, *bitmap* will highlight the line from the square where the mouse button was initially pressed to the square where the mouse pointer is located. By releasing the mouse button you will cause the change to take effect, and the highlighted line will disappear.

Rectangle
This command will change the gird squares in a rectangle between two squares. Once you press a mouse button in the grid window, *bitmap* will highlight the rectangle from the square where the mouse button was initially pressed to the square where the mouse pointer is located. By releasing the mouse button you will cause the change to take effect, and the highlighted rectangle will disappear.

Filled Rectangle
This command is identical to **Rectangle**, except at the end the rectangle will be filled rather than outlined.

Circle
This command will change the gird squares in a circle between two squares. Once you press a mouse button in the grid window, *bitmap* will highlight the circle from the square where the mouse button was initially pressed to the square where the mouse pointer is located. By releasing the mouse button you will cause the change to take effect, and the highlighted circle will disappear.

Filled Circle
This command is identical to **Circle**, except at the end the circle will be filled rather than outlined.

Flood Fill
This command will flood fill the connected area underneath the mouse pointer when you click on the desired square. Diagonally adjacent squares are not considered to be connected.

Set Hot Spot
This command designates one square in the grid as the hot spot if this bitmap image is to be used for defining a cursor. Pressing a mouse button in the desired square will cause a diamond shape to be displayed.

Clear Hot Spot
This command removes any designated hot spot from the bitmap image.

Undo
This command will undo the last executed command. It has depth one, that is,

pressing **Undo** after **Undo** will undo itself.

FILE MENU

The File menu commands can be accessed by pressing the File button and selecting the appropriate menu entry, or by pressing Ctrl key with another key. These commands deal with files and global bitmap parameters, such as size, basename, filename etc.

New

This command will clear the editing area and prompt for the name of the new file to be edited. It will not load in the new file.

Load

This command is used to load a new bitmap file into the bitmap editor. If the current image has not been saved, user will be asked whether to save or ignore the changes. The editor can edit only one file at a time. If you need interactive editing, run a number of editors and use cut and paste mechanism as described below.

Insert

This command is used to insert a bitmap file into the image being currently edited. After being prompted for the filename, click inside the grid window and drag the outlined rectangle to the location where you want to insert the new file.

Save

This command will save the bitmap image. It will not prompt for the filename unless it is said to be <none>. If you leave the filename undesignated or −, the output will be piped to stdout.

Save As

This command will save the bitmap image after prompting for a new filename. It should be used if you want to change the filename.

Resize

This command is used to resize the editing area to the new number of pixels. The size should be entered in the WIDTHxHEIGHT format. The information in the image being edited will not be lost unless the new size is smaller that the current image size. The editor was not designed to edit huge files.

Rescale

This command is used to rescale the editing area to the new width and height. The size should be entered in the WIDTHxHEIGHT format. It will not do antialiasing and information will be lost if you rescale to the smaller sizes. Feel free to add you own algorithms for better rescaling.

Filename

This command is used to change the filename without changing the basename nor saving the file. If you specify − for a filename, the output will be piped to stdout.

Basename

This command is used to change the basename, if a different one from the specified filename is desired.

Quit

This command will terminate the bitmap application. If the file was not saved, user will be prompted and asked whether to save the image or not. This command is preferred over killing the process.

EDIT MENU

The Edit menu commands can be accessed by pressing the Edit button and selecting the appropriate menu entry, or by pressing Meta key with another key. These commands deal with editing facilities such as grid, axes, zooming, cut and paste, etc.

Image
This command will display the image being edited and its inverse in its actual size in a separate window. The window can be moved away to continue with editing. Pressing the left mouse button in the image window will cause it to disappear from the screen.

Grid
This command controls the grid in the editing area. If the grid spacing is below the value specified by gridTolerance resource (8 by default), the grid will be automatically turned off. It can be enforced by explicitly activating this command.

Dashed
This command controls the stipple for drawing the grid lines. The stipple specified by dashes resource can be turned on or off by activating this command.

Axes
This command controls the highlighting of the main axes of the image being edited. The actual lines are not part of the image. They are provided to aid user when constructing symmetrical images, or whenever having the main axes highlighted helps your editing.

Stippled
This command controls the stippling of the highlighted areas of the bitmap image. The stipple specified by stipple resource can be turned on or off by activating this command.

Proportional
This command controls the proportional mode. If the proportional mode is on, width and height of all image squares are forced to be equal, regardless of the proportions of the bitmap window.

Zoom
This command controls the zoom mode. If there is a marked area of the image already displayed, bitmap will automatically zoom into it. Otherwise, user will have to highlight an area to be edited in the zoom mode and bitmap will automatically switch into it. One can use all the editing commands and other utilities in the zoom mode. When you zoom out, undo command will undo the whole zoom session.

Cut This commands cuts the contents of the highlighted image area into the internal cut and paste buffer.

Copy
This command copies the contents of the highlighted image area into the internal cut and paste buffer.

Paste
This command will check if there are any other bitmap applications with a highlighted image area, or if there is something in the internal cut and paste buffer and copy it to the image. To place the copied image, click in the editing window and drag the outlined image to the position where you want to place i, and then release the button.

CUT AND PASTE

Bitmap supports two cut and paste mechanisms; the internal cut and paste and the global X selection cut and paste. The internal cut and paste is used when executing copy and move drawing commands and also cut and copy commands from the edit menu. The global X selection cut and paste is used whenever there is a highlighted area of a bitmap image displayed anywhere on the screen. To copy a part of image from another bitmap editor simply highlight the desired area by using the Mark command or pressing the shift key and dragging the area with the left mouse button. When the selected area becomes highlighted, any other applications (such as xterm, etc.) that use primary selection will discard their selection values and unhighlight the appropriate information. Now, use the Paste command for the Edit menu or control mouse button to copy the selected part of image into another (or the same) bitmap application. If you attempt to do this without a visible highlighted image area, the bitmap will fall back to the internal cut and paste buffer and paste whatever was there stored at the moment.

WIDGETS

Below is the widget structure of the *bitmap* application. Indentation indicates hierarchical structure. The widget class name is given first, followed by the widget instance name. All widgets except the bitmap widget are from the standard Athena widget set.

```
Bitmap bitmap
        TransientShell image
                Box box
                        Label normalImage
                        Label invertedImage
        TransientShell input
                Dialog dialog
                        Command okay
                        Command cancel
        TransientShell error
                Dialog dialog
                        Command abort
                        Command retry
        TransientShell qsave
                Dialog dialog
                        Command yes
                        Command no
                        Command cancel
        Paned parent
                Form formy
                        MenuButton fileButton
                        SimpleMenu fileMenu
                                SmeBSB new
                                SmeBSB load
                                SmeBSB insert
                                SmeBSB save
                                SmeBSB saveAs
                                SmeBSB resize
                                SmeBSB rescale
                                SmeBSB filename
                                SmeBSB basename
                                SmeLine line
                                SmeBSB quit
                        MenuButton editButton
                        SimpleMenu editMenu
```

```
                SmeBSB  image
                SmeBSB  grid
                SmeBSB  dashed
                SmeBSB  axes
                SmeBSB  stippled
                SmeBSB  proportional
                SmeBSB  zoom
                SmeLine line
                SmeBSB  cut
                SmeBSB  copy
                SmeBSB  paste
        Label status
Pane pane
        Bitmap bitmap
        Form form
                Command clear
                Command set
                Command invert
                Toggle  mark
                Command unmark
                Toggle  copy
                Toggle  move
                Command flipHoriz
                Command up
                Command flipVert
                Command left
                Command fold
                Command right
                Command rotateLeft
                Command down
                Command rotateRight
                Toggle  point
                Toggle  curve
                Toggle  line
                Toggle  rectangle
                Toggle  filledRectangle
                Toggle  circle
                Toggle  filledCircle
                Toggle  floodFill
                Toggle  setHotSpot
                Command clearHotSpot
                Command undo
```

COLORS

If you would like bitmap to be viewable in color, include the following in the #ifdef COLOR section of the file you read with xrdb:

```
*customization:        –color
```

This will cause bitmap to pick up the colors in the app-defaults color customization file:

<XRoot>/lib/X11/app-defaults/Bitmap-color

where <XRoot> refers to the root of the X11 install tree.

BITMAP WIDGET

Bitmap widget is a stand-alone widget for editing raster images. It is not designed to edit large images, although it may be used in that purpose as well. It can be freely incorporated with other applications and used as a standard editing tool. The following are the resources provided by the bitmap widget.

Bitmap Widget

Header file	Bitmap.h
Class	bitmapWidgetClass
Class Name	Bitmap
Superclass	Bitmap

All the Simple Widget resources plus . . .

Name	Class	Type	Default Value
foreground	Foreground	Pixel	XtDefaultForeground
highlight	Highlight	Pixel	XtDefaultForeground
framing	Framing	Pixel	XtDefaultForeground
gridTolerance	GridTolerance	Dimension	8
size	Size	String	32x32
dashed	Dashed	Boolean	True
grid	Grid	Boolean	True
stippled	Stippled	Boolean	True
proportional	Proportional	Boolean	True
axes	Axes	Boolean	False
squareWidth	SquareWidth	Dimension	16
squareHeight	SquareHeight	Dimension	16
margin	Margin	Dimension	16
xHot	XHot	Position	NotSet (−1)
yHot	YHot	Position	NotSet (−1)
button1Function	Button1Function	DrawingFunction	Set
button2Function	Button2Function	DrawingFunction	Invert
button3Function	Button3Function	DrawingFunction	Clear
button4Function	Button4Function	DrawingFunction	Invert
button5Function	Button5Function	DrawingFunction	Invert
filename	Filename	String	None ("")
basename	Basename	String	None ("")

AUTHOR

Davor Matic, MIT X Consortium

NAME

editres – a dynamic resource editor for X Toolkit applications

SYNTAX

editres [–*toolkitoption* ...]

OPTIONS

Editres accepts all of the standard X Toolkit command line options (see *X(1)*). The order of the command line options is not important.

DESCRIPTION

Editres is a tool that allows users and application developers to view the full widget hierarchy of any X Toolkit application that speaks the Editres protocol. In addition, editres will help the user construct resource specifications, allow the user to apply the resource to the application and view the results dynamically. Once the user is happy with a resource specification editres will append the resource string to the user's X Resources file.

USING EDITRES

Editres provides a window consisting of the following four areas:

Menu Bar	A set of popup menus that allow you full access to editres's features.
Panner	The panner allows a more intuitive way to scroll the application tree display.
Message Area	Displays information to the user about the action that editres expects of her.
Application Widget Tree	This area will be used to display the selected application's widget tree. To begin an editres session select the **Get Widget Tree** menu item from the command menu. This will change the pointer cursor to cross hair. You should now select the application you wish look at by clicking on any of its windows. If this application understands the editres protocol then editres will display the application's widget tree in its tree window. If the application does not understand the editres protocol editres will inform you of this fact in the message area after a few seconds delay. Once you have a widget tree you may now select any of the other menu options. The effect of each of these is described below.

COMMANDS

Get Widget Tree

Allows the user to click on any application that speaks the editres protocol and receive its widget tree.

Refresh Current Widget Tree

Editres only knows about the widgets that exist at the present time. Many applications create and destroy widgets on the fly. Selecting this menu item will cause editres to ask the application to resend its widget tree, thus updating its information to the new state of the application.

For example, xman only creates the widgets for its *topbox* when it starts up. None of the widgets for the manual page window are created until the user actually clicks on the *Manual Page* button. If you retrieved xman's widget tree before the the manual page is active, you may wish to refresh the widget tree after the manual page has been displayed. This will allow you to also edit the manual page's resources.

Dump Widget Tree to a File
: For documenting applications it is often useful to be able to dump the entire application widget tree to an ASCII file. This file can then be included in the manual page. When this menu item is selected a popup dialog is activated. Type the name of the file in this dialog, and either select *okay*, or type a carriage-return. Editres will now dump the widget tree to this file. To cancel the file dialog, select the *cancel* button.

Show Resource Box
: This command will popup a resource box for the current application. This resource box (described in detail below) will allow the user to see exactly which resources can be set for the widget that is currently selected in the widget tree display. Only one widget may be currently selected; if greater or fewer are selected editres will refuse to pop up the resource box and put an error message in the **Message Area**.

Set Resource
: This command will popup a simple dialog box for setting an arbitrary resource on all selected widgets. You must type in the resource name, as well as the value. You can use the Tab key to switch between the resource name field the resource value field.

Quit
: Exits editres.

TREE COMMANDS

The **Tree** menu contains several commands that allow operations to be performed on the widget tree.

Select Widget in Client
: This menu item allows you to select any widget in the application; editres will then highlight the corresponding element the widget tree display. Once this menu item is selected the pointer cursor will again turn to a crosshair, and you must click any pointer button in the widget you wish to have displayed. Since some widgets are fully obscured by their children, it is not possible to get to every widget this way, but this mechanism does give very useful feedback between the elements in the widget tree and those in the actual application.

Select All
Unselect All
Invert All
: These functions allow the user to select, unselect, or invert all widgets in the widget tree.

Select Children
Select Parents
: These functions select the immediate parent or children of each of the currently selected widgets.

Select Descendants
Select Ancestors
: These functions select all parents or children of each of the currently selected widgets. This is a recursive search.

Show Widget Names
Show Class Names
Show Widget Windows

When the tree widget is initially displayed the labels of each widget in the tree correspond to the widget names. These functions will cause the label of **all** widgets in the tree to be changed to show the class name, IDs, or window associated with each widget in the application. The widget IDs, and windows are shown as hex numbers. In addition there are keyboard accelerators for each of the Tree operations. If the input focus is over an individual widget in the tree, then that operation will only effect that widget. If the input focus is in the Tree background it will have exactly the same effect as the corresponding menu item. The translation entries shown may be applied to any widget in the application. If that widget is a child of the Tree widget, then it will only affect that widget, otherwise it will have the same effect as the commands in the tree menu.

Flash Active Widgets

This command is the inverse of the **Select Widget in Client** command, it will show the user each widget that is currently selected in the widget tree, by flashing the corresponding widget in the application *numFlashes* (three by default) times in the *flashColor*.

Key	**Option**	**Translation Entry**
space	Unselect	Select(nothing)
w	Select	Select(widget)
s	Select	Select(all)
i	Invert	Select(invert)
c	Select Children	Select(children)
d	Select Descendants	Select(descendants)
p	Select Parent	Select(parent)
a	Select Ancestors	Select(ancestors)
N	Show Widget Names	Relabel(name)
C	Show Class Names	Relabel(class)
I	Show Widget IDs	Relabel(id)
W	Show Widget Windows	Relabel(window)
T	Toggle Widget/Class Name	Relabel(toggle)

Clicking button 1 on a widget adds it to the set of selected widgets. Clicking button 2 on a widget deselects all other widgets and then selects just that widget. Clicking button 3 on a widget toggles its label between the widget's instance name the widget's class name.

USING THE RESOURCE BOX

The resource box contains five different areas. Each of the areas, as they appear on the screen, from top to bottom will be discussed.

The Resource Line

This area at the top of the resource box shows the current resource name exactly as it would appear if you were to save it to a file or apply it.

The Widget Names and Classes

This area allows you to select exactly which widgets this resource will apply to. The area contains four lines, the first contains the name of the selected widget and all its ancestors, and the more restrictive dot (**.**) separator. The second line contains less specific the Class names of each widget, and well as the less restrictive star (*****) separator. The third line contains a set of special buttons called **Any Widget** which will generalize this level to match any widget. The last line contains a set of special buttons called **Any Widget Chain** which will turn the single level into something that matches zero or more levels.

The initial state of this area is the most restrictive, using the resource names and the dot separator. By selecting the other buttons in this area you can ease the restrictions to allow more and more widgets to match the specification. The extreme case is to select all the **Any Widget Chain** buttons, which will match every widget in the application. As you select different buttons the tree display will update to show you exactly which widgets will be effected by the current resource specification.

Normal and Constraint Resources

The next area allows you to select the name of the normal or constraint resources you wish to set. Some widgets may not have constraint resources, so that area will not appear.

Resource Value

This next area allows you to enter the resource value. This value should be entered exactly as you would type a line into your resource file. Thus it should contain no unescaped new-lines. There are a few special character sequences for this file:

\n - This will be replaced with a newline.

\### - Where # is any octal digit. This will be replaced with a single byte that contains this sequence interpreted as an octal number. For example, a value containing a NULL byte can be stored by specifying \000.

\<new-line> - This will compress to nothing.

\\ - This will compress to a single backslash.

Command Area

This area contains several command buttons, described in this section.

Set Save File

This button allows the user to modify file that the resources will be saved to. This button will bring up a dialog box that will ask you for a filename; once the filename has been entered, either hit carriage-return or click on the *okay* button. To pop down the dialog box without changing the save file, click the *cancel* button.

Save

This button will append the **resource line** described above to the end of the current save file. If no save file has been set the **Set Save File** dialog box will be popped up to prompt the user for a filename.

Apply

This button attempts to perform a XtSetValues call on all widgets that match the **resource line** described above. The value specified is applied directly to all matching widgets. This behavior is an attempt to give a dynamic feel to the resource editor. Since this feature allows users to put an application in states it may not be willing to handle, a hook has been provided to allow specific applications to block these SetValues requests (see **Blocking Editres Requests** below).

Unfortunately due to design constraints imposed on the widgets by the X Toolkit and the Resource Manager, trying to coerce an inherently static system into dynamic behavior can cause strange results. There is no guarantee that the results of an apply will be the same as what will happen when you save the value and restart the application. This functionality is provided to try to give you a rough feel for what your changes will accomplish, and the results obtained should be considered suspect at best. Having said that, this is one of the neatest features of editres, and I strongly suggest that you play with it, and see what it can do.

Save and Apply
: This button combines the Save and Apply actions described above into one button.

Popdown Resource Box
: This button will remove the resource box from the display.

BLOCKING EDITRES REQUESTS

The editres protocol has been built into the Athena Widget set. This allows all applications that are linked against Xaw to be able to speak to the resource editor. While this provides great flexibility, and is a useful tool, it can quite easily be abused. It is therefore possible for any Xaw application to specify a value for the **editresBlock** resource described below, to keep editres from divulging information about its internals, or to disable the **SetValues** part of the protocol.

editresBlock (Class **EditresBlock**)
: Specifies which type of blocking this application wishes to impose on the editres protocol. The accepted values are:

all
: Block all requests.

setValues
: Block all SetValues requests. As this is the only editres request that actually modifies the application, this is in effect stating that the application is read-only.

none
: Allow all editres requests. Remember that these resources are set on any Xaw application, **not editres**. They allow individual applications to keep all or some of the requests editres makes from ever succeeding. Of course, editres is also an Xaw application, so it may also be viewed and modified by editres (rather recursive, I know), these commands can be blocked by setting the **editresBlock** resource on editres itself.

RESOURCES

For *editres* the available application resources are:

numFlashes (Class **NumFlashes**)
: Specifies the number of times the widgets in the application will be flashed when the **Show Active Widgets** command in invoked.

flashTime (Class **FlashTime**)
: Amount of time between the flashes described above.

flashColor (Class **flashColor**)
: Specifies the color used to flash application widgets. A bright color should be used that will immediately draw your attention to the area being flashed, such as red or yellow.

saveResourcesFile (Class **SaveResourcesFile**)
: This is the file the resource line will be append to when the **Save** button

activated in the resource box.

WIDGETS

In order to specify resources, it is useful to know the hierarchy of the widgets which compose *editres*. In the notation below, indentation indicates hierarchical structure. The widget class name is given first, followed by the widget instance name.

```
Editres  editres
        Paned  paned
                Box  box
                        MenuButton  commands
                                SimpleMenu  menu
                                SmeBSB  sendTree
                                SmeBSB  refreshTree
                                SmeBSB  dumpTreeToFile
                                SmeLine  line
                                SmeBSB  getResourceList
                                SmeLine  line
                                SmeBSB  quit
                        MenuButton  treeCommands
                                SimpleMenu  menu
                                SmeBSB  showClientWidget
                                SmeBSB  selectAll
                                SmeBSB  unselectAll
                                SmeBSB  invertAll
                                SmeLine  line
                                SmeBSB  selectChildren
                                SmeBSB  selectParent
                                SmeBSB  selectDescendants
                                SmeBSB  selectAncestors
                                SmeLine  line
                                SmeBSB  showWidgetNames
                                SmeBSB  showClassNames
                                SmeBSB  showWidgetIDs
                                SmeBSB  showWidgetWindows
                                SmeLine  line
                                SmeBSB  flashActiveWidgets
                Paned  hPane
                        Panner  panner
                        Label  userMessage
                        Grip  grip
                Porthole  porthole
                        Tree  tree
                                Toggle  <name of widget in application>
                                .
                                .
                                .
                                TransientShell  resourceBox
                                Paned  pane
                                Label  resourceLabel
                                Form  namesAndClasses
                                Toggle  dot
```

```
                    Toggle  star
                    Toggle  any
                    Toggle  name
                    Toggle  class
                       .
                       .
                       .
                    Label  namesLabel
                    List  namesList
                    Label  constraintLabel
                    List  constraintList
                    Form  valueForm
                    Label  valueLabel
                    Text  valueText
                    Box  commandBox
                    Command  setFile
                    Command  save
                    Command  apply
                    Command  saveAndApply
                    Command  cancel
                    Grip  grip
            Grip  grip
```

ENVIRONMENT

DISPLAY
to get the default host and display number.

XENVIRONMENT
to get the name of a resource file that overrides the global resources stored in the RESOURCE_MANAGER property.

FILES

<XRoot>/lib/X11/app-defaults/Editres - specifies required resources

SEE ALSO

X(1), xrdb(1), Athena Widget Set

RESTRICTIONS

This is a prototype, there are lots of nifty features I would love to add, but I hope this will give you some ideas about what a resource editor can do.

AUTHOR

Chris D. Peterson, formerly MIT X Consortium

NAME
fsinfo – X font server information utility

SYNOPSIS
fsinfo [−server *servername*]

DESCRIPTION
Fsinfo is a utility for displaying information about an X font server. It is used to examine the capabilities of a server, the predefined values for various parameters used in communicating between clients and the server, and the font catalogues and alternate servers that are available.

EXAMPLE
The following shows a sample produced by *fsinfo.*

```
name of server:    hansen:7100
version number:    1
vendor string:     Font Server Prototype
vendor release number:     17
maximum request size:      16384 longwords (65536 bytes)
number of catalogues:      1
        all
Number of alternate servers: 2
    #0     hansen:7101
    #1     hansen:7102
number of extensions:      0
```

ENVIRONMENT
FONTSERVER
To get the default fontserver.

SEE ALSO
xfs(1), fslsfonts(1)

AUTHOR
Dave Lemke, Network Computing Devices, Inc

NAME

fslsfonts – list fonts served by X font server

SYNOPSIS

fslsfonts [–options ...] [–fn pattern]

DESCRIPTION

Fslsfonts lists the fonts that match the given *pattern*. The wildcard character "*" may be used to match any sequence of characters (including none), and "?" to match any single character. If no pattern is given, "*" is assumed.

The "*" and "?" characters must be quoted to prevent them from being expanded by the shell.

OPTIONS

–server *host***:***port*
: This option specifies the X font server to contact.

–l Lists some attributes of the font on one line in addition to its name.

–ll Lists font properties in addition to **–l** output.

–lll Supported for compatibility with *xlsfonts*, but output is the same as for **–ll**.

–m This option indicates that long listings should also print the minimum and maximum bounds of each font.

–C This option indicates that listings should use multiple columns. This is the same as **–n 0**.

–1 This option indicates that listings should use a single column. This is the same as **–n 1**.

–w *width*
: This option specifies the width in characters that should be used in figuring out how many columns to print. The default is 79.

–n *columns*
: This option specifies the number of columns to use in displaying the output. The default is 0, which will attempt to fit as many columns of font names into the number of character specified by **–w** *width*.

–u This option indicates that the output should be left unsorted.

SEE ALSO

xfs(1), showfont(1), xlsfonts(1)

ENVIRONMENT

FONTSERVER
: to get the default host and port to use.

BUGS

Doing ‘‘fslsfonts –l’’ can tie up your server for a very long time. This is really a bug with single-threaded non-preemptable servers, not with this program.

AUTHOR

Dave Lemke, Network Computing Devices, Inc

NAME

fstobdf – generate BDF font from X font server

SYNOPSIS

fstobdf [**–server** *server*] **–fn** *fontname*

DESCRIPTION

The *fstobdf* program reads a font from a font server and prints a BDF file on the standard output that may be used to recreate the font. This is useful in testing servers, debugging font metrics, and reproducing lost BDF files.

OPTIONS

–server *servername*
: This option specifies the server from which the font should be read.

–fn *fontname*
: This option specifies the font for which a BDF file should be generated.

ENVIRONMENT

FONTSERVER
: default server to use

SEE ALSO

xfs(1), bdftopcf(1), fslsfonts(1)

AUTHOR

Olaf Brandt, Network Computing Devices
Dave Lemke, Network Computing Devices

Jim Fulton, MIT X Consortium

NAME

iceauth – ICE authority file utility

SYNOPSIS

iceauth [**–f** *authfile*] [**–vqib**] [*command arg* ...]

DESCRIPTION

The *iceauth* program is used to edit and display the authorization information used in connecting with ICE. This program is usually used to extract authorization records from one machine and merge them in on another (as is the case when using remote logins or granting access to other users). Commands (described below) may be entered interactively, on the *iceauth* command line, or in scripts.

AUTHOR

Ralph Mor, X Consortium

NAME

lbxproxy - Low BandWidth X proxy

SYNOPSIS

lbxproxy [:<display>] [option]

DESCRIPTION

Applications that would like to take advantage of the Low Bandwidth extension to X (LBX) must make their connections to an lbxproxy. These applications need to know nothing about LBX, they simply connect to the lbxproxy as if were a regular server. The lbxproxy accepts client connections, multiplexes them over a single connection to the X server, and performs various optimizations on the X protocol to make it faster over low bandwidth and/or high latency connections.

With regard to authentication/authorization, lbxproxy simply passes along to the server the credentials presented by the client. Since X clients will connect to lbxproxy, it is important that the user's .Xauthority file contain entries with valid keys associated with the network ID of the proxy. lbxproxy does not get involved with how these entries are added to the .Xauthority file. The user is responsible for setting it up.

The lbxproxy program has various options, all of which are optional.

If **:<display>** is specified, the proxy will use the given display port when listening for connections. The display port is an offset from port 6000, identical to the way in which regular X display connections are specified. If no port is specified on the command line option, lbxproxy will default to port 63. If the port number that the proxy tries to listen on is in use, the proxy will attempt to use another port number. If the proxy is not using the Proxy Manager and the default port number cannot be used, the port number that is used will be written to stderr.

The other command line options that can be specified are:

–help Prints a brief help message about the command line options.

–display *dpy*
: Specifies the address of the X server supporting the LBX extension. If this option is not specified, the display is obtained by the DISPLAY environment variable.

–motion *count*
: A limited number of pointer motion events are allowed to be in flight between the server and the proxy at any given time. The maximimum number of motion events that can be in flight is set with this option; the default is 8.

–maxservers *number*
: The default behavior of lbxproxy is to manage a single server. However, lbxproxy can manage more than one server. The default maximum number of servers is 20. The number of servers can be overridden by setting the environment variable LBXPROXY_MAXSERVERS to the desired number. The order of precedence from highest to lowest: command line, environment variable, default number.

–[terminate|reset]
: The default behavior of lbxproxy is to continue running as usual when it's last client exits. The **–terminate** option will cause lbxproxy to exit when the last client exits. The **–reset** option will cause lbxproxy to reset itself when the last client exits. Resetting causes lbxproxy to clean up it's state and reconnect to the server.

−reconnect
The default behavior of lbxproxy is to exit when its connection to the server is broken. The **−reconnect** option will cause lbxproxy to just reset instead (see **−reset** above) and attempt to reconnect to the server.

−I Causes all remaining arguments to be ignored.

−nolbx Disables all LBX optimizations.

−nocomp
Disables stream compression.

−nodelta Disables delta request substitutions.

−notags Disables usage of tags.

−nogfx Disables reencoding of graphics requests (not including image related requests).

−noimage
Disables image compression.

−nosquish
Disables squishing of X events.

−nointernsc
Disables short circuiting of InternAtom requests.

−noatomsfile
Disables reading of the atoms control file. See the section on "Atom Control" for more details.

−atomsfile *file*
Overrides the default AtomControl file. See the section on "Atom Control" for more details.

−nowinattr
Disables GetWindowAttributes/GetGeometry grouping into one round trip.

−nograbcmap
Disables colormap grabbing.

−norgbfile
Disables color name to RGB resolution in proxy.

−rgbfile *path*
Specifies an alternate RGB database for color name to RGB resolution.

−tagcachesize
Set the size of the proxy's tag cache (in bytes).

−zlevel *level*
Set the Zlib compression level (used for stream compression).
default is 6
1 = worst compression, fastest
9 = best compression, slowest

−compstats
Report stream compression statistics every time the proxy resets or receives a SIGHUP signal.

−nozeropad
Don't zero out unused pad bytes in X requests, replies, and events.

−cheaterrors

Allows cheating on X protocol for the sake of improved performance. The X protocol guarantees that any replies, events or errors generated by a previous request will be sent before those of a later request. This puts substantial restrictions on when lbxproxy can short circuit a request. The -cheaterrors option allows lbxproxy to violate X protocol rules with respect to errors. Use at your own risk.

−cheatevents

The -cheatevents option allows lbxproxy to violate X protocol rules with respect to events as well as errors. Use at your own risk.

ATOM CONTROL

At startup, lbxproxy "pre-interns" a configurable list of atoms. This allows lbxproxy to intern a group of atoms in a single round trip and immediately store the results in its cache.

While running, lbxproxy uses heuristics to decide when to delay sending window property data to the server. The heuristics depend on the size of the data, the name of the property, and whether a window manager is running through the same lbxproxy.

Atom control is specified in the "AtomControl" file, set up during installation of lbxproxy, with command line overrides.

The file is a simple text file. There are three forms of lines: comments, length control, and name control. Lines starting with a '!' are treated as comments. A line of the form

z *length*

specifies the minimum length in bytes before property data will be delayed. A line of the form

options atomname

controls the given atom, where *options* is any combination of the following characters: 'i' means the atom should be pre-interned; and 'w' means data for properties with this name should be delayed only if a window manager is also running through the same lbxproxy.

BUGS

When the authorization protocol XDM-AUTHORIZATION-1 is used:

A client must be on the same host as lbxproxy for the client to be authorized to connect to the server.

If a client is not on the same host as lbxproxy, the client will not be authorized to connect to the server.

NAME

mkfontdir, fonts.dir, fonts.scale, fonts.alias – create an index of X font files in a directory

SYNOPSIS

mkfontdir [*directory-name* ...]

DESCRIPTION

For each directory argument, *mkfontdir* reads all of the font files in the directory searching for properties named "FONT", or (failing that) the name of the file stripped of its suffix. These are converted to lower case and used as font names, and, along with the name of the font file, are written out to the file "fonts.dir" in the directory. The X server and font server use "fonts.dir" to find font files.

The kinds of font files read by *mkfontdir* depend on configuration parameters, but typically include PCF (suffix ".pcf"), SNF (suffix ".snf") and BDF (suffix ".bdf"). If a font exists in multiple formats, *mkfontdir* will first choose PCF, then SNF and finally BDF.

The first line of fonts.dir gives the number of fonts in the file. The remaining lines list the fonts themselves, one per line, in two fields. First is the name of the font file, followed by a space and the name of the font.

SCALABLE FONTS

Because scalable font files do not usually include the X font name, the file "fonts.scale" can be used to name the scalable fonts in the directory. The fonts listed in it are copied to fonts.dir by *mkfontdir*. "fonts.scale" has the same format as the "fonts.dir" file.

FONT NAME ALIASES

The file "fonts.alias", which can be put in any directory of the font-path, is used to map new names to existing fonts, and should be edited by hand. The format is two white-space separated columns, the first containing aliases and the second containing font-name patterns. Lines beginning with "!" are comment lines and are ignored.

If neither the alias nor the value specifies the size fields of the font name, this is a scalable alias. A font name of any size that matches this alias will be mapped to the same size of the font that the alias resolves to.

When a font alias is used, the name it references is searched for in the normal manner, looking through each font directory in turn. This means that the aliases need not mention fonts in the same directory as the alias file.

To embed white space in either name, simply enclose it in double-quote marks; to embed double-quote marks (or any other character), precede them with back-slash:

```
"magic-alias with spaces"  "\"font name\" with quotes"
regular-alias              fixed
```

If the string "FILE_NAMES_ALIASES" stands alone on a line, each file-name in the directory (stripped of its suffix) will be used as an alias for that font.

FILES

fonts.dir List of fonts in the directory and the files they are stored in. Created by *mkfontdir*. Read by the X server and font server each time the font path is set (see xset(1)).

fonts.scale List of scalable fonts in the directory. Contents are copied to fonts.dir by *mkfontdir*.

fonts.alias List of font name aliases. Read by the X server and font server each time the font path is set (see xset(1)).

SEE ALSO

X(1), Xserver(1), xfs(1), xset(1)

NAME

oclock – round X clock

SYNOPSIS

oclock [–option ...]

DESCRIPTION

Oclock simply displays the current time on an analog display.

OPTIONS

–fg *color*
: choose a different color for the both hands and the jewel of the clock

–bg *color*
: choose a different color for the background.

–jewel *color*
: choose a different color for the jewel on the clock.

–minute *color*
: choose a different color for the minute hand of the clock.

–hour *color*
: choose a different color for the hour hand of the clock.

–backing *{ WhenMapped Always NotUseful }*
: selects an appropriate level of backing store.

–geometry *geometry*
: define the initial window geometry; see *X(1)*.

–display *display*
: specify the display to use; see *X(1)*.

–bd *color*
: choose a different color for the window border.

–bw *width*
: choose a different width for the window border. As the Clock widget changes its border around quite a bit, this is most usefully set to zero.

–shape causes the clock to use the Shape extension to create an oval window. This is the default unless the shapeWindow resource is set to false.

–noshape
: causes the clock to not reshape itself and ancestors to exactly fit the outline of the clock.

–transparent
: causes the clock to consist only of the jewel, the hands, and the border.

COLORS

If you would like your clock to be viewable in color, include the following in the #ifdef COLOR section you read with xrdb:

*customization: -color

This will cause oclock to pick up the colors in the app-defaults color customization file: <XRoot>/lib/X11/app-defaults/Clock-color. Below are the default colors:

Clock*Background: grey

Clock*BorderColor: light blue
Clock*hour: yellow
Clock*jewel: yellow
Clock*minute: yellow

SEE ALSO

X(1), X Toolkit documentation

AUTHOR

Keith Packard, MIT X Consortium

NAME

proxymngr - proxy manager service

SYNOPSIS

proxymngr [**−config** *filename*] [**−timeout** *seconds*] [**−retries** *#*] [**−verbose**]

DESCRIPTION

The proxy manager (proxymngr) is responsible for resolving requests from xfindproxy (and other similar clients), starting new proxies when appropriate, and keeping track of all of the available proxy services. The proxy manager strives to reuse existing proxies whenever possible.

There are two types of proxies that the proxy manager deals with, *managed* and *unmanaged* proxies.

A *managed* proxy is a proxy that is started "on demand" by the proxy manager.

An *unmanaged* proxy, on the other hand, is started either at system boot time, or manually by a system administrator. The proxy manager is made aware of its existence, but no attempt is made by the proxy manager to start unmanaged proxies.

The command line options that can be specified to **proxymngr** are:

−config Used to override the default proxymngr config file. See below for more details about the config file.

−timeout
Sets the number of seconds between attempts made by the proxy manager to find an unmanaged proxy. The default is 10.

−retries Sets the maximum number of retries made by the proxy manager to find an an unmanaged proxy. The default is 3.

−verbose
Causes various debugging and tracing records to be displayed as requests are received and proxies are started.

Proxy Manager Config File

The proxy manager maintains a local configuration file describing the proxy services available. This configuration file is installed in /usr/X11R6.4/lib/X11/proxymngr/pmconfig during the installation of proxymngr. The location of the configuration file can be overwritten using the **−config** command line option.

Aside from lines starting with an exclamation point for comments, each line of the configuration file describes either an unmanaged or managed proxy service.

For unmanaged proxies, the format is:

<service-name> unmanaged <proxy-address>

service-name is the name of the unmanaged proxy service, and must not contain any spaces, for example "XFWP". service-name is case insenstive.

proxy-address is the network address of the unmanaged proxy. The format of the address is specific to the service-name. For example, for the "XFWP" service, the proxy-address might be "firewall.x.org:100".

If there is more than one entry in the config file with the same unmanaged service-name, the proxy manager will try to use the proxies in the order presented in the config file.

For managed proxies, the format is:

<service-name> managed <command-to-start-proxy>

service-name is the name of the managed proxy service, and must not contain any spaces, for example ‘‘LBX’’. service-name is case insensitive.

command-to-start-proxy is the command executed by the proxy manager to start a new instance of the proxy. If command-to-start-proxy contains spaces, the complete command should be surrounded by single quotes. If desired, command-to-start-proxy can be used to start a proxy on a remote machine. The specifics of the remote execution method used to do this is not specified here.

EXAMPLE

Here is a sample configuration file:

```
! proxy manager config file
!
! Each line has the format:
!    <serviceName> managed <startCommand>
!        or
!    <serviceName> unmanaged <proxyAddress>
!
lbx managed /usr/X11R6.4/bin/lbxproxy
!
! substitute site-specific info
xfwp unmanaged firewall:4444
```

PROXY MANAGER DETAILS

When the proxy manager gets a request from xfindproxy (or another similar client), its course of action will depend on the service-name in question.

For a managed proxy service, the proxy manager will find out if any of the already running proxies for this service can handle a new request. If not, the proxy manager will attempt to start up a new instance of the proxy (using the command-to-start-proxy found in the config file). If that fails, an error will be returned to the caller.

For an unmanaged proxy service, the proxy manager will look in the config file to find all unmanaged proxies for this service. If there is more than one entry in the config file with the same unmanaged service-name, the proxy manager will try to use the proxies in the order presented in the config file. If none of the unmanged proxies can satisfy the request, the proxy manager will timeout for a configurable amount of time (specified by **−timeout** or default of 10) and reattempt to find an unmanaged proxy willing to satisfy the request. The number of retries can be specified by the **−retries** argument, or a default of 3 will be used. If the retries fail, the proxy manager has no choice but to return an error to the caller (since the proxy manager can not start unmanaged proxy services).

BUGS

proxy manager listen port should be configurable.

−timeout and **−retries** is not implemented in proxymngr.

proxymngr does not utilize the ‘‘options’’ and ‘‘host’’ fields in the proxy management protocol GetProxyAddr request.

SEE ALSO

xfindproxy (1), xfwp (1), Proxy Management Protocol spec V1.0

AUTHOR

Ralph Mor, X Consortium

NAME

showrgb − uncompile an rgb color-name database

SYNOPSIS

showrgb [*database*]

DESCRIPTION

The *showrgb* program reads an rgb color-name database compiled for use with the dbm database routines and converts it back to source form, printing the result to standard output. The default database is the one that X was built with, and may be overridden on the command line. Specify the database name without the *.pag* or *.dir* suffix.

FILES

<XRoot>/lib/X11/rgb default database.

NAME

rstart - a sample implementation of a Remote Start client

SYNOPSIS

rstart [−c *context*] [−g] [−l *username*] [−v] *hostname command args ...*

DESCRIPTION

Rstart is a simple implementation of a Remote Start client as defined in "A Flexible Remote Execution Protocol Based on **rsh**". It uses *rsh* as its underlying remote execution mechanism.

OPTIONS

−c *context*
: This option specifies the *context* in which the command is to be run. A *context* specifies a general environment the program is to be run in. The details of this environment are host-specific; the intent is that the client need not know how the environment must be configured. If omitted, the context defaults to **X**. This should be suitable for running X programs from the host's "usual" X installation.

−g
: Interprets *command* as a *generic command*, as discussed in the protocol document. This is intended to allow common applications to be invoked without knowing what they are called on the remote system. Currently, the only generic commands defined are **Terminal**, **LoadMonitor**, **ListContexts**, and **ListGenericCommands**.

−l *username*
: This option is passed to the underlying *rsh*; it requests that the command be run as the specified user.

−v
: This option requests that *rstart* be verbose in its operation. Without this option, *rstart* discards output from the remote's *rstart* helper, and directs the *rstart* helper to detach the program from the *rsh* connection used to start it. With this option, responses from the helper are displayed and the resulting program is not detached from the connection.

NOTES

This is a trivial implementation. Far more sophisticated implementations are possible and should be developed.

Error handling is nonexistant. Without **−v**, error reports from the remote are discarded silently. With **−v**, error reports are displayed.

The $DISPLAY environment variable is passed. If it starts with a colon, the local hostname is prepended. The local domain name should be appended to unqualified host names, but isn't.

The $SESSION_MANAGER environment variable should be passed, but isn't.

X11 authority information is passed for the current display.

ICE authority information should be passed, but isn't. It isn't completely clear how *rstart* should select what ICE authority information to pass.

Even without **−v**, the sample *rstart* helper will leave a shell waiting for the program to complete. This causes no real harm and consumes relatively few resources, but if it is undesirable it can be avoided by explicitly specifying the "exec" command to the shell, eg

rstart somehost exec xterm

This is obviously dependent on the command interpreter being used on the remote system; the example given will work for the Bourne and C shells.

SEE ALSO

rstartd(1), rsh(1), A Flexible Remote Execution Protocol Based on **rsh**

AUTHOR

Jordan Brown, Quarterdeck Office Systems

NAME

rstartd - a sample implementation of a Remote Start rsh helper

SYNOPSIS

rstartd

rstartd.real [−c *configfilename*]

DESCRIPTION

Rstartd is an implementation of a Remote Start "helper" as defined in "A Flexible Remote Execution Protocol Based on **rsh**".

This document describes the peculiarities of *rstartd* and how it is configured.

OPTIONS

−c *configfilename*

This option specifies the "global" configuration file that *rstartd* is to read. Normally, *rstartd* is a shell script that invokes *rstartd.real* with the **-c** switch, allowing local configuration of the location of the configuration file. If *rstartd.real* is started without the -c option, it reads <XRoot>/lib/X11/rstart/config, where <XRoot> refers to the root of the X11 install tree.

INSTALLATION

It is critical to successful interoperation of the Remote Start protocol that *rstartd* be installed in a directory which is in the "default" search path, so that default rsh requests and the ilk will be able to find it.

CONFIGURATION AND OPERATION

Rstartd is by design highly configurable. One would like things like configuration file locations to be fixed, so that users and administrators can find them without searching, but reality is that no two vendors will agree on where things should go, and nobody thinks the original location is "right". Thus, *rstartd* allows one to relocate **all** of its files and directories.

Rstartd has a hierarchy of configuration files which are executed in order when a request is made. They are:

global config
per-user ("local") config
global per-context config
per-user ("local") per-context config
config from request

As you might guess from the presence of "config from request", all of the config files are in the format of an *rstart* request. *Rstartd* defines a few additional keywords with the INTERNAL- prefix for specifying its configuration.

Rstartd starts by reading and executing the global config file. This file will normally specify the locations of the other configuration files and any systemwide defaults.

Rstartd will then read the user's local config file, default name $HOME/.rstart.

Rstartd will then start interpreting the request.

Presumably one of the first lines in the request will be a CONTEXT line. The context name is converted to lower case.

Rstartd will read the global config file for that context, default name <XRoot>/lib/X11/rstart/contexts/<name>, if any.

It will then read the user's config file for that context, default name $HOME/.rstart.contexts/<name>, if any.

(If neither of these exists, *rstartd* aborts with a Failure message.)

Rstartd will finish interpreting the request, and execute the program specified.

This allows the system administrator and the user a large degree of control over the operation of *rstartd*. The administrator has final say, because the global config file doesn't need to specify a per-user config file. If it does, however, the user can override anything from the global file, and can even completely replace the global context config files.

The config files have a somewhat more flexible format than requests do; they are allowed to contain blank lines and lines beginning with "#" are comments and ignored. (#s in the middle of lines are data, not comment markers.)

Any commands run are provided a few useful pieces of information in environment variables. The exact names are configurable, but the supplied defaults are:

$RSTART_CONTEXT	the name of the context
$RSTART_GLOBAL_CONTEXTS	the global contexts directory
$RSTART_LOCAL_CONTEXTS	the local contexts directory
$RSTART_GLOBAL_COMMANDS	the global generic commands directory
$RSTART_LOCAL_COMMANDS	the local generic commands directory

$RSTART_{GLOBAL,LOCAL}_CONTEXTS should contain one special file, @List, which contains a list of the contexts in that directory in the format specified for ListContexts. The supplied version of ListContexts will cat both the global and local copies of @List.

Generic commands are searched for in several places: (defaults)

per-user per-context directory ($HOME/.rstart.commands/<context>)
global per-context directory (<XRoot>/lib/X11/rstart/commands/<context>)
per-user all-contexts directory ($HOME/.rstart.commands)
global all-contexts directory (<XRoot>/lib/X11/rstart/commands)

(Yes, this means you can't have an all-contexts generic command with the same name as a context. It didn't seem like a big deal.)

Each of these directories should have a file called @List that gives the names and descriptions of the commands in that directory in the format specified for ListGenericCommands.

CONFIGURATION KEYWORDS

There are several "special" *rstart* keywords defined for *rstartd* configuration. Unless otherwise specified, there are no defaults; related features are disabled in this case.

INTERNAL-REGISTRIES name ...
Gives a space-separated list of "MISC" registries that this system understands. (Registries other than this are accepted but generate a Warning.)

INTERNAL-LOCAL-DEFAULT relative_filename
Gives the name ($HOME relative) of the per-user config file.

INTERNAL-GLOBAL-CONTEXTS absolute_directory_name
Gives the name of the system-wide contexts directory.

INTERNAL-LOCAL-CONTEXTS relative_directory_name
Gives the name ($HOME relative) of the per-user contexts directory.

INTERNAL-GLOBAL-COMMANDS absolute_directory_name
Gives the name of the system-wide generic commands directory.

INTERNAL-LOCAL-COMMANDS relative_directory_name
Gives the name ($HOME relative) of the per-user generic commands directory.

INTERNAL-VARIABLE-PREFIX prefix
Gives the prefix for the configuration environment variables *rstartd* passes to its kids.

INTERNAL-AUTH-PROGRAM authscheme program argv[0] argv[1] ...
Specifies the program to run to set up authentication for the specified authentication scheme. "program argv[0] ..." gives the program to run and its arguments, in the same form as the EXEC keyword.

INTERNAL-AUTH-INPUT authscheme
Specifies the data to be given to the authorization program as its standard input. Each argument is passed as a single line. $n, where n is a number, is replaced by the n'th argument to the "AUTH authscheme arg1 arg2 ..." line.

INTERNAL-PRINT arbitrary text
Prints its arguments as a Debug message. Mostly for *rstartd* debugging, but could be used to debug config files.

NOTES

When using the C shell, or any other shell which runs a script every time the shell is started, the script may get run several times. In the worst case, the script may get run **three** times:

By rsh, to run *rstartd*
By *rstartd*, to run the specified command
By the command, eg *xterm*

rstartd currently limits lines, both from config files and requests, to BUFSIZ bytes.

DETACH is implemented by redirecting file descriptors 0,1, and 2 to /dev/null and forking before executing the program.

CMD is implemented by invoking $SHELL (default /bin/sh) with "-c" and the specified command as arguments.

POSIX-UMASK is implemented in the obvious way.

The authorization programs are run in the same context as the target program - same environment variables, path, etc. Long term this might be a problem.

In the X context, GENERIC-CMD Terminal runs xterm. In the OpenWindows context, GENERIC-CMD Terminal runs cmdtool.

In the X context, GENERIC-CMD LoadMonitor runs xload. In the OpenWindows context, GENERIC-CMD LoadMonitor runs perfmeter.

GENERIC-CMD ListContexts lists the contents of @List in both the system-wide and per-user contexts directories. It is available in all contexts.

GENERIC-CMD ListGenericCommands lists the contents of @List in the system-wide and per-user commands directories, including the per-context subdirectories for the current context. It is available in all contexts.

CONTEXT None is not implemented.

CONTEXT Default is really dull.

For installation ease, the "contexts" directory in the distribution contains a file "@Aliases" which lists a context name and aliases for that context. This file is used to make symlinks in the contexts and commands directories.

All **MISC** values are passed unmodified as environment variables.

One can mistreat *rstartd* in any number of ways, resulting in anything from stupid behavior to core dumps. Other than by explicitly running programs I don't think it can write or delete any files, but there's no guarantee of that. The important thing is that (a) it probably won't do anything REALLY stupid and (b) it runs with the user's permissions, so it can't do anything catastrophic.

@List files need not be complete; contexts or commands which are dull or which need not or should not be advertised need not be listed. In particular, per-user @List files should not list things which are in the system-wide @List files. In the future, perhaps ListContexts and ListGenericCommands will automatically suppress lines from the system-wide files when there are per-user replacements for those lines.

Error handling is OK to weak. In particular, no attempt is made to properly report errors on the exec itself. (Perversely, exec errors could be reliably reported when detaching, but not when passing the stdin/out socket to the app.)

If compiled with -DODT1_DISPLAY_HACK, *rstartd* will work around a bug in SCO ODT version 1. (1.1?) (The bug is that the X clients are all compiled with a bad library that doesn't know how to look host names up using DNS. The fix is to look up a host name in $DISPLAY and substitute an IP address.) This is a trivial example of an incompatibility that *rstart* can hide.

SEE ALSO

rstart(1), rsh(1), A Flexible Remote Execution Protocol Based on **rsh**

AUTHOR

Jordan Brown, Quarterdeck Office Systems

NAME

xon – start an X program on a remote machine

SYNOPSIS

xon remote-host [-access] [-debug] [-name window-name] [-nols] [-screen screen-no] [-user user-name] [command ...]

DESCRIPTION

Xon runs the specified command (default xterm -ls) on the remote machine using rsh, remsh, or rcmd. Xon passes the DISPLAY, XAUTHORITY and XUSERFILESEARCHPATH environment variables to the remote command.

When no command is specified, xon runs 'xterm -ls'. It additionally specifies the application name to be 'xterm-*remote-host*' and the window title to be '-fIremote-host'.

Xon can only work when the remote host will allow you to log in without a password, by having an entry in the .rhosts file permitting access.

OPTIONS

Note that the options follow the remote host name (as they do with rlogin).

-access
Runs xhost locally to add the remote host to the host access list in the X server. This won't work unless xhost is given permission to modify the access list.

-debug
Normally, xon disconnects the remote process from stdin, stdout and stderr to eliminate the daemon processes which usually connect them across the network. Specifying the **-debug** option leaves them connected so that error messages from the remote execution are sent back to the originating host.

-name window-name
This specifies a different application name and window title for the default command (xterm).

-nols Normally xon passes the -ls option to the remote xterm; this option suspends that behaviour.

-screen screen-no
This changes the screen number of the DISPLAY variable passed to the remote command.

-user user-name
By default, xon simply uses rsh/remsh/rcmd to connect to the remote machine using the same user name as on the local machine. This option cause xon to specify an alternative user name. This will not work unless you have authorization to access the remote account, by placing an appropriate entry in the remote users .rhosts file.

BUGS

Xon can get easily confused when the remote-host, user-name or various environment variable values contain white space.

Xon has no way to send the appropriate X authorization information to the remote host.

NAME

smproxy − Session Manager Proxy

SYNOPSIS

smproxy [-clientId id] [-restore saveFile]

OPTIONS

−clientId *id*

Specifies the session ID used by *smproxy* in the previous session.

−restore *saveFile*

Specifies the file used by *smproxy* to save state in the previous session.

DESCRIPTION

smproxy allows X applications that do not support X11R6 session management to participate in an X11R6 session.

In order for *smproxy* to act as a proxy for an X application, one of the following must be true:

- The application maps a top level window containing the **WM_CLIENT_LEADER** property. This property provides a pointer to the client leader window which contains the **WM_CLASS**, **WM_NAME**, **WM_COMMAND**, and **WM_CLIENT_MACHINE** properties.

or ...

- The application maps a top level window which does not contain the **WM_CLIENT_LEADER** property. However, this top level window contains the **WM_CLASS**, **WM_NAME**, **WM_COMMAND**, and **WM_CLIENT_MACHINE** properties.

An application that support the **WM_SAVE_YOURSELF** protocol will receive a **WM_SAVE_YOURSELF** client message each time the session manager issues a checkpoint or shutdown. This allows the application to save state. If an application does not support the **WM_SAVE_YOURSELF** protocol, then the proxy will provide enough information to the session manager to restart the application (using **WM_COMMAND**), but no state will be restored.

SEE ALSO

xsm(1)

AUTHOR

Ralph Mor, X Consortium

NAME

twm – Tab Window Manager for the X Window System

SYNTAX

twm [**–display** *dpy*] [**–s**] [**–f** *initfile*] [**–v**]

DESCRIPTION

Twm is a window manager for the X Window System. It provides titlebars, shaped windows, several forms of icon management, user-defined macro functions, click-to-type and pointer-driven keyboard focus, and user-specified key and pointer button bindings.

This program is usually started by the user's session manager or startup script. When used from *xdm(1)* or *xinit(1)* without a session manager, *twm* is frequently executed in the foreground as the last client. When run this way, exiting *twm* causes the session to be terminated (i.e., logged out).

By default, application windows are surrounded by a ''frame'' with a titlebar at the top and a special border around the window. The titlebar contains the window's name, a rectangle that is lit when the window is receiving keyboard input, and function boxes known as ''titlebuttons'' at the left and right edges of the titlebar.

Pressing pointer Button1 (usually the left-most button unless it has been changed with *xmodmap*) on a titlebutton will invoke the function associated with the button. In the default interface, windows are iconified by clicking (pressing and then immediately releasing) the left titlebutton (which looks like a Dot). Conversely, windows are deiconified by clicking in the associated icon or entry in the icon manager (see description of the variable **ShowIconManager** and of the function **f.showiconmgr**).

Windows are resized by pressing the right titlebutton (which resembles a group of nested squares), dragging the pointer over edge that is to be moved, and releasing the pointer when the outline of the window is the desired size. Similarly, windows are moved by pressing in the title or highlight region, dragging a window outline to the new location, and then releasing when the outline is in the desired position. Just clicking in the title or highlight region raises the window without moving it.

When new windows are created, *twm* will honor any size and location information requested by the user (usually through *-geometry* command line argument or resources for the individual applications). Otherwise, an outline of the window's default size, its titlebar, and lines dividing the window into a 3x3 grid that track the pointer are displayed. Clicking pointer Button1 will position the window at the current position and give it the default size. Pressing pointer Button2 (usually the middle pointer button) and dragging the outline will give the window its current position but allow the sides to be resized as described above. Clicking pointer Button3 (usually the right pointer button) will give the window its current position but attempt to make it long enough to touch the bottom the screen.

OPTIONS

Twm accepts the following command line options:

–display *dpy*

This option specifies the X server to use.

–s This option indicates that only the default screen (as specified by **–display** or by the **DISPLAY** environment variable) should be managed. By default, *twm* will attempt to manage all screens on the display.

–f *filename*

This option specifies the name of the startup file to use. By default, *twm* will look in the user's home directory for files named *.twmrc.num* (where *num* is a screen number) or *.twmrc*.

−v This option indicates that *twm* should print error messages whenever an unexpected X Error event is received. This can be useful when debugging applications but can be distracting in regular use.

CUSTOMIZATION

Much of *twm*'s appearance and behavior can be controlled by providing a startup file in one of the following locations (searched in order for each screen being managed when *twm* begins):

$HOME/.twmrc.*screennumber*
The *screennumber* is a small positive number (e.g. 0, 1, etc.) representing the screen number (e.g. the last number in the DISPLAY environment variable *host:displaynum.screennum*) that would be used to contact that screen of the display. This is intended for displays with multiple screens of differing visual types.

$HOME/.twmrc
This is the usual name for an individual user's startup file.

<XRoot>/lib/X11/twm/system.twmrc
If neither of the preceding files are found, *twm* will look in this file for a default configuration. This is often tailored by the site administrator to provide convenient menus or familiar bindings for novice users. <XRoot> refers to the root of the X11 install tree.

If no startup files are found, *twm* will use the built-in defaults described above. The only resource used by *twm* is *bitmapFilePath* for a colon-separated list of directories to search when looking for bitmap files (for more information, see the *Athena Widgets* manual and *xrdb(1)*).

Twm startup files are logically broken up into three types of specifications: *Variables*, *Bindings*, *Menus*. The *Variables* section must come first and is used to describe the fonts, colors, cursors, border widths, icon and window placement, highlighting, autoraising, layout of titles, warping, use of the icon manager. The *Bindings* section usually comes second and is used to specify the functions that should be to be invoked when keyboard and pointer buttons are pressed in windows, icons, titles, and frames. The *Menus* section gives any user-defined menus (containing functions to be invoked or commands to be executed).

Variable names and keywords are case-insensitive. Strings must be surrounded by double quote characters (e.g. "blue") and are case-sensitive. A pound sign (#) outside of a string causes the remainder of the line in which the character appears to be treated as a comment.

VARIABLES

Many of the aspects of *twm*'s user interface are controlled by variables that may be set in the user's startup file. Some of the options are enabled or disabled simply by the presence of a particular keyword. Other options require keywords, numbers, strings, or lists of all of these.

Lists are surrounded by braces and are usually separated by whitespace or a newline. For example:

> **AutoRaise** { "emacs" "XTerm" "Xmh" }

or

```
AutoRaise
{
        "emacs"
        "XTerm"
        "Xmh"
}
```

When a variable containing a list of strings representing windows is searched (e.g. to determine whether or not to enable autoraise as shown above), a string must be an exact, case-sensitive match to the window's name (given by the WM_NAME window property), resource name or class name (both given by the WM_CLASS window property). The preceding example would enable autoraise on windows named "emacs" as well as any *xterm* (since they are of class "XTerm") or xmh windows (which are of class "Xmh").

String arguments that are interpreted as filenames (see the **Pixmaps**, **Cursors**, and **IconDirectory** below) will prepend the user's directory (specified by the **HOME** environment variable) if the first character is a tilde (~). If, instead, the first character is a colon (:), the name is assumed to refer to one of the internal bitmaps that are used to create the default titlebars symbols: **:xlogo** or **:delete** (both refer to the X logo), **:dot** or **:iconify** (both refer to the dot), **:resize** (the nested squares used by the resize button), **:menu** (a page with lines), and **:question** (the question mark used for non-existent bitmap files).

The following variables may be specified at the top of a *twm* startup file. Lists of Window name prefix strings are indicated by *win-list*. Optional arguments are shown in square brackets:

AutoRaise { *win-list* }
: This variable specifies a list of windows that should automatically be raised whenever the pointer enters the window. This action can be interactively enabled or disabled on individual windows using the function **f.autoraise**.

AutoRelativeResize
: This variable indicates that dragging out a window size (either when initially sizing the window with pointer Button2 or when resizing it) should not wait until the pointer has crossed the window edges. Instead, moving the pointer automatically causes the nearest edge or edges to move by the same amount. This allows the resizing of windows that extend off the edge of the screen. If the pointer is in the center of the window, or if the resize is begun by pressing a titlebutton, *twm* will still wait for the pointer to cross a window edge (to prevent accidents). This option is particularly useful for people who like the press-drag-release method of sweeping out window sizes.

BorderColor *string* [{ *wincolorlist* }]
: This variable specifies the default color of the border to be placed around all non-iconified windows, and may only be given within a **Color**, **Grayscale** or **Monochrome** list. The optional *wincolorlist* specifies a list of window and color name pairs for specifying particular border colors for different types of

windows. For example:

```
BorderColor "gray50"
{
        "XTerm"         "red"
        "xmh"   "green"
}
```

The default is "black".

BorderTileBackground *string* [{ *wincolorlist* }]
This variable specifies the default background color in the gray pattern used in unhighlighted borders (only if **NoHighlight** hasn't been set), and may only be given within a **Color**, **Grayscale** or **Monochrome** list. The optional *wincolorlist* allows per-window colors to be specified. The default is "white".

BorderTileForeground *string* [{ *wincolorlist* }]
This variable specifies the default foreground color in the gray pattern used in unhighlighted borders (only if **NoHighlight** hasn't been set), and may only be given within a **Color**, **Grayscale** or **Monochrome** list. The optional *wincolorlist* allows per-window colors to be specified. The default is "black".

BorderWidth *pixels*
This variable specifies the width in pixels of the border surrounding all client window frames if **ClientBorderWidth** has not been specified. This value is also used to set the border size of windows created by *twm* (such as the icon manager). The default is 2.

ButtonIndent *pixels*
This variable specifies the amount by which titlebuttons should be indented on all sides. Positive values cause the buttons to be smaller than the window text and highlight area so that they stand out. Setting this and the **TitleButtonBorderWidth** variables to 0 makes titlebuttons be as tall and wide as possible. The default is 1.

ClientBorderWidth
This variable indicates that border width of a window's frame should be set to the initial border width of the window, rather than to the value of **BorderWidth**.

Color { *colors-list* }
This variable specifies a list of color assignments to be made if the default display is capable of displaying more than simple black and white. The *colors-list* is made up of the following color variables and their values: **DefaultBackground**, **DefaultForeground**, **MenuBackground**, **MenuForeground**, **MenuTitleBackground**, **MenuTitleForeground**, **MenuShadowColor**, **PointerForeground**, and **PointerBackground**. The following color variables may also be given a list of window and color name pairs to allow per-window colors to be specified (see **BorderColor** for details): **BorderColor**, **IconManagerHighlight**, **BorderTitleBackground**, **BorderTitleForeground**, **TitleBackground**, **TitleForeground**, **IconBackground**, **IconForeground**, **IconBorderColor**, **IconManagerBackground**, and **IconManagerForeground**. For exam-

ple:

```
Color
{
        MenuBackground          "gray50"
        MenuForeground          "blue"
        BorderColor             "red" { "XTerm" "yellow" }
        TitleForeground         "yellow"
        TitleBackground         "blue"
}
```

All of these color variables may also be specified for the **Monochrome** variable, allowing the same initialization file to be used on both color and monochrome displays.

ConstrainedMoveTime *milliseconds*

This variable specifies the length of time between button clicks needed to begin a constrained move operation. Double clicking within this amount of time when invoking **f.move** will cause the window to be moved only in a horizontal or vertical direction. Setting this value to 0 will disable constrained moves. The default is 400 milliseconds.

Cursors { *cursor-list* }

This variable specifies the glyphs that *twm* should use for various pointer cursors. Each cursor may be defined either from the **cursor** font or from two bitmap files. Shapes from the **cursor** font may be specified directly as:

cursorname "*string*"

where *cursorname* is one of the cursor names listed below, and *string* is the name of a glyph as found in the file <XRoot>/include/X11/cursorfont.h (without the "XC_" prefix). If the cursor is to be defined from bitmap files, the following syntax is used instead:

cursorname "*image*" "*mask*"

The *image* and *mask* strings specify the names of files containing the glyph image and mask in *bitmap(1)* form. The bitmap files are located in the same manner as icon bitmap files. The following example shows the default cursor definitions:

```
Cursors
{
        Frame           "top_left_arrow"
        Title           "top_left_arrow"
        Icon            "top_left_arrow"
        IconMgr "top_left_arrow"
        Move            "fleur"
        Resize          "fleur"
        Menu            "sb_left_arrow"
        Button          "hand2"
        Wait            "watch"
        Select          "dot"
        Destroy "pirate"
}
```

DecorateTransients
This variable indicates that transient windows (those containing a WM_TRANSIENT_FOR property) should have titlebars. By default, transients are not reparented.

DefaultBackground *string*
This variable specifies the background color to be used for sizing and information windows. The default is "white".

DefaultForeground *string*
This variable specifies the foreground color to be used for sizing and information windows. The default is "black".

DontIconifyByUnmapping { *win-list* }
This variable specifies a list of windows that should not be iconified by simply unmapping the window (as would be the case if **IconifyByUnmapping** had been set). This is frequently used to force some windows to be treated as icons while other windows are handled by the icon manager.

DontMoveOff
This variable indicates that windows should not be allowed to be moved off the screen. It can be overridden by the **f.forcemove** function.

DontSqueezeTitle [{ *win-list* }]
This variable indicates that titlebars should not be squeezed to their minimum size as described under **SqueezeTitle** below. If the optional window list is supplied, only those windows will be prevented from being squeezed.

ForceIcons
This variable indicates that icon pixmaps specified in the **Icons** variable should override any client-supplied pixmaps.

FramePadding *pixels*
This variable specifies the distance between the titlebar decorations (the button and text) and the window frame. The default is 2 pixels.

Grayscale { *colors* }
This variable specifies a list of color assignments that should be made if the screen has a GrayScale default visual. See the description of **Colors**.

IconBackground *string* [{ *win-list* }]
This variable specifies the background color of icons, and may only be specified inside of a **Color**, **Grayscale** or **Monochrome** list. The optional *win-list* is a list of window names and colors so that per-window colors may be specified. See the **BorderColor** variable for a complete description of the *win-list*. The default is "white".

IconBorderColor *string* [{ *win-list* }]
This variable specifies the color of the border used for icon windows, and may only be specified inside of a **Color**, **Grayscale** or **Monochrome** list. The optional *win-list* is a list of window names and colors so that per-window colors may be specified. See the **BorderColor** variable for a complete description of the *win-list*. The default is "black".

IconBorderWidth *pixels*
This variable specifies the width in pixels of the border surrounding icon windows. The default is 2.

IconDirectory *string*
This variable specifies the directory that should be searched if if a bitmap file cannot be found in any of the directories in the **bitmapFilePath** resource.

IconFont *string*
This variable specifies the font to be used to display icon names within icons. The default is "variable".

IconForeground *string* [{ *win-list* }]
This variable specifies the foreground color to be used when displaying icons, and may only be specified inside of a **Color**, **Grayscale** or **Monochrome** list. The optional *win-list* is a list of window names and colors so that per-window colors may be specified. See the **BorderColor** variable for a complete description of the *win-list*. The default is "black".

IconifyByUnmapping [{ *win-list* **}]**
This variable indicates that windows should be iconified by being unmapped without trying to map any icons. This assumes that the user will remap the window through the icon manager, the **f.warpto** function, or the *TwmWindows* menu. If the optional *win-list* is provided, only those windows will be iconified by simply unmapping. Windows that have both this and the **IconManagerDontShow** options set may not be accessible if no binding to the *TwmWindows* menu is set in the user's startup file.

IconManagerBackground *string* [{ *win-list* }]
This variable specifies the background color to use for icon manager entries, and may only be specified inside of a **Color**, **Grayscale** or **Monochrome** list. The optional *win-list* is a list of window names and colors so that per-window colors may be specified. See the **BorderColor** variable for a complete description of the *win-list*. The default is "white".

IconManagerDontShow [{ *win-list* }]
This variable indicates that the icon manager should not display any windows. If the optional *win-list* is given, only those windows will not be displayed. This variable is used to prevent windows that are rarely iconified (such as *xclock* or *xload*) from taking up space in the icon manager.

IconManagerFont *string*
This variable specifies the font to be used when displaying icon manager entries. The default is "variable".

IconManagerForeground *string* [{ *win-list* }]
This variable specifies the foreground color to be used when displaying icon manager entries, and may only be specified inside of a **Color**, **Grayscale** or **Monochrome** list. The optional *win-list* is a list of window names and colors so that per-window colors may be specified. See the **BorderColor** variable for a complete description of the *win-list*. The default is "black".

IconManagerGeometry *string* [*columns*]
This variable specifies the geometry of the icon manager window. The *string* argument is standard geometry specification that indicates the initial full size of the icon manager. The icon manager window is then broken into *columns* pieces and scaled according to the number of entries in the icon manager. Extra entries are wrapped to form additional rows. The default number of columns is 1.

IconManagerHighlight *string* [{ *win-list* }]

This variable specifies the border color to be used when highlighting the icon manager entry that currently has the focus, and can only be specified inside of a **Color**, **Grayscale** or **Monochrome** list. The optional *win-list* is a list of window names and colors so that per-window colors may be specified. See the **BorderColor** variable for a complete description of the *win-list*. The default is "black".

IconManagers { *iconmgr-list* }

This variable specifies a list of icon managers to create. Each item in the *iconmgr-list* has the following format:

"*winname*" ["*iconname*"] "*geometry*" *columns*

where *winname* is the name of the windows that should be put into this icon manager, *iconname* is the name of that icon manager window's icon, *geometry* is a standard geometry specification, and *columns* is the number of columns in this icon manager as described in **IconManagerGeometry**. For example:

```
IconManagers
{
        "XTerm"         "=300x5+800+5" 5
        "myhost"        "=400x5+100+5" 2
}
```

Clients whose name or class is "XTerm" will have an entry created in the "XTerm" icon manager. Clients whose name was "myhost" would be put into the "myhost" icon manager.

IconManagerShow { *win-list* }

This variable specifies a list of windows that should appear in the icon manager. When used in conjunction with the **IconManagerDontShow** variable, only the windows in this list will be shown in the icon manager.

IconRegion *geomstring vgrav hgrav gridwidth gridheight*

This variable specifies an area on the root window in which icons are placed if no specific icon location is provided by the client. The *geomstring* is a quoted string containing a standard geometry specification. If more than one **IconRegion** lines are given, icons will be put into the succeeding icon regions when the first is full. The *vgrav* argument should be either **North** or **South** and control and is used to control whether icons are first filled in from the top or bottom of the icon region. Similarly, the *hgrav* argument should be either **East** or **West** and is used to control whether icons should be filled in from left from the right. Icons are laid out within the region in a grid with cells *gridwidth* pixels wide and *gridheight* pixels high.

Icons { *win-list* }

This variable specifies a list of window names and the bitmap filenames that should be used as their icons. For example:

```
Icons
{
        "XTerm"         "xterm.icon"
        "xfd"           "xfd_icon"
}
```

Windows that match "XTerm" and would not be iconified by unmapping, and would try to use the icon bitmap in the file “xterm.icon”. If **ForceIcons** is specified, this bitmap will be used even if the client has requested its own icon pixmap.

InterpolateMenuColors
This variable indicates that menu entry colors should be interpolated between entry specified colors. In the example below:

```
Menu "mymenu"
{
        "Title"         ("black":"red")         f.title
        "entry1"                        f.nop
        "entry2"                        f.nop
        "entry3" ("white":"green") f.nop
        "entry4"                        f.nop
        "entry5" ("red":"white")        f.nop
}
```

the foreground colors for “entry1” and “entry2” will be interpolated between black and white, and the background colors between red and green. Similarly, the foreground for “entry4” will be half-way between white and red, and the background will be half-way between green and white.

MakeTitle { *win-list* }
This variable specifies a list of windows on which a titlebar should be placed and is used to request titles on specific windows when **NoTitle** has been set.

MaxWindowSize *string*
This variable specifies a geometry in which the width and height give the maximum size for a given window. This is typically used to restrict windows to the size of the screen. The default width is 32767 - screen width. The default height is 32767 - screen height.

MenuBackground *string*
This variable specifies the background color used for menus, and can only be specified inside of a **Color** or **Monochrome** list. The default is "white".

MenuFont *string*
This variable specifies the font to use when displaying menus. The default is "variable".

MenuForeground *string*
This variable specifies the foreground color used for menus, and can only be specified inside of a **Color**, **Grayscale** or **Monochrome** list. The default is "black".

MenuShadowColor *string*
This variable specifies the color of the shadow behind pull-down menus and can only be specified inside of a **Color**, **Grayscale** or **Monochrome** list. The default is "black".

MenuTitleBackground *string*
This variable specifies the background color for **f.title** entries in menus, and can only be specified inside of a **Color**, **Grayscale** or **Monochrome** list. The default is "white".

MenuTitleForeground *string*
This variable specifies the foreground color for **f.title** entries in menus and can

only be specified inside of a **Color** or **Monochrome** list. The default is "black".

Monochrome { *colors* }
This variable specifies a list of color assignments that should be made if the screen has a depth of 1. See the description of **Colors**.

MoveDelta *pixels*
This variable specifies the number of pixels the pointer must move before the **f.move** function starts working. Also see the **f.deltastop** function. The default is zero pixels.

NoBackingStore
This variable indicates that *twm*'s menus should not request backing store to minimize repainting of menus. This is typically used with servers that can repaint faster than they can handle backing store.

NoCaseSensitive
This variable indicates that case should be ignored when sorting icon names in an icon manager. This option is typically used with applications that capitalize the first letter of their icon name.

NoDefaults
This variable indicates that *twm* should not supply the default titlebuttons and bindings. This option should only be used if the startup file contains a completely new set of bindings and definitions.

NoGrabServer
This variable indicates that *twm* should not grab the server when popping up menus and moving opaque windows.

NoHighlight [{ *win-list* }]
This variable indicates that borders should not be highlighted to track the location of the pointer. If the optional *win-list* is given, highlighting will only be disabled for those windows. When the border is highlighted, it will be drawn in the current **BorderColor**. When the border is not highlighted, it will be stippled with a gray pattern using the current **BorderTileForeground** and **BorderTileBackground** colors.

NoIconManagers
This variable indicates that no icon manager should be created.

NoMenuShadows
This variable indicates that menus should not have drop shadows drawn behind them. This is typically used with slower servers since it speeds up menu drawing at the expense of making the menu slightly harder to read.

NoRaiseOnDeiconify
This variable indicates that windows that are deiconified should not be raised.

NoRaiseOnMove
This variable indicates that windows should not be raised when moved. This is typically used to allow windows to slide underneath each other.

NoRaiseOnResize
This variable indicates that windows should not be raised when resized. This is typically used to allow windows to be resized underneath each other.

NoRaiseOnWarp
This variable indicates that windows should not be raised when the pointer is warped into them with the **f.warpto** function. If this option is set, warping to an occluded window may result in the pointer ending up in the occluding window instead the desired window (which causes unexpected behavior with **f.warpring**).

NoSaveUnders
This variable indicates that menus should not request save-unders to minimize window repainting following menu selection. It is typically used with displays that can repaint faster than they can handle save-unders.

NoStackMode [{ *win-list* }]
This variable indicates that client window requests to change stacking order should be ignored. If the optional *win-list* is given, only requests on those windows will be ignored. This is typically used to prevent applications from relentlessly popping themselves to the front of the window stack.

NoTitle [{ *win-list* }]
This variable indicates that windows should not have titlebars. If the optional *win-list* is given, only those windows will not have titlebars. **MakeTitle** may be used with this option to force titlebars to be put on specific windows.

NoTitleFocus
This variable indicates that *twm* should not set keyboard input focus to each window as it is entered. Normally, *twm* sets the focus so that focus and key events from the titlebar and icon managers are delivered to the application. If the pointer is moved quickly and *twm* is slow to respond, input can be directed to the old window instead of the new. This option is typically used to prevent this ‘‘input lag’’ and to work around bugs in older applications that have problems with focus events.

NoTitleHighlight [{ *win-list* }]
This variable indicates that the highlight area of the titlebar, which is used to indicate the window that currently has the input focus, should not be displayed. If the optional *win-list* is given, only those windows will not have highlight areas. This and the **SqueezeTitle** options can be set to substantially reduce the amount of screen space required by titlebars.

OpaqueMove
This variable indicates that the **f.move** function should actually move the window instead of just an outline so that the user can immediately see what the window will look like in the new position. This option is typically used on fast displays (particularly if **NoGrabServer** is set).

Pixmaps { *pixmaps* }
This variable specifies a list of pixmaps that define the appearance of various images. Each entry is a keyword indicating the pixmap to set, followed by a string giving the name of the bitmap file. The following pixmaps may be specified:

```
Pixmaps
{
        TitleHighlight      "gray1"
}
```

The default for *TitleHighlight* is to use an even stipple pattern.

Priority *priority*
This variable sets *twm*'s priority. *priority* should be an unquoted, signed number (e.g. 999). This variable has an effect only if the server supports the SYNC extension.

RandomPlacement
This variable indicates that windows with no specified geometry should be placed in a pseudo-random location instead of having the user drag out an outline.

ResizeFont *string*
This variable specifies the font to be used for in the dimensions window when resizing windows. The default is "fixed".

RestartPreviousState
This variable indicates that *twm* should attempt to use the WM_STATE property on client windows to tell which windows should be iconified and which should be left visible. This is typically used to try to regenerate the state that the screen was in before the previous window manager was shutdown.

SaveColor { *colors-list* }
This variable indicates a list of color assignments to be stored as pixel values in the root window property _MIT_PRIORITY_COLORS. Clients may elect to preserve these values when installing their own colormap. Note that use of this mechanism is a way an for application to avoid the "technicolor" problem, whereby useful screen objects such as window borders and titlebars disappear when a programs custom colors are installed by the window manager. For example:

```
SaveColor
{
    BorderColor
    TitleBackground
    TitleForeground
    "red"
    "green"
    "blue"
}
```

This would place on the root window 3 pixel values for borders and titlebars, as well as the three color strings, all taken from the default colormap.

ShowIconManager
This variable indicates that the icon manager window should be displayed when *twm* is started. It can always be brought up using the **f.showiconmgr** function.

SortIconManager
This variable indicates that entries in the icon manager should be sorted alphabetically rather than by simply appending new windows to the end.

SqueezeTitle [{ *squeeze-list* }]
This variable indicates that *twm* should attempt to use the SHAPE extension to make titlebars occupy only as much screen space as they need, rather than extending all the way across the top of the window. The optional *squeeze-list* may be used to control the location of the squeezed titlebar along the top of the

window. It contains entries of the form:

"*name*" *justification* *num* *denom*

where *name* is a window name, *justification* is either **left**, **center**, or **right**, and *num* and *denom* are numbers specifying a ratio giving the relative position about which the titlebar is justified. The ratio is measured from left to right if the numerator is positive, and right to left if negative. A denominator of 0 indicates that the numerator should be measured in pixels. For convenience, the ratio 0/0 is the same as 1/2 for **center** and -1/1 for **right**. For example:

```
SqueezeTitle
{
        "XTerm"     left        0    0
        "xterm1"    left        1    3
        "xterm2"    left        2    3
        "oclock" center     0   0
        "emacs" right       0   0
}
```

The **DontSqueezeTitle** list can be used to turn off squeezing on certain titles.

StartIconified [{ *win-list* }]
This variable indicates that client windows should initially be left as icons until explicitly deiconified by the user. If the optional *win-list* is given, only those windows will be started iconic. This is useful for programs that do not support an *-iconic* command line option or resource.

TitleBackground *string* [{ *win-list* }]
This variable specifies the background color used in titlebars, and may only be specified inside of a **Color**, **Grayscale** or **Monochrome** list. The optional *win-list* is a list of window names and colors so that per-window colors may be specified. The default is "white".

TitleButtonBorderWidth *pixels*
This variable specifies the width in pixels of the border surrounding titlebuttons. This is typically set to 0 to allow titlebuttons to take up as much space as possible and to not have a border. The default is 1.

TitleFont *string*
This variable specifies the font to be used for displaying window names in titlebars. The default is "variable".

TitleForeground *string* [{ *win-list* }]
This variable specifies the foreground color used in titlebars, and may only be specified inside of a **Color**, **Grayscale** or **Monochrome** list. The optional *win-list* is a list of window names and colors so that per-window colors may be specified. The default is "black".

TitlePadding *pixels*
This variable specifies the distance between the various buttons, text, and highlight areas in the titlebar. The default is 8 pixels.

UnknownIcon *string*
This variable specifies the filename of a bitmap file to be used as the default icon. This bitmap will be used as the icon of all clients which do not provide an icon bitmap and are not listed in the **Icons** list.

UsePPosition *string*

This variable specifies whether or not *twm* should honor program-requested locations (given by the **PPosition** flag in the WM_NORMAL_HINTS property) in the absence of a user-specified position. The argument *string* may have one of three values: **"off"** (the default) indicating that *twm* should ignore the program-supplied position, **"on"** indicating that the position should be used, and **"non-zero"** indicating that the position should used if it is other than (0,0). The latter option is for working around a bug in older toolkits.

WarpCursor [{ *win-list* }]

This variable indicates that the pointer should be warped into windows when they are deiconified. If the optional *win-list* is given, the pointer will only be warped when those windows are deiconified.

WindowRing { *win-list* }

This variable specifies a list of windows along which the **f.warpring** function cycles.

WarpUnmapped

This variable indicates that the **f.warpto** function should deiconify any iconified windows it encounters. This is typically used to make a key binding that will pop a particular window (such as *xmh*), no matter where it is. The default is for **f.warpto** to ignore iconified windows.

XorValue *number*

This variable specifies the value to use when drawing window outlines for moving and resizing. This should be set to a value that will result in a variety of of distinguishable colors when exclusive-or'ed with the contents of the user's typical screen. Setting this variable to 1 often gives nice results if adjacent colors in the default colormap are distinct. By default, *twm* will attempt to cause temporary lines to appear at the opposite end of the colormap from the graphics.

Zoom [*count*]

This variable indicates that outlines suggesting movement of a window to and from its iconified state should be displayed whenever a window is iconified or deiconified. The optional *count* argument specifies the number of outlines to be drawn. The default count is 8.

The following variables must be set after the fonts have been assigned, so it is usually best to put them at the end of the variables or beginning of the bindings sections:

DefaultFunction *function*

This variable specifies the function to be executed when a key or button event is received for which no binding is provided. This is typically bound to **f.nop**, **f.beep**, or a menu containing window operations.

WindowFunction *function*

This variable specifies the function to execute when a window is selected from the **TwmWindows** menu. If this variable is not set, the window will be deiconified and raised.

BINDINGS

After the desired variables have been set, functions may be attached titlebuttons and key and pointer buttons. Titlebuttons may be added from the left or right side and appear in the titlebar from left-to-right according to the order in which they are specified. Key and pointer button bindings may be given in any order.

Titlebuttons specifications must include the name of the pixmap to use in the button box and the function to be invoked when a pointer button is pressed within them:

LeftTitleButton "*bitmapname*" = *function*

or

RightTitleButton "*bitmapname*" = *function*

The *bitmapname* may refer to one of the built-in bitmaps (which are scaled to match **TitleFont**) by using the appropriate colon-prefixed name described above.

Key and pointer button specifications must give the modifiers that must be pressed, over which parts of the screen the pointer must be, and what function is to be invoked. Keys are given as strings containing the appropriate keysym name; buttons are given as the keywords **Button1**-**Button5**:

"FP1" = *modlist* : *context* : *function*
Button1 = *modlist* : *context* : *function*

The *modlist* is any combination of the modifier names **shift**, **control**, **lock**, **meta**, **mod1**, **mod2**, **mod3**, **mod4**, or **mod5** (which may be abbreviated as **s**, **c**, **l**, **m**, **m1**, **m2**, **m3**, **m4**, **m5**, respectively) separated by a vertical bar (|). Similarly, the *context* is any combination of **window**, **title**, **icon**, **root**, **frame**, **iconmgr**, their first letters (**iconmgr** abbreviation is **m**), or **all**, separated by a vertical bar. The *function* is any of the **f.** keywords described below. For example, the default startup file contains the following bindings:

```
Button1 =      : root           : f.menu "TwmWindows"
Button1 = m    : window | icon  : f.function "move-or-lower"
Button2 = m    : window | icon  : f.iconify
Button3 = m    : window | icon  : f.function "move-or-raise"
Button1 =      : title          : f.function "move-or-raise"
Button2 =      : title          : f.raiselower
Button1 =      : icon           : f.function "move-or-iconify"
Button2 =      : icon           : f.iconify
Button1 =      : iconmgr        : f.iconify
Button2 =      : iconmgr        : f.iconify
```

A user who wanted to be able to manipulate windows from the keyboard could use the following bindings:

```
"F1"     =       : all    : f.iconify
"F2"     =       : all    : f.raiselower
"F3"     =       : all    : f.warpring "next"
"F4"     =       : all    : f.warpto "xmh"
"F5"     =       : all    : f.warpto "emacs"
"F6"     =       : all    : f.colormap "next"
"F7"     =       : all    : f.colormap "default"
"F20"    =       : all    : f.warptoscreen "next"
"Left"   = m     : all    : f.backiconmgr
"Right"  = m | s : all    : f.forwiconmgr
"Up"     = m     : all    : f.upiconmgr
"Down"   = m | s : all    : f.downiconmgr
```

Twm provides many more window manipulation primitives than can be conveniently stored in a titlebar, menu, or set of key bindings. Although a small set of defaults are supplied (unless the **NoDefaults** is specified), most users will want to have their most common operations bound to key and button strokes. To do this, *twm* associates names with each of the primitives and provides *user-defined functions* for building higher level primitives and *menus* for interactively selecting among groups of functions.

User-defined functions contain the name by which they are referenced in calls to **f.function** and a list of other functions to execute. For example:

```
Function "move-or-lower" { f.move f.deltastop f.lower }
Function "move-or-raise" { f.move f.deltastop f.raise }
Function "move-or-iconify"        { f.move f.deltastop f.iconify }
Function "restore-colormap"       { f.colormap "default" f.lower }
```

The function name must be used in **f.function** exactly as it appears in the function specification.

In the descriptions below, if the function is said to operate on the selected window, but is invoked from a root menu, the cursor will be changed to the **Select** cursor and the next window to receive a button press will be chosen:

! *string* This is an abbreviation for **f.exec** *string*.

f.autoraise
This function toggles whether or not the selected window is raised whenever entered by the pointer. See the description of the variable **AutoRaise**.

f.backiconmgr
This function warps the pointer to the previous column in the current icon manager, wrapping back to the previous row if necessary.

f.beep This function sounds the keyboard bell.

f.bottomzoom
This function is similar to the **f.fullzoom** function, but resizes the window to fill only the bottom half of the screen.

f.circledown
This function lowers the top-most window that occludes another window.

f.circleup
This function raises the bottom-most window that is occluded by another window.

f.colormap *string*
This function rotates the colormaps (obtained from the WM_COLORMAP_WINDOWS property on the window) that *twm* will display when the pointer is in this window. The argument *string* may have one of the following values: **"next"**, **"prev"**, and **"default"**. It should be noted here that in general, the installed colormap is determined by keyboard focus. A pointer driven keyboard focus will install a private colormap upon entry of the window owning the colormap. Using the click to type model, private colormaps will not be installed until the user presses a mouse button on the target window.

f.deiconify
This function deiconifies the selected window. If the window is not an icon, this function does nothing.

f.delete This function sends the WM_DELETE_WINDOW message to the selected window if the client application has requested it through the WM_PROTOCOLS window property. The application is supposed to respond to the message by removing the indicated window. If the window has not requested WM_DELETE_WINDOW messages, the keyboard bell will be rung indicating that the user should choose an alternative method. Note this is very different from f.destroy. The intent here is to delete a single window, not necessarily the entire application.

f.deltastop
This function allows a user-defined function to be aborted if the pointer has been moved more than *MoveDelta* pixels. See the example definition given for **Function "move-or-raise"** at the beginning of the section.

f.destroy This function instructs the X server to close the display connection of the client that created the selected window. This should only be used as a last resort for shutting down runaway clients. See also f.delete.

f.downiconmgr
This function warps the pointer to the next row in the current icon manger, wrapping to the beginning of the next column if necessary.

f.exec *string*
This function passes the argument *string* to /bin/sh for execution. In multi-screen mode, if *string* starts a new X client without giving a display argument, the client will appear on the screen from which this function was invoked.

f.focus This function toggles the keyboard focus of the server to the selected window, changing the focus rule from pointer-driven if necessary. If the selected window already was focused, this function executes an **f.unfocus**.

f.forcemove
This function is like **f.move** except that it ignores the **DontMoveOff** variable.

f.forwiconmgr
This function warps the pointer to the next column in the current icon manager, wrapping to the beginning of the next row if necessary.

f.fullzoom
This function resizes the selected window to the full size of the display or else restores the original size if the window was already zoomed.

f.function *string*
This function executes the user-defined function whose name is specified by the argument *string*.

f.hbzoom
This function is a synonym for **f.bottomzoom**.

f.hideiconmgr
This function unmaps the current icon manager.

f.horizoom
This variable is similar to the **f.zoom** function except that the selected window is resized to the full width of the display.

f.htzoom This function is a synonym for **f.topzoom**.

f.hzoom This function is a synonym for **f.horizoom**.

f.iconify This function iconifies or deiconifies the selected window or icon, respectively.

f.identify
This function displays a summary of the name and geometry of the selected

window. If the server supports the SYNC extension, the priority of the client owning the window is also displayed. Clicking the pointer or pressing a key in the window will dismiss it.

f.lefticonmgr
This function similar to **f.backiconmgr** except that wrapping does not change rows.

f.leftzoom
This variable is similar to the **f.bottomzoom** function but causes the selected window is only resized to the left half of the display.

f.lower This function lowers the selected window.

f.menu *string*
This function invokes the menu specified by the argument *string*. Cascaded menus may be built by nesting calls to **f.menu**.

f.move This function drags an outline of the selected window (or the window itself if the **OpaqueMove** variable is set) until the invoking pointer button is released. Double clicking within the number of milliseconds given by **ConstrainedMoveTime** warps the pointer to the center of the window and constrains the move to be either horizontal or vertical depending on which grid line is crossed. To abort a move, press another button before releasing the first button.

f.nexticonmgr
This function warps the pointer to the next icon manager containing any windows on the current or any succeeding screen.

f.nop This function does nothing and is typically used with the **DefaultFunction** or **WindowFunction** variables or to introduce blank lines in menus.

f.previconmgr
This function warps the pointer to the previous icon manager containing any windows on the current or preceding screens.

f.priority *string*
This function sets the priority of the client owning the selected window to the numeric value of the argument *string*, which should be a signed integer in double quotes (e.g. "999"). This function has an effect only if the server supports the SYNC extension.

f.quit This function causes *twm* to restore the window's borders and exit. If *twm* is the first client invoked from *xdm*, this will result in a server reset.

f.raise This function raises the selected window.

f.raiselower
This function raises the selected window to the top of the stacking order if it is occluded by any windows, otherwise the window will be lowered.

f.refresh This function causes all windows to be refreshed.

f.resize This function displays an outline of the selected window. Crossing a border (or setting **AutoRelativeResize**) will cause the outline to begin to rubber band until the invoking button is released. To abort a resize, press another button before releasing the first button.

f.restart This function kills and restarts *twm*.

f.righticonmgr
This function is similar to **f.nexticonmgr** except that wrapping does not change rows.

f.rightzoom
This variable is similar to the **f.bottomzoom** function except that the selected window is only resized to the right half of the display.

f.saveyourself
This function sends a WM_SAVEYOURSELF message to the selected window if it has requested the message in its WM_PROTOCOLS window property. Clients that accept this message are supposed to checkpoint all state associated with the window and update the WM_COMMAND property as specified in the ICCCM. If the selected window has not selected for this message, the keyboard bell will be rung.

f.showiconmgr
This function maps the current icon manager.

f.sorticonmgr
This function sorts the entries in the current icon manager alphabetically. See the variable **SortIconManager**.

f.title This function provides a centered, unselectable item in a menu definition. It should not be used in any other context.

f.topzoom
This variable is similar to the **f.bottomzoom** function except that the selected window is only resized to the top half of the display.

f.unfocus
This function resets the focus back to pointer-driven. This should be used when a focused window is no longer desired.

f.upiconmgr
This function warps the pointer to the previous row in the current icon manager, wrapping to the last row in the same column if necessary.

f.vlzoom This function is a synonym for **f.leftzoom**.

f.vrzoom
This function is a synonym for **f.rightzoom**.

f.warpring *string*
This function warps the pointer to the next or previous window (as indicated by the argument *string*, which may be **"next"** or **"prev"**) specified in the **WindowRing** variable.

f.warpto *string*
This function warps the pointer to the window which has a name or class that matches *string*. If the window is iconified, it will be deiconified if the variable **WarpUnmapped** is set or else ignored.

f.warptoiconmgr *string*
This function warps the pointer to the icon manager entry associated with the window containing the pointer in the icon manager specified by the argument *string*. If *string* is empty (i.e. ""), the current icon manager is chosen.

f.warptoscreen *string*
This function warps the pointer to the screen specified by the argument *string*.

String may be a number (e.g. **"0"** or **"1"**), the word **"next"** (indicating the current screen plus 1, skipping over any unmanaged screens), the word **"back"** (indicating the current screen minus 1, skipping over any unmanaged screens), or the word **"prev"** (indicating the last screen visited.

f.winrefresh
This function is similar to the **f.refresh** function except that only the selected window is refreshed.

f.zoom This function is similar to the **f.fullzoom** function, except that the only the height of the selected window is changed.

MENUS

Functions may be grouped and interactively selected using pop-up (when bound to a pointer button) or pull-down (when associated with a titlebutton) menus. Each menu specification contains the name of the menu as it will be referred to by **f.menu**, optional default foreground and background colors, the list of item names and the functions they should invoke, and optional foreground and background colors for individual items:

```
Menu "menuname" [ ("deffore":"defback") ]
{
        string1   [ ("fore1":"backn")]        function1
        string2   [ ("fore2":"backn")]        function2
                  .
                  .
                  .
        stringN   [ ("foreN":"backN")]        functionN
}
```

The *menuname* is case-sensitive. The optional *deffore* and *defback* arguments specify the foreground and background colors used on a color display to highlight menu entries. The *string* portion of each menu entry will be the text which will appear in the menu. The optional *fore* and *back* arguments specify the foreground and background colors of the menu entry when the pointer is not in the entry. These colors will only be used on a color display. The default is to use the colors specified by the **MenuForeground** and **MenuBackground** variables. The *function* portion of the menu entry is one of the functions, including any user-defined functions, or additional menus.

There is a special menu named **TwmWindows** which contains the names of all of the client and *twm*-supplied windows. Selecting an entry will cause the **WindowFunction** to be executed on that window. If **WindowFunction** hasn't been set, the window will be deiconified and raised.

ICONS

Twm supports several different ways of manipulating iconified windows. The common pixmap-and-text style may be laid out by hand or automatically arranged as described by the **IconRegion** variable. In addition, a terse grid of icon names, called an icon manager, provides a more efficient use of screen space as well as the ability to navigate among windows from the keyboard.

An icon manager is a window that contains names of selected or all windows currently on the display. In addition to the window name, a small button using the default iconify symbol will be displayed to the left of the name when the window is iconified. By default, clicking on an entry in the icon manager performs **f.iconify**. To change the actions taken in the icon manager, use the the **iconmgr** context when specifying button and keyboard bindings.

Moving the pointer into the icon manager also directs keyboard focus to the indicated window (setting the focus explicitly or else sending synthetic events **NoTitleFocus** is set). Using the **f.upiconmgr**, **f.downiconmgr f.lefticonmgr**, and **f.righticonmgr** functions, the input focus can be changed between windows directly from the keyboard.

BUGS

The resource manager should have been used instead of all of the window lists.

The **IconRegion** variable should take a list.

Double clicking very fast to get the constrained move function will sometimes cause the window to move, even though the pointer is not moved.

If **IconifyByUnmapping** is on and windows are listed in **IconManagerDontShow** but not in **DontIconifyByUnmapping**, they may be lost if they are iconified and no bindings to **f.menu "TwmWindows"** or **f.warpto** are setup.

FILES

$HOME/.twmrc.<screen number>
$HOME/.twmrc
<XRoot>/lib/X11/twm/system.twmrc

ENVIRONMENT VARIABLES

DISPLAY
: This variable is used to determine which X server to use. It is also set during **f.exec** so that programs come up on the proper screen.

HOME
: This variable is used as the prefix for files that begin with a tilde and for locating the *twm* startup file.

SEE ALSO

X(1), Xserver(1), xdm(1), xrdb(1)

AUTHORS

Tom LaStrange, Solbourne Computer; Jim Fulton, MIT X Consortium; Steve Pitschke, Stardent Computer; Keith Packard, MIT X Consortium; Dave Sternlicht, MIT X Consortium; Dave Payne, Apple Computer.

NAME

x11perfcomp – X11 server performance comparison program

SYNTAX

x11perfcomp [–r | –ro] [–l label_file] files

DESCRIPTION

The *x11perfcomp* program merges the output of several *x11perf(1)* runs into a nice tabular format. It takes the results in each file, fills in any missing test results if necessary, and for each test shows the objects/second rate of each server. If invoked with the -r or -ro options, it shows the relative performance of each server to the first server.

Normally, *x11perfcomp* uses the first file specified to determine which specific tests it should report on. Some (non-DEC :) servers may fail to perform all tests. In this case, *x11perfcomp* automatically substitutes in a rate of 0.0 objects/second. Since the first file determines which tests to report on, this file must contain a superset of the tests reported in the other files, else *x11perfcomp* will fail.

You can provide an explicit list of tests to report on by using the -l switch to specify a file of labels. You can create a label file by using the -label option in *x11perf.*

OPTIONS

x11perfcomp accepts the options listed below:

–r Specifies that the output should also include relative server performance.

–ro Specifies that the output should include only relative server performance.

–l label_file Specifies a label file to use.

X DEFAULTS

There are no X defaults used by this program.

SEE ALSO

X(1), x11perf(1)

AUTHORS

Mark Moraes wrote the original scripts to compare servers.
Joel McCormack just munged them together a bit.

NAME

x11perf − X11 server performance test program

SYNTAX

x11perf [−option ...]

DESCRIPTION

The *x11perf* program runs one or more performance tests and reports how fast an X server can execute the tests.

Many graphics benchmarks assume that the graphics device is used to display the output of a single fancy graphics application, and that the user gets his work done on some other device, like a terminal. Such benchmarks usually measure drawing speed for lines, polygons, text, etc.

Since workstations are not used as standalone graphics engines, but as super-terminals, *x11perf* measures window management performance as well as traditional graphics performace. *x11perf* includes benchmarks for the time it takes to create and map windows (as when you start up an application); to map a pre-existing set of windows onto the screen (as when you deiconify an application or pop up a menu); and to rearrange windows (as when you slosh windows to and fro trying to find the one you want).

x11perf also measures graphics performance for operations not normally used in standalone graphics displays, but are nonetheless used frequently by X applications. Such operations include CopyPlane (used to map bitmaps into pixels), scrolling (used in text windows), and various stipples and tiles (used for CAD and color half-toning, respectively).

x11perf should be used to analyze particular strengths and weaknesses of servers, and is most useful to a server writer who wants to analyze and improve a server. *x11perf* is meant to comprehensively exercise just about every X11 operation you can perform; it does not purport to be a representative sample of the operations that X11 applications actually use. While it can be used as a benchmark, it was written and is intended as a performance testing tool.

As such, *x11perf* DOES NOT whittle down measurements to a single ‘‘HeXStones’’ or ‘‘MeXops’’ number. We consider such numbers to be uninformative at best and misleading at worst. Some servers which are very fast for certain applications can be very slow for others. No single number or small set of numbers are sufficient to characterize how an X implementation will perform on all applications. However, by knowledge of your favorite application, you may be able to use the numbers *x11perf* reports to predict its performance on a given X implementation.

That said, you might also want to look at *x11perfcomp(1),* a program to compare the outputs of different *x11perf* runs. You provide a list of files containing results from *x11perf,* and it lays them out in a nice tabular format.

For repeatable results, *x11perf* should be run using a local connection on a freshly-started server. The default configuration runs each test 5 times, in order to see if each trial takes approximately the same amount of time. Strange glitches should be examined; if nonrepeatable one might chalk them up to daemons and network traffic. Each trial is run for 5 seconds, in order to reduce random time differences. The number of objects processed per second is displayed to 3 significant digits, but you’ll be lucky on most UNIX system if the numbers are actually consistent to 2 digits. *x11perf* moves the cursor out of the test window; you should be careful not to bump the mouse and move it back into the window. (A prize to people who correctly explain why!!).

Before running a test, *x11perf* determines what the round trip time to the server is, and factors this out of the final timing reported. It ensures that the server has actually performed the work requested by fetching a pixel back from the test window, which means that servers talking to graphics accelerators can't claim that they are done, while in the meantime the accelerator is painting madly.

By default *x11perf* automatically calibrates the number of repetitions of each test, so that each should take approximately the same length of time to run across servers of widely differing speeds. However, since each test must be run to completion at least once, some slow servers may take a very long time, particularly on the window moving and resizing tests, and on the arc drawing tests.

All timing reports are for the smallest object involved. For example, the line tests use a PolyLine request to paint several lines at once, but report how many lines per second the server can paint, not how many PolyLine requests per second. Text tests paint a line of characters, but report on the number of characters per second. Some window tests map, unmap, or move a single parent window, but report on how many children windows per second the server can map, unmap, or move.

The current program is mostly the responsibility of Joel McCormack. It is based upon the x11perf developed by Phil Karlton, Susan Angebranndt, Chris Kent, Mary Walker, and Todd Newman, who wanted to assess performance differences between various servers. Several tests were added in order to write and tune the PMAX (DECStation 3100) servers. For a general release to the world, *x11perf* was rewritten to ease making comparisons between widely varying machines, to cover most important (and unimportant) X functionality, and to exercise graphics operations in as many different orientations and alignments as possible.

OPTIONS

x11perf is solely Xlib based, and accepts the options listed below:

−display host:dpy
Specifies which display to use.

−sync Runs the tests in synchronous mode. Normally only useful for debugging *x11perf* .

−pack Runs rectangle tests so that they pack rectangles right next to each other. This makes it easy to debug server code for stipples and tiles...if the pattern looks ugly, you've got alignment problems.

−repeat <*n*> Repeats each test *n* times (by default each test is run 5 times).

−time <*s*> Specifies how long in seconds each test should be run (default 5 seconds).

−all Runs all tests. This may take a while.

−range <*test1*>**[,**<*test2*>**]**
Runs all the tests starting from the specified name *test1* until the name *test2*, including both the specified tests. The testnames should be one of the options starting from -dot. (eg) -range line100 will peform the tests from the 100 pixel line test, and go on till the last test, -range line100,dline10 will do the tests from line100 to dline10.

−labels Generates just the descriptive labels for each test specified. See *x11perfcomp* for more details.

–fg *color-or-pixel*
Specifies the foreground color or pixel value to use.

–bg *color-or-pixel*
Specifies the background color or pixel value to use.

–clips *default* Default number of clip windows.

–ddbg *color-or-pixel*
Specifies the color or pixel value to use for drawing the odd segments of a DoubleDashed line or arc. This will default to the bg color.

–rop <rop0 rop1 ...>
Use specified raster ops (default is GXcopy). This option only affects graphics benchmarks in which the graphics function is actually used.

–pm <pm0 pm1 ...>
Use specified planemasks (default is ˜0). This option only affects graphics benchmarks in which the planemask is actually used.

–depth <depth>
Use a visual with <depth> planes per pixel (default is the default visual).

–vclass <vclass>
Use a visual with of class <vclass>. <vclass> can be StaticGray, GrayScale, StaticColor, PseudoColor, TrueColor, or DirectColor. (default is the default visual).

–reps <n> Specify the repetion count (Default is number that takes aprox. 5 seconds)

–subs <s0 s1 ...>
Specify the number of sub windows to use in the Window tests. Default is 4, 16, 25, 50, 75, 100 and 200.

–v1.2 Perform only x11perf Version 1.2 tests using Version 1.2 semantics.

–v1.3 Perform only x11perf Version 1.3 tests using Version 1.3 semantics.

–su Set the save_under window attribute to True on all windows created by x11perf. Default is False.

–bs <backing_store_hint>
Set the backing_store window attribute to the given value on all windows created by x11perf. <backing_store_hint> can be WhenMapped or Always. Default is NotUseful.

–dot Dot.

–rect1 1x1 solid-filled rectangle.

–rect10 10x10 solid-filled rectangle.

–rect100 100x100 solid-filled rectangle.

–rect500 500x500 solid-filled rectangle.

–srect1 1x1 transparent stippled rectangle, 8x8 stipple pattern.

–srect10 10x10 transparent stippled rectangle, 8x8 stipple pattern.

–srect100 100x100 transparent stippled rectangle, 8x8 stipple pattern.

−srect500 500x500 transparent stippled rectangle, 8x8 stipple pattern.
−osrect1 1x1 opaque stippled rectangle, 8x8 stipple pattern.
−osrect10 10x10 opaque stippled rectangle, 8x8 stipple pattern.
−osrect100 100x100 opaque stippled rectangle, 8x8 stipple pattern.
−osrect500 500x500 opaque stippled rectangle, 8x8 stipple pattern.
−tilerect1 1x1 tiled rectangle, 4x4 tile pattern.
−tilerect10 10x10 tiled rectangle, 4x4 tile pattern.
−tilerect100 100x100 tiled rectangle, 4x4 tile pattern.
−tilerect500 500x500 tiled rectangle, 4x4 tile pattern.
−oddsrect1 1x1 transparent stippled rectangle, 17x15 stipple pattern.
−oddsrect10 10x10 transparent stippled rectangle, 17x15 stipple pattern.
−oddsrect100 100x100 transparent stippled rectangle, 17x15 stipple pattern.
−oddsrect500 500x500 transparent stippled rectangle, 17x15 stipple pattern.
−oddosrect1 1x1 opaque stippled rectangle, 17x15 stipple pattern.
−oddosrect10 10x10 opaque stippled rectangle, 17x15 stipple pattern.
−oddosrect100 100x100 opaque stippled rectangle, 17x15 stipple pattern.
−oddosrect500 500x500 opaque stippled rectangle, 17x15 stipple pattern.
−oddtilerect1 1x1 tiled rectangle, 17x15 tile pattern.
−oddtilerect10 10x10 tiled rectangle, 17x15 tile pattern.
−oddtilerect100 100x100 tiled rectangle, 17x15 tile pattern.
−oddtilerect500 500x500 tiled rectangle, 17x15 tile pattern.
−bigsrect1 1x1 stippled rectangle, 161x145 stipple pattern.
−bigsrect10 10x10 stippled rectangle, 161x145 stipple pattern.
−bigsrect100 100x100 stippled rectangle, 161x145 stipple pattern.
−bigsrect500 500x500 stippled rectangle, 161x145 stipple pattern.
−bigosrect1 1x1 opaque stippled rectangle, 161x145 stipple pattern.
−bigosrect10 10x10 opaque stippled rectangle, 161x145 stipple pattern.
−bigosrect100 100x100 opaque stippled rectangle, 161x145 stipple pattern.
−bigosrect500 500x500 opaque stippled rectangle, 161x145 stipple pattern.
−bigtilerect1 1x1 tiled rectangle, 161x145 tile pattern.
−bigtilerect10 10x10 tiled rectangle, 161x145 tile pattern.
−bigtilerect100 100x100 tiled rectangle, 161x145 tile pattern.
−bigtilerect500 500x500 tiled rectangle, 161x145 tile pattern.
−eschertilerect1
1x1 tiled rectangle, 215x208 tile pattern.
−eschertilerect10
10x10 tiled rectangle, 215x208 tile pattern.

–eschertilerect100
100x100 tiled rectangle, 215x208 tile pattern.

–eschertilerect500
500x500 tiled rectangle, 215x208 tile pattern.

–seg1 1-pixel thin line segment.

–seg10 10-pixel thin line segment.

–seg100 100-pixel thin line segment.

–seg500 500-pixel thin line segment.

–seg100c1 100-pixel thin line segment (1 obscuring rectangle).

–seg100c2 100-pixel thin line segment (2 obscuring rectangles).

–seg100c3 100-pixel thin line segment (3 obscuring rectangles).

–dseg10 10-pixel thin dashed segment (3 on, 2 off).

–dseg100 100-pixel thin dashed segment (3 on, 2 off).

–ddseg100 100-pixel thin double-dashed segment (3 fg, 2 bg).

–hseg10 10-pixel thin horizontal line segment.

–hseg100 100-pixel thin horizontal line segment.

–hseg500 500-pixel thin horizontal line segment.

–vseg10 10-pixel thin vertical line segment.

–vseg100 100-pixel thin vertical line segment.

–vseg500 500-pixel thin vertical line segment.

–whseg10 10-pixel wide horizontal line segment.

–whseg100 100-pixel wide horizontal line segment.

–whseg500 500-pixel wide horizontal line segment.

–wvseg10 10-pixel wide vertical line segment.

–wvseg100 100-pixel wide vertical line segment.

–wvseg500 500-pixel wide vertical line segment.

–line1 1-pixel thin (width 0) line.

–line10 10-pixel thin line.

–line100 100-pixel thin line.

–line500 500-pixel thin line.

–dline10 10-pixel thin dashed line (3 on, 2 off).

–dline100 100-pixel thin dashed line (3 on, 2 off).

–ddline100 100-pixel thin double-dashed line (3 fg, 2 bg).

–wline10 10-pixel line, line width 1.

–wline100 100-pixel line, line width 10.

–wline500 500-pixel line, line width 50.

–wdline100 100-pixel dashed line, line width 10 (30 on, 20 off).

–wddline100	100-pixel double-dashed line, line width 10 (30 fg, 20 bg).
–orect10	10x10 thin rectangle outline.
–orect100	100-pixel thin vertical line segment.
–orect500	500-pixel thin vertical line segment.
–worect10	10x10 wide rectangle outline.
–worect100	100-pixel wide vertical line segment.
–worect500	500-pixel wide vertical line segment.
–circle1	1-pixel diameter thin (line width 0) circle.
–circle10	10-pixel diameter thin circle.
–circle100	100-pixel diameter thin circle.
–circle500	500-pixel diameter thin circle.
–dcircle100	100-pixel diameter thin dashed circle (3 on, 2 off).
–ddcircle100	100-pixel diameter thin double-dashed circle (3 fg, 2 bg).
–wcircle10	10-pixel diameter circle, line width 1.
–wcircle100	100-pixel diameter circle, line width 10.
–wcircle500	500-pixel diameter circle, line width 50.
–wdcircle100	100-pixel diameter dashed circle, line width 10 (30 on, 20 off).
–wddcircle100	100-pixel diameter double-dashed circle, line width 10 (30 fg, 20 bg).
–pcircle10	10-pixel diameter thin partial circle, orientation and arc angle evenly distributed.
–pcircle100	100-pixel diameter thin partial circle.
–wpcircle10	10-pixel diameter wide partial circle.
–wpcircle100	100-pixel diameter wide partial circle.
–fcircle1	1-pixel diameter filled circle.
–fcircle10	10-pixel diameter filled circle.
–fcircle100	100-pixel diameter filled circle.
–fcircle500	500-pixel diameter filled circle.
–fcpcircle10	10-pixel diameter partial filled circle, chord fill, orientation and arc angle evenly distributed.
–fcpcircle100	100-pixel diameter partial filled circle, chord fill.
–fspcircle10	10-pixel diameter partial filled circle, pie slice fill, orientation and arc angle evenly distributed.
–fspcircle100	100-pixel diameter partial filled circle, pie slice fill.
–ellipse10	10-pixel diameter thin (line width 0) ellipse, major and minor axis sizes evenly distributed.
–ellipse100	100-pixel diameter thin ellipse.
–ellipse500	500-pixel diameter thin ellipse.

–dellipse100 100-pixel diameter thin dashed ellipse (3 on, 2 off).
–ddellipse100 100-pixel diameter thin double-dashed ellipse (3 fg, 2 bg).
–wellipse10 10-pixel diameter ellipse, line width 1.
–wellipse100 100-pixel diameter ellipse, line width 10.
–wellipse500 500-pixel diameter ellipse, line width 50.
–wdellipse100 100-pixel diameter dashed ellipse, line width 10 (30 on, 20 off).
–wddellipse100 100-pixel diameter double-dashed ellipse, line width 10 (30 fg, 20 bg).
–pellipse10 10-pixel diameter thin partial ellipse.
–pellipse100 100-pixel diameter thin partial ellipse.
–wpellipse10 10-pixel diameter wide partial ellipse.
–wpellipse100 100-pixel diameter wide partial ellipse.
–fellipse10 10-pixel diameter filled ellipse.
–fellipse100 100-pixel diameter filled ellipse.
–fellipse500 500-pixel diameter filled ellipse.
–fcpellipse10 10-pixel diameter partial filled ellipse, chord fill.
–fcpellipse100 100-pixel diameter partial filled ellipse, chord fill.
–fspellipse10 10-pixel diameter partial filled ellipse, pie slice fill.
–fspellipse100 100-pixel diameter partial filled ellipse, pie slice fill.
–triangle1 Fill 1-pixel/side triangle.
–triangle10 Fill 10-pixel/side triangle.
–triangle100 Fill 100-pixel/side triangle.
–trap1 Fill 1x1 trapezoid.
–trap10 Fill 10x10 trapezoid.
–trap100 Fill 100x100 trapezoid.
–trap300 Fill 300x300 trapezoid.
–strap1 Fill 1x1 transparent stippled trapezoid, 8x8 stipple pattern.
–strap10 Fill 10x10 transparent stippled trapezoid, 8x8 stipple pattern.
–strap100 Fill 100x100 transparent stippled trapezoid, 8x8 stipple pattern.
–strap300 Fill 300x300 transparent stippled trapezoid, 8x8 stipple pattern.
–ostrap1 Fill 10x10 opaque stippled trapezoid, 8x8 stipple pattern.
–ostrap10 Fill 10x10 opaque stippled trapezoid, 8x8 stipple pattern.
–ostrap100 Fill 100x100 opaque stippled trapezoid, 8x8 stipple pattern.
–ostrap300 Fill 300x300 opaque stippled trapezoid, 8x8 stipple pattern.
–tiletrap1 Fill 10x10 tiled trapezoid, 4x4 tile pattern.
–tiletrap10 Fill 10x10 tiled trapezoid, 4x4 tile pattern.
–tiletrap100 Fill 100x100 tiled trapezoid, 4x4 tile pattern.

−tiletrap300 Fill 300x300 tiled trapezoid, 4x4 tile pattern.
−oddstrap1 Fill 1x1 transparent stippled trapezoid, 17x15 stipple pattern.
−oddstrap10 Fill 10x10 transparent stippled trapezoid, 17x15 stipple pattern.
−oddstrap100 Fill 100x100 transparent stippled trapezoid, 17x15 stipple pattern.
−oddstrap300 Fill 300x300 transparent stippled trapezoid, 17x15 stipple pattern.
−oddostrap1 Fill 10x10 opaque stippled trapezoid, 17x15 stipple pattern.
−oddostrap10 Fill 10x10 opaque stippled trapezoid, 17x15 stipple pattern.
−oddostrap100 Fill 100x100 opaque stippled trapezoid, 17x15 stipple pattern.
−oddostrap300 Fill 300x300 opaque stippled trapezoid, 17x15 stipple pattern.
−oddtiletrap1 Fill 10x10 tiled trapezoid, 17x15 tile pattern.
−oddtiletrap10 Fill 10x10 tiled trapezoid, 17x15 tile pattern.
−oddtiletrap100
Fill 100x100 tiled trapezoid, 17x15 tile pattern.
−oddtiletrap300
Fill 300x300 tiled trapezoid, 17x15 tile pattern.
−bigstrap1 Fill 1x1 transparent stippled trapezoid, 161x145 stipple pattern.
−bigstrap10 Fill 10x10 transparent stippled trapezoid, 161x145 stipple pattern.
−bigstrap100 Fill 100x100 transparent stippled trapezoid, 161x145 stipple pattern.
−bigstrap300 Fill 300x300 transparent stippled trapezoid, 161x145 stipple pattern.
−bigostrap1 Fill 10x10 opaque stippled trapezoid, 161x145 stipple pattern.
−bigostrap10 Fill 10x10 opaque stippled trapezoid, 161x145 stipple pattern.
−bigostrap100 Fill 100x100 opaque stippled trapezoid, 161x145 stipple pattern.
−bigostrap300 Fill 300x300 opaque stippled trapezoid, 161x145 stipple pattern.
−bigtiletrap1 Fill 10x10 tiled trapezoid, 161x145 tile pattern.
−bigtiletrap10 Fill 10x10 tiled trapezoid, 161x145 tile pattern.
−bigtiletrap100 Fill 100x100 tiled trapezoid, 161x145 tile pattern.
−bigtiletrap300 Fill 300x300 tiled trapezoid, 161x145 tile pattern.
−eschertiletrap1
Fill 1x1 tiled trapezoid, 216x208 tile pattern.
−eschertiletrap10
Fill 10x10 tiled trapezoid, 216x208 tile pattern.
−eschertiletrap100
Fill 100x100 tiled trapezoid, 216x208 tile pattern.
−eschertiletrap300
Fill 300x300 tiled trapezoid, 216x208 tile pattern.
−complex10 Fill 10-pixel/side complex polygon.
−complex100 Fill 100-pixel/side complex polygon.
−64poly10convex
Fill 10x10 convex 64-gon.

–64poly100convex
Fill 100x100 convex 64-gon.

–64poly10complex
Fill 10x10 complex 64-gon.

–64poly100complex
Fill 100x100 complex 64-gon.

–ftext Character in 80-char line (6x13).

–f8text Character in 70-char line (8x13).

–f9text Character in 60-char line (9x15).

–f14text16 2-byte character in 40-char line (k14).

–tr10text Character in 80-char line (Times-Roman 10).

–tr24text Character in 30-char line (Times-Roman 24).

–polytext Character in 20/40/20 line (6x13, Times-Roman 10, 6x13).

–polytext16 2-byte character in 7/14/7 line (k14, k24).

–fitext Character in 80-char image line (6x13).

–f8itext Character in 70-char image line (8x13).

–f9itext Character in 60-char image line (9x15).

–f14itext16 2-byte character in 40-char image line (k14).

–f24itext16 2-byte character in 23-char image line (k24).

–tr10itext Character in 80-char image line (Times-Roman 10).

–tr24itext Character in 30-char image line (Times-Roman 24).

–scroll10 Scroll 10x10 pixels vertically.

–scroll100 Scroll 100x100 pixels vertically.

–scroll500 Scroll 500x500 pixels vertically.

–copywinwin10 Copy 10x10 square from window to window.

–copywinwin100
Copy 100x100 square from window to window.

–copywinwin500
Copy 500x500 square from window to window.

–copypixwin10 Copy 10x10 square from pixmap to window.

–copypixwin100
Copy 100x100 square from pixmap to window.

–copypixwin500
Copy 500x500 square from pixmap to window.

–copywinpix10 Copy 10x10 square from window to pixmap.

–copywinpix100
Copy 100x100 square from window to pixmap.

–copywinpix500
Copy 500x500 square from window to pixmap.

–copypixpix10 Copy 10x10 square from pixmap to pixmap.
–copypixpix100 Copy 100x100 square from pixmap to pixmap.
–copypixpix500 Copy 500x500 square from pixmap to pixmap.
–copyplane10 Copy 10x10 1-bit deep plane.
–copyplane100 Copy 100x100 1-bit deep plane.
–copyplane500 Copy 500x500 1-bit deep plane.
–putimage10 PutImage 10x10 square.
–putimage100 PutImage 100x100 square.
–putimage500 PutImage 500x500 square.
–putimagexy10 PutImage XY format 10x10 square.
–putimagexy100
PutImage XY format 100x100 square.
–putimagexy500
PutImage XY format 500x500 square.
–shmput10 PutImage 10x10 square, MIT shared memory extension.
–shmput100 PutImage 100x100 square, MIT shared memory extension.
–shmput500 PutImage 500x500 square, MIT shared memory extension.
–shmputxy10 PutImage XY format 10x10 square, MIT shared memory extension.
–shmputxy100 PutImage XY format 100x100 square, MIT shared memory extension.
–shmputxy500 PutImage XY format 500x500 square, MIT shared memory extension.
–getimage10 GetImage 10x10 square.
–getimage100 GetImage 100x100 square.
–getimage500 GetImage 500x500 square.
–getimagexy10 GetImage XY format 10x10 square.
–getimagexy100
GetImage XY format 100x100 square.
–getimagexy500
GetImage XY format 500x500 square.
–noop X protocol NoOperation.
–atom GetAtomName.
–pointer QueryPointer.
–prop GetProperty.
–gc Change graphics context.
–create Create child window and map using MapSubwindows.
–ucreate Create unmapped window.
–map Map child window via MapWindow on parent.
–unmap Unmap child window via UnmapWindow on parent.

–destroy	Destroy child window via DestroyWindow parent.
–popup	Hide/expose window via Map/Unmap popup window.
–move	Move window.
–umove	Moved unmapped window.
–movetree	Move window via MoveWindow on parent.
–resize	Resize window.
–uresize	Resize unmapped window.
–circulate	Circulate lowest window to top.
–ucirculate	Circulate unmapped window to top.

X DEFAULTS

There are no X defaults used by this program.

SEE ALSO

X(1), xbench(1), x11perfcomp(1)

AUTHORS

Joel McCormack
Phil Karlton
Susan Angebranndt
Chris Kent
Keith Packard
Graeme Gill

NAME

xauth − X authority file utility

SYNOPSIS

xauth [**−f** *authfile*] [**−vqib**] [*command arg ...*]

DESCRIPTION

The *xauth* program is used to edit and display the authorization information used in connecting to the X server. This program is usually used to extract authorization records from one machine and merge them in on another (as is the case when using remote logins or granting access to other users). Commands (described below) may be entered interactively, on the *xauth* command line, or in scripts. Note that this program does **not** contact the X server except when the generate command is used. Normally *xauth* is not used to create the authority file entry in the first place; *xdm* does that.

OPTIONS

The following options may be used with *xauth*. They may be given individually (e.g., *−q −i*) or may combined (e.g., *−qi*).

−f *authfile*
: This option specifies the name of the authority file to use. By default, *xauth* will use the file specified by the XAUTHORITY environment variable or *.Xauthority* in the user's home directory.

−q
: This option indicates that *xauth* should operate quietly and not print unsolicited status messages. This is the default if an *xauth* command is is given on the command line or if the standard output is not directed to a terminal.

−v
: This option indicates that *xauth* should operate verbosely and print status messages indicating the results of various operations (e.g., how many records have been read in or written out). This is the default if *xauth* is reading commands from its standard input and its standard output is directed to a terminal.

−i
: This option indicates that *xauth* should ignore any authority file locks. Normally, *xauth* will refuse to read or edit any authority files that have been locked by other programs (usually *xdm* or another *xauth*).

−b
: This option indicates that *xauth* should attempt to break any authority file locks before proceeding. Use this option only to clean up stale locks.

COMMANDS

The following commands may be used to manipulate authority files:

add *displayname protocolname hexkey*
: An authorization entry for the indicated display using the given protocol and key data is added to the authorization file. The data is specified as an even-lengthed string of hexadecimal digits, each pair representing one octet. The first digit of each pair gives the most significant 4 bits of the octet, and the second digit of the pair gives the least significant 4 bits. For example, a 32 character hexkey would represent a 128-bit value. A protocol name consisting of just a single period is treated as an abbreviation for *MIT-MAGIC-COOKIE-1*.

generate *displayname protocolname* [**trusted|untrusted**]
[**timeout** *seconds*] [**group** *group-id*] [**data** *hexdata*]

: This command is similar to add. The main difference is that instead of requiring the user to supply the key data, it connects to the server specified in *displayname* and uses the SECURITY extension in order to get the key data to store in the authorization file. If the server cannot be contacted or if it does not support the SECURITY extension, the command fails. Otherwise, an authorization entry for the indicated display using the given protocol is added to the authorization file. A protocol name consisting of just a single period is treated as an abbreviation for *MIT-MAGIC-COOKIE-1*.

If the **trusted** option is used, clients that connect using this authorization will have full run of the display, as usual. If **untrusted** is used, clients that connect using this authorization will be considered untrusted and prevented from stealing or tampering with data belonging to trusted clients. See the SECURITY extension specification for full details on the restrictions imposed on untrusted clients. The default is **untrusted**.

The **timeout** option specifies how long in seconds this authorization will be valid. If the authorization remains unused (no clients are connected with it) for longer than this time period, the server purges the authorization, and future attempts to connect using it will fail. Note that the purging done by the server does **not** delete the authorization entry from the authorization file. The default timeout is 60 seconds.

The **group** option specifies the application group that clients connecting with this authorization should belong to. See the application group extension specification for more details. The default is to not belong to an application group.

The **data** option specifies data that the server should use to generate the authorization. Note that this is **not** the same data that gets written to the authorization file. The interpretation of this data depends on the authorization protocol. The *hexdata* is in the same format as the *hexkey* described in the add command. The default is to send no data.

[n]extract *filename displayname...*
Authorization entries for each of the specified displays are written to the indicated file. If the *nextract* command is used, the entries are written in a numeric format suitable for non-binary transmission (such as secure electronic mail). The extracted entries can be read back in using the *merge* and *nmerge* commands. If the filename consists of just a single dash, the entries will be written to the standard output.

[n]list [*displayname*...]
Authorization entries for each of the specified displays (or all if no displays are named) are printed on the standard output. If the *nlist* command is used, entries will be shown in the numeric format used by the *nextract* command; otherwise, they are shown in a textual format. Key data is always displayed in the hexadecimal format given in the description of the *add* command.

[n]merge [*filename*...]
Authorization entries are read from the specified files and are merged into the authorization database, superceding any matching existing entries. If the *nmerge* command is used, the numeric format given in the description of the *extract* command is used. If a filename consists of just a single dash, the standard input will be read if it hasn't been read before.

remove *displayname...*
Authorization entries matching the specified displays are removed from the authority file.

source *filename*
The specified file is treated as a script containing *xauth* commands to execute. Blank lines and lines beginning with a sharp sign (#) are ignored. A single dash may be used to indicate the standard input, if it hasn't already been read.

info Information describing the authorization file, whether or not any changes have been made, and from where *xauth* commands are being read is printed on the standard output.

exit If any modifications have been made, the authority file is written out (if allowed), and the program exits. An end of file is treated as an implicit *exit* command.

quit The program exits, ignoring any modifications. This may also be accomplished by pressing the interrupt character.

help [*string***]**
A description of all commands that begin with the given string (or all commands if no string is given) is printed on the standard output.

? A short list of the valid commands is printed on the standard output.

DISPLAY NAMES

Display names for the *add*, *[n]extract*, *[n]list*, *[n]merge*, and *remove* commands use the same format as the DISPLAY environment variable and the common *−display* command line argument. Display-specific information (such as the screen number) is unnecessary and will be ignored. Same-machine connections (such as local-host sockets, shared memory, and the Internet Protocol hostname *localhost*) are referred to as *hostname*/unix:*displaynumber* so that local entries for different machines may be stored in one authority file.

EXAMPLE

The most common use for *xauth* is to extract the entry for the current display, copy it to another machine, and merge it into the user's authority file on the remote machine:

```
% xauth extract − $DISPLAY | rsh otherhost xauth merge −
```

The following command contacts the server :0 to create an authorization using the MIT-MAGIC-COOKIE-1 protocol. Clients that connect with this authorization will be untrusted.

```
% xauth generate :0 .
```

ENVIRONMENT

This *xauth* program uses the following environment variables:

XAUTHORITY
to get the name of the authority file to use if the *−f* option isn't used.

HOME to get the user's home directory if XAUTHORITY isn't defined.

FILES

$HOME/.Xauthority
default authority file if XAUTHORITY isn't defined.

BUGS

Users that have unsecure networks should take care to use encrypted file transfer mechanisms to copy authorization entries between machines. Similarly, the *MIT-MAGIC-COOKIE-1* protocol is not very useful in unsecure environments. Sites that are interested in additional security may need to use encrypted authorization mechanisms such as Kerberos.

Spaces are currently not allowed in the protocol name. Quoting could be added for the truly perverse.

AUTHOR

Jim Fulton, MIT X Consortium

NAME

xclipboard − X clipboard client

SYNOPSIS

xclipboard [−*toolkitoption* ...] [**−w**] [**−nw**]

DESCRIPTION

The *xclipboard* program is used to collect and display text selections that are sent to the CLIPBOARD by other clients. It is typically used to save CLIPBOARD selections for later use. It stores each CLIPBOARD selection as a separate string, each of which can be selected. Each time CLIPBOARD is asserted by another application, *xclipboard* transfers the contents of that selection to a new buffer and displays it in the text window. Buffers are never automatically deleted, so you'll want to use the delete button to get rid of useless items.

Since *xclipboard* uses a Text Widget to display the contents of the clipboard, text sent to the CLIPBOARD may be re-selected for use in other applications. *xclipboard* also responds to requests for the CLIPBOARD selection from other clients by sending the entire contents of the currently displayed buffer.

An *xclipboard* window has the following buttons across the top:

quit When this button is pressed, *xclipboard* exits.

delete When this button is pressed, the current buffer is deleted and the next one displayed.

new Creates a new buffer with no contents. Useful in constructing a new CLIPBOARD selection by hand.

save Displays a File Save dialog box. Pressing the Accept button saves the currently displayed buffer to the file specified in the text field.

next Displays the next buffer in the list.

previous Displays the previous buffer.

OPTIONS

The *xclipboard* program accepts all of the standard X Toolkit command line options as well as the following:

−w This option indicates that lines of text that are too long to be displayed on one line in the clipboard should wrap around to the following lines.

−nw This option indicates that long lines of text should not wrap around. This is the default behavior.

WIDGETS

In order to specify resources, it is useful to know the hierarchy of the widgets which compose *xclipboard*. In the notation below, indentation indicates hierarchical structure. The widget class name is given first, followed by the widget instance name.

```
XClipboard  xclipboard
        Form  form
                Command  Quit
                Command  delete
                Command  new
                Command  Save
                Command  next
                Command  prev
```

```
        Label  index
        Text  text
    TransientShell  fileDialogShell
        Dialog  fileDialog
            Label  label
            Command  accept
            Command  cancel
            Text value
    TransientShell  failDialogShell
        Dialog  failDialog
            Label  label
            Command  continue
```

SENDING/RETRIEVING CLIPBOARD CONTENTS

Text is copied to the clipboard whenever a client asserts ownership of the **CLIPBOARD** selection. Text is copied from the clipboard whenever a client requests the contents of the **CLIPBOARD** selection. Examples of event bindings that a user may wish to include in a resource configuration file to use the clipboard are:

```
*VT100.Translations: #override \
    <Btn3Up>:           select-end(CLIPBOARD) \n\
    <Btn2Up>:           insert-selection(PRIMARY,CLIPBOARD) \n\
    <Btn2Down>:         ignore ()
```

SEE ALSO

X(1), xcutsel(1), xterm(1), individual client documentation for how to make a selection and send it to the CLIPBOARD.

ENVIRONMENT

DISPLAY
to get the default host and display number.

XENVIRONMENT
to get the name of a resource file that overrides the global resources stored in the RESOURCE_MANAGER property.

FILES

<XRoot>/lib/X11/app-defaults/XClipboard - specifies required resources

AUTHOR

Ralph R. Swick, DEC/MIT Project Athena
Chris D. Peterson, MIT X Consortium
Keith Packard, MIT X Consortium

NAME

xcutsel - interchange between cut buffer and selection

SYNOPSIS

xcutsel [*-toolkitoption* ...] [-selection *selection*] [-cutbuffer *number*]

DESCRIPTION

The *xcutsel* program is used to copy the current selection into a cut buffer and to make a selection that contains the current contents of the cut buffer. It acts as a bridge between applications that don't support selections and those that do.

By default, *xcutsel* will use the selection named PRIMARY and the cut buffer CUT_BUFFER0. Either or both of these can be overridden by command line arguments or by resources.

An *xcutsel* window has the following buttons:

quit
: When this button is pressed, *xcutsel* exits. Any selections held by *xcutsel* are automatically released.

copy PRIMARY to 0
: When this button is pressed, *xcutsel* copies the current selection into the cut buffer.

copy 0 to PRIMARY
: When this button is pressed, *xcutsel* converts the current contents of the cut buffer into the selection.

The button labels reflect the selection and cutbuffer selected by command line options or through the resource database.

When the "copy 0 to PRIMARY" button is activated, the button will remain inverted as long as *xcutsel* remains the owner of the selection. This serves to remind you which client owns the current selection. Note that the value of the selection remains constant; if the cutbuffer is changed, you must again activate the copy button to retrieve the new value when desired.

OPTIONS

Xcutsel accepts all of the standard X Toolkit command line options as well as the following:

–selection *name*
: This option specifies the name of the selection to use. The default is PRIMARY. The only supported abbreviations for this option are "-select", "-sel" and "-s", as the standard toolkit option "-selectionTimeout" has a similar name.

–cutbuffer *number*
: This option specifies the cut buffer to use. The default is cut buffer 0.

X DEFAULTS

This program accepts all of the standard X Toolkit resource names and classes as well as:

selection (class **Selection)**
: This resource specifies the name of the selection to use. The default is PRIMARY.

cutBuffer (class **CutBuffer)**
: This resource specifies the number of the cut buffer to use. The default is 0.

WIDGET NAMES

The following instance names may be used when user configuration of the labels in them is desired:

sel-cut (class **Command)**
This is the ‘‘copy SELECTION to BUFFER’’ button.

cut-sel (class **Command)**
This is the ‘‘copy BUFFER to SELECTION’’ button.

quit (class **Command)**
This is the ‘‘quit’’ button.

SEE ALSO

X(1), xclipboard(1), xterm(1), text widget documentation, individual client documentation for how to make a selection.

BUGS

There is no way to change the name of the selection or the number of the cut buffer while the program is running.

AUTHOR

Ralph R. Swick, DEC/MIT Project Athena

NAME

xclock − analog / digital clock for X

SYNOPSIS

xclock [**−help**] [**−analog**] [**−digital**] [**−chime**] [**−hd** *color*] [**−hl** *color*] [**−update** *seconds*] [**−padding** *number*]

DESCRIPTION

The *xclock* program displays the time in analog or digital form. The time is continuously updated at a frequency which may be specified by the user.

OPTIONS

Xclock accepts all of the standard X Toolkit command line options along with the additional options listed below:

−help This option indicates that a brief summary of the allowed options should be printed on the standard error.

−analog This option indicates that a conventional 12 hour clock face with tick marks and hands should be used. This is the default.

−digital or **−d**
This option indicates that a 24 hour digital clock should be used.

−chime This option indicates that the clock should chime once on the half hour and twice on the hour.

−hands *color* (or **−hd** *color*)
This option specifies the color of the hands on an analog clock. The default is *black*.

−highlight *color* (or **−hl** *color*)
This option specifies the color of the edges of the hands on an analog clock, and is only useful on color displays. The default is *black*.

−update *seconds*
This option specifies the frequency in seconds at which *xclock* should update its display. If the clock is obscured and then exposed, it will be updated immediately. A value of 30 seconds or less will enable a second hand on an analog clock. The default is 60 seconds.

−padding *number*
This option specifies the width in pixels of the padding between the window border and clock text or picture. The default is 10 on a digital clock and 8 on an analog clock.

X DEFAULTS

This program uses the *Clock* widget. It understands all of the core resource names and classes as well as:

width (class **Width)**
Specifies the width of the clock. The default for analog clocks is 164 pixels; the default for digital clocks is whatever is needed to hold the clock when displayed in the chosen font.

height (class **Height)**
Specifies the height of the clock. The default for analog clocks is 164 pixels; the default for digital clocks is whatever is needed to hold the clock when displayed in the chosen font.

update (class **Interval)**
Specifies the frequency in seconds at which the time should be redisplayed.

foreground (class **Foreground)**
Specifies the color for the tic marks. The default is depends on whether *reverse-Video* is specified. If *reverseVideo* is specified the default is *lwhite*, otherwise the default is *black.*

hands (class **Foreground)**
Specifies the color of the insides of the clock's hands. The default is depends on whether *reverseVideo* is specified. If *reverseVideo* is specified the default is *lwhite*, otherwise the default is *black.*

highlight (class **Foreground)**
Specifies the color used to highlight the clock's hands. The default is depends on whether *reverseVideo* is specified. If *reverseVideo* is specified the default is *lwhite*, otherwise the default is *black.*

analog (class **Boolean)**
Specifies whether or not an analog clock should be used instead of a digital one. The default is True.

chime (class **Boolean)**
Specifies whether or not a bell should be rung on the hour and half hour.

padding (class **Margin)**
Specifies the amount of internal padding in pixels to be used. The default is 8.

font (class **Font)**
Specifies the font to be used for the digital clock. Note that variable width fonts currently will not always display correctly.

WIDGETS

In order to specify resources, it is useful to know the hierarchy of the widgets which compose *xclock*. In the notation below, indentation indicates hierarchical structure. The widget class name is given first, followed by the widget instance name.

```
XClock  xclock
        Clock  clock
```

ENVIRONMENT

DISPLAY
to get the default host and display number.

XENVIRONMENT
to get the name of a resource file that overrides the global resources stored in the RESOURCE_MANAGER property.

FILES

<XRoot>/lib/X11/app-defaults/XClock - specifies required resources

SEE ALSO

X(1), xrdb(1), time(3C)

BUGS

Xclock believes the system clock.

When in digital mode, the string should be centered automatically.

AUTHORS

Tony Della Fera (MIT-Athena, DEC)
Dave Mankins (MIT-Athena, BBN)
Ed Moy (UC Berkeley)

NAME

xcmsdb – Device Color Characterization utility for X Color Management System

SYNOPSIS

xcmsdb [**–query**] [**–remove**] [**–format 32|16|8**] [*filename*]

DESCRIPTION

xcmsdb is used to load, query, or remove Device Color Characterization data stored in properties on the root window of the screen as specified in section 7, Device Color Characterization, of the ICCCM. Device Color Characterization data (also called the Device Profile) is an integral part of Xlib's X Color Management System (Xcms), necessary for proper conversion of color specification between device-independent and device-dependent forms. Xcms uses 3x3 matrices stored in the XDCCC_LINEAR_RGB_MATRICES property to convert color specifications between CIEXYZ and RGB Intensity (XcmsRGBi, also referred to as linear RGB). Xcms then uses display gamma information stored in the XDCCC_LINEAR_RGB_CORRECTION property to convert color specifications between RGBi and RGB device (XcmsRGB, also referred to as device RGB). Note that Xcms allows clients to register *function sets* in addition to its built-in function set for CRT color monitors. Additional function sets may store their device profile information in other properties in function set specific format. This utility is unaware of these non-standard properties. The ASCII readable contents of *filename* (or the standard input if no input file is given) are appropriately transformed for storage in properties, provided the **–query** or **–remove** options are not specified.

OPTIONS

xcmsdb program accepts the following options:

–query This option attempts to read the XDCCC properties off the screen's root window. If successful, it transforms the data into a more readable format, then sends the data to standard out.

–remove This option attempts to remove the XDCCC properties on the screen's root window.

–format 32|16|8
Specifies the property format (32, 16, or 8 bits per entry) for the XDCCC_LINEAR_RGB_CORRECTION property. Precision of encoded floating point values increases with the increase in bits per entry. The default is 32 bits per entry.

SEE ALSO

xprop(1), Xlib documentation

ENVIRONMENT

DISPLAY
to figure out which display and screen to use.

AUTHOR

Chuck Adams, Tektronix Inc. Al Tabayoyon, SynChromatics Inc. (added multi-visual support)

NAME

xconsole − monitor system console messages with X

SYNOPSIS

xconsole [-*toolkitoption* ...] [-file *file-name*] [-notify] [-stripNonprint] [-daemon] [-verbose] [-exitOnFail]

DESCRIPTION

The *xconsole* program displays messages which are usually sent to /dev/console.

OPTIONS

Xconsole accepts all of the standard X Toolkit command line options along with the additional options listed below:

−file *file-name*
: To monitor some other device, use this option to specify the device name. This does not work on regular files as they are always ready to be read from.

−notify −nonotify
: When new data are received from the console and the notify option is set, the icon name of the application has " *" appended, so that it is evident even when the application is iconified. −notify is the default.

−daemon
: This option causes *xconsole* to place itself in the background, using fork/exit.

−verbose
: When set, this option directs *xconsole* to display an informative message in the first line of the text buffer.

−exitOnFail
: When set, this option directs *xconsole* to exit when it is unable to redirect the console output.

X DEFAULTS

This program uses the *Athena Text* widget, look in the *Athena Widget Set* documentation for controlling it.

WIDGETS

In order to specify resources, it is useful to know the hierarchy of the widgets which compose *xconsole*. In the notation below, indentation indicates hierarchical structure. The widget class name is given first, followed by the widget instance name.

```
XConsole  xconsole
        XConsole  text
```

ENVIRONMENT

DISPLAY
: to get the default host and display number.

XENVIRONMENT
: to get the name of a resource file that overrides the global resources stored in the RESOURCE_MANAGER property.

FILES

<XRoot>/lib/X11/app-defaults/XConsole - specifies required resources

SEE ALSO

X(1), xrdb(1), Athena Text widget

AUTHOR

Keith Packard (MIT X Consortium)

NAME

sessreg – manage utmp/wtmp entries for non-init clients

SYNOPSIS

sessreg [-w *wtmp-file*] [-u *utmp-file*] [-l *line-name*] [-h *host-name*] [-s *slot-number*] [-x *Xservers-file*] [-t *ttys-file*] [-a] [-d] *user-name*

DESCRIPTION

Sessreg is a simple program for managing utmp/wtmp entries for xdm sessions.

System V has a better interface to /etc/utmp than BSD; it dynamically allocates entries in the file, instead of writing them at fixed positions indexed by position in /etc/ttys.

To manage BSD-style utmp files, *sessreg* has two strategies. In conjunction with xdm, the -x option counts the number of lines in /etc/ttys and then adds to that the number of the line in the Xservers file which specifies the display. The display name must be specified as the "line-name" using the -l option. This sum is used as the "slot-number" in /etc/utmp that this entry will be written at. In the more general case, the -s option specifies the slot-number directly. If for some strange reason your system uses a file other that /etc/ttys to manage init, the -t option can direct *sessreg* to look elsewhere for a count of terminal sessions.

Conversely, System V managers will not ever need to use these options (-x, -s and -t). To make the program easier to document and explain, *sessreg* accepts the BSD-specific flags in the System V environment and ignores them.

BSD also has a host-name field in the utmp file which doesn't exist in System V. This option is also ignored by the System V version of *sessreg*.

USAGE

In Xstartup, place a call like:

```
sessreg -a -l $DISPLAY -x /usr/X11R6/lib/xdm/Xservers $USER
```

and in Xreset:

```
sessreg -d -l $DISPLAY -x /usr/X11R6/lib/xdm/Xservers $USER
```

OPTIONS

-w *wtmp-file*
This specifies an alternate wtmp file, instead of /usr/adm/wtmp for BSD or /etc/wtmp for sysV. The special name "none" disables writing records to /usr/adm/wtmp.

-u *utmp-file*
This specifies an alternate utmp file, instead of "/etc/utmp". The special name "none" disables writing records to /etc/utmp.

-l *line-name*
This describes the "line" name of the entry. For terminal sessions, this is the final pathname segment of the terminal device filename (e.g. ttyd0). For X sessions, it should probably be the local display name given to the users session (e.g. :0). If none is specified, the terminal name will be determined with ttyname(3) and stripped of leading components.

-h *host-name*
This is set for BSD hosts to indicate that the session was initiated from a remote host. In typical xdm usage, this options is not used.

-s *slot-number*
Each potential session has a unique slot number in BSD systems, most are identified by the position of the *line-name* in the /etc/ttys file. This option overrides the default position determined with ttyslot(3). This option is inappropriate for use with xdm, the -x option is more useful.

-x *Xservers-file*
As X sessions are one-per-display, and each display is entered in this file, this options sets the *slot-number* to be the number of lines in the *ttys-file* plus the index into this file that the *line-name* is found.

-t *ttys-file*
This specifies an alternate file which the *-x* option will use to count the number of terminal sessions on a host.

-a This session should be added to utmp/wtmp.

-d This session should be deleted from utmp/wtmp. One of -a/-d must be specified.

SEE ALSO

xdm(1)

AUTHOR

Keith Packard, MIT X Consortium

NAME

xdm – X Display Manager with support for XDMCP, host chooser

SYNOPSIS

xdm [**–config** *configuration_file*] [**–nodaemon**] [**–debug** *debug_level*] [**–error** *error_log_file*] [**–resources** *resource_file*] [**–server** *server_entry*] [**–session** *session_program*]

DESCRIPTION

Xdm manages a collection of X displays, which may be on the local host or remote servers. The design of *xdm* was guided by the needs of X terminals as well as the X Consortium standard XDMCP, the *X Display Manager Control Protocol*. *Xdm* provides services similar to those provided by *init*, *getty* and *login* on character terminals: prompting for login name and password, authenticating the user, and running a "session."

A "session" is defined by the lifetime of a particular process; in the traditional character-based terminal world, it is the user's login shell. In the *xdm* context, it is an arbitrary session manager. This is because in a windowing environment, a user's login shell process does not necessarily have any terminal-like interface with which to connect. When a real session manager is not available, a window manager or terminal emulator is typically used as the "session manager," meaning that termination of this process terminates the user's session.

When the session is terminated, *xdm* resets the X server and (optionally) restarts the whole process.

When *xdm* receives an Indirect query via XDMCP, it can run a *chooser* process to perform an XDMCP BroadcastQuery (or an XDMCP Query to specified hosts) on behalf of the display and offer a menu of possible hosts that offer XDMCP display management. This feature is useful with X terminals that do not offer a host menu themselves.

Because *xdm* provides the first interface that users will see, it is designed to be simple to use and easy to customize to the needs of a particular site. *Xdm* has many options, most of which have reasonable defaults. Browse through the various sections of this manual, picking and choosing the things you want to change. Pay particular attention to the **Session Program** section, which will describe how to set up the style of session desired.

OVERVIEW

xdm is highly configurable, and most of its behavior can be controlled by resource files and shell scripts. The names of these files themselves are resources read from the file *xdm-config* or the file named by the **–config** option.

xdm offers display management two different ways. It can manage X servers running on the local machine and specified in *Xservers*, and it can manage remote X servers (typically X terminals) using XDMCP (the XDM Control Protocol) as specified in the *Xaccess* file.

The resources of the X clients run by *xdm* outside the user's session, including *xdm*'s own login window, can be affected by setting resources in the *Xresources* file.

For X terminals that do not offer a menu of hosts to get display management from, *xdm* can collect willing hosts and run the *chooser* program to offer the user a menu. For X displays attached to a host, this step is typically not used, as the local host does the display management.

After resetting the X server, *xdm* runs the *Xsetup* script to assist in setting up the screen the user sees along with the *xlogin* widget.

The *xlogin* widget, which *xdm* presents, offers the familiar login and password prompts.

After the user logs in, *xdm* runs the *Xstartup* script as root.

Then *xdm* runs the *Xsession* script as the user. This system session file may do some additional startup and typically runs the *.xsession* script in the user's home directory. When the *Xsession* script exits, the session is over.

At the end of the session, the *Xreset* script is run to clean up, the X server is reset, and the cycle starts over.

The file */usr/X11R6/lib/X11/xdm/xdm-errors* will contain error messages from *xdm* and anything output to stderr by *Xsetup, Xstartup, Xsession* or *Xreset*. When you have trouble getting *xdm* working, check this file to see if *xdm* has any clues to the trouble.

OPTIONS

All of these options, except **−config** itself, specify values that can also be specified in the configuration file as resources.

−config *configuration_file*
: Names the configuration file, which specifies resources to control the behavior of *xdm*. *<XRoot>/lib/X11/xdm/xdm-config* is the default. See the section **Configuration File**.

−nodaemon
: Specifies "false" as the value for the **DisplayManager.daemonMode** resource. This suppresses the normal daemon behavior, which is for *xdm* to close all file descriptors, disassociate itself from the controlling terminal, and put itself in the background when it first starts up.

−debug *debug_level*
: Specifies the numeric value for the **DisplayManager.debugLevel** resource. A non-zero value causes *xdm* to print lots of debugging statements to the terminal; it also disables the **DisplayManager.daemonMode** resource, forcing *xdm* to run synchronously. To interpret these debugging messages, a copy of the source code for *xdm* is almost a necessity. No attempt has been made to rationalize or standardize the output.

−error *error_log_file*
: Specifies the value for the **DisplayManager.errorLogFile** resource. This file contains errors from *xdm* as well as anything written to stderr by the various scripts and programs run during the progress of the session.

−resources *resource_file*
: Specifies the value for the **DisplayManager*resources** resource. This file is loaded using *xrdb* to specify configuration parameters for the authentication widget.

−server *server_entry*
: Specifies the value for the **DisplayManager.servers** resource. See the section **Local Server Specification** for a description of this resource.

−udpPort *port_number*
: Specifies the value for the **DisplayManager.requestPort** resource. This sets the port-number which *xdm* will monitor for XDMCP requests. As XDMCP uses the registered well-known UDP port 177, this resource should not be changed except for debugging.

–session *session_program*
Specifies the value for the **DisplayManager*session** resource. This indicates the program to run as the session after the user has logged in.

–xrm *resource_specification*
Allows an arbitrary resource to be specified, as in most X Toolkit applications.

RESOURCES

At many stages the actions of *xdm* can be controlled through the use of its configuration file, which is in the X resource format. Some resources modify the behavior of *xdm* on all displays, while others modify its behavior on a single display. Where actions relate to a specific display, the display name is inserted into the resource name between ‘‘Display-Manager’’ and the final resource name segment.

For local displays, the resource name and class are as read from the *Xservers* file.

For remote displays, the resource name is what the network address of the display resolves to. See the **removeDomain** resource. The name must match exactly; *xdm* is not aware of all the network aliases that might reach a given display. If the name resolve fails, the address is used. The resource class is as sent by the display in the XDMCP Manage request.

Because the resource manager uses colons to separate the name of the resource from its value and dots to separate resource name parts, *xdm* substitutes underscores for both dots and colons when generating the resource name. For example, **DisplayManager.expo_x_org_0.startup** is the name of the resource which defines the startup shell file for the ‘‘expo.x.org:0’’ display.

DisplayManager.servers
This resource either specifies a file name full of server entries, one per line (if the value starts with a slash), or a single server entry. See the section **Local Server Specification** for the details.

DisplayManager.requestPort
This indicates the UDP port number which *xdm* uses to listen for incoming XDMCP requests. Unless you need to debug the system, leave this with its default value of 177.

DisplayManager.errorLogFile
Error output is normally directed at the system console. To redirect it, set this resource to a file name. A method to send these messages to *syslog* should be developed for systems which support it; however, the wide variety of interfaces precludes any system-independent implementation. This file also contains any output directed to stderr by the *Xsetup, Xstartup, Xsession* and *Xreset* files, so it will contain descriptions of problems in those scripts as well.

DisplayManager.debugLevel
If the integer value of this resource is greater than zero, reams of debugging information will be printed. It also disables daemon mode, which would redirect the information into the bit-bucket, and allows non-root users to run *xdm,* which would normally not be useful.

DisplayManager.daemonMode
Normally, *xdm* attempts to make itself into a daemon process unassociated with any terminal. This is accomplished by forking and leaving the parent process to exit, then closing file descriptors and releasing the controlling terminal. In some environments this is not desired (in particular, when debugging). Setting this

resource to ‘‘false’’ will disable this feature.

DisplayManager.pidFile
The filename specified will be created to contain an ASCII representation of the process-id of the main *xdm* process. *Xdm* also uses file locking on this file to attempt to eliminate multiple daemons running on the same machine, which would cause quite a bit of havoc.

DisplayManager.lockPidFile
This is the resource which controls whether *xdm* uses file locking to keep multiple display managers from running amok. On System V, this uses the *lockf* library call, while on BSD it uses *flock.*

DisplayManager.authDir
This names a directory under which *xdm* stores authorization files while initializing the session. The default value is *<XRoot>/lib/X11/xdm.* Can be overridden for specific displays by DisplayManager.*DISPLAY*.authFile.

DisplayManager.autoRescan
This boolean controls whether *xdm* rescans the configuration, servers, access control and authentication keys files after a session terminates and the files have changed. By default it is ‘‘true.’’ You can force *xdm* to reread these files by sending a SIGHUP to the main process.

DisplayManager.removeDomainname
When computing the display name for XDMCP clients, the name resolver will typically create a fully qualified host name for the terminal. As this is sometimes confusing, *xdm* will remove the domain name portion of the host name if it is the same as the domain name of the local host when this variable is set. By default the value is ‘‘true.’’

DisplayManager.keyFile
XDM-AUTHENTICATION-1 style XDMCP authentication requires that a private key be shared between *xdm* and the terminal. This resource specifies the file containing those values. Each entry in the file consists of a display name and the shared key. By default, *xdm* does not include support for XDM-AUTHENTICATION-1, as it requires DES which is not generally distributable because of United States export restrictions.

DisplayManager.accessFile
To prevent unauthorized XDMCP service and to allow forwarding of XDMCP IndirectQuery requests, this file contains a database of hostnames which are either allowed direct access to this machine, or have a list of hosts to which queries should be forwarded to. The format of this file is described in the section **XDMCP Access Control.**

DisplayManager.exportList
A list of additional environment variables, separated by white space, to pass on to the *Xsetup*, *Xstartup*, *Xsession*, and *Xreset* programs.

DisplayManager.randomFile
A file to checksum to generate the seed of authorization keys. This should be a file that changes frequently. The default is */dev/mem*.

DisplayManager.greeterLib
On systems that support a dynamically-loadable greeter library, the name of the library. Default is *<XRoot>/lib/X11/xdm/libXdmGreet.so*.

DisplayManager.choiceTimeout
Number of seconds to wait for display to respond after user has selected a host from the chooser. If the display sends an XDMCP IndirectQuery within this time, the request is forwarded to the chosen host. Otherwise, it is assumed to be from a new session and the chooser is offered again. Default is 15.

DisplayManager.*DISPLAY*.resources
This resource specifies the name of the file to be loaded by *xrdb* as the resource database onto the root window of screen 0 of the display. The *Xsetup* program, the Login widget, and *chooser* will use the resources set in this file. This resource data base is loaded just before the authentication procedure is started, so it can control the appearance of the login window. See the section **Authentication Widget,** which describes the various resources that are appropriate to place in this file. There is no default value for this resource, but *<XRoot>/lib/X11/xdm/Xresources* is the conventional name.

DisplayManager.*DISPLAY*.chooser
Specifies the program run to offer a host menu for Indirect queries redirected to the special host name CHOOSER. *<XRoot>/lib/X11/xdm/chooser* is the default. See the sections **XDMCP Access Control** and **Chooser**.

DisplayManager.*DISPLAY*.xrdb
Specifies the program used to load the resources. By default, *xdm* uses *<XRoot>/bin/xrdb*.

DisplayManager.*DISPLAY*.cpp
This specifies the name of the C preprocessor which is used by *xrdb*.

DisplayManager.*DISPLAY*.setup
This specifies a program which is run (as root) before offering the Login window. This may be used to change the appearance of the screen around the Login window or to put up other windows (e.g., you may want to run *xconsole* here). By default, no program is run. The conventional name for a file used here is *Xsetup*. See the section **Setup Program.**

DisplayManager.*DISPLAY*.startup
This specifies a program which is run (as root) after the authentication process succeeds. By default, no program is run. The conventional name for a file used here is *Xstartup*. See the section **Startup Program.**

DisplayManager.*DISPLAY*.session
This specifies the session to be executed (not running as root). By default, *<XRoot>/bin/xterm* is run. The conventional name is *Xsession*. See the section **Session Program.**

DisplayManager.*DISPLAY*.reset
This specifies a program which is run (as root) after the session terminates. By default, no program is run. The conventional name is *Xreset*. See the section **Reset Program.**

DisplayManager.*DISPLAY*.openDelay

DisplayManager.*DISPLAY*.openRepeat

DisplayManager.*DISPLAY*.openTimeout

DisplayManager.*DISPLAY*.startAttempts
These numeric resources control the behavior of *xdm* when attempting to open

intransigent servers. **openDelay** is the length of the pause (in seconds) between successive attempts, **openRepeat** is the number of attempts to make, **openTimeout** is the amount of time to wait while actually attempting the open (i.e., the maximum time spent in the *connect*(2) system call) and **startAttempts** is the number of times this entire process is done before giving up on the server. After **openRepeat** attempts have been made, or if **openTimeout** seconds elapse in any particular attempt, *xdm* terminates and restarts the server, attempting to connect again. This process is repeated **startAttempts** times, at which point the display is declared dead and disabled. Although this behavior may seem arbitrary, it has been empirically developed and works quite well on most systems. The default values are 5 for **openDelay**, 5 for **openRepeat**, 30 for **openTimeout** and 4 for **startAttempts**.

DisplayManager.*DISPLAY*.pingInterval

DisplayManager.*DISPLAY*.pingTimeout

To discover when remote displays disappear, *xdm* occasionally pings them, using an X connection and *XSync* calls. **pingInterval** specifies the time (in minutes) between each ping attempt, **pingTimeout** specifies the maximum amount of time (in minutes) to wait for the terminal to respond to the request. If the terminal does not respond, the session is declared dead and terminated. By default, both are set to 5 minutes. If you frequently use X terminals which can become isolated from the managing host, you may wish to increase this value. The only worry is that sessions will continue to exist after the terminal has been accidentally disabled. *xdm* will not ping local displays. Although it would seem harmless, it is unpleasant when the workstation session is terminated as a result of the server hanging for NFS service and not responding to the ping.

DisplayManager.*DISPLAY*.terminateServer

This boolean resource specifies whether the X server should be terminated when a session terminates (instead of resetting it). This option can be used when the server tends to grow without bound over time, in order to limit the amount of time the server is run. The default value is ‘‘false.’’

DisplayManager.*DISPLAY*.userPath

Xdm sets the PATH environment variable for the session to this value. It should be a colon separated list of directories; see *sh*(1) for a full description. ‘‘:/bin:/usr/bin:/usr/X11R6/bin:/usr/ucb’’ is a common setting. The default value can be specified at build time in the X system configuration file with DefaultUserPath.

DisplayManager.*DISPLAY*.systemPath

Xdm sets the PATH environment variable for the startup and reset scripts to the value of this resource. The default for this resource is specified at build time by the DefaultSystemPath entry in the system configuration file; ‘‘/etc:/bin:/usr/bin:/usr/X11R6/bin:/usr/ucb’’ is a common choice. Note the absence of ‘‘.’’ from this entry. This is a good practice to follow for root; it avoids many common Trojan Horse system penetration schemes.

DisplayManager.*DISPLAY*.systemShell

Xdm sets the SHELL environment variable for the startup and reset scripts to the value of this resource. It is */bin/sh* by default.

DisplayManager.*DISPLAY*.failsafeClient

If the default session fails to execute, *xdm* will fall back to this program. This program is executed with no arguments, but executes using the same environment variables as the session would have had (see the section **Session Program**). By default, *<XRoot>/bin/xterm* is used.

DisplayManager.*DISPLAY*.grabServer

DisplayManager.*DISPLAY*.grabTimeout
To improve security, *xdm* grabs the server and keyboard while reading the login name and password. The **grabServer** resource specifies if the server should be held for the duration of the name/password reading. When ‘‘false,’’ the server is ungrabbed after the keyboard grab succeeds, otherwise the server is grabbed until just before the session begins. The default is ‘‘false.’’ The **grabTimeout** resource specifies the maximum time *xdm* will wait for the grab to succeed. The grab may fail if some other client has the server grabbed, or possibly if the network latencies are very high. This resource has a default value of 3 seconds; you should be cautious when raising it, as a user can be spoofed by a look-alike window on the display. If the grab fails, *xdm* kills and restarts the server (if possible) and the session.

DisplayManager.*DISPLAY*.authorize

DisplayManager.*DISPLAY*.authName
authorize is a boolean resource which controls whether *xdm* generates and uses authorization for the local server connections. If authorization is used, **authName** is a list of authorization mechanisms to use, separated by white space. XDMCP connections dynamically specify which authorization mechanisms are supported, so **authName** is ignored in this case. When **authorize** is set for a display and authorization is not available, the user is informed by having a different message displayed in the login widget. By default, **authorize** is ‘‘true.’’ **authName** is ‘‘MIT-MAGIC-COOKIE-1,’’ or, if XDM-AUTHORIZATION-1 is available, ‘‘XDM-AUTHORIZATION-1 MIT-MAGIC-COOKIE-1.’’

DisplayManager.*DISPLAY*.authFile
This file is used to communicate the authorization data from *xdm* to the server, using the **−auth** server command line option. It should be kept in a directory which is not world-writable as it could easily be removed, disabling the authorization mechanism in the server. If not specified, a name is generated from DisplayManager.authDir and the name of the display.

DisplayManager.*DISPLAY*.authComplain
If set to ‘‘false,’’ disables the use of the **unsecureGreeting** in the login window. See the section **Authentication Widget.** The default is ‘‘true.’’

DisplayManager.*DISPLAY*.resetSignal
The number of the signal *xdm* sends to reset the server. See the section **Controlling the Server.** The default is 1 (SIGHUP).

DisplayManager.*DISPLAY*.termSignal
The number of the signal *xdm* sends to terminate the server. See the section **Controlling the Server.** The default is 15 (SIGTERM).

DisplayManager.*DISPLAY*.resetForAuth
The original implementation of authorization in the sample server reread the authorization file at server reset time, instead of when checking the initial connection. As *xdm* generates the authorization information just before connecting to the display, an old server would not get up-to-date authorization information. This resource causes *xdm* to send SIGHUP to the server after setting up the file,

causing an additional server reset to occur, during which time the new authorization information will be read. The default is ‘‘false,’’ which will work for all MIT servers.

DisplayManager.*DISPLAY*.userAuthDir

When *xdm* is unable to write to the usual user authorization file ($HOME/.Xauthority), it creates a unique file name in this directory and points the environment variable XAUTHORITY at the created file. It uses */tmp* by default.

CONFIGURATION FILE

First, the *xdm* configuration file should be set up. Make a directory (usually *<XRoot>/lib/X11/xdm*, where <XRoot> refers to the root of the X11 install tree) to contain all of the relevant files. In the examples that follow, we use /usr/X11R6 as the value of <XRoot>. Here is a reasonable configuration file, which could be named *xdm-config*:

```
DisplayManager.servers:          /usr/X11R6/lib/X11/xdm/Xservers
DisplayManager.errorLogFile:     /usr/X11R6/lib/X11/xdm/xdm-errors
DisplayManager*resources:        /usr/X11R6/lib/X11/xdm/Xresources
DisplayManager*startup:          /usr/X11R6/lib/X11/xdm/Xstartup
DisplayManager*session:          /usr/X11R6/lib/X11/xdm/Xsession
DisplayManager.pidFile:          /usr/X11R6/lib/X11/xdm/xdm-pid
DisplayManager._0.authorize:     true
DisplayManager*authorize:        false
```

Note that this file mostly contains references to other files. Note also that some of the resources are specified with ‘‘*’’ separating the components. These resources can be made unique for each different display, by replacing the ‘‘*’’ with the display-name, but normally this is not very useful. See the **Resources** section for a complete discussion.

XDMCP ACCESS CONTROL

The database file specified by the **DisplayManager.accessFile** provides information which *xdm* uses to control access from displays requesting XDMCP service. This file contains three types of entries: entries which control the response to Direct and Broadcast queries, entries which control the response to Indirect queries, and macro definitions.

The format of the Direct entries is simple, either a host name or a pattern, which is distinguished from a host name by the inclusion of one or more meta characters (‘*’ matches any sequence of 0 or more characters, and ‘?’ matches any single character) which are compared against the host name of the display device. If the entry is a host name, all comparisons are done using network addresses, so any name which converts to the correct network address may be used. For patterns, only canonical host names are used in the comparison, so ensure that you do not attempt to match aliases. Preceding either a host name or a pattern with a ‘!’ character causes hosts which match that entry to be excluded.

An Indirect entry also contains a host name or pattern, but follows it with a list of host names or macros to which indirect queries should be sent.

A macro definition contains a macro name and a list of host names and other macros that the macro expands to. To distinguish macros from hostnames, macro names start with a ‘%’ character. Macros may be nested.

Indirect entries may also specify to have *xdm* run *chooser* to offer a menu of hosts to connect to. See the section **Chooser**.

When checking access for a particular display host, each entry is scanned in turn and the first matching entry determines the response. Direct and Broadcast entries are ignored when scanning for an Indirect entry and vice-versa.

Blank lines are ignored, '#' is treated as a comment delimiter causing the rest of that line to be ignored, and '*newline*' causes the newline to be ignored, allowing indirect host lists to span multiple lines.

Here is an example Xaccess file:

```
#
# Xaccess – XDMCP access control file
#

#
# Direct/Broadcast query entries
#

!xtra.lcs.mit.edu          # disallow direct/broadcast service for xtra
bambi.ogi.edu              # allow access from this particular display
*.lcs.mit.edu              # allow access from any display in LCS

#
# Indirect query entries
#

%HOSTS                     expo.lcs.mit.edu xenon.lcs.mit.edu \
                           excess.lcs.mit.edu kanga.lcs.mit.edu

extract.lcs.mit.edu        xenon.lcs.mit.edu        #force extract to contact xenon
!xtra.lcs.mit.edu          dummy                    #disallow indirect access
*.lcs.mit.edu              %HOSTS                   #all others get to choose
```

CHOOSER

For X terminals that do not offer a host menu for use with Broadcast or Indirect queries, the *chooser* program can do this for them. In the *Xaccess* file, specify "CHOOSER" as the first entry in the Indirect host list. *Chooser* will send a Query request to each of the remaining host names in the list and offer a menu of all the hosts that respond.

The list may consist of the word "BROADCAST," in which case *chooser* will send a Broadcast instead, again offering a menu of all hosts that respond. Note that on some operating systems, UDP packets cannot be broadcast, so this feature will not work.

Example *Xaccess* file using *chooser*:

```
extract.lcs.mit.edu        CHOOSER %HOSTS           #offer a menu of these hosts
xtra.lcs.mit.edu           CHOOSER BROADCAST        #offer a menu of all hosts
```

The program to use for *chooser* is specified by the **DisplayManager.*DISPLAY*.chooser** resource. For more flexibility at this step, the chooser could be a shell script. *Chooser* is the session manager here; it is run instead of a child *xdm* to manage the display.

Resources for this program can be put into the file named by **DisplayManager.*DISPLAY*.resources**.

When the user selects a host, *chooser* prints the host chosen, which is read by the parent *xdm*, and exits. *xdm* closes its connection to the X server, and the server resets and sends another **Indirect** XDMCP request. *xdm* remembers the user's choice (for **DisplayManager.choiceTimeout** seconds) and forwards the request to the chosen host, which starts a session on that display.

LOCAL SERVER SPECIFICATION

The resource **DisplayManager.servers** gives a server specification or, if the values starts with a slash (/), the name of a file containing server specifications, one per line.

Each specification indicates a display which should constantly be managed and which is not using XDMCP. This method is used typically for local servers only. If the resource or the file named by the resource is empty, *xdm* will offer XDMCP service only.

Each specification consists of at least three parts: a display name, a display class, a display type, and (for local servers) a command line to start the server. A typical entry for local display number 0 would be:

```
:0 Digital-QV local /usr/X11R6/bin/X :0
```

The display types are:

local	local display: *xdm* must run the server
foreign	remote display: *xdm* opens an X connection to a running server

The display name must be something that can be passed in the **−display** option to an X program. This string is used to generate the display-specific resource names, so be careful to match the names (e.g., use ":0 Sun-CG3 local /usr/X11R6/bin/X :0" instead of "localhost:0 Sun-CG3 local /usr/X11R6/bin/X :0" if your other resources are specified as "DisplayManager._0.session"). The display class portion is also used in the display-specific resources, as the class of the resource. This is useful if you have a large collection of similar displays (such as a corral of X terminals) and would like to set resources for groups of them. When using XDMCP, the display is required to specify the display class, so the manual for your particular X terminal should document the display class string for your device. If it doesn't, you can run *xdm* in debug mode and look at the resource strings which it generates for that device, which will include the class string.

When *xdm* starts a session, it sets up authorization data for the server. For local servers, *xdm* passes "**−auth** *filename*" on the server's command line to point it at its authorization data. For XDMCP servers, *xdm* passes the authorization data to the server via the **Accept** XDMCP request.

RESOURCES FILE

The *Xresources* file is loaded onto the display as a resource database using *xrdb.* As the authentication widget reads this database before starting up, it usually contains parameters for that widget:

```
xlogin*login.translations: #override\
        Ctrl<Key>R: abort-display()\n\
        <Key>F1: set-session-argument(failsafe) finish-field()\n\
        <Key>Return: set-session-argument() finish-field()
xlogin*borderWidth: 3
xlogin*greeting: CLIENTHOST
#ifdef COLOR
xlogin*greetColor: CadetBlue
xlogin*failColor: red
```

```
#endif
```

Please note the translations entry; it specifies a few new translations for the widget which allow users to escape from the default session (and avoid troubles that may occur in it). Note that if #override is not specified, the default translations are removed and replaced by the new value, not a very useful result as some of the default translations are quite useful (such as "<Key>: insert-char ()" which responds to normal typing).

This file may also contain resources for the setup program and *chooser*.

SETUP PROGRAM

The *Xsetup* file is run after the server is reset, but before the Login window is offered. The file is typically a shell script. It is run as root, so should be careful about security. This is the place to change the root background or bring up other windows that should appear on the screen along with the Login widget.

In addition to any specified by **DisplayManager.exportList**, the following environment variables are passed:

DISPLAY	the associated display name
PATH	the value of **DisplayManager.*DISPLAY*.systemPath**
SHELL	the value of **DisplayManager.*DISPLAY*.systemShell**
XAUTHORITY	may be set to an authority file

Note that since *xdm* grabs the keyboard, any other windows will not be able to receive keyboard input. They will be able to interact with the mouse, however; beware of potential security holes here. If **DisplayManager.*DISPLAY*.grabServer** is set, *Xsetup* will not be able to connect to the display at all. Resources for this program can be put into the file named by **DisplayManager.*DISPLAY*.resources**.

Here is a sample *Xsetup* script:

```
#!/bin/sh
# Xsetup_0 – setup script for one workstation
xcmsdb < /usr/X11R6/lib/monitors/alex.0
xconsole –geometry 480x130–0–0 –notify –verbose –exitOnFail &
```

AUTHENTICATION WIDGET

The authentication widget reads a name/password pair from the keyboard. Nearly every imaginable parameter can be controlled with a resource. Resources for this widget should be put into the file named by **DisplayManager.*DISPLAY*.resources**. All of these have reasonable default values, so it is not necessary to specify any of them.

xlogin.Login.width, xlogin.Login.height, xlogin.Login.x, xlogin.Login.y
The geometry of the Login widget is normally computed automatically. If you wish to position it elsewhere, specify each of these resources.

xlogin.Login.foreground
The color used to display the typed-in user name.

xlogin.Login.font
The font used to display the typed-in user name.

xlogin.Login.greeting
A string which identifies this window. The default is "X Window System."

xlogin.Login.unsecureGreeting
When X authorization is requested in the configuration file for this display and none is in use, this greeting replaces the standard greeting. The default is ‘‘This is an unsecure session’’

xlogin.Login.greetFont
The font used to display the greeting.

xlogin.Login.greetColor
The color used to display the greeting.

xlogin.Login.namePrompt
The string displayed to prompt for a user name. *Xrdb* strips trailing white space from resource values, so to add spaces at the end of the prompt (usually a nice thing), add spaces escaped with backslashes. The default is ‘‘Login: ’’

xlogin.Login.passwdPrompt
The string displayed to prompt for a password. The default is ‘‘Password: ’’

xlogin.Login.promptFont
The font used to display both prompts.

xlogin.Login.promptColor
The color used to display both prompts.

xlogin.Login.fail
A message which is displayed when the authentication fails. The default is ‘‘Login incorrect’’

xlogin.Login.failFont
The font used to display the failure message.

xlogin.Login.failColor
The color used to display the failure message.

xlogin.Login.failTimeout
The number of seconds that the failure message is displayed. The default is 30.

xlogin.Login.translations
This specifies the translations used for the login widget. Refer to the X Toolkit documentation for a complete discussion on translations. The default translation table is:

```
Ctrl<Key>H:          delete-previous-character() \n\
Ctrl<Key>D:          delete-character() \n\
Ctrl<Key>B:          move-backward-character() \n\
Ctrl<Key>F:          move-forward-character() \n\
Ctrl<Key>A:          move-to-begining() \n\
Ctrl<Key>E:          move-to-end() \n\
Ctrl<Key>K:          erase-to-end-of-line() \n\
Ctrl<Key>U:          erase-line() \n\
Ctrl<Key>X:          erase-line() \n\
Ctrl<Key>C:          restart-session() \n\
Ctrl<Key>\\:         abort-session() \n\
<Key>BackSpace:      delete-previous-character() \n\
<Key>Delete:         delete-previous-character() \n\
<Key>Return:         finish-field() \n\
<Key>:               insert-char() \
```

The actions which are supported by the widget are:

delete-previous-character
: Erases the character before the cursor.

delete-character
: Erases the character after the cursor.

move-backward-character
: Moves the cursor backward.

move-forward-character
: Moves the cursor forward.

move-to-begining
: (Apologies about the spelling error.) Moves the cursor to the beginning of the editable text.

move-to-end
: Moves the cursor to the end of the editable text.

erase-to-end-of-line
: Erases all text after the cursor.

erase-line
: Erases the entire text.

finish-field
: If the cursor is in the name field, proceeds to the password field; if the cursor is in the password field, checks the current name/password pair. If the name/password pair is valid, *xdm* starts the session. Otherwise the failure message is displayed and the user is prompted again.

abort-session
: Terminates and restarts the server.

abort-display
: Terminates the server, disabling it. This action is not accessible in the default configuration. There are various reasons to stop *xdm* on a system console, such as when shutting the system down, when using *xdmshell*, to start another type of server, or to generally access the console. Sending *xdm* a SIGHUP will restart the display. See the section **Controlling XDM**.

restart-session
: Resets the X server and starts a new session. This can be used when the resources have been changed and you want to test them or when the screen has been overwritten with system messages.

insert-char
: Inserts the character typed.

set-session-argument
: Specifies a single word argument which is passed to the session at startup. See the section **Session Program**.

allow-all-access
: Disables access control in the server. This can be used when the .Xauthority file cannot be created by *xdm*. Be very careful using this; it might be better to disconnect the machine from the network before doing this.

STARTUP PROGRAM

The *Xstartup* program is run as root when the user logs in. It is typically a shell script. Since it is run as root, *Xstartup* should be very careful about security. This is the place to put commands which add entries to */etc/utmp* (the *sessreg* program may be useful here), mount users' home directories from file servers, or abort the session if logins are not allowed.

In addition to any specified by **DisplayManager.exportList**, the following environment variables are passed:

DISPLAY	the associated display name
HOME	the initial working directory of the user
LOGNAME	the user name
USER	the user name
PATH	the value of **DisplayManager.***DISPLAY***.systemPath**
SHELL	the value of **DisplayManager.***DISPLAY***.systemShell**
XAUTHORITY	may be set to an authority file

No arguments are passed to the script. *Xdm* waits until this script exits before starting the user session. If the exit value of this script is non-zero, *xdm* discontinues the session and starts another authentication cycle.

The sample *Xstartup* file shown here prevents login while the file */etc/nologin* exists. Thus this is not a complete example, but simply a demonstration of the available functionality.

Here is a sample *Xstartup* script:

```
#!/bin/sh
#
# Xstartup
#
# This program is run as root after the user is verified
#
if [ −f /etc/nologin ]; then
        xmessage −file /etc/nologin −timeout 30 −center
        exit 1
fi
sessreg −a −l $DISPLAY −x /usr/X11R6/lib/xdm/Xservers $LOGNAME
/usr/X11R6/lib/xdm/GiveConsole
exit 0
```

SESSION PROGRAM

The *Xsession* program is the command which is run as the user's session. It is run with the permissions of the authorized user.

In addition to any specified by **DisplayManager.exportList**, the following environment variables are passed:

DISPLAY	the associated display name
HOME	the initial working directory of the user
LOGNAME	the user name
USER	the user name
PATH	the value of **DisplayManager.***DISPLAY***.userPath**

SHELL	the user's default shell (from *getpwnam*)
XAUTHORITY	may be set to a non-standard authority file
KRB5CCNAME	may be set to a Kerberos credentials cache name

At most installations, *Xsession* should look in $HOME for a file *.xsession,* which contains commands that each user would like to use as a session. *Xsession* should also implement a system default session if no user-specified session exists. See the section **Typical Usage**.

An argument may be passed to this program from the authentication widget using the 'set-session-argument' action. This can be used to select different styles of session. One good use of this feature is to allow the user to escape from the ordinary session when it fails. This allows users to repair their own *.xsession* if it fails, without requiring administrative intervention. The example following demonstrates this feature.

This example recognizes the special ''failsafe'' mode, specified in the translations in the *Xresources* file, to provide an escape from the ordinary session. It also requires that the .xsession file be executable so we don't have to guess what shell it wants to use.

```
#!/bin/sh
#
# Xsession
#
# This is the program that is run as the client
# for the display manager.

case $# in
1)
	case $1 in
	failsafe)
		exec xterm −geometry 80x24−0−0
		;;
	esac
esac

startup=$HOME/.xsession
resources=$HOME/.Xresources

if [ −f "$startup" ]; then
	exec "$startup"
else
	if [ −f "$resources" ]; then
		xrdb −load "$resources"
	fi
	twm &
	xman −geometry +10−10 &
	exec xterm −geometry 80x24+10+10 −ls
fi
```

The user's *.xsession* file might look something like this example. Don't forget that the file must have execute permission.

```
#! /bin/csh
```

```
# no −f in the previous line so .cshrc gets run to set $PATH
twm &
xrdb −merge "$HOME/.Xresources"
emacs −geometry +0+50 &
xbiff −geometry −430+5 &
xterm −geometry −0+50 -ls
```

RESET PROGRAM

Symmetrical with *Xstartup*, the *Xreset* script is run after the user session has terminated. Run as root, it should contain commands that undo the effects of commands in *Xstartup,* removing entries from */etc/utmp* or unmounting directories from file servers. The environment variables that were passed to *Xstartup* are also passed to *Xreset*.

A sample *Xreset* script:

```
#!/bin/sh
#
# Xreset
#
# This program is run as root after the session ends
#
sessreg −d −l $DISPLAY −x /usr/X11R6/lib/xdm/Xservers $LOGNAME
/usr/X11R6/lib/xdm/TakeConsole
exit 0
```

CONTROLLING THE SERVER

Xdm controls local servers using POSIX signals. SIGHUP is expected to reset the server, closing all client connections and performing other cleanup duties. SIGTERM is expected to terminate the server. If these signals do not perform the expected actions, the resources **DisplayManager.***DISPLAY***.resetSignal** and **DisplayManager.***DISPLAY***.termSignal** can specify alternate signals.

To control remote terminals not using XDMCP, *xdm* searches the window hierarchy on the display and uses the protocol request KillClient in an attempt to clean up the terminal for the next session. This may not actually kill all of the clients, as only those which have created windows will be noticed. XDMCP provides a more sure mechanism; when *xdm* closes its initial connection, the session is over and the terminal is required to close all other connections.

CONTROLLING XDM

Xdm responds to two signals: SIGHUP and SIGTERM. When sent a SIGHUP, *xdm* rereads the configuration file, the access control file, and the servers file. For the servers file, it notices if entries have been added or removed. If a new entry has been added, *xdm* starts a session on the associated display. Entries which have been removed are disabled immediately, meaning that any session in progress will be terminated without notice and no new session will be started.

When sent a SIGTERM, *xdm* terminates all sessions in progress and exits. This can be used when shutting down the system.

Xdm attempts to mark its various sub-processes for *ps*(1) by editing the command line argument list in place. Because *xdm* can't allocate additional space for this task, it is useful to start *xdm* with a reasonably long command line (using the full path name should be enough). Each process which is servicing a display is marked **−***display.*

ADDITIONAL LOCAL DISPLAYS

To add an additional local display, add a line for it to the *Xservers* file. (See the section **Local Server Specification**.)

Examine the display-specific resources in *xdm-config* (e.g., **DisplayManager._0.authorize**) and consider which of them should be copied for the new display. The default *xdm-config* has all the appropriate lines for displays **:0** and **:1**.

OTHER POSSIBILITIES

You can use *xdm* to run a single session at a time, using the 4.3 *init* options or other suitable daemon by specifying the server on the command line:

```
xdm −server ":0 SUN-3/60CG4 local /usr/X11R6/bin/X :0"
```

Or, you might have a file server and a collection of X terminals. The configuration for this is identical to the sample above, except the *Xservers* file would look like

```
extol:0 VISUAL-19 foreign
exalt:0 NCD-19 foreign
explode:0 NCR-TOWERVIEW3000 foreign
```

This directs *xdm* to manage sessions on all three of these terminals. See the section **Controlling Xdm** for a description of using signals to enable and disable these terminals in a manner reminiscent of *init*(8).

LIMITATIONS

One thing that *xdm* isn't very good at doing is coexisting with other window systems. To use multiple window systems on the same hardware, you'll probably be more interested in *xinit.*

FILES

<XRoot>/lib/X11/xdm/xdm-config
  the default configuration file

$HOME/.Xauthority user authorization file where *xdm* stores keys for clients to read

<XRoot>/lib/X11/xdm/chooser
  the default chooser

<XRoot>/bin/xrdb the default resource database loader

<XRoot>/bin/X the default server

<XRoot>/bin/xterm the default session program and failsafe client

<XRoot>/lib/X11/xdm/A<display>−<suffix>
  the default place for authorization files

/tmp/K5C<display> Kerberos credentials cache Note: <XRoot> refers to the root of the X11 install tree.

SEE ALSO

X(1), *xinit*(1), *xauth*(1), *Xsecurity*(1), *sessreg*(1), *Xserver*(1),
X Display Manager Control Protocol

AUTHOR

Keith Packard, MIT X Consortium

NAME

xdpyinfo − display information utility for X

SYNOPSIS

xdpyinfo [−display *displayname*] [−queryExtensions] [−ext *extension-name*]

DESCRIPTION

Xdpyinfo is a utility for displaying information about an X server. It is used to examine the capabilities of a server, the predefined values for various parameters used in communicating between clients and the server, and the different types of screens and visuals that are available.

By default, numeric information (opcode, base event, base error) about protocol extensions is not displayed. This information can be obtained with the **−queryExtensions** option. Use of this option on servers that dynamically load extensions will likely cause all possible extensions to be loaded, which can be slow and can consume significant server resources.

Detailed information about a particular extension is displayed with the **−ext** *extensionName* option. If *extensionName* is **all**, information about all extensions supported by both *xdpyinfo* and the server is displayed.

ENVIRONMENT

DISPLAY

To get the default host, display number, and screen.

SEE ALSO

X(1), xwininfo(1), xprop(1), xrdb(1)

AUTHOR

Jim Fulton, MIT X Consortium

NAME

xfd – display all the characters in an X font

SYNOPSIS

xfd [–options ...] **–fn** *fontname*

DESCRIPTION

The *xfd* utility creates a window containing the name of the font being displayed, a row of command buttons, several lines of text for displaying character metrics, and a grid containing one glyph per cell. The characters are shown in increasing order from left to right, top to bottom. The first character displayed at the top left will be character number 0 unless the **–start** option has been supplied in which case the character with the number given in the **–start** option will be used.

The characters are displayed in a grid of boxes, each large enough to hold any single character in the font. Each character glyph is drawn using the PolyText16 request (used by the *Xlib* routine **XDrawString16**). If the **–box** option is given, a rectangle will be drawn around each character, showing where an ImageText16 request (used by the *Xlib* routine **XDrawImageString16**) would cause background color to be displayed.

The origin of each glyph is normally set so that the character is drawn in the upper left hand corner of the grid cell. However, if a glyph has a negative left bearing or an unusually large ascent, descent, or right bearing (as is the case with *cursor* font), some character may not appear in their own grid cells. The **–center** option may be used to force all glyphs to be centered in their respective cells.

All the characters in the font may not fit in the window at once. To see the next page of glyphs, press the *Next* button at the top of the window. To see the previous page, press *Prev*. To exit *xfd*, press *Quit*.

Individual character metrics (index, width, bearings, ascent and descent) can be displayed at the top of the window by clicking on the desired character.

The font name displayed at the top of the window is the full name of the font, as determined by the server. See *xlsfonts* for ways to generate lists of fonts, as well as more detailed summaries of their metrics and properties.

OPTIONS

xfd accepts all of the standard toolkit command line options along with the additional options listed below:

–fn *font* This option specifies the font to be displayed. This can also be set with the FontGrid **font** resource. A font must be specified.

–box This option indicates that a box should be displayed outlining the area that would be filled with background color by an ImageText request. This can also be set with the FontGrid **boxChars** resource. The default is False.

–center This option indicates that each glyph should be centered in its grid. This can also be set with the FontGrid **centerChars** resource. The default is False.

–start *number*
This option specifies the glyph index of the upper left hand corner of the grid. This is used to view characters at arbitrary locations in the font. This can also be set with the FontGrid **startChar** resource. The default is 0.

–bc *color*
This option specifies the color to be used if ImageText boxes are drawn. This can also be set with the FontGrid **boxColor** resource.

−rows *numrows*
This option specifies the number of rows in the grid. This can also be set with the FontGrid **cellRows** resource.

−columns *numcols*
This option specifies the number of columns in the grid. This can also be set with the FontGrid **cellColumns** resource.

WIDGETS

In order to specify resources, it is useful to know the widgets which compose *xfd*. In the notation below, indentation indicates hierarchical structure. The widget class name is given first, followed by the widget instance name. The application class name is Xfd.

```
Xfd  xfd
        Paned  pane
                Label  fontname
                Box  box
                        Command  quit
                        Command  prev
                        Command  next
                Label  select
                Label  metrics
                Label  range
                Label  start
                Form  form
                        FontGrid  grid
```

FONTGRID RESOURCES

The FontGrid widget is an application-specific widget, and a subclass of the Simple widget in the Athena widget set. The effects and instance names of this widget's resources are given in the **OPTIONS** section. Capitalize the first letter of the resource instance name to get the corresponding class name.

APPLICATION SPECIFIC RESOURCES

The instance names of the application specific resources are given below. Capitalize the first letter of the resource instance name to get the corresponding class name. These resources are unlikely to be interesting unless you are localizing xfd for a different language.

selectFormat
Specifies a printf-style format string used to display information about the selected character. The default is "character 0x%02x%02x (%u,%u) (%#o,%#o)". The arguments that will come after the format string are char.byte1, char.byte2, char.byte1, char.byte2, char.byte1, char.byte2. char.byte1 is byte 1 of the selected character. char.byte2 is byte 2 of the selected character.

metricsFormat
Specifies a printf-style format string used to display character metrics. The default is "width %d; left %d, right %d; ascent %d, descent %d (font %d, %d)". The arguments that will come after the format string are the character metrics width, lbearing, rbearing, character ascent, character descent, font ascent, and font descent.

rangeFormat
Specifies a printf-style format string used to display the range of characters currently being displayed. The default is "range: 0x%02x%02x (%u,%u) thru 0x%02x%02x (%u,%u)". The arguments that will come after the format string are the following fields from the XFontStruct that is returned from opening the font: min_byte1, min_char_or_byte2, min_byte1, min_char_or_byte2, max_byte1, max_char_or_byte2, max_byte1, max_char_or_byte2.

startFormat
Specifies a printf-style format string used to display information about the character at the upper left corner of the font grid. The default is "upper left: 0x%04x (%d,%d)". The arguments that will come after the format string are the new character, the high byte of the new character, and the low byte of the new character.

nocharFormat
Specifies a printf-style format string to display when the selected character does not exist. The default is "no such character 0x%02x%02x (%u,%u) (%#o,%#o)". The arguments that will come after the format string are the same as for the **selectFormat** resource.

SEE ALSO

X(1), xlsfonts(1), xrdb(1), xfontsel(1), *X Logical Font Description Conventions*

BUGS

The program should skip over pages full of non-existent characters.

AUTHOR

Jim Fulton, MIT X Consortium; previous program of the same name by Mark Lillibridge, MIT Project Athena.

NAME

xfindproxy - locate proxy services

SYNOPSIS

xfindproxy **−manager** *managerAddr* **−name** *serviceName* **−server** *serverAddr* [**−auth**] [**−host** *hostAddr*] [**−options** *opts*]

DESCRIPTION

xfindproxy is a program used to locate available proxy services. It utilizes the Proxy Management Protocol to communicate with a proxy manager. The proxy manager keeps track of all available proxy services, starts new proxies when necessary, and makes sure that proxies are shared whenever possible.

The **−manager** argument is required, and it specifies the network address of the proxy manager. The format of the address is a standard ICE network id (for example, "tcp/blah.x.org:6500").

The **−name** argument is required, and it specifies the name of the desired proxy service (for example, "LBX"). The name is case insensitive.

The **−server** argument is also required, and it specifies the address of the target server. The format of the address is specific to the proxy service specified with the -name argument. For example, for a proxy service of "LBX", the address would be an X display address (e.g, "blah.x.org:0").

The **−auth** argument is optional. If specified, xfindproxy will read 2 lines from standard input. The first line is an authorization/authentication name. The second line is the authorization/authentication data in hex format (the same format used by xauth). xfindproxy will pass this auth data to the proxy, and in most cases, will be used by the proxy to authorize/authenticate itself to the target server.

The **−host** argument is optional. If xfindproxy starts a new proxy service, it will pass the host specified. The proxy may choose to restrict all connections to this host. In the event that xfindproxy locates an already existing proxy, the host will be passed, but the semantics of how the proxy uses this host are undefined.

The **−options** argument is optional. If xfindproxy starts a new proxy service, it will pass any options specified. The semantics of the options are specific to each proxy server and are not defined here. In the event that xfindproxy locates an already existing proxy, the options will be passed, but the semantics of how the proxy uses these options are undefined.

If xfindproxy is successful in obtaining a proxy address, it will print it to stdout. The format of the proxy address is specific to the proxy service being used. For example, for a proxy service of "LBX", the proxy address would be the X display address of the proxy (e.g, "blah.x.org:63").

If xfindproxy is unsuccessful in obtaining a proxy address, it will print an error to stderr.

SEE ALSO

proxymngr (1), Proxy Management Protocol spec V1.0

AUTHOR

Ralph Mor, X Consortium

NAME

xfs – X font server

SYNOPSIS

xfs [−config *configuration_file*] [−port *tcp_port*]

DESCRIPTION

Xfs is the X Window System font server. It supplies fonts to X Window System display servers.

STARTING THE SERVER

The server is usually run by a system administrator, and started via boot files like */etc/rc.local*. Users may also wish to start private font servers for specific sets of fonts.

OPTIONS

−config configuration_file
Specifies the configuration file the font server will use.

−ls listen-socket
Specifies a file descriptor which is already set up to be used as the listen socket. This option is only intended to be used by the font server itself when automatically spawning another copy of itself to handle additional connections.

−port tcp_port
Specifies the TCP port number on which the server will listen for connections. The default port number is 7100.

SIGNALS

SIGTERM
This causes the font server to exit cleanly.

SIGUSR1
This signal is used to cause the server to re-read its configuration file.

SIGUSR2
This signal is used to cause the server to flush any cached data it may have.

SIGHUP This signal is used to cause the server to reset, closing all active connections and re-reading the configuration file.

CONFIGURATION

The configuration language is a list of keyword and value pairs. Each keyword is followed by an '=' and then the desired value.

Recognized keywords include:

catalogue (list of string)
Ordered list of font path element names. Use of the keyword "catalogue" is very misleading at present, the current implementation only supports a single catalogue ("all"), containing all of the specified fonts.

alternate-servers (list of string)
List of alternate servers for this font server.

client-limit (cardinal)
Number of clients this font server will support before refusing service. This is useful for tuning the load on each individual font server.

clone-self (boolean)
Whether this font server should attempt to clone itself when it reachs the client-

limit.

default-point-size (cardinal)
: The default pointsize (in decipoints) for fonts that don't specify. The default is 120.

default-resolutions (list of resolutions)
: Resolutions the server supports by default. This information may be used as a hint for pre-rendering, and substituted for scaled fonts which do not specify a resolution. A resolution is a comma-separated pair of x and y resolutions in pixels per inch. Multiple resolutions are separated by commas.

error-file (string)
: Filename of the error file. All warnings and errors will be logged here.

port (cardinal)
: TCP port on which the server will listen for connections.

use-syslog (boolean)
: Whether syslog(3) (on supported systems) is to be used for errors.

deferglyphs (string)
: Set the mode for delayed fetching and caching of glyphs. Value is "none", meaning deferred glyphs is disabled, "all", meaning it is enabled for all fonts, and "16", meaning it is enabled only for 16-bits fonts.

EXAMPLE

```
#
# sample font server configuration file
#

# allow a max of 10 clients to connect to this font server
client-limit = 10

# when a font server reaches its limit, start up a new one
clone-self = on

# alternate font servers for clients to use
alternate-servers = hansen:7101,hansen:7102

# where to look for fonts
# the first is a set of Speedo outlines, the second is a set of
# misc bitmaps and the last is a set of 100dpi bitmaps
#
catalogue = /usr/X11R6/lib/X11/fonts/speedo,
        /usr/X11R6/lib/X11/fonts/misc,
        /usr/X11R6/lib/X11/fonts/100dpi/

# in 12 points, decipoints
default-point-size = 120

# 100 x 100 and 75 x 75
default-resolutions = 100,100,75,75
use-syslog = off
```

FONT SERVER NAMES

One of the following forms can be used to name a font server that accepts TCP connections:

tcp/*hostname*:*port*
tcp/*hostname*:*port*/*cataloguelist*

The *hostname* specifies the name (or decimal numeric address) of the machine on which the font server is running. The *port* is the decimal TCP port on which the font server is listening for connections. The *cataloguelist* specifies a list of catalogue names, with '+' as a separator.

Examples: *tcp/fs.x.org:7100*, *tcp/18.30.0.212:7101/all*.

One of the following forms can be used to name a font server that accepts DECnet connections:

decnet/*nodename*::font$*objname*
decnet/*nodename*::font$*objname*/*cataloguelist*

The *nodename* specifies the name (or decimal numeric address) of the machine on which the font server is running. The *objname* is a normal, case-insensitive DECnet object name. The *cataloguelist* specifies a list of catalogue names, with '+' as a separator.

Examples: *DECnet/SRVNOD::FONT$DEFAULT*, *decnet/44.70::font$special/symbols*.

SEE ALSO

X(1), *The X Font Service Protocol*,
Font server implementation overview

BUGS

Multiple catalogues should be supported.

AUTHORS

Dave Lemke, Network Computing Devices, Inc
Keith Packard, Massachusetts Institute of Technology

NAME

xfwp - X firewall proxy

SYNOPSIS

xfwp [option ...]

COMMAND LINE OPTIONS

The command line options that can be specified are:

–cdt *num_secs*
Used to override the default time-to-close (604800 seconds) for xfwp client data connections on which there is no activity (connections over which X protocol is already being relayed by xfwp)

–clt *num_secs*
Used to override the default time-to-close (86400 seconds) for xfwp client listen ports (ports on xfwp to which X clients first connect when trying to reach an X server)

–pdt *num_secs*
Used to override the default time-to-close (3600 seconds) for Proxy Manager connections on which there is no activity

–config *file_name*
Used to specify the configuration the name of the configuration file

–pmport *port_number*
Used to override the default port address (4444) for proxy manager connections

–verify Used to display the configuration file rule that was actually matched for each service request

–logfile *file_name*
Used to specify the name of a file where audit information should be logged. The format of a logged entry is: time of day; event code; source IP address; destination IP address; and configuration rule number. The event codes are: "0" for a successful connection; "1" if a connection is denied because of a configuration rule; and "2" if a connection is denied because of an authorization failure. If the event code is "1", and a configuration file is used, the configuration rule number is the line number of the configuration file where the match was made (see the section CONFIGURATION FILE for more information). If the event code is not "1", or if no configuration file is used, the configuration rule number is "-1".

–loglevel *{0,1}*
Used to specify the amount of audit detail that should be logged. If "0", all connections are logged. If "1", only unsuccessful connections are logged.

–max_pm_conns *num_connections*
Used to specify the maximum number of Proxy Manager connections. The default is 10.

–max_pm_conns *num_connections*
Used to specify the maximum number of X server connections. The default is 100.

DESCRIPTION

The X firewall proxy (xfwp) is an application layer gateway proxy that may be run on a network firewall host to forward X traffic across the firewall. Used in conjunction with the X server Security extension and authorization checking, xfwp constitutes a safe, simple, and reliable mechanism both to hide the addresses of X servers located on the Intranet and to enforce a server connection policy. Xfwp cannot protect against mischief originating on the Intranet; however, when properly configured it can guarantee that only trusted clients originating on authorized external Internet hosts will be allowed inbound access to local X servers.

To use xfwp there must be an X proxy manager running in the local environment which has been configured at start-up to know the location of the xfwp. [NOTE: There may be more than one xfwp running in a local environment; see notes below on load balancing for further discussion.] Using the xfindproxy utility (which relays its requests through the proxy manager) a user asks xfwp to allocate a client listen port for a particular X server, which is internally associated with all future connection requests for that server. This client listen port address is returned by the proxy manager through xfindproxy. The xfwp hostname and port number is then passed out-of-band (i.e., via a Web browser) to some remote X client, which will subsequently connect to xfwp instead of to the target X server.

When an X client connection request appears on one of xfwp's listen ports, xfwp connects to the X server associated with this listen port and performs authorization checks against the server as well as against its own configurable access control list for requesting clients. If these checks fail, or if the requested server does not support the X Security Extension, the client connection is refused. Otherwise, the connection is accepted and all ensuing data between client and server is relayed by xfwp until the client terminates the connection or, in the case of an inactive client, until a configured timeout period is exceeded. Xfwp is designed to block while waiting for activity on its connections, thereby minimizing demand for system cycles.

If xfwp is run without a configuration file and thus no sitepolicy is defined, if xfwp is using an X server where xhost + has been run to turn off host-based authorization checks, when a client tries to connect to this X server via xfwp, the X server will deny the connection. If xfwp does not define a sitepolicy, host-based authorization must be turned on for clients to connect to an X server via the xfwp.

INTEROPERATION WITH IP PACKET-FILTERING ROUTERS

The whole purpose of the xfwp is to provide reliable control over access to Intranet X servers by clients originating outside the firewall. At the present time, such access control is typically achieved by firewall configurations incorporating IP packet-filtering routers. Frequently, the rules for such filters deny access to X server ports (range 6000 - 6xxx) for all Intranet host machines.

In order for xfwp to do its job, restrictions on access for ports 6001 - 6xxx must be removed from the rule-base of the IP packet-filtering router. [NOTE: xfwp only assigns ports in the range beginning with 6001; access to port 6000 on all Intranet hosts may continue to be denied.] This does not mean the Intranet firewall will be opened for indiscriminate entry by X clients. Instead, xfwp supports a fully configurable rule-based access control system, similar to that of the IP packet-filter router itself. Xfwp in effect adds another level of packet-filtering control which is fully configurable and applies specifically to X traffic. See section entitled CONFIGURATION FILE, below, for further details.

INSTALLATION, SETUP AND TROUBLESHOOTING

Xfwp is typically run as a background process on the Intranet firewall host. It can be launched using any of the command-line options described above. As noted above, xfwp works only in conjunction with proxy manager and the xfindproxy utility. It can also be configured to support a user-defined X server site security policy, in which the X server is required to indicate to xfwp whether or not it supports the particular policy. Consult the X server man pages for further information on these components. Xfwp diagnostics can be turned on by compiling with the -DDEBUG switch. Connection status can be recorded by using the -logfile and -loglevel command line options.

PERFORMANCE, LOAD BALANCING AND RESOURCE MANAGEMENT

Xfwp manages four different kinds of connections: proxy manager (PM) data, X client listen, X client data, and X server. The sysadmin employing xfwp must understand how the resources for each of these connection types are allocated and reclaimed by xfwp in order to optimize the availability of xfwp service.

Each connection-type has a default number of allocation slots and a default timeout. The number of allocation slots for PM connections and X server connections is configurable via command line options. Connection timeouts are also configurable via command line options. Each connection timeout represents the period the connection will be allowed to remain open in the absence of any activity on that connection. Whenever there is activity on a connection, the time-to-close is automatically reset. The default distribution of total process connection slots across the four connection types, as well as the choice of default timeouts for the connection types, is governed by a number of assumptions embedded in the xfwp use model.

The default number of PM connections is 10 and the default duration for PM connections is 3,600 seconds (1 hour) for each connection after time of last activity. At start-up, xfwp listens for PM connection requests on any non-reserved port (default of 4444 if not specified on the xfwp command-line). The PM normally connects to xfwp only when a call is made to the PM with xfindproxy. Thereafter, the PM remains connected to xfwp, even after the messaging between them has been completed, for the default connection duration period. In some cases this may result in depletion of available PM connection slots. If the sysadmin expects connections to a single xfwp from many PM's, xfwp should be started using the -pdt command line option, with a timeout value reflecting the desired duration that inactive connections will be permitted to remain open.

Xfwp client listeners are set up by a call to xfindproxy and continue to listen for X client connection requests for a default duration of 86,400 seconds (24 hours) from the point of last activity. After this time they are automatically closed and their fd's recovered for future allocation. In addressing the question of how to choose some alternative timeout value which will guarantee the availability of client listen ports, sysadmins should take into consideration the expected delay between the time when the listener was allocated (using xfindproxy) and the time when a client actually attempts to connect to xfwp, as well the likelihood that client listeners will be re-used after the initial client data connection is closed.

Each client connection is allocated a default lifetime of 604,800 seconds (7 * 24 hours) from the point when it last saw activity. After this time it is automatically closed and its fd's recovered for future allocation. Because server connections are not actually established until a connection request from a remote X client arrives at one of the xfwp's client listen ports, the client data timeout applies both to client-xfwp connections as well as to xfwp-server connections. If the system administrator expects many client data connections through xfwp, an overriding of the default timeout should be considered.

CONFIGURATION FILE

The xfwp configuration file resides on the xfwp host machine and is used to determine whether X client data connection requests will be permitted or denied. The path to the file is specified at start-up time. If no configuration file is specified, all X client data connection requests routed through xfwp will be by default permitted, assuming that other X server authorization checks are successful. If a configuration file is supplied but none of its entries matches the connection request then the connection is by default denied.

If a line in the configuration file begins with the '#' character or a new-line character, the line is ignored and the evaluator will skip the line.

The configuration file supports two entirely independent authorization checks: one which is performed by xfwp itself, and a second which is the result of xfwp's querying the target X server. For the first of these, the configuration file employs a syntax and semantic similar to that of IP packet-filtering routers. It contains zero or more source-destination rules of the following form:

{permit | deny} <src> <src mask> [<dest> <dest mask> [<operator> <service>]]

permit/deny
: the keywords "permit" or "deny" indicate whether the rule will enable or disable access, respectively

src
: the IP address against the host who originated the connection request will be matched, expressed in IP format (x.x.x.x)

src mask
: a subnet mask, also in IP format, for further qualifying the source mask. Bits set in the mask indicate bits of the incoming address to be *ignored* when comparing to the specified src

dest
: the IP address against which the destination of the incoming connection request (i.e. the host IP of the X server to which the incoming client is attempting to connect) will be matched

dest mask
: a subnet mask, also in IP format, for further qualifying the destination mask. Bits set in the mask indicate bits of the destination address to be *ignored* when comparing to the specified dest

operator
: always "eq" (if the service field is not NULL)

service
: one of the following three strings: "pm", "fp", or "cd", corresponding to proxy manager, xfindproxy, or client data, respectively

For the second type of authorization check, the configuration file contains zero or more site policy rules of the following form:

{require | disallow} sitepolicy <site_policy>

require
: specifies that the X server *must* be configured with *at least one* of the corresponding site policies, else it must refuse the connection.

disallow
: specifies that the X server *must not* be configured with *any* of the corresponding site policies, else it must refuse the connection.

sitepolicy a required keyword

<site_policy> specifies the policy string. The string may contain any combination of alphanumeric characters subject only to interpretation by the target X server

RULES FOR EVALUATING THE XFWP CONFIGURATION

For the first type of configurable authorization checking, access can be permitted or denied for each connection type based upon source and, optionally, destination and service. Each file entry must at a minimum specify the keyword ‘‘permit’’ or ‘‘deny’’ and the two source fields. The destination and service fields can be used to provide finer-grained access control if desired.

The algorithm for rule-matching is as follows:

```
while (more entries to check)
{
  if ((<originator IP> AND (NOT <src mask>)) == src)
    [if ((<dest X server IP> AND (NOT <dest mask>)) == dest)]
      [if (service fields present and matching)]
        do either permit or deny connection depending on keyword
  else
    continue
}
if (no rule matches)
  deny connection
```

If a permit or deny rule does not specify a service and operation, then the rule applies to all services. If a configuration file is specified and it contains at least one valid deny or permit rule, then a host that is not explicitly permitted will be denied a connection.

Site policy configuration checking constitutes a separate (and X server only) authorization check on incoming connection requests. Any number of require or disallow rules may be specified, but all rules must be of the same type; that is, a single rule file cannot have both ‘‘require’’ and ‘‘disallow’’ keywords. The algorithm for this check is as follows:

```
if (X server recognizes any of the site policy strings)
  if (keyword == require)
    permit connection
  else
    deny connection
else
  if (keyword == require)
    deny connection
  else
    permit connection
```

The site policy check is performed by xfwp only if the source-destination rules permit the connection.

EXAMPLES

```
# if and only if server supports one of these policies then authorize
# connections, but still subject to applicable rule matches
#
```

```
require sitepolicy policy1
require sitepolicy policy2
#
# deny pm connections originating on 8.7.6.5 [NOTE:  If pm service
# is explicitly qualified, line must include destination fields as
# shown.]
#
deny  8.7.6.5  0.0.0.0  0.0.0.0  255.255.255.255  eq  pm
#
# permit xfindproxy X server connects to anywhere [NOTE:  If
# fp service is explicitly qualified, line must include source fields
# as shown.]
#
permit  0.0.0.0  255.255.255.255   0.0.0.0  255.255.255.255  eq  fp
#
# permit all connection types originating from the 192.0.0.0
# IP domain only
#
permit  192.0.0.0   0.255.255.255
```

Care should be taken that source-destination rules are written in the correct order, as the first matching rule will be applied. In addition to parser syntax checking, a special command-line switch (-verify) has been provided to assist the sysadmin in determining which rule was actually matched.

BUGS

Xfwp should check server site policy and security extension before allocating a listen port.

SEE ALSO

xfindproxy (1), Proxy Management Protocol spec V1.0, proxymngr(1), Xserver(1)

AUTHOR

Reed Augliere, consulting to X Consortium, Inc.

NAME

xhost − server access control program for X

SYNOPSIS

xhost [[+−]name ...]

DESCRIPTION

The *xhost* program is used to add and delete host names or user names to the list allowed to make connections to the X server. In the case of hosts, this provides a rudimentary form of privacy control and security. It is only sufficient for a workstation (single user) environment, although it does limit the worst abuses. Environments which require more sophisticated measures should implement the user-based mechanism or use the hooks in the protocol for passing other authentication data to the server.

OPTIONS

Xhost accepts the following command line options described below. For security, the options that effect access control may only be run from the "controlling host". For workstations, this is the same machine as the server. For X terminals, it is the login host.

−help Prints a usage message.

[+]*name* The given *name* (the plus sign is optional) is added to the list allowed to connect to the X server. The name can be a host name or a user name.

−*name* The given *name* is removed from the list of allowed to connect to the server. The name can be a host name or a user name. Existing connections are not broken, but new connection attempts will be denied. Note that the current machine is allowed to be removed; however, further connections (including attempts to add it back) will not be permitted. Resetting the server (thereby breaking all connections) is the only way to allow local connections again.

+ Access is granted to everyone, even if they aren't on the list (i.e., access control is turned off).

− Access is restricted to only those on the list (i.e., access control is turned on).

nothing If no command line arguments are given, a message indicating whether or not access control is currently enabled is printed, followed by the list of those allowed to connect. This is the only option that may be used from machines other than the controlling host.

NAMES

A complete name has the syntax ‘‘family:name’’ where the families are as follows:

inet	Internet host
dnet	DECnet host
nis	Secure RPC network name
krb	Kerberos V5 principal
local	contains only one name, the empty string

The family is case insensitive. The format of the name varies with the family.

When Secure RPC is being used, the network independent netname (e.g., "nis:unix.*uid@domainname*") can be specified, or a local user can be specified with just the username and a trailing at-sign (e.g., "nis:pat@").

For backward compatibility with pre-R6 *xhost*, names that contain an at-sign (@) are assumed to be in the nis family. Otherwise the inet family is assumed.

DIAGNOSTICS

For each name added to the access control list, a line of the form "*name* being added to access control list" is printed. For each name removed from the access control list, a line of the form "*name* being removed from access control list" is printed.

FILES

/etc/X*.hosts

SEE ALSO

X(1), Xsecurity(1), Xserver(1), xdm(1)

ENVIRONMENT

DISPLAY
to get the default host and display to use.

BUGS

You can't specify a display on the command line because **−display** is a valid command line argument (indicating that you want to remove the machine named *"display"* from the access list).

The X server stores network addresses, not host names. This is not really a bug. If somehow you change a host's network address while the server is still running, *xhost* must be used to add the new address and/or remove the old address.

AUTHORS

Bob Scheifler, MIT Laboratory for Computer Science,
Jim Gettys, MIT Project Athena (DEC).

NAME

xieperf - XIE server extension test and demo program

SYNTAX

xieperf [-option ...]

DESCRIPTION

The *xieperf* program is based upon R5 *x11perf(1)* , and while not entirely comprehensive in its coverage of the XIE protocol (see BUGS, below), it is intended to be useful in the evaluation of XIE implementations in the areas of protocol adherance and performance. The *xieperf* program includes tests which execute each of the protocol requests and photoflo elements specified by revision 5.0 of the XIE protocol. In addition, *xieperf* provides a set of tests which can be used to validate the detection and transmission of XIE protocol request errors, such as FloMatch, FloValue, and so forth. Finally, *xieperf* provides a customizable demonstration program for XIE.

A test is made up of three components executed in sequence - an initialization function, a test function, and an end function. The initialization function is responsible for allocating and populating test resources, such as photomaps and LUTs, and for creating a stored photoflo which will be executed by the test function. The test function, in most cases, simply executes the stored photoflo for a specified number of repetitions. The end function, which is called following the test function, is used primarily to destroy any non-cachable server resources used by the test, and to free any memory which was dynamically allocated by the client. Some tests, such as -modify1, -await, -abort, and -redefine, perform additional steps within the test function inner loop, as required by the element being tested, or in an attempt to make the test more visually appealing.

Evaluating the performance of individual XIE elements is not as simple as measuring Core X drawing times. The XIE protocol requires elements to be embedded within photoflos in order to be exercised, and the minimum possible photoflo size is two. This implies that it is impossible to measure performance of a single element in isolation - the time it takes to run the flo depends on what other elements exist in the flo. Extrapolating performance of a single elment (or technique) in a flo must be done carefully, on a case-by-case basis, since in general measured element performance depends on input image size, data type, and other factors, all of which can be influenced by upstream flo elements. Note further that the number and type of elements in a flo can be influenced by the visuals available on the display, so even flo-flo comparisons on machines with different visuals must be done with caution.

Many test labels contain an abbreviated pipeline description. For instance. IP/IL/P/ED indicates ImportPhotomap, ImportLUT, Point, and ExportDrawable. Pipelines ending in ED (ExportDrawable) often include hidden elements such as BandExtract, ConvertToIndex, Dither, or Point to match the flo output to the screen visual. Pipelines ending in EP (ExportPhotomap) will result in a blank window.

xieperf is compatible with *x11perfcomp(1),* which is used to compare the outputs of different *xieperf* and *x11perf* runs in a nice, tabular format. In *xieperf* you will need to use the -labels option (see OPTIONS, below), and provide the resulting labels file to *x11perfcomp(1)* to obtain correct output. See the *x11perfcomp(1)* man pages for more details on this.

OPTIONS

xieperf accepts the options listed below:

−display host:dpy

Specifies which display to use.

–images *<path>*
Normally, *xieperf* references image files located in the directory ‘‘images’’, which *xieperf* assumes is located in your current directory. If the ‘‘images’’ directory is not in your current directory, or the file has been renamed, use this option to specify its location.

–timeout *<s>* Some tests require the reception of an event such as FloNotify to continue, and may cause *xieperf* to hang should these events not be received. This option allows the user to specify a timeout value which, if exceeded, will cause *xieperf* to give up waiting for an event, and continue on with the next test in sequence. Should an event timeout, a warning message will be printed to stderr. The default timeout value is 60 seconds.

–sync Runs the tests in synchronous mode.

–script *<file>* Using this option gives the user the ability to run a subset of the available tests and control the number of times the tests are executed on an individual basis. This is thought to be especially useful for those running *xieperf* for demonstration purposes. Using this option causes *xieperf* to read commands specified in a script file, or from stdin if *<file>* is ‘‘-’’. Tests are specified by newline-terminated input lines of the form "command [-reps n] [-repeat m]". Characters following and including "#" are treated as comments. See the -mkscript option.

–repeat *<n>* Repeats each test *n* times (by default each test is run 2 times). This option may be used in script files also, in which case the script file -repeat overrides the command line option.

–time *<s>* Specifies how long in seconds each test should be run (default 5 seconds).

–depth <depth>
Use a visual with <depth> planes per pixel (default is the default visual).

–GrayScale Use a GrayScale visual (default is the default visual).

–PseudoColor Use a PseudoColor visual (default is the default visual).

–StaticGray Use a StaticGray visual (default is the default visual).

–StaticColor Use a StaticColor visual (default is the default visual).

–TrueColor Use a TrueColor visual (default is the default visual).

–DirectColor Use a DirectColor visual (default is the default visual).

–WMSafe If *xieperf* must be run in a window manager environment, use this flag to make *xieperf* aware of this. If specified, *xieperf* will create a window, identical to the size of the root window, and all further windows created by *xieperf* will be transient popup children of this window. If this flag is omitted, *xieperf* will set the override_redirect attribute of all windows to ‘‘True’’ and will also do evil things such as calling XInstallColormap. Using this option will cause the window manager to (hopefully) obey window geometry hints specified by *xieperf*

–showtechs Display a comprehensive list of techniques, by category, indicating which of the techniques are supported by the XIE server.

–showlabels Print test label to screen prior to calling any of the test code. This allows the user to know which test is executing in case the test hangs for some reason.

–showevents Be verbose when running event and error tests. Also, can be used to catch and display information on any signals received during execution of *xieperf* that the error events received by *xieperf* are valid the first time the tests are executed on a new platform.

–events Run tests which test for event generation.

–errors Run tests which test for error event generation.

–loCal Skip test calibration. This may be used when running *xieperf* in situations where execution timing is not important. Execution times will not be reported by *xieperf* when this option is enabled. The inner loop repeat count, additionally, is set to a value of 5 (but can be overridden by the -reps option).

–all Runs all tests. This may take a while, depending on the speed of your machine, and its floating point capabilities. This option is ignored if a script file is used.

–tests Generate a list of the available tests for the *xieperf* program. In *x11perf* , this list is normally displayed in the usage statement. It was yanked from the usage of *xieperf* because it was too lengthy.

–mkscript Generate a script file suitable for use with the script option. If -repeat or -reps are also specified, they will be automatically placed at the end of each command in the script. The script is generated to stderr. See the -script command, above.

–cache *<n>* Most test flos utilize a photomap resource for a source. A photomap cache of up to n entries is controlled by *xieperf* to avoid having to constantly reload these images during test initialization. The default cache size is 4. If a value less than the default is specified, the cache size will be set to the default.

–labels Generates just the descriptive labels for each test specified. Use -all or -range to specify which tests are included. See *x11perfcomp(1)* for more details.

–DIS Pretend we are running *xieperf* while connected to a DIS-only capable implementation of XIE. This will cause *xieperf* to execute those tests which only use protocol requests found in the DIS subset of XIE, and bypass those which are not DIS compatible. If *xieperf* detects a DIS server, it will do this automatically, and this option is ignored. Use -all or -range to specify the initial range of tests.

–range *<test1>***[,***<test2>***]**
Runs all the tests starting from the specified name *test1* until the name *test2*, including both the specified tests. Some tests, like the event and error tests, also require the -errors or -events options to specified. This option is ignored if a script it used.

–reps *<n>* Fix the inner loop repetitions to n. This indicates how many time the photoflo will be executed each time the test is run. This option is overridden on a per-test basis if specified in a script. Typically, *xieperf* determines the ideal number of reps during each test's calibration period.

–ImportObscuredEvent through –ExportAvailable
Test generation of events. Requires -events flag.

–BadValue through –FloValueError
Test generation of errors. Requires -errors flag. **–ColorList** Create and destroy ColorList resource test.

–LUT Create and destroy LUT resource test.

–Photomap Create and destroy Photomap resource test.

–ROI Create and destroy ROI resource test.

–Photospace Create and destroy Photospace test.

–Photoflo Create and destroy Photoflo test.

–QueryPhotomap
Query Photomap resource test.

–QueryColorList
Query ColorList resource test.

–QueryTechniquesDefault through –QueryTechniquesWhiteAdjust
Query techniques as specified by test name.

–QueryPhotoflo
Query photoflo test.

–PurgeColorList
PurgeColorList test.

–Abort This tests creates a photoflo which is started and blocks for data provided by PutClientData(). Instead of sending the data, the test uses XieAbort() to stop the photoflo, and then waits for PhotofloDone event to be send by the server. If the test times out waiting for the event, a error message is sent to stderr.

–Await This test creates a flo of the form ImportClientLUT -> ExportLUT, and starts the flo executing. *xieperf* then forks, and the child process streams the LUT data to the flo using PutClientData, while the parent blocks in XieAwait. If the flo successfully finishes, XieAwait will return and the flo state, after query, will indicate that it has completed. If XieAwait does not complete naturally, or after return from XieAwait the flo is still active, an error is reported to stderr. Note, on a really slow machine, it is possible that XieAwait will return before the flo has a chance to finish. In this case, use the -timeout option to increase the timeout for this test.

–importclientlut1
ImportClientLUT -> ExportLUT test.

–importclientphoto1 through –importclientphoto9
Flos of the form ImportClientPhoto -> ExportPhotomap using various decode techniques, e.g. G32D, TIFF2, UncompressedTriple, and so forth.

–importclientroi1
ImportClientROI with 10 rectangles.

–importclientroi2
ImportClientROI with 100 rectangles.

–encodephoto1 through –encodephoto14
Flos of the form ImportPhotomap -> ExportPhotomap using various encode techniques, e.g. G32D, TIFF2, UncompressedTriple, and so forth. Original encoding is shown in left window, image after encoding is shown in right window.

–encodeclientphoto1 through –encodeclientphoto11
Two flos, one of the form ImportPhotomap -> ExportClientPhoto, and the other of the form ImportClientPhoto -> ExportPhotomap, where ExportClientPhoto in the first flo uses various encode techniques, e.g. G32D, TIFF2, UncompressedTriple, and so forth. The image before encoding is displayed in the left window, while the right window shows the image which was encoded in the first flo and read back in the second flo.

–exportclientlut1
ExportClientLUT test. LUT is displayed in a histogram window.

–exportclientroi1
ExportClientROI test, 10 ROIs. The ROIs which are sent to the server are represented by the filled rectangles. The ROIs which are received back from the server by the client are drawn as white bordered non-filled rectangles. The resulting output illustrates how the server combined the rectangles sent to it.

–exportclientroi2
Same as exportclientroi1, except using 100 rectangles.

–exportclienthistogram1 through –exportclienthistogram4
ExportClientHistogram tests using various images. The histogram is displayed in a window which overlaps the image.

–exportclienthistogramroi1 through –exportclienthistogramroi4
Same as the ExportClientHistogram test, but using a ROI to identify the area of interest.

–exportclienthistogramcplane1 through –exportclienthistogramcplane4
Same as the ExportClientHistogram test, but using a Control Plane to identify the area of interest.

–importlut1 Test ImportLUT element, LUT size is 256.

–importphoto1 ImportPhotomap -> ExportPhotomap, with source and destination equal.

–importphoto2 ImportPhotomap -> ExportDrawable, window destination.

–importroi1 ImportROI -> ExportROI, 10 rectangles, source and destination ROIs equal.

–importroi2 ImportROI -> ExportROI, 100 rectangles, source and destination ROIs equal.

–importdrawable1
ImportDrawable -> ExportDrawable, Source is pixmap, destination is window.

–importdrawable2
ImportDrawable -> ExportDrawable, Source and destination is window.

–importdrawable3
ImportDrawable -> ExportDrawable, Destination window obscured by source window.

–importdrawable4
ImportDrawable -> ExportDrawable, Source window obscured by destination window.

–importdrawable5
ImportDrawablePlane -> ExportDrawablePlane, pixmap, source = destination.

–importdrawable6
ImportDrawablePlane -> ExportDrawablePlane, window, source = destination.

–importdrawable7
ImportDrawablePlane -> ExportDrawablePlane, window, source obscures destination.

–importdrawable8
ImportDrawablePlane -> ExportDrawablePlane, window, destination obscures source.

–constrain1 Constrain HardClip technique test, drawable destination.

–constrain2 Constrain ClipScale technique test, drawable destination.

–constrainphoto1
Constrain HardClip technique test, photomap destination.

–constrainphoto2
Constrain ClipScale technique test, photomap destination.

–convolve1 Boxcar 3x3 convolution test. Smoothing or lowpass filter.

–convolve2 Boxcar 5x5 convolution test. Smoothing or lowpass filter.

–convolve3 LaPlacian 3x3 convolution test. Edge or highpass filter.

–convolve4 LaPlacian 5x5 convolution test. Edge or highpass filter.

–convolveroi1 LaPlacian 3x3 convolution test, with ROI.

–convolveroi2 LaPlacian 5x5 convolution test, with ROI.

–convolvecplane1
LaPlacian 3x3 convolution test, with Control Plane.

–convolvecplane2
LaPlacian 5x5 convolution test, with Control Plane.

–math1 through –mathcplane7
Various tests which exercise the Math element, some tests using ROIs and control planes.

–arithmeticdyadic1 through –arithmeticdyadic5
Arithmetic element tests, using photomaps as the operands.

–arithmeticmonadic1 through –arithmeticmonadic9
Arithmetic element tests, photomap and constant operands.

–arithmeticdyadicroi1 through –arithmeticdyadicroi5
Arithmetic element tests, using photomaps as the operands, with ROIs.

–arithmeticmonadicroi1 through –arithmeticmonadicroi9
Arithmetic element tests, photomap and constant operands, with ROIs.

–arithmeticdyadiccplane1 through –arithmeticdyadiccplane5
Arithmetic element tests, using photomaps as the operands, with Control Planes.

–arithmeticmonadiccplane1 through –arithmeticmonadiccplane9
Arithmetic element tests, photomap and constant operands, with Control Planes.

–arithmeticfloatdyadic1 though –arithmeticfloatdyadic5
Arithmetic element tests, using photomaps as the operands, unconstrained.

–arithmeticfloatmonadic1 though –arithmeticfloatmonadic9
Arithmetic element tests, photomap and constant operands, unconstrained.

–arithmeticroifloatdyadic1 to –arithmeticroifloatdyadic5
Arithmetic element tests, photomaps as the operands, ROIs, unconstrained.

–arithmeticroifloatmonadic1 to - .rithmeticroifloatmonadic9
Arithmetic element tests, photomap and constant operands, ROIs, unconstrained.

–band1 BandSelect element test. Image input is triple band. If visual of *xieperf* window is a color visual, then three BandSelect elements are used to extract the indivudual bands, they are combined once again using BandCombine, and displayed using ConvertToIndex. If the visual is not color, e.g. GrayScale or StaticGray, then the flo simply uses one BandSelect element to extract a single band for display.

–band2 BandCombine test. Input bands are made of of three separate single band photomaps. These are combined using a BandCombine element, which is followed by a BandExtract and ExportDrawable. CCIR 601-1 coefficients.

–band3 BandExtract test. Input is a triple band photomap. CCIR 601-1 coefficients. Destination window colormap is gray ramp.

–band4 BandExtract test. Input is a triple band photomap. CCIR 601-1 coefficients. Destination window colormap is RGB_BEST_MAP standard colormap.

–band5 BandExtract test. Input is a triple band photomap. CCIR 601-1 coefficients. Destination window colormap is RGB_DEFAULT_MAP standard colormap.

−comparedyadic1 through −comparedyadic6
Test various Compare operators with dyadic photomap operands.

−comparemonadic1 through −comparemonadic6
Test various compare operators with photomap, constant operands.

−compareroidyadic1 through −compareroidyadic6
Test various Compare operators with dyadic photomap operands, using ROIs.

−compareroimonadic1 through −compareroimonadic6
Test various compare operators with photomap, constant operands, using ROIs.

−comparecplanedyadic1 through −comparecplanedyadic6
Test various Compare operators with dyadic photomap operands, Control Planes.

−comparecplanemonadic1 through −comparecplanemonadic6
Test various compare operators with photomap, constant operands, Control Planes.

−matchhistogram1 through −matchhistogram18
MatchHistogram element tests, using various images and histogram matching techniques.

−matchhistogramroi1 through −matchhistogramroi6
A selection of MatchHistogram element tests, with ROIs.

−matchhistogramcplane1 through −matchhistogramcplane6
A selection of MatchHistogram element tests, with Control Planes.

−unconstrain1 ImportPhotomap, Unconstrain, Constrain(ClipScale), ExportDrawable test.

−pasteup1 through −pasteup2
PasteUp element tests.

−geometry1 through −geometry14
Geometry element tests, including rotations, scales, and mirroring. NearestNeighbor technique.

−geometry15 through −geometry28
Geometry element tests, including rotations, scales, and mirroring. AntiAlias technique.

−geometry29 through −geometry42
Geometry element tests, including rotations, scales, and mirroring. BilinearInterpolation technique.

−geomg31dscale1 through −geometryfaxradio1
Tests to exercise the various FAX decoders and the Geometry element.

−dither1 Dither test, ErrorDiffusion dither technique, ExportDrawable.

−dither2 Dither test, ErrorDiffusion dither technique, ExportDrawablePlane.

−dither3 Dither test, Ordered(4) dither technique, ExportDrawable.

−dither4 Dither test, Ordered(4) dither technique, ExportDrawablePlane.

−dither5 Dither test, Ordered(8) dither technique, ExportDrawable.

–dither6 Dither test, Ordered(8) dither technique, ExportDrawablePlane.

–dither7 Dither test, Default dither technique, ExportDrawable.

–dither8 Dither test, Default dither technique, ExportDrawablePlane.

–logicalmonadic1 through –logicalmonadic16
Logical element, photomap and a constant of 0 as operands, various operators.

–logicaldyadic1 through –logicaldyadic16
Logical element tests, dyadic photomaps as operands, various operators.

–logicalmonadicroi1 through –logicalmonadicroi16
Logical element, photomap and constant of 0 operands, various operators, ROIs.

–logicaldyadicroi1 through –logicaldyadicroi16
Logical element, dyadic photomaps as operands, various operators, ROIs.

–logicalmonadiccplane1 through –logicalmonadiccplane16
Logical element, photomap and constant of 0 operands, various operators, Control Planes.

–logicaldyadiccplane1 through –logicaldyadiccplane16
Logical element, dyadic photomaps as operands, various operators, Control Planes.

–blend1 Blend element test. Monadic source, 0.1 source constant. Alpha constant of 0.5.

–blend2 Blend element test. Dyadic sources. Alpha constant of 0.5.

–blendroi1 Blend test. Monadic source, 0.1 source constant. Alpha constant of 0.5. ROIs.

–blendroi2 Blend element test. Dyadic sources. Alpha constant of 0.5. Uses ROIs.

–blendcplane1 Blend test. Monadic source, 0.1 source constant. Alpha constant of 0.5. Control Plane.

–blendcplane2 Blend element test. Dyadic sources. Alpha constant of 0.5. Control Plane.

–blendalpha1 Blend test. Monadic source, 220 source constant. Alpha plane is a photomap.

–blendalpha2 Blend test. Dyadic sources. Alpha plane is a constant 220.

–blendalpharoi1
Blend test. Monadic source, 220 source constant. Alpha plane photomap. ROIs.

–blendalpharoi2
Blend test. Dyadic sources. Alpha plane is a constant 220. ROIs.

–triplepoint1 through –triplepoint2
Illustrate use of point and Standard colormaps for rendering triple band images.

–funnyencode1 through –funnyencode8
These tests are design to perform limited exercising of XIE's capability

of dealing with various encodings of flo source data. The test init function obtains a photomap using ICP -> EP. A series of independent permanent flo pairs, one of the form IP -> EP, and the other of the basic form IP -> ED, are constructed. The encoding parameters for the ExportPhotomap (EP) element in the first flo are derived from test configuration. The number of flo pairs created is also dependent upon test configuration. The tests can be configured so that the test init function will constrain the input photomap to a specified number of levels, on a per-band basis, so that word-sized and quad-sized pixels are passed through the flos. Some tests below take advantage of this. See tests.c for test configuration, and hints on how to add similar tests.

−point1 through −point3
Simple Point element tests. Drawable destination. **−pointroi1** Simple Point element test which uses ROIs. Drawable destination.

−pointcplane1 Simple Point element test which uses a Control Plane. Drawable destination.

−pointphoto1 Simple Point element test. Photomap destination.

−pointroiphoto1
Simple Point element test which uses a ROIs. Photomap destination.

−pointcplanephoto1
Simple Point element test which uses a Control Plane. Photomap destination.

−redefine Two flographs are created which are the same in structure, expect for the x and y offsets specified for the ExportDrawable flo elements. The test init function creates a photoflo based upon one of the two flographs. The inner loop of the test function uses XieRedefinePhotoflo() to alternate between each of the flographs. Make sure that your ineer loop reps are 2 or greater in order to exercise this test fully (see -reps).

−modify1 Test XieModifyPhotoflo() by adjust ROI offsets and size.

−modify2 Test XieModifyPhotoflo() by changing the LUT input to a Point element.

−modify3 Test XieModifyPhotoflo() by changing ExportDrawable x and y offsets.

−modify4 This test creates a rather long flo of arithmetic elements, each which does nothing more than add 1 to a small image. The test init function scales the input photomap. The ExportDrawable x and y offset is modified randomly during each iteration of the test function inner loop.

−modify5 This test creates a rather long flo of arithmetic elements, each which does nothing more than add 1 to a large image. Each rep, the Geometry and ExportDrawable elements at the end of the flo are modifed to crop a small piece of the input into its appropriate place in the larger image.

−rgb1 through −rgb16
These tests all basically take an UncompressedTriple image as input, send it to ConvertFromRGB which converts the image to some configured colorspace, and then send the converted image on to ConvertToRGB prior to display. The original image is displayed in the lefthand window, and the image which has passed through the flo is shown in the

righthand window. The goal of these test is to show that ConvertFromRGB -> ConvertToRGB is lossless.

−converttoindexpixel
ConvertToIndex test, TripleBand BandByPixel.

−converttoindexplane
ConvertToIndex test, TripleBand BandByPlane.

−convertfromindex
The test init function uses a flo containing ConvertToIndex to display an image in the left window. The test function uses this drawable as input to a flo which does ConvertFromIndex -> ConvertToIndex and sends the resulting image to the right window. The result should be lossless.

−complex A somewhat large flo which uses control planes, LUTs, Point, PasteUp, Logical, Constrain, Dither, Geometry, MatchHistogram, BandCombine, and BandSelect elements. See the Postscript file ''complex.ps'' for a rendition of the photoflo which is executed.

X DEFAULTS

There are no X defaults used by this program.

SEE ALSO

X(1), x11perf(1), x11perfcomp(1)

BUGS

There should be a IMAGES environment variable to augment the -images option.

Many tests only scratch the surface of possible test cases. Some of the options available for certain flo elements are either inadequately tested, or ignored altogether. There are insufficent tests for bitonal, large pixel, or triple band tests.

Some of the test names are inconsistently cased, e.g. -Abort and -dither1.

Some tests are hopelessly slow when run against machines with slow FPUs.

Bitonal images are for the most part displayed using the ExportDrawable flo element, however, ExportDrawablePlane would be a better choice.

AUTHOR

Syd Logan, AGE Logic, Inc.

NAME

startx − initialize an X session

SYNOPSIS

startx [[*client*] *options* ...] [**−−** [*server*] *options* ...]

DESCRIPTION

NOTE: The *startx* script supplied with the X11 distribution is a sample designed more as a base for customization than as a finished product. Site administrators are urged to customize it for their site. And to update this manual page when they do!

The *startx* script is a front end to *xinit* that provides a somewhat nicer user interface for running a single session of the X Window System. It is typically run with no arguments.

To determine the client to run, *startx* first looks for a file called *.xinitrc* in the user's home directory. If that is not found, it uses the file *xinitrc* in the *xinit* library directory. If command line client options are given, they override this behavior. To determine the server to run, *startx* first looks for a file called *.xserverrc* in the user's home directory. If that is not found, it uses the file *xserverrc* in the *xinit* library directory. If command line server options are given, they override this behavior. Users rarely need to provide a *.xserverrc* file. See the *xinit*(1) manual page for more details on the arguments.

The *.xinitrc* is typically a shell script which starts many clients according to the user's preference. When this shell script exits, *startx* kills the server and performs any other session shutdown needed. Most of the clients started by *.xinitrc* should be run in the background. The last client should run in the foreground; when it exits, the session will exit. People often choose a session manager, window manager, or *xterm* as the "magic" client.

EXAMPLE

Below is a sample *.xinitrc* that starts several applications and leaves the window manager running as the "last" application. Assuming that the window manager has been configured properly, the user then chooses the "Exit" menu item to shut down X.

```
xrdb −load $HOME/.Xresources
xsetroot −solid gray &
xbiff −geometry −430+5 &
oclock −geometry 75x75−0−0 &
xload −geometry −80−0 &
xterm −geometry +0+60 −ls &
xterm −geometry +0−100 &
xconsole −geometry −0+0 −fn 5x7 &
exec twm
```

ENVIRONMENT VARIABLES

DISPLAY
: This variable gets set to the name of the display to which clients should connect. Note that this gets *set,* not read.

FILES

$(HOME)/.xinitrc
: Client to run. Typically a shell script which runs many programs in the background.

$(HOME)/.xserverrc
: Server to run. The default is *X.*

<XRoot>/lib/X11/xinit/xinitrc
: Client to run if the user has no *.xinitrc* file. <XRoot> refers to the root of the X11 install tree.

<XRoot>/lib/X11/xinit/xserverrc
Client to run if the user has no *.xserverrc* file. This is only needed if the server needs special arguments or is not named. <XRoot> refers to the root of the X11 install tree.

SEE ALSO
xinit(1)

NAME

xinit − X Window System initializer

SYNOPSIS

xinit [[*client*] *options*] [**−−** [*server*] [*display*] *options*]

DESCRIPTION

The *xinit* program is used to start the X Window System server and a first client program on systems that cannot start X directly from */etc/init* or in environments that use multiple window systems. When this first client exits, *xinit* will kill the X server and then terminate.

If no specific client program is given on the command line, *xinit* will look for a file in the user's home directory called *.xinitrc* to run as a shell script to start up client programs. If no such file exists, *xinit* will use the following as a default:

```
xterm −geometry +1+1 −n login −display :0
```

If no specific server program is given on the command line, *xinit* will look for a file in the user's home directory called *.xserverrc* to run as a shell script to start up the server. If no such file exists, *xinit* will use the following as a default:

```
X :0
```

Note that this assumes that there is a program named *X* in the current search path. However, servers are usually named *Xdisplaytype* where *displaytype* is the type of graphics display which is driven by this server. The site administrator should, therefore, make a link to the appropriate type of server on the machine, or create a shell script that runs *xinit* with the appropriate server.

An important point is that programs which are run by *.xinitrc* should be run in the background if they do not exit right away, so that they don't prevent other programs from starting up. However, the last long-lived program started (usually a window manager or terminal emulator) should be left in the foreground so that the script won't exit (which indicates that the user is done and that *xinit* should exit).

An alternate client and/or server may be specified on the command line. The desired client program and its arguments should be given as the first command line arguments to *xinit*. To specify a particular server command line, append a double dash (−−) to the *xinit* command line (after any client and arguments) followed by the desired server command.

Both the client program name and the server program name must begin with a slash (/) or a period (.). Otherwise, they are treated as an arguments to be appended to their respective startup lines. This makes it possible to add arguments (for example, foreground and background colors) without having to retype the whole command line.

If an explicit server name is not given and the first argument following the double dash (−−) is a colon followed by a digit, *xinit* will use that number as the display number instead of zero. All remaining arguments are appended to the server command line.

EXAMPLES

Below are several examples of how command line arguments in *xinit* are used.

xinit This will start up a server named *X* and run the user's *.xinitrc*, if it exists, or else start an *xterm*.

xinit −− /usr/X11R6/bin/Xqdss :1
This is how one could start a specific type of server on an alternate display.

xinit −geometry =80x65+10+10 −fn 8x13 −j −fg white −bg navy
This will start up a server named *X*, and will append the given arguments to the default *xterm* command. It will ignore *.xinitrc*.

xinit −e widgets −− ./Xsun −l −c
This will use the command *.Xsun −l −c* to start the server and will append the arguments *−e widgets* to the default *xterm* command.

xinit /usr/ucb/rsh fasthost cpupig −display ws:1 −− :1 −a 2 −t 5
This will start a server named *X* on display 1 with the arguments *−a 2 −t 5*. It will then start a remote shell on the machine **fasthost** in which it will run the command *cpupig*, telling it to display back on the local workstation.

Below is a sample *.xinitrc* that starts a clock, several terminals, and leaves the window manager running as the ‘‘last’’ application. Assuming that the window manager has been configured properly, the user then chooses the ‘‘Exit’’ menu item to shut down X.

```
xrdb −load $HOME/.Xresources
xsetroot −solid gray &
xclock −g 50x50−0+0 −bw 0 &
xload −g 50x50−50+0 −bw 0 &
xterm −g 80x24+0+0 &
xterm −g 80x24+0−0 &
twm
```

Sites that want to create a common startup environment could simply create a default *.xinitrc* that references a site-wide startup file:

```
#!/bin/sh
. /usr/local/lib/site.xinitrc
```

Another approach is to write a script that starts *xinit* with a specific shell script. Such scripts are usually named *x11*, *xstart*, or *startx* and are a convenient way to provide a simple interface for novice users:

```
#!/bin/sh
xinit /usr/local/lib/site.xinitrc −− /usr/X11R6/bin/X bc
```

ENVIRONMENT VARIABLES

DISPLAY This variable gets set to the name of the display to which clients should connect.

XINITRC This variable specifies an init file containing shell commands to start up the initial windows. By default, *.xinitrc* in the home directory will be used.

FILES

.xinitrc default client script

xterm client to run if *.xinitrc* does not exist

.xserverrc default server script

X server to run if *.xserverrc* does not exist

SEE ALSO

X(1), *startx*(1), *Xserver*(1), *xterm*(1)

AUTHOR

Bob Scheifler, MIT Laboratory for Computer Science

NAME

xkbcomp – compile XKB keyboard description

SYNOPSIS

xkbcomp [option] source [destination]

DESCRIPTION

The *xkbcomp* keymap compiler converts a description of an XKB keymap into one of several output formats. The most common use for xkbcomp is to create a compiled keymap file (.xkm extension) which can be read directly by XKB-capable X servers or utilities. The keymap compiler can also produce C header files or XKB source files. The C header files produced by xkbcomp can be included by X servers or utilities that need a built-in default keymap. The XKB source files produced by *xkbcomp* are fully resolved and can be used to verify that the files which typically make up an XKB keymap are merged correctly or to create a single file which contains a complete description of the keymap.

The *source* may specify an X display, or an *.xkb* or *depends on the format of the source. Compiling a .xkb (keymap source) file generates a .xkm (compiled keymap file) by default. If the source is a .xkm file or an X display, xkbcomp generates a keymap source file by default.*

If the *destination* is an X display, the keymap for the display is updated with the compiled keymap.

The name of the *destination* is usually computed from the name of the source, with the extension replaced as appropriate. When compiling a single map from a file which contains several maps, *xkbcom* constructs the destination file name by appending an appropriate extension to the name of the map to be used.

OPTIONS

–a Show all keyboard information, reporting implicit or dervied information as a comment. Only affects *.xkb* format output.

–C Produce a C header file as output (.h extension).

–dflts Compute defaults for any missing components, such as key names.

–I*dir* Specifies top-level directories to be searched for files included by the keymap description.

–l List maps that specify the *map* pattern in any files listed on the command line (not implemented yet).

–m *name*
Specifies a map to be compiled from an file with multiple entries.

–merge Merge the compiled information with the map from the server (not implemented yet).

–o *name* Specifies a name for the generated output file. The default is the name of the source file with an appropriate extension for the output format.

–opt *parts*
Specifies a list of optional parts. Compilation errors in any optional parts are not fatal. Parts may consist of any combination of the letters *c*, *g*,*k*,*s*,*t* which specify the compatibility map, geometry, keycodes, symbols and types, respectively.

-R*dir* Specifies the root directory for relative path names.

-synch Force synchonization for X requests.

–w *lvl* Controls the reporting of warnings during compilation. A warning level of 0 disables all warnings; a warning level of 10 enables them all.

–xkb Generate a source description of the keyboard as output (.xkb extension).

–xkm Generate a compiled keymap file as output (.xkm extension).

SEE ALSO

X(1)

COPYRIGHT

AUTHOR

Erik Fortune, Silicon Graphics

NAME
: xkbevd − XKB event daemon

SYNOPSIS
: **xkbevd** [options]

DESCRIPTION

This command is very raw and is therefore only partially implemented; we present it here as a rough prototype for developers, not as a general purpose tool for end users. Something like this might make a suitable replacement for xev; I'm not signing up, mind you, but it's an interesting idea.

The *xkbevd* event daemon listens for specified XKB events and executes requested commands if they occur. The configuration file consists of a list of event specification/action pairs and/or variable definitions.

An event specification consists of a short XKB event name followed by a string or identifier which serves as a qualifier in parentheses; empty parentesis indicate no qualification and serve to specify the default command which is applied to events which do not match any of the other specifications. The interpretation of the qualifier depends on the type of the event: Bell events match using the name of the bell, message events match on the contents of the message string and slow key events accept any of *press*, *release*, *accept*, or *reject*. No other events are currently recognized.

An action consists of an optional keyword followed by an optional string argument. Currently, *xkbev* recognizes the actions: *none*, *ignore*, *echo*, *printEvent*, *sound*, and *shell*. If the action is not specified, the string is taken as the name of a sound file to be played unless it begins with an exclamation point, in which case it is taken as a shell command.

Variable definitions in the argument string are expanded with fields from the event in question before the argument string is passed to the action processor. The general syntax for a variable is either $*cP or $(str), where c is a single character and str is a string of arbitrary length. All parameters have both single-character and long names.*

The list of recognized parameters varies from event to event and is too long to list here right now. This is a developer release anyway, so you can be expected to look at the source code (evargs.c is of particular interest).

The *ignore*, *echo*, *printEvent*, *sound*,and *shell* actions do what you would expect commands named *ignore*, *echo*, *printEvent*, *sound*, and *shell* to do, except that the sound command has only been implemented and tested for SGI machines. It launches an external program right now, so it should be pretty easy to adapt, especially if you like audio cues that arrive about a half-second after you expect them.

The only currently recognized variables are *soundDirectory* and *soundCmd*. I'm sure you can figure out what they do.

OPTIONS

−help Prints a usage message that is far more up-to-date than anything in this man page.

−cfg *file* Specifies the configuration file to read. If no configuration file is specified, *xkbevd* looks for ˜/.xkb/xkbevd.cf and $(LIBDIR)/xkb/xkbevd.cf in that order.

−sc *cmd* Specifies the command used to play sounds.

−sd *directory*
: Specifies a top-level directory for sound files.

–display *display*
Specifies the display to use. If not present, *xkbevd* uses $DISPLAY.

–bg Tells *xkbevd* to fork itself (and run in the background).

–synch Forces synchronization of all X requests. Slow.

–v Print more information, including debugging messages. Multiple specifications of *-v* cause more output, to a point.

SEE ALSO

X(1)

COPYRIGHT

AUTHOR

Erik Fortune, Silicon Graphics

NAME

xkbprint − print an XKB keyboard description

SYNOPSIS

xkbprint [options] source [output_file]

DESCRIPTION

The *xkbprint* comman generates a printable or encapsulated PostScript description of the XKB keyboard description specifed by *source*. The *source* can be any compiled keymap (.xkm) file that includes a geometry description or an X display specification. If an *output_file* is specified, xkbprint writes to it. If no output file is specified, xkbprint creates replaces the extension of the source file with *.ps* or *.eps* depending on the requested format. If the source is a non-local X display (e.g.:0), xkbprint appends the appropriate prefix to the display specification, replacing the colon with a dash. For a local display, xkprint uses server-*n* where *n* is the number of the display.

OPTIONS

−?, -help Prints a usage message.

−color Print using the colors specified in the geometry file; by default, xkbprint prints a black-and-white image of the keyboard.

−dflts Attempt to compute default names for any missing components, such as keys.

−diffs Show symbols only where they are explicitly bound.

−eps Generate an encapsulated PostScript file.

−fit Fit the keyboard image on the page (default).

−full Print the keyboard at full size.

−grid *res*
Print a grid with *res*mm resolution over the keyboard.

−if *fontName*
Specifies an internal PostScript type 1 font to dump to the specified output file or to *fontName*.pfa, if no output file is specified. No keyboard description is printed if an internal font is dumped.

−label *type*
Specifies the labels to be printed on keys; legal types are: *none*, *name*,*code*,*symbols*.

−lc *<locale>*
Specifies a locale in which KeySyms should be resolved.

−level1 Generate level 1 PostScript.

−level2 Generate level 2 PostScript.

−lg *group*
Print symbols in keyboard groups starting from *group*.

−ll *level* Print symbols starting from shift level *level*.

−mono Generate black-and-white image of keyboard (default).

−n *num* Print *num* copies.

−nkg *num*
Print the symbols in *num* keyboard groups.

–npk *num*
Number of keyboard images to print on each page; for EPS files, this specifies the total number of keyboard images to print.

–o *file* Write output to *file*.

–R*directory*
Use *directory* as the root directory; all path names are interpreted relative to *directory*.

-pict *which*
Controls use of pictographs instead of keysym names where available. *which* can be any of *all*, *none*, or *common*(default).

-synch Forces synchronization for X requests.

-w *level* Sets warning level (0 for no warning, 10 for all warnings).

SEE ALSO

X(1),xkbcomp(1)

COPYRIGHT

AUTHOR

Erik Fortune, Silicon Graphics

NAME

xkill - kill a client by its X resource

SYNOPSIS

xkill [−display *displayname*] [−id *resource*] [−button number] [−frame] [−all]

DESCRIPTION

Xkill is a utility for forcing the X server to close connections to clients. This program is very dangerous, but is useful for aborting programs that have displayed undesired windows on a user's screen. If no resource identifier is given with *-id*, *xkill* will display a special cursor as a prompt for the user to select a window to be killed. If a pointer button is pressed over a non-root window, the server will close its connection to the client that created the window.

OPTIONS

−display *displayname*
: This option specifies the name of the X server to contact.

−id *resource*
: This option specifies the X identifier for the resource whose creator is to be aborted. If no resource is specified, *xkill* will display a special cursor with which you should select a window to be kill.

−button *number*
: This option specifies the number of pointer button that should be used in selecting a window to kill. If the word "any" is specified, any button on the pointer may be used. By default, the first button in the pointer map (which is usually the leftmost button) is used.

−all
: This option indicates that all clients with top-level windows on the screen should be killed. *Xkill* will ask you to select the root window with each of the currently defined buttons to give you several chances to abort. Use of this option is highly discouraged.

−frame
: This option indicates that xkill should ignore the standard conventions for finding top-level client windows (which are typically nested inside a window manager window), and simply believe that you want to kill direct children of the root.

XDEFAULTS

Button
: Specifies a specific pointer button number or the word "any" to use when selecting windows.

SEE ALSO

X(1), xwininfo(1), XKillClient and XGetPointerMapping in the Xlib Programmers Manual, KillClient in the X Protocol Specification

AUTHOR

Jim Fulton, MIT X Consortium
Dana Chee, Bellcore

NAME

xlogo - X Window System logo

SYNOPSIS

xlogo [-*toolkitoption* ...]

DESCRIPTION

The *xlogo* program displays the X Window System logo.

OPTIONS

Xlogo accepts all of the standard X Toolkit command line options, as well as the following:

–shape This option indicates that the logo window should be shaped rather than rectangular.

RESOURCES

The default width and the default height are each 100 pixels. This program uses the *Logo* widget in the Athena widget set. It understands all of the Simple widget resource names and classes as well as:

foreground (class **Foreground)**
Specifies the color for the logo. The default is depends on whether *reverseVideo* is specified. If *reverseVideo* is specified the default is *XtDefaultForeground*, otherwise the default is *XtDefaultBackground*.

shapeWindow (class **ShapeWindow)**
Specifies that the window is shaped to the X logo. The default is False.

WIDGETS

In order to specify resources, it is useful to know the hierarchy of the widgets which compose *xlogo*. In the notation below, indentation indicates hierarchical structure. The widget class name is given first, followed by the widget instance name.

```
XLogo  xlogo
        Logo  xlogo
```

ENVIRONMENT

DISPLAY
to get the default host and display number.

XENVIRONMENT
to get the name of a resource file that overrides the global resources stored in the RESOURCE_MANAGER property.

FILES

<XRoot>/lib/X11/app-defaults/XLogo - specifies required resources

SEE ALSO

X(1), xrdb(1)

AUTHORS

Ollie Jones of Apollo Computer and Jim Fulton of the MIT X Consortium wrote the logo graphics routine, based on a graphic design by Danny Chong and Ross Chapman of Apollo Computer.

NAME

xlsatoms - list interned atoms defined on server

SYNOPSIS

xlsatoms [-options ...]

DESCRIPTION

Xlsatoms lists the interned atoms. By default, all atoms starting from 1 (the lowest atom value defined by the protocol) are listed until unknown atom is found. If an explicit range is given, *xlsatoms* will try all atoms in the range, regardless of whether or not any are undefined.

OPTIONS

–display *dpy*
: This option specifies the X server to which to connect.

–format *string*
: This option specifies a *printf*-style string used to list each atom *<value,name>* pair, printed in that order (*value* is an *unsigned long* and *name* is a *char* *). *Xlsatoms* will supply a newline at the end of each line. The default is *%ld\t%s*.

–range *[low]-[high]*
: This option specifies the range of atom values to check. If *low* is not given, a value of 1 assumed. If *high* is not given, *xlsatoms* will stop at the first undefined atom at or above *low*.

–name *string*
: This option specifies the name of an atom to list. If the atom does not exist, a message will be printed on the standard error.

SEE ALSO

X(1), Xserver(1), xprop(1)

ENVIRONMENT

DISPLAY
: to get the default host and display to use.

AUTHOR

Jim Fulton, MIT X Consortium

NAME

xlsclients - list client applications running on a display

SYNOPSIS

xlsclients [-display *displayname*] [-a] [-l] [-m maxcmdlen]

DESCRIPTION

Xlsclients is a utility for listing information about the client applications running on a display. It may be used to generate scripts representing a snapshot of the user's current session.

OPTIONS

−display *displayname*
: This option specifies the X server to contact.

−a
: This option indicates that clients on all screens should be listed. By default, only those clients on the default screen are listed.

−l
: List in long format, giving the window name, icon name, and class hints in addition to the machine name and command string shown in the default format.

−m *maxcmdlen*
: This option specifies the maximum number of characters in a command to print out. The default is 10000.

ENVIRONMENT

DISPLAY
: To get the default host, display number, and screen.

SEE ALSO

X(1), xwininfo(1), xprop(1)

AUTHOR

Jim Fulton, MIT X Consortium

NAME

xlsfonts – server font list displayer for X

SYNOPSIS

xlsfonts [–options ...] [–fn pattern]

DESCRIPTION

Xlsfonts lists the fonts that match the given *pattern*. The wildcard character "*" may be used to match any sequence of characters (including none), and "?" to match any single character. If no pattern is given, "*" is assumed.

The "*" and "?" characters must be quoted to prevent them from being expanded by the shell.

OPTIONS

–display *host***:***dpy*
: This option specifies the X server to contact.

–l Lists some attributes of the font on one line in addition to its name.

–ll Lists font properties in addition to **–l** output.

–lll Lists character metrics in addition to **–ll** output.

–m This option indicates that long listings should also print the minimum and maximum bounds of each font.

–C This option indicates that listings should use multiple columns. This is the same as **–n 0**.

–1 This option indicates that listings should use a single column. This is the same as **–n 1**.

–w *width*
: This option specifies the width in characters that should be used in figuring out how many columns to print. The default is 79.

–n *columns*
: This option specifies the number of columns to use in displaying the output. By default, it will attempt to fit as many columns of font names into the number of character specified by **–w** *width*.

–u This option indicates that the output should be left unsorted.

–o This option indicates that *xlsfonts* should do an **OpenFont** (and **QueryFont**, if appropriate) rather than a **ListFonts**. This is useful if **ListFonts** or **ListFontsWithInfo** fail to list a known font (as is the case with some scaled font systems).

–fn *pattern*
: This option specifies the font name pattern to match.

SEE ALSO

X(1), Xserver(1), xset(1), xfd(1), *X Logical Font Description Conventions*

ENVIRONMENT

DISPLAY
: to get the default host and display to use.

BUGS

Doing “xlsfonts -l” can tie up your server for a very long time. This is really a bug with single-threaded non-preemptable servers, not with this program.

AUTHOR

Mark Lillibridge, MIT Project Athena; Jim Fulton, MIT X Consortium; Phil Karlton, SGI

NAME

xmag − magnify parts of the screen

SYNOPSIS

xmag [**−mag** *magfactor*] [**−source** *geom*] [−*toolkitoption* ...]

DESCRIPTION

The *xmag* program allows you to magnify portions of an X screen. If no explicit region is specified, a square with the pointer in the upper left corner is displayed indicating the area to be enlarged. The area can be dragged out to the desired size by pressing Button 2. Once a region has been selected, a window is popped up showing a blown up version of the region in which each pixel in the source image is represented by a small square of the same color. Pressing Button1 in the enlargement window shows the position and RGB value of the pixel under the pointer until the button is released. Typing ‘‘Q’’ or ‘‘ˆC’’ in the enlargement window exits the program. The application has 5 buttons across its top. *Close* deletes this particular magnification instance. *Replace* brings up the rubber band selector again to select another region for this magnification instance. *New* brings up the rubber band selector to create a new magnification instance. *Cut* puts the magnification image into the primary selection. *Paste* copies the primary selection buffer into *xmag*. Note that you can cut and paste between *xmag* and the *bitmap* program. Resizing *xmag* resizes the magnification area. *xmag* preserves the colormap, visual, and window depth of the source.

WIDGETS

xmag uses the X Toolkit and the Athena Widget Set. The magnified image is displayed in the Scale widget. For more information, see the Athena Widget Set documentation. Below is the widget structure of the *xmag* application. Indentation indicates hierarchical structure. The widget class name is given first, followed by the widget instance name.

```
Xmag xmag
        RootWindow root
        TopLevelShell xmag
                Paned pane1
                        Paned pane2
                                Command close
                                Command replace
                                Command new
                                Command select
                                Command paste
                                Label xmag label
                        Paned pane2
                                Scale scale
        OverrideShell pixShell
                Label pixLabel
```

OPTIONS

−source *geom* This option specifies the size and/or location of the source region on the screen. By default, a 64x64 square is provided for the user to select an area of the screen.

−mag *integer* This option indicates the magnification to be used. 5 is the default.

AUTHORS

Dave Sternlicht and Davor Matic, MIT X Consortium.

NAME

xmodmap - utility for modifying keymaps (and pointer buttons) in X

SYNOPSIS

xmodmap [-options ...] [filename]

DESCRIPTION

The *xmodmap* program is used to edit and display the keyboard *modifier map* and *keymap table* that are used by client applications to convert event keycodes into keysyms. It is usually run from the user's session startup script to configure the keyboard according to personal tastes.

OPTIONS

The following options may be used with *xmodmap*:

−display *display*
: This option specifies the host and display to use.

−help
: This option indicates that a brief description of the command line arguments should be printed on the standard error channel. This will be done whenever an unhandled argument is given to *xmodmap.*

−grammar
: This option indicates that a help message describing the expression grammar used in files and with −e expressions should be printed on the standard error.

−verbose
: This option indicates that *xmodmap* should print logging information as it parses its input.

−quiet
: This option turns off the verbose logging. This is the default.

−n
: This option indicates that *xmodmap* should not change the mappings, but should display what it would do, like *make(1)* does when given this option.

−e *expression*
: This option specifies an expression to be executed. Any number of expressions may be specified from the command line.

−pm
: This option indicates that the current modifier map should be printed on the standard output.

−pk
: This option indicates that the current keymap table should be printed on the standard output.

−pke
: This option indicates that the current keymap table should be printed on the standard output in the form of expressions that can be fed back to *xmodmap*.

−pp
: This option indicates that the current pointer map should be printed on the standard output.

−
: A lone dash means that the standard input should be used as the input file.

The *filename* specifies a file containing *xmodmap* expressions to be executed. This file is usually kept in the user's home directory with a name like *.xmodmaprc*.

EXPRESSION GRAMMAR

The *xmodmap* program reads a list of expressions and parses them all before attempting to execute any of them. This makes it possible to refer to keysyms that are being redefined in a natural way without having to worry as much about name conflicts.

keycode *NUMBER = KEYSYMNAME ...*
The list of keysyms is assigned to the indicated keycode (which may be specified in decimal, hex or octal and can be determined by running the *xev* program.

keycode any = *KEYSYMNAME ...*
If no existing key has the specified list of keysyms assigned to it, a spare key on the keyboard is selected and the keysyms are assigned to it. The list of keysyms may be specified in decimal, hex or octal.

keysym *KEYSYMNAME = KEYSYMNAME ...*
The *KEYSYMNAME* on the left hand side is translated into matching keycodes used to perform the corresponding set of **keycode** expressions. The list of keysym names may be found in the header file *<X11/keysymdef.h>* (without the *XK_* prefix) or the keysym database *<XRoot>/lib/X11/XKeysymDB*, where <XRoot> refers to the root of the X11 install tree. Note that if the same keysym is bound to multiple keys, the expression is executed for each matching keycode.

clear *MODIFIERNAME*
This removes all entries in the modifier map for the given modifier, where valid name are: **Shift**, **Lock**, **Control**, **Mod1**, **Mod2**, **Mod3**, **Mod4**, and **Mod5** (case does not matter in modifier names, although it does matter for all other names). For example, ‘‘clear Lock’’ will remove all any keys that were bound to the shift lock modifier.

add *MODIFIERNAME = KEYSYMNAME ...*
This adds all keys containing the given keysyms to the indicated modifier map. The keysym names are evaluated after all input expressions are read to make it easy to write expressions to swap keys (see the EXAMPLES section).

remove *MODIFIERNAME = KEYSYMNAME ...*
This removes all keys containing the given keysyms from the indicated modifier map. Unlike **add,** the keysym names are evaluated as the line is read in. This allows you to remove keys from a modifier without having to worry about whether or not they have been reassigned.

pointer = default
This sets the pointer map back to its default settings (button 1 generates a code of 1, button 2 generates a 2, etc.).

pointer = *NUMBER ...*
This sets to pointer map to contain the indicated button codes. The list always starts with the first physical button.

Lines that begin with an exclamation point (!) are taken as comments.

If you want to change the binding of a modifier key, you must also remove it from the appropriate modifier map.

EXAMPLES

Many pointers are designed such that the first button is pressed using the index finger of the right hand. People who are left-handed frequently find that it is more comfortable to reverse the button codes that get generated so that the primary button is pressed using the index finger of the left hand. This could be done on a 3 button pointer as follows:

```
% xmodmap -e "pointer = 3 2 1"
```

Many applications support the notion of Meta keys (similar to Control keys except that Meta is held down instead of Control). However, some servers do not have a Meta keysym in the default keymap table, so one needs to be added by hand. The following command will attach Meta to the Multi-language key (sometimes labeled Compose Character). It also takes advantage of the fact that applications that need a Meta key simply need to get the keycode and don't require the keysym to be in the first column of the keymap table. This means that applications that are looking for a Multi_key (including the default modifier map) won't notice any change.

```
% xmodmap -e "keysym Multi_key = Multi_key Meta_L"
```

Similarly, some keyboards have an Alt key but no Meta key. In that case the following may be useful:

```
% xmodmap -e "keysym Alt_L = Meta_L Alt_L"
```

One of the more simple, yet convenient, uses of *xmodmap* is to set the keyboard's "rubout" key to generate an alternate keysym. This frequently involves exchanging Backspace with Delete to be more comfortable to the user. If the *ttyModes* resource in *xterm* is set as well, all terminal emulator windows will use the same key for erasing characters:

```
% xmodmap -e "keysym BackSpace = Delete"
% echo "XTerm*ttyModes: erase ^?" | xrdb -merge
```

Some keyboards do not automatically generate less than and greater than characters when the comma and period keys are shifted. This can be remedied with *xmodmap* by resetting the bindings for the comma and period with the following scripts:

```
!
! make shift-, be < and shift-. be >
!
keysym comma = comma less
keysym period = period greater
```

One of the more irritating differences between keyboards is the location of the Control and Shift Lock keys. A common use of *xmodmap* is to swap these two keys as follows:

```
!
! Swap Caps_Lock and 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
```

The *keycode* command is useful for assigning the same keysym to multiple keycodes. Although unportable, it also 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 shown above), flushes all existing caps lock bindings, makes the CapsLock key be a control key, make F5 generate Escape, and makes Break/Reset be a shift lock.

```
!
! On the HP, the following keycodes have key caps as listed:
!
!    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
```

ENVIRONMENT

DISPLAY
to get default host and display number.

SEE ALSO

X(1), xev(1), *Xlib* documentation on key and pointer events

BUGS

Every time a **keycode** expression is evaluated, the server generates a *MappingNotify* event on every client. This can cause some thrashing. All of the changes should be batched together and done at once. Clients that receive keyboard input and ignore *MappingNotify* events will not notice any changes made to keyboard mappings.

Xmodmap should generate "add" and "remove" expressions automatically whenever a keycode that is already bound to a modifier is changed.

There should be a way to have the *remove* expression accept keycodes as well as keysyms for those times when you really mess up your mappings.

AUTHOR

Jim Fulton, MIT X Consortium, rewritten from an earlier version by David Rosenthal of Sun Microsystems.

NAME

xprop - property displayer for X

SYNOPSIS

xprop [-help] [-grammar] [-id *id*] [-root] [-name *name*] [-frame] [-font *font*] [-display *display*] [-len *n*] [-notype] [-fs *file*] [-remove *property-name*] [-spy] [-f *atom format* [*dformat*]]* [*format* [*dformat*] *atom*]*

SUMMARY

The *xprop* utility is for displaying window and font properties in an X server. One window or font is selected using the command line arguments or possibly in the case of a window, by clicking on the desired window. A list of properties is then given, possibly with formatting information.

OPTIONS

-help Print out a summary of command line options.

-grammar
Print out a detailed grammar for all command line options.

-id *id* This argument allows the user to select window *id* on the command line rather than using the pointer to select the target window. This is very useful in debugging X applications where the target window is not mapped to the screen or where the use of the pointer might be impossible or interfere with the application.

-name *name*
This argument allows the user to specify that the window named *name* is the target window on the command line rather than using the pointer to select the target window.

-font *font*
This argument allows the user to specify that the properties of font *font* should be displayed.

-root This argument specifies that X's root window is the target window. This is useful in situations where the root window is completely obscured.

-display *display*
This argument allows you to specify the server to connect to; see *X(1)*.

-len *n* Specifies that at most *n* bytes of any property should be read or displayed.

-notype Specifies that the type of each property should not be displayed.

-fs *file* Specifies that file *file* should be used as a source of more formats for properties.

-frame Specifies that when selecting a window by hand (i.e. if none of **-name**, **-root**, or **-id** are given), look at the window manager frame (if any) instead of looking for the client window.

-remove *property-name*
Specifies the name of a property to be removed from the indicated window.

-spy Examine window properties forever, looking for property change events.

-f *name format* [*dformat*]
Specifies that the *format* for *name* should be *format* and that the *dformat* for *name* should be *dformat*. If *dformat* is missing, " = $0+\n" is assumed.

DESCRIPTION

For each of these properties, its value on the selected window or font is printed using the supplied formatting information if any. If no formatting information is supplied, internal defaults are used. If a property is not defined on the selected window or font, "not defined" is printed as the value for that property. If no property list is given, all the properties possessed by the selected window or font are printed.

A window may be selected in one of four ways. First, if the desired window is the root window, the -root argument may be used. If the desired window is not the root window, it may be selected in two ways on the command line, either by id number such as might be obtained from *xwininfo*, or by name if the window possesses a name. The -id argument selects a window by id number in either decimal or hex (must start with 0x) while the -name argument selects a window by name.

The last way to select a window does not involve the command line at all. If none of -font, -id, -name, and -root are specified, a crosshairs cursor is displayed and the user is allowed to choose any visible window by pressing any pointer button in the desired window. If it is desired to display properties of a font as opposed to a window, the -font argument must be used.

Other than the above four arguments and the -help argument for obtaining help, and the -grammar argument for listing the full grammar for the command line, all the other command line arguments are used in specifying both the format of the properties to be displayed and how to display them. The -len *n* argument specifies that at most *n* bytes of any given property will be read and displayed. This is useful for example when displaying the cut buffer on the root window which could run to several pages if displayed in full.

Normally each property name is displayed by printing first the property name then its type (if it has one) in parentheses followed by its value. The -notype argument specifies that property types should not be displayed. The -fs argument is used to specify a file containing a list of formats for properties while the -f argument is used to specify the format for one property.

The formatting information for a property actually consists of two parts, a *format* and a *dformat*. The *format* specifies the actual formatting of the property (i.e., is it made up of words, bytes, or longs?, etc.) while the *dformat* specifies how the property should be displayed.

The following paragraphs describe how to construct *format*s and *dformat*s. However, for the vast majority of users and uses, this should not be necessary as the built in defaults contain the *format*s and *dformat*s necessary to display all the standard properties. It should only be necessary to specify *format*s and *dformat*s if a new property is being dealt with or the user dislikes the standard display format. New users especially are encouraged to skip this part.

A *format* consists of one of 0, 8, 16, or 32 followed by a sequence of one or more format characters. The 0, 8, 16, or 32 specifies how many bits per field there are in the property. Zero is a special case meaning use the field size information associated with the property itself. (This is only needed for special cases like type INTEGER which is actually three different types depending on the size of the fields of the property)

A value of 8 means that the property is a sequence of bytes while a value of 16 would mean that the property is a sequence of words. The difference between these two lies in the fact that the sequence of words will be byte swapped while the sequence of bytes will not be when read by a machine of the opposite byte order of the machine that originally wrote the property. For more information on how properties are formatted and stored, consult the Xlib manual.

Once the size of the fields has been specified, it is necessary to specify the type of each field (i.e., is it an integer, a string, an atom, or what?) This is done using one format character per field. If there are more fields in the property than format characters supplied, the last character will be repeated as many times as necessary for the extra fields. The format characters and their meaning are as follows:

a The field holds an atom number. A field of this type should be of size 32.

b The field is an boolean. A 0 means false while anything else means true.

c The field is an unsigned number, a cardinal.

i The field is a signed integer.

m The field is a set of bit flags, 1 meaning on.

s This field and the next ones until either a 0 or the end of the property represent a sequence of bytes. This format character is only usable with a field size of 8 and is most often used to represent a string.

x The field is a hex number (like 'c' but displayed in hex - most useful for displaying window ids and the like)

An example *format* is 32ica which is the format for a property of three fields of 32 bits each, the first holding a signed integer, the second an unsigned integer, and the third an atom.

The format of a *dformat* unlike that of a *format* is not so rigid. The only limitations on a *dformat* is that one may not start with a letter or a dash. This is so that it can be distinguished from a property name or an argument. A *dformat* is a text string containing special characters instructing that various fields be printed at various points in a manner similar to the formatting string used by printf. For example, the *dformat* " is ($0, $1 \)\n" would render the POINT 3, -4 which has a *format* of 32ii as " is (3, -4)\n".

Any character other than a $, ?, \, or a (in a *dformat* prints as itself. To print out one of $, ?, \, or (precede it by a \. For example, to print out a $, use \$. Several special backslash sequences are provided as shortcuts. \n will cause a newline to be displayed while \t will cause a tab to be displayed. *o* where *o* is an octal number will display character number *o*.

A $ followed by a number *n* causes field number *n* to be displayed. The format of the displayed field depends on the formatting character used to describe it in the corresponding *format*. I.e., if a cardinal is described by 'c' it will print in decimal while if it is described by a 'x' it is displayed in hex.

If the field is not present in the property (this is possible with some properties), <field not available> is displayed instead. $*n*+ will display field number *n* then a comma then field number *n*+1 then another comma then ... until the last field defined. If field *n* is not defined, nothing is displayed. This is useful for a property that is a list of values.

A ? is used to start a conditional expression, a kind of if-then statement. ?*exp*(*text*) will display *text* if and only if *exp* evaluates to non-zero. This is useful for two things. First, it allows fields to be displayed if and only if a flag is set. And second, it allows a value such as a state number to be displayed as a name rather than as just a number. The syntax of *exp* is as follows:

exp ::= *term* | *term*=*exp* | !*exp*

term ::= *n* | $*n* | m*n*

The ! operator is a logical "not", changing 0 to 1 and any non-zero value to 0. = is an equality operator. Note that internally all expressions are evaluated as 32 bit numbers so -1 is not equal to 65535. = returns 1 if the two values are equal and 0 if not. *n* represents the constant value *n* while $*n* represents the value of field number *n*. m*n* is 1 if flag number *n* in the first field having format character 'm' in the corresponding *format* is 1, 0 otherwise.

Examples: ?m3(count: $3\n) displays field 3 with a label of count if and only if flag number 3 (count starts at 0!) is on. ?$2=0(True)?!$2=0(False) displays the inverted value of field 2 as a boolean.

In order to display a property, *xprop* needs both a *format* and a *dformat*. Before *xprop* uses its default values of a *format* of 32x and a *dformat* of " = { $0+ }\n", it searches several places in an attempt to find more specific formats. First, a search is made using the name of the property. If this fails, a search is made using the type of the property. This allows type STRING to be defined with one set of formats while allowing property WM_NAME which is of type STRING to be defined with a different format. In this way, the display formats for a given type can be overridden for specific properties.

The locations searched are in order: the format if any specified with the property name (as in 8x WM_NAME), the formats defined by -f options in last to first order, the contents of the file specified by the -fs option if any, the contents of the file specified by the environmental variable XPROPFORMATS if any, and finally *xprop*'s built in file of formats.

The format of the files referred to by the -fs argument and the XPROPFORMATS variable is one or more lines of the following form:

name format [*dformat*]

Where *name* is either the name of a property or the name of a type, *format* is the *format* to be used with *name* and *dformat* is the *dformat* to be used with *name*. If *dformat* is not present, " = $0+\n" is assumed.

EXAMPLES

To display the name of the root window: *xprop* -root WM_NAME

To display the window manager hints for the clock: *xprop* -name xclock WM_HINTS

To display the start of the cut buffer: *xprop* -root -len 100 CUT_BUFFER0

To display the point size of the fixed font: *xprop* -font fixed POINT_SIZE

To display all the properties of window # 0x200007: *xprop* -id 0x200007

ENVIRONMENT

DISPLAY
To get default display.

XPROPFORMATS
Specifies the name of a file from which additional formats are to be obtained.

SEE ALSO

X(1), xwininfo(1)

AUTHOR

Mark Lillibridge, MIT Project Athena

NAME

xrdb - X server resource database utility

SYNOPSIS

xrdb [-option ...] [*filename*]

DESCRIPTION

Xrdb is used to get or set the contents of the RESOURCE_MANAGER property on the root window of screen 0, or the SCREEN_RESOURCES property on the root window of any or all screens, or everything combined. You would normally run this program from your X startup file. Most X clients use the RESOURCE_MANAGER and SCREEN_RESOURCES properties to get user preferences about color, fonts, and so on for applications. Having this information in the server (where it is available to all clients) instead of on disk, solves the problem in previous versions of X that required you to maintain *defaults* files on every machine that you might use. It also allows for dynamic changing of defaults without editing files. The RESOURCE_MANAGER property is used for resources that apply to all screens of the display. The SCREEN_RESOURCES property on each screen specifies additional (or overriding) resources to be used for that screen. (When there is only one screen, SCREEN_RESOURCES is normally not used, all resources are just placed in the RESOURCE_MANAGER property.) The file specified by *filename* (or the contents from standard input if - or no filename is given) is optionally passed through the C preprocessor with the following symbols defined, based on the capabilities of the server being used:

SERVERHOST=*hostname*
: the hostname portion of the display to which you are connected.

SRVR_*name*
: the SERVERHOST hostname string turned into a legal identifier. For example, "my-dpy.lcs.mit.edu" becomes SRVR_my_dpy_lcs_mit_edu.

HOST=*hostname*
: the same as **SERVERHOST**.

DISPLAY_NUM=*num*
: the number of the display on the server host.

CLIENTHOST=*hostname*
: the name of the host on which *xrdb* is running.

CLNT_*name*
: the CLIENTHOST hostname string turned into a legal identifier. For example, "expo.lcs.mit.edu" becomes CLNT_expo_lcs_mit_edu.

RELEASE=*num*
: the vendor release number for the server. The interpretation of this number will vary depending on VENDOR.

REVISION=*num*
: the X protocol minor version supported by this server (currently 0).

VERSION=*num*
: the X protocol major version supported by this server (should always be 11).

VENDOR="*vendor***"**
: a string literal specifying the vendor of the server.

VNDR_*name*
: the VENDOR name string turned into a legal identifier. For example, "MIT X

Consortium" becomes VNDR_MIT_X_Consortium.

EXT_*name*
A symbol is defined for each protocol extension supported by the server. Each extension string name is turned into a legal identifier. For example, "X3D-PEX" becomes EXT_X3D_PEX.

NUM_SCREENS=*num*
the total number of screens.

SCREEN_NUM=*num*
the number of the current screen (from zero).

BITS_PER_RGB=*num*
the number of significant bits in an RGB color specification. This is the log base 2 of the number of distinct shades of each primary that the hardware can generate. Note that it usually is not related to PLANES.

CLASS=*visualclass*
one of StaticGray, GrayScale, StaticColor, PseudoColor, TrueColor, DirectColor. This is the visual class of the root window.

CLASS_*visualclass***=***visualid*
the visual class of the root window in a form you can *#ifdef* on. The value is the numeric id of the visual.

COLOR defined only if CLASS is one of StaticColor, PseudoColor, TrueColor, or DirectColor.

CLASS_*visualclass_depth***=***num*
A symbol is defined for each visual supported for the screen. The symbol includes the class of the visual and its depth; the value is the numeric id of the visual. (If more than one visual has the same class and depth, the numeric id of the first one reported by the server is used.)

HEIGHT=*num*
the height of the root window in pixels.

WIDTH=*num*
the width of the root window in pixels.

PLANES=*num*
the number of bit planes (the depth) of the root window.

X_RESOLUTION=*num*
the x resolution of the screen in pixels per meter.

Y_RESOLUTION=*num*
the y resolution of the screen in pixels per meter. SRVR_*name*, CLNT_*name*, VNDR_*name*, and EXT_*name* identifiers are formed by changing all characters other than letters and digits into underscores (_). Lines that begin with an exclamation mark (!) are ignored and may be used as comments. Note that since *xrdb* can read from standard input, it can be used to the change the contents of properties directly from a terminal or from a shell script.

OPTIONS

xrdb program accepts the following options:

−help This option (or any unsupported option) will cause a brief description of the allowable options and parameters to be printed.

−display *display*
This option specifies the X server to be used; see *X(1).* It also specifies the screen to use for the *-screen* option, and it specifies the screen from which preprocessor symbols are derived for the *-global* option.

−all This option indicates that operation should be performed on the screen-independent resource property (RESOURCE_MANAGER), as well as the screen-specific property (SCREEN_RESOURCES) on every screen of the display. For example, when used in conjunction with *-query*, the contents of all properties are output. For *-load*, *-override* and *-merge*, the input file is processed once for each screen. The resources which occur in common in the output for every screen are collected, and these are applied as the screen-independent resources. The remaining resources are applied for each individual per-screen property. This the default mode of operation.

−global This option indicates that the operation should only be performed on the screen-independent RESOURCE_MANAGER property.

−screen This option indicates that the operation should only be performed on the SCREEN_RESOURCES property of the default screen of the display.

−screens This option indicates that the operation should be performed on the SCREEN_RESOURCES property of each screen of the display. For *-load*, *-override* and *-merge*, the input file is processed for each screen.

−n This option indicates that changes to the specified properties (when used with *-load*, *-override* or *-merge*) or to the resource file (when used with *-edit*) should be shown on the standard output, but should not be performed.

−quiet This option indicates that warning about duplicate entries should not be displayed.

-cpp *filename*
This option specifies the pathname of the C preprocessor program to be used. Although *xrdb* was designed to use CPP, any program that acts as a filter and accepts the -D, -I, and -U options may be used.

-nocpp This option indicates that *xrdb* should not run the input file through a preprocessor before loading it into properties.

−symbols
This option indicates that the symbols that are defined for the preprocessor should be printed onto the standard output.

−query This option indicates that the current contents of the specified properties should be printed onto the standard output. Note that since preprocessor commands in the input resource file are part of the input file, not part of the property, they won't appear in the output from this option. The **−edit** option can be used to merge the contents of properties back into the input resource file without damaging preprocessor commands.

−load This option indicates that the input should be loaded as the new value of the specified properties, replacing whatever was there (i.e. the old contents are removed). This is the default action.

−override
This option indicates that the input should be added to, instead of replacing, the current contents of the specified properties. New entries override previous

entries.

–merge This option indicates that the input should be merged and lexicographically sorted with, instead of replacing, the current contents of the specified properties.

–remove This option indicates that the specified properties should be removed from the server.

–retain This option indicates that the server should be instructed not to reset if *xrdb* is the first client. This never be necessary under normal conditions, since *xdm* and *xinit* always act as the first client.

–edit *filename*
This option indicates that the contents of the specified properties should be edited into the given file, replacing any values already listed there. This allows you to put changes that you have made to your defaults back into your resource file, preserving any comments or preprocessor lines.

–backup *string*
This option specifies a suffix to be appended to the filename used with **–edit** to generate a backup file.

–D*name[=value]*
This option is passed through to the preprocessor and is used to define symbols for use with conditionals such as

–U*name* This option is passed through to the preprocessor and is used to remove any definitions of this symbol.

–I*directory*
This option is passed through to the preprocessor and is used to specify a directory to search for files that are referenced with *#include.*

FILES

Generalizes *~/.Xdefaults* files.

SEE ALSO

X(1), Xlib Resource Manager documentation, Xt resource documentation

ENVIRONMENT

DISPLAY
to figure out which display to use.

BUGS

The default for no arguments should be to query, not to overwrite, so that it is consistent with other programs.

AUTHORS

Bob Scheifler, Phil Karlton, rewritten from the original by Jim Gettys

NAME

xrefresh - refresh all or part of an X screen

SYNOPSIS

xrefresh [-option ...]

DESCRIPTION

Xrefresh is a simple X program that causes all or part of your screen to be repainted. This is useful when system messages have messed up your screen. *Xrefresh* maps a window on top of the desired area of the screen and then immediately unmaps it, causing refresh events to be sent to all applications. By default, a window with no background is used, causing all applications to repaint ‘‘smoothly.’’ However, the various options can be used to indicate that a solid background (of any color) or the root window background should be used instead.

ARGUMENTS

–white Use a white background. The screen just appears to flash quickly, and then repaint.

–black Use a black background (in effect, turning off all of the electron guns to the tube). This can be somewhat disorienting as everything goes black for a moment.

–solid *color*
Use a solid background of the specified color. Try green.

–root Use the root window background.

–none This is the default. All of the windows simply repaint.

–geometry *WxH+X+Y*
Specifies the portion of the screen to be repainted; see *X(1)*.

–display *display*
This argument allows you to specify the server and screen to refresh; see *X(1)*.

X DEFAULTS

The *xrefresh* program uses the routine *XGetDefault(3X)* to read defaults, so its resource names are all capitalized.

Black, **White**, **Solid**, **None**, **Root**
Determines what sort of window background to use.

Geometry
Determines the area to refresh. Not very useful.

ENVIRONMENT

DISPLAY - To get default host and display number.

SEE ALSO

X(1)

BUGS

It should have just one default type for the background.

AUTHORS

Jim Gettys, Digital Equipment Corp., MIT Project Athena

NAME

xrx - RX helper program

SYNOPSIS

xrx [−*toolkitoption* ...] *filename*

DESCRIPTION

The helper program may be used with any Web browser to interpret documents in the RX MIME type format and start remote applications.

xrx reads in the RX document specified by its *filename*, from which it gets the list of services the application wants to use. Based on this information, **xrx** sets the various requested services, including creating authorization keys if your X server supports the SECURITY extension. It then passes the relevant data, such as the X display name, to the application through an HTTP GET request of the associated CGI script. The Web server then executes the CGI script to start the application. The client runs on the web server host connected to your X server.

INSTALLATION

You need to configure your web browser to use **xrx** for RX documents. Generally the following line in your $HOME/.mailcap is enough:

```
application/x-rx; xrx %s
```

However, you may need to refer to your web browser's documentation for exact instructions on configuring helper applications.

Once correctly configured, your browser will activate the helper program whenever you retrieve any document of the MIME type *application/x-rx*.

OPTIONS

The *xrx* helper program accepts all of the standard X Toolkit command line options such as:

−xrm *resourcestring*
: This option specifies a resource string to be used. There may be several instances of this option on the command line.

RESOURCES

The application class name of the *xrx* program is Xrx and it understands the following application resource names and classes:

xrxHasFirewallProxy (class **XrxHasFirewallProxy**)
: Specifies whether an X server firewall proxy (see xfwp) is running and should be used. Default is "False."

xrxInternalWebServers (class **XrxInternalWebServers**)
: The web servers for which the X server firewall proxy should not be used (only relevant when **xrxHasFirewallProxy** is "True"). Its value is a comma separated list of mask/value pairs to be used to filter internal web servers, based on their address. The mask part specifies which segments of the address are to be considered and the value part specifies what the result should match. For instance the following list:

255.255.255.0/198.112.45.0, 255.255.255.0/198.112.46.0

matches the address sets: 198.112.45.* and 198.112.46.*. More precisely, the test is (address & mask) == value.

xrxFastWebServers (class **XrxFastWebServers)**
The web servers for which LBX should not be used. The resource value is a list of address mask/value pairs, as previously described.

xrxTrustedWebServers (class **XrxTrustedWebServers)**
The web servers from which remote applications should be run as trusted clients. The default is to run remote applications as untrusted clients. The resource value is a list of address mask/value pairs, as previously described.

ENVIRONMENT

The *xrx* helper program uses the standard X environment variables such as ‘‘DISPLAY’’ to get the default X server host and display number. If the RX document requests X-UI-LBX service and the default X server does not advertise the LBX extension, *xrx* will look for the environment variable ‘‘XREALDISPLAY’’ to get a second address for your X server and look for the LBX extension there. When running your browser through *lbx-proxy* you will need to set XREALDISPLAY to the actual address of your server if you wish remote applications to be able to use LBX across the Internet.

If the RX document requests XPRINT service, *xrx* looks for the variable ‘‘XPRINTER’’ to get the printer name and X Print server address to use. If the server address is not specified as part of XPRINTER, *xrx* uses the first one specified through the variable ‘‘XPSERVERLIST’’ when it is set. When it is not *xrx* then tries to use the video server as the print server. If the printer name is not specified via XPRINTER, *xrx* looks for it in the variables ‘‘PDPRINTER’’, then ‘‘LPDEST’’, and finally ‘‘PRINTER’’,

Finally, if you are using a firewall proxy, *xrx* will look for ‘‘PROXY_MANAGER’’ to get the address of your proxy manager (see proxymngr). When not specified it will use ":6500" as the default.

KNOWN BUG

When an authorization key is created for a remote application to use the X Print service, the helper program has to create the key with an infinite timeout since nobody knows when the application will actually connect to the X Print server. Therefore, in this case, the helper program stays around to revoke the key when the application goes away (that is when its video key expires). However, if the helper program dies unexpectedly the print authorization key will never get revoked.

SEE ALSO

libxrx (1), xfwp (1), lbxproxy (1), proxymngr (1), The RX Document specification

AUTHOR

Arnaud Le Hors, X Consortium

NAME

libxrx - RX Netscape Navigator Plug-in

DESCRIPTION

The **RX Plug-in** may be used with Netscape Navigator (3.0 or later) to interpret documents in the RX MIME type format and start remote applications.

The **RX Plug-in** reads an RX document, from which it gets the list of services the application wants to use. Based on this information, the **RX Plug-in** sets the various requested services, including creating authorization keys if your X server supports the SECURITY extension. It then passes the relevant data, such as the X display name, to the application through an HTTP GET request of the associated CGI script. The Web server then executes the CGI script to start the application. The client runs on the web server host connected to your X server. In addition when the RX document is used within the EMBED tag (a Netscape extension to HTML), the **RX Plug-in** uses the XC-APPGROUP extension, if it is supported by your X server, to cause the remote application to be embedded within the browser page from which it was launched.

INSTALLATION

To install the **RX Plug-in** so that Netscape Navigator can use it, find the file named libxrx.so.6.3 or libxrx.sl.6.3 (or similar, depending on your platform) in <ProjectRoot>/lib (e.g. /usr/X11R6.4/lib) and copy it to either /usr/local/lib/netscape/plugins or $HOME/.netscape/plugins. Do not install the symlinks libxrx.so or libxrx.sl; they would confuse Netscape.

If you have configured Netscape Navigator to use the RX helper program (**xrx**), you must reconfigure it. Generally you simply need to remove or comment out the line you may have previously added in your mailcap file to use the RX helper program. Otherwise the plug-in will not be enabled. (The usual comment character for mailcap is "#".)

If you are already running Netscape Navigator, you need to exit and restart it after copying the plug-in library so the new plug-in will be found. Once this is done you can check that Navigator has successfully loaded the plug-in by checking the "About Plug-ins" page from the Help menu. This should show something like:

```
                RX Plug-in

 File name: /usr/local/lib/netscape/plugins/libxrx.sl.6.3

 X Remote Activation Plug-in

 Mime Type        Description                 Suffixes  Enabled
 application/x-rx X Remote Activation Plug-in xrx       Yes
```

Once correctly configured, Netscape Navigator will activate the **RX Plug-in** whenever you retrieve any document of the MIME type *application/x-rx*.

RESOURCES

The **RX Plug-in** looks for resources associated with the widget **netscape.Navigator** (class **Netscape.TopLevelShell**) and understands the following resource names and classes:

xrxHasFirewallProxy (class **XrxHasFirewallProxy**)

Specifies whether an X server firewall proxy (see xfwp) is running and should be used. Default is "False." The X firewall proxy uses the X Security

Extension and this extension will only allow clients to connect to the X server if host-based authentication is turned on. See **xfwp(1)** for more information.

xrxInternalWebServers (class **XrxInternalWebServers)**
The web servers for which the X server firewall proxy should not be used (only relevant when **xrxHasFirewallProxy** is ‘‘True’’). Its value is a comma separated list of mask/value pairs to be used to filter internal web servers, based on their address. The mask part specifies which segments of the address are to be considered and the value part specifies what the result should match. For instance the following list:

255.255.255.0/198.112.45.0, 255.255.255.0/198.112.46.0

matches the address sets: 198.112.45.* and 198.112.46.*. More precisely, the test is (address & mask) == value.

xrxFastWebServers (class **XrxFastWebServers)**
The web servers for which LBX should not be used. The resource value is a list of address mask/value pairs, as previously described.

xrxTrustedWebServers (class **XrxTrustedWebServers)**
The web servers from which remote applications should be run as trusted clients. The default is to run remote applications as untrusted clients. The resource value is a list of address mask/value pairs, as previously described.

ENVIRONMENT

If the RX document requests X-UI-LBX service and the default X server does not advertise the LBX extension, the *RX Plug-in* will look for the environment variable ‘‘XREALDISPLAY’’ to get a second address for your X server and look for the LBX extension there. When running your browser through *lbxproxy* you will need to set XREALDISPLAY to the actual address of your server if you wish remote applications to be able to use LBX across the Internet.

If the RX document requests XPRINT service, *RX Plug-in* looks for the variable ‘‘XPRINTER’’ to get the printer name and X Print server address to use. If the server address is not specified as part of XPRINTER, *RX Plug-in* uses the first one specified through the variable ‘‘XPSERVERLIST’’ when it is set. When it is not *RX Plug-in* then tries to use the video server as the print server. If the printer name is not specified via XPRINTER, *RX Plug-in* looks for it in the variables ‘‘PDPRINTER’’, then ‘‘LPDEST’’, and finally ‘‘PRINTER’’,

Finally, if you are using a firewall proxy, *RX Plug-in* will look for ‘‘PROXY_MANAGER’’ to get the address of your proxy manager (see proxymngr). When not specified it will use ":6500" as the default.

KNOWN BUG

When an authorization key is created for a remote application to use the X Print service, the **RX Plug-in** has to create the key with an infinite timeout since nobody knows when the application will actually connect to the X Print server. It then revokes the key when its instance is destroyed (that is when you go to another page). However, if the Plug-in does not get destroyed properly, which happens when Netscape Navigator dies unexpectedly, the print authorization key will never get revoked.

SEE ALSO

xrx (1), xfwp (1), lbxproxy (1), proxymngr (1), The RX Document specification

AUTHORS

Arnaud Le Hors and Kaleb Keithley, X Consortium

NAME

testplugin - a Netscape Plug-in test bed utility

SYNOPSIS

testplugin src=*url* [width=*width*] [height=*height*] [*name*=*value*] ...

DESCRIPTION

This program is designed to provide a means for testing Netscape Navigator UNIX plug-ins. It exercises the plug-in in a way close (I hope) to how Navigator does, and because it is a standalone program it allows you to run it through debugger and to use various program analysis tools.

The line-mode browser **www**, must be in your **PATH** to be able to use testplugin successfully.

ARGUMENTS

src=*url* This argument specifies the source document to use. It should be of the MIME type your plug-in is expecting since unlike within Netscape, no type checking is done and the document is simply streamed to the plug-in.

[width=*width***] [height=***height***]**
These options allow to specify the plug-in window size.

[*name***=***value***]...**
In addition to the arguments described above, any other argument can be specified and will be passed uninterpreted to the plug-in (through the NPP_New method).

CURRENT LIMITATIONS

I've not implemented all of the "Netscape Methods", but only the ones I've needed so far. Also the "Plug-in Methods" are not called in every possible manner so it does not provide a 100% testing coverage.

SEE ALSO

www(1), Netscape Navigator Documentation

AUTHOR

Arnaud Le Hors, X Consortium

NAME

xset - user preference utility for X

SYNOPSIS

xset [-display *display*] [-b] [b on/off] [b [*volume* [*pitch* [*duration*]]] [[-]bc] [-c] [c on/off] [c [*volume*]] [[-+]fp[-+=] *path*[,*path*[,...]]] [fp default] [fp rehash] [[-]led [*integer*]] [led on/off] [m[ouse] [*accel_mult*[/*accel_div*] [*threshold*]]] [m[ouse] default] [p *pixel color*] [[-]r [keycode]] [r on/off] [s [*length* [*period*]]] [s blank/noblank] [s expose/noexpose] [s on/off] [s default] [s activate] [s reset] [q]

DESCRIPTION

This program is used to set various user preference options of the display.

OPTIONS

−display *display*
: This option specifies the server to use; see *X(1)*.

b
: The **b** option controls bell volume, pitch and duration. This option accepts up to three numerical parameters, a preceding dash(-), or a 'on/off' flag. If no parameters are given, or the 'on' flag is used, the system defaults will be used. If the dash or 'off' are given, the bell will be turned off. If only one numerical parameter is given, the bell volume will be set to that value, as a percentage of its maximum. Likewise, the second numerical parameter specifies the bell pitch, in hertz, and the third numerical parameter specifies the duration in milliseconds. Note that not all hardware can vary the bell characteristics. The X server will set the characteristics of the bell as closely as it can to the user's specifications.

bc
: The **bc** option controls *bug compatibility* mode in the server, if possible; a preceding dash(-) disables the mode, otherwise the mode is enabled. Various pre-R4 clients pass illegal values in some protocol requests, and pre-R4 servers did not correctly generate errors in these cases. Such clients, when run against an R4 server, will terminate abnormally or otherwise fail to operate correctly. Bug compatibility mode explicitly reintroduces certain bugs into the X server, so that many such clients can still be run. This mode should be used with care; new application development should be done with this mode disabled. The server must support the MIT-SUNDRY-NONSTANDARD protocol extension in order for this option to work.

c
: The **c** option controls key click. This option can take an optional value, a preceding dash(-), or an 'on/off' flag. If no parameter or the 'on' flag is given, the system defaults will be used. If the dash or 'off' flag is used, keyclick will be disabled. If a value from 0 to 100 is given, it is used to indicate volume, as a percentage of the maximum. The X server will set the volume to the nearest value that the hardware can support.

fp= *path,...*
: The **fp=** sets the font path to the entries given in the path argument. The entries are interpreted by the server, not by the client. Typically they are directory names or font server names, but the interpretation is server-dependent.

fp default
: The **default** argument causes the font path to be reset to the server's default.

fp rehash
: The **rehash** argument resets the font path to its current value, causing the server to reread the font databases in the current font path. This is generally only used

when adding new fonts to a font directory (after running *mkfontdir* to recreate the font database).

−fp or **fp−**
: The **−fp** and **fp−** options remove elements from the current font path. They must be followed by a comma-separated list of entries.

+fp or **fp+**
: This **+fp** and **fp+** options prepend and append elements to the current font path, respectively. They must be followed by a comma-separated list of entries.

led
: The **led** option controls the keyboard LEDs. This controls the turning on or off of one or all of the LEDs. It accepts an optional integer, a preceding dash(-) or an 'on/off' flag. If no parameter or the 'on' flag is given, all LEDs are turned on. If a preceding dash or the flag 'off' is given, all LEDs are turned off. If a value between 1 and 32 is given, that LED will be turned on or off depending on the existence of a preceding dash. A common LED which can be controlled is the "Caps Lock" LED. "xset led 3" would turn led #3 on. "xset -led 3" would turn it off. The particular LED values may refer to different LEDs on different hardware.

m
: The **m** option controls the mouse parameters. The parameters for the mouse are 'acceleration' and 'threshold'. The acceleration can be specified as an integer, or as a simple fraction. The mouse, or whatever pointer the machine is connected to, will go 'acceleration' times as fast when it travels more than 'threshold' pixels in a short time. This way, the mouse can be used for precise alignment when it is moved slowly, yet it can be set to travel across the screen in a flick of the wrist when desired. One or both parameters for the **m** option can be omitted, but if only one is given, it will be interpreted as the acceleration. If no parameters or the flag 'default' is used, the system defaults will be set.

p
: The **p** option controls pixel color values. The parameters are the color map entry number in decimal, and a color specification. The root background colors may be changed on some servers by altering the entries for BlackPixel and WhitePixel. Although these are often 0 and 1, they need not be. Also, a server may choose to allocate those colors privately, in which case an error will be generated. The map entry must not be a read-only color, or an error will result.

r
: The **r** option controls the autorepeat. If a preceding dash or the 'off' flag is used, autorepeat will be disabled. If no parameters or the 'on' flag is used, autorepeat will be enabled. If a specific keycode is specified as a parameter, autorepeat for that keycode is enabled or disabled.

s
: The **s** option lets you set the screen saver parameters. This option accepts up to two numerical parameters, a 'blank/noblank' flag, an 'expose/noexpose' flag, an 'on/off' flag, an 'activate/reset' flag, or the 'default' flag. If no parameters or the 'default' flag is used, the system will be set to its default screen saver characteristics. The 'on/off' flags simply turn the screen saver functions on or off. The 'activate' flag forces activation of screen saver even if the screen saver had been turned off. The 'reset' flag forces deactivation of screen saver if it is active. The 'blank' flag sets the preference to blank the video (if the hardware can do so) rather than display a background pattern, while 'noblank' sets the preference to display a pattern rather than blank the video. The 'expose' flag sets the preference to allow window exposures (the server can freely discard

window contents), while 'noexpose' sets the preference to disable screen saver unless the server can regenerate the screens without causing exposure events. The length and period parameters for the screen saver function determines how long the server must be inactive for screen saving to activate, and the period to change the background pattern to avoid burn in. The arguments are specified in seconds. If only one numerical parameter is given, it will be used for the length.

q The **q** option gives you information on the current settings.

These settings will be reset to default values when you log out.

Note that not all X implementations are guaranteed to honor all of these options.

SEE ALSO

X(1), Xserver(1), xmodmap(1), xrdb(1), xsetroot(1)

AUTHOR

Bob Scheifler, MIT Laboratory for Computer Science
David Krikorian, MIT Project Athena (X11 version)

NAME

xsetroot – root window parameter setting utility for X

SYNOPSIS

xsetroot [-help] [-def] [-display *display*] [-cursor *cursorfile maskfile*] [-cursor_name *cursorname*] [-bitmap *filename*] [-mod *x y*] [-gray] [-grey] [-fg *color*] [-bg *color*] [-rv] [-solid *color*] [-name *string*]

DESCRIPTION

The *setroot* program allows you to tailor the appearance of the background ("root") window on a workstation display running X. Normally, you experiment with *xsetroot* until you find a personalized look that you like, then put the *xsetroot* command that produces it into your X startup file. If no options are specified, or if *-def* is specified, the window is reset to its default state. The *-def* option can be specified along with other options and only the non-specified characteristics will be reset to the default state.

Only one of the background color/tiling changing options (-solid, -gray, -grey, -bitmap, and -mod) may be specified at a time.

OPTIONS

The various options are as follows:

-help Print a usage message and exit.

-def Reset unspecified attributes to the default values. (Restores the background to the familiar gray mesh and the cursor to the hollow x shape.)

-cursor *cursorfile maskfile*
This lets you change the pointer cursor to whatever you want when the pointer cursor is outside of any window. Cursor and mask files are bitmaps (little pictures), and can be made with the *bitmap(1)* program. You probably want the mask file to be all black until you get used to the way masks work.

-cursor_name *cursorname*
This lets you change the pointer cursor to one of the standard cursors from the cursor font. Refer to appendix B of the X protocol for the names (except that the XC_ prefix is elided for this option).

-bitmap *filename*
Use the bitmap specified in the file to set the window pattern. You can make your own bitmap files (little pictures) using the *bitmap(1)* program. The entire background will be made up of repeated "tiles" of the bitmap.

-mod *x y*
This is used if you want a plaid-like grid pattern on your screen. x and y are integers ranging from 1 to 16. Try the different combinations. Zero and negative numbers are taken as 1.

-gray Make the entire background gray. (Easier on the eyes.)

-grey Make the entire background grey.

-fg *color*
Use ‘‘color’’ as the foreground color. Foreground and background colors are meaningful only in combination with -cursor, -bitmap, or -mod.

-bg *color*
Use ‘‘color’’ as the background color.

-rv This exchanges the foreground and background colors. Normally the foreground color is black and the background color is white.

-solid *color*
This sets the background of the root window to the specified color. This option is only useful on color servers.

-name *string*
Set the name of the root window to ‘‘string’’. There is no default value. Usually a name is assigned to a window so that the window manager can use a text representation when the window is iconified. This option is unused since you can’t iconify the background.

-display *display*
Specifies the server to connect to; see *X(1)*.

SEE ALSO
X(1), xset(1), xrdb(1)

AUTHOR
Mark Lillibridge, MIT Project Athena

NAME

xsm – X Session Manager

SYNOPSIS

xsm [-display display] [-session sessionName] [-verbose]

DESCRIPTION

xsm is a session manager. A session is a group of applications, each of which has a particular state. *xsm* allows you to create arbitrary sessions - for example, you might have a "light" session, a "development" session, or an "xterminal" session. Each session can have its own set of applications. Within a session, you can perform a "checkpoint" to save application state, or a "shutdown" to save state and exit the session. When you log back in to the system, you can load a specific session, and you can delete sessions you no longer want to keep.

Some session managers simply allow you to manually specify a list of applications to be started in a session. *xsm* is more powerful because it lets you run applications and have them automatically become part of the session. On a simple level, *xsm* is useful because it gives you this ability to easily define which applications are in a session. The true power of *xsm*, however, can be taken advantage of when more and more applications learn to save and restore their state.

OPTIONS

–display *display*
Causes *xsm* to connect to the specified X display.

–session *sessionName*
Causes *xsm* to load the specified session, bypassing the session menu.

–verbose
Turns on debugging information.

SETUP

.xsession file

Using *xsm* requires a change to your *.xsession* file:

The last program executed by your *.xsession* file should be *xsm*. With this configuration, when the user chooses to shut down the session using *xsm*, the session will truly be over.

Since the goal of the session manager is to restart clients when logging into a session, your .xsession file, in general, should not directly start up applications. Rather, the applications should be started within a session. When *xsm* shuts down the session, *xsm* will know to restart these applications. Note however that there are some types of applications that are not "session aware". *xsm* allows you to manually add these applications to your session (see the section titled *Client List*).

SM_SAVE_DIR environment variable

If the *SM_SAVE_DIR* environment variable is defined, *xsm* will save all configuration files in this directory. Otherwise, they will be stored in the user's home directory. Session aware applications are also encouraged to save their checkpoint files in the *SM_SAVE_DIR* directory, although the user should not depend on this convention.

Default Startup Applications

The first time *xsm* is started, it will need to locate a list of applications to start up. For example, this list might include a window manager, a session management proxy, and an xterm. *xsm* will first look for the file *.xsmstartup* in the user's home directory. If that file does not exists, it will look for the *system.xsm* file that was set up at installation time.

Note that *xsm* provides a "fail safe" option when the user chooses a session to start up. The fail safe option simply loads the default applications described above.

Each line in the startup file should contain a command to start an application. A sample startup file might look this:

<start of file>
twm
smproxy
xterm
<end of file>

STARTING A SESSION

When *xsm* starts up, it first checks to see if the user previously saved any sessions. If no saved sessions exist, *xsm* starts up a set of default applications (as described above in the section titled *Default Startup Applications*). If at least one session exists, a session menu is presented. The **[-session sessionName]** option forces the specified session to be loaded, bypassing the session menu.

The session menu

The session menu presents the user with a list of sessions to choose from. The user can change the currently selected session with the mouse, or by using the up and down arrows on the keyboard. Note that sessions which are locked (i.e. running on a different display) can not be loaded or deleted.

The following operations can be performed from the session menu:

Load Session Pressing this button will load the currently selected session. Alternatively, hitting the Return key will also load the currently selected session, or the user can double click a session from the list.

Delete Session This operation will delete the currently selected session, along with all of the application checkpoint files associated with the session. After pressing this button, the user will be asked to press the button a second time in order to confirm the operation.

Default/Fail Safe *xsm* will start up a set of default applications (as described above in the section titled *Default Startup Applications*). This is useful when the user wants to start a fresh session, or if the session configuration files were corrupted and the user wants a "fail safe" session.

Cancel Pressing this button will cause *xsm* to exit. It can also be used to cancel a "Delete Session" operation.

CONTROLLING A SESSION

After *xsm* determines which session to load, it brings up its main window, then starts up all applications that are part of the session. The title bar for the session manager's main window will contain the name of the session that was loaded.

The following options are available from *xsm*'s main window:

Client List Pressing this button brings up a window containing a list of all clients that are in the current session. For each client, the host machine that the client is running on is presented. As clients are added and removed from the session, this list is updated to reflect the changes. The user is able to control how these clients are restarted (see below).

By pressing the **View Properties** button, the user can view the session management properties associated with the currently selected client.

By pressing the **Clone** button, the user can start a copy of the selected application.

By pressing the **Kill Client** button, the user can remove a client from the session.

By selecting a restart hint from the **Restart Hint** menu, the user can control the restarting of a client. The following hints are available:

– The **Restart If Running** hint indicates that the client should be restarted in the next session if it is connected to the session manager at the end of the current session.

– The **Restart Anyway** hint indicates that the client should be restarted in the next session even if it exits before the current session is terminated.

– The **Restart Immediately** hint is similar to the **Restart Anyway** hint, but in addition, the client is meant to run continuously. If the client exits, the session manager will try to restart it in the current session.

– The **Restart Never** hint indicates that the client should not be restarted in the next session.

Note that all X applications may not be "session aware". Applications that are not session aware are ones that do not support the X Session Management Protocol or they can not be detected by the Session Management Proxy (see the section titled *THE PROXY*). *xsm* allows the user to manually add such applications to the session. The bottom of the *Client List* window contains a text entry field in which application commands can be typed in. Each command should go on its own line. This information will be saved with the session at checkpoint or shutdown time. When the session is restarted, *xsm* will restart these applications in addition to the regular "session aware" applications.

Pressing the **Done** button removes the **Client List** window.

Session Log... The Session Log window presents useful information about the session. For example, when a session is restarted, all of the restart commands will be displayed in the log window.

Checkpoint

By performing a checkpoint, all applications that are in the session are asked to save their state. Not every application will save its complete state, but at a minimum, the session manager is guaranteed that it will receive the command required to restart the application (along with all command line options). A window manager participating in the session should guarantee that the applications will come back up with the same window configurations.

If the session being checkpointed was never assigned a name, the user will be required to specify a session name. Otherwise, the user can perform the checkpoint using the current session name, or a new session name can be specified. If the session name specified already exists, the user will be given the opportunity to specify a different name or to overwrite the already existing session. Note that a session which is locked can not be overwritten.

When performing a checkpoint, the user must specify a **Save Type** which informs the applications in the session how much state they should save.

The **Local** type indicates that the application should save enough information to restore the state as seen by the user. It should not affect the state as seen by other users. For example, an editor would create a temporary file containing the contents of its editing buffer, the location of the cursor, etc...

The **Global** type indicates that the application should commit all of its data to permanent, globally accessible storage. For example, the editor would simply save the edited file.

The **Both** type indicates that the application should do both of these. For example, the editor would save the edited file, then create a temporary file with information such as the location of the cursor, etc...

In addition to the **Save Type**, the user must specify an **Interact Style**.

The **None** type indicates that the application should not interact with the user while saving state.

The **Errors** type indicates that the application may interact with the user only if an error condition arises.

The **Any** type indicates that the application may interact with the user for any purpose. Note that *xsm* will only allow one application to interact with the user at a time.

After the checkpoint is completed, *xsm* will, if necessary, display a window containing the list of applications which did not report a successful save of state.

Shutdown A shutdown provides all of the options found in a checkpoint, but in addition, can cause the session to exit. Note that if the interaction style is **Errors** or **Any**, the user may cancel the shutdown. The user may also cancel the shutdown if any of the applications report an unsuccessful save of state.

The user may choose to shutdown the session with our without performing a checkpoint.

HOW *XSM* RESPONDS TO SIGNALS

xsm will respond to a SIGTERM signal by performing a shutdown with the following options: fast, no interaction, save type local. This allows the user's session to be saved when the system is being shutdown. It can also be used to perform a remote shutdown of a session.

xsm will respond to a SIGUSR1 signal by performing a checkpoint with the following options: no interaction, save type local. This signal can be used to perform a remote checkpoint of a session.

THE PROXY

Since not all applications have been ported to support the X Session Management Protocol, a proxy service exists to allow "old" clients to work with the session manager. In order for the proxy to detect an application joining a session, one of the following must be true:

- The application maps a top level window containing the **WM_CLIENT_LEADER** property. This property provides a pointer to the client leader window which contains the **WM_CLASS**, **WM_NAME**, **WM_COMMAND**, and **WM_CLIENT_MACHINE** properties.

or ...

- The application maps a top level window which does not contain the **WM_CLIENT_LEADER** property. However, this top level window contains the **WM_CLASS**, **WM_NAME**, **WM_COMMAND**, and **WM_CLIENT_MACHINE** properties.

An application that support the **WM_SAVE_YOURSELF** protocol will receive a **WM_SAVE_YOURSELF** client message each time the session manager issues a checkpoint or shutdown. This allows the application to save state. If an application does not support the **WM_SAVE_YOURSELF** protocol, then the proxy will provide enough information to the session manager to restart the application (using **WM_COMMAND**), but no state will be restored.

REMOTE APPLICATIONS

xsm requires a remote execution protocol in order to restart applications on remote machines. Currently, *xsm* supports the *rstart* protocol. In order to restart an application on remote machine **X**, machine **X** must have *rstart* installed. In the future, additional remote execution protocols may be supported.

SEE ALSO

smproxy(1), rstart(1)

AUTHORS

Ralph Mor, X Consortium
Jordan Brown, Quarterdeck Office Systems

NAME

xstdcmap - X standard colormap utility

SYNOPSIS

xstdcmap [-all] [-best] [-blue] [-default] [-delete *map*] [-display *display*] [-gray] [-green] [-help] [-red] [-verbose]

DESCRIPTION

The *xstdcmap* utility can be used to selectively define standard colormap properties. It is intended to be run from a user's X startup script to create standard colormap definitions in order to facilitate sharing of scarce colormap resources among clients. Where at all possible, colormaps are created with read-only allocations.

OPTIONS

The following options may be used with *xstdcmap*:

−all This option indicates that all six standard colormap properties should be defined on each screen of the display. Not all screens will support visuals under which all six standard colormap properties are meaningful. *xstdcmap* will determine the best allocations and visuals for the colormap properties of a screen. Any previously existing standard colormap properties will be replaced.

−best This option indicates that the RGB_BEST_MAP should be defined.

−blue This option indicates that the RGB_BLUE_MAP should be defined.

−default This option indicates that the RGB_DEFAULT_MAP should be defined.

−delete *map*
This option specifies that a specific standard colormap property, or all such properties, should be removed. *map* may be one of: default, best, red, green, blue, gray, or all.

−display *display*
This option specifies the host and display to use; see *X(1)*.

−gray This option indicates that the RGB_GRAY_MAP should be defined.

−green This option indicates that the RGB_GREEN_MAP should be defined.

−help This option indicates that a brief description of the command line arguments should be printed on the standard error. This will be done whenever an unhandled argument is given to *xstdcmap.*

−red This option indicates that the RGB_RED_MAP should be defined.

−verbose
This option indicates that *xstdcmap* should print logging information as it parses its input and defines the standard colormap properties.

ENVIRONMENT

DISPLAY
to get default host and display number.

SEE ALSO

X(1)

AUTHOR

Donna Converse, MIT X Consortium

NAME

resize − set TERMCAP and terminal settings to current xterm window size

SYNOPSIS

resize [**−u** | **−c**] [**−s** [*row col*]]

DESCRIPTION

Resize prints a shell command for setting the TERM and TERMCAP environment variables to indicate the current size of *xterm* window from which the command is run. For this output to take effect, *resize* must either be evaluated as part of the command line (usually done with a shell alias or function) or else redirected to a file which can then be read in. From the C shell (usually known as */bin/csh*), the following alias could be defined in the user's *.cshrc*:

```
% alias rs 'set noglob; eval `resize`'
```

After resizing the window, the user would type:

```
% rs
```

Users of versions of the Bourne shell (usually known as */bin/sh*) that don't have command functions will need to send the output to a temporary file and the read it back in with the "." command:

```
$ resize > /tmp/out
$ . /tmp/out
```

OPTIONS

The following options may be used with *resize*:

−u This option indicates that Bourne shell commands should be generated even if the user's current shell isn't */bin/sh*.

−c This option indicates that C shell commands should be generated even if the user's current shell isn't */bin/csh*.

−s [*rows columns*]
This option indicates that Sun console escape sequences will be used instead of the special *xterm* escape code. If *rows* and *columns* are given, *resize* will ask the *xterm* to resize itself. However, the window manager may choose to disallow the change.

FILES

/etc/termcap	for the base termcap entry to modify.
~/.cshrc	user's alias for the command.

SEE ALSO

csh(1), tset(1), xterm(1)

AUTHORS

Mark Vandevoorde (MIT-Athena), Edward Moy (Berkeley)

BUGS

The **−u** or **−c** must appear to the left of **−s** if both are specified.

NAME

xterm – terminal emulator for X

SYNOPSIS

xterm [–*toolkitoption* ...] [–option ...]

DESCRIPTION

The *xterm* program is a terminal emulator for the X Window System. It provides DEC VT102 and Tektronix 4014 compatible terminals for programs that can't use the window system directly. If the underlying operating system supports terminal resizing capabilities (for example, the SIGWINCH signal in systems derived from 4.3bsd), *xterm* will use the facilities to notify programs running in the window whenever it is resized.

The VT102 and Tektronix 4014 terminals each have their own window so that you can edit text in one and look at graphics in the other at the same time. To maintain the correct aspect ratio (height/width), Tektronix graphics will be restricted to the largest box with a 4014's aspect ratio that will fit in the window. This box is located in the upper left area of the window.

Although both windows may be displayed at the same time, one of them is considered the "active" window for receiving keyboard input and terminal output. This is the window that contains the text cursor. The active window can be chosen through escape sequences, the "VT Options" menu in the VT102 window, and the "Tek Options" menu in the 4014 window.

EMULATIONS

The VT102 emulation is fairly complete, but does not support smooth scrolling, VT52 mode, the blinking character attribute nor the double-wide and double-size character sets. *Termcap*(5) entries that work with *xterm* include "xterm," "vt102," "vt100" and "ansi," and *xterm* automatically searches the termcap file in this order for these entries and then sets the "TERM" and the "TERMCAP" environment variables.

Many of the special *xterm* features may be modified under program control through a set of escape sequences different from the standard VT102 escape sequences. (See the *Xterm Control Sequences* document.)

The Tektronix 4014 emulation is also fairly good. It supports 12-bit graphics addressing, scaled to the window size. Four different font sizes and five different lines types are supported. There is no write-through or defocused mode support. The Tektronix text and graphics commands are recorded internally by *xterm* and may be written to a file by sending the COPY escape sequence (or through the **Tektronix** menu; see below). The name of the file will be "**COPY***yy***–***MM***–***dd***.***hh***:***mm***:***ss*", where *yy*, *MM*, *dd*, *hh*, *mm* and *ss* are the year, month, day, hour, minute and second when the COPY was performed (the file is created in the directory *xterm* is started in, or the home directory for a login *xterm*).

OTHER FEATURES

Xterm automatically highlights the text cursor when the pointer enters the window (selected) and unhighlights it when the pointer leaves the window (unselected). If the window is the focus window, then the text cursor is highlighted no matter where the pointer is.

In VT102 mode, there are escape sequences to activate and deactivate an alternate screen buffer, which is the same size as the display area of the window. When activated, the current screen is saved and replaced with the alternate screen. Saving of lines scrolled off the top of the window is disabled until the normal screen is restored. The *termcap*(5) entry for *xterm* allows the visual editor *vi*(1) to switch to the alternate screen for editing and to restore the screen on exit.

In either VT102 or Tektronix mode, there are escape sequences to change the name of the windows. See *Xterm Control Sequences* for details.

OPTIONS

The *xterm* terminal emulator accepts all of the standard X Toolkit command line options as well as the following (if the option begins with a '**+**' instead of a '**−**', the option is restored to its default value):

−help This causes *xterm* to print out a verbose message describing its options.

−132 Normally, the VT102 DECCOLM escape sequence that switches between 80 and 132 column mode is ignored. This option causes the DECCOLM escape sequence to be recognized, and the *xterm* window will resize appropriately.

−ah This option indicates that *xterm* should always highlight the text cursor. By default, *xterm* will display a hollow text cursor whenever the focus is lost or the pointer leaves the window.

+ah This option indicates that *xterm* should do text cursor highlighting based on focus.

−ai This option disables active icon support if that feature was compiled into xterm. This is equivalent to setting the *vt100* resource **activeIcon** to FALSE.

+ai This option enables active icon support if that feature was compiled into xterm. This is equivalent to setting the *vt100* resource **activeIcon** to TRUE.

−b *number*
This option specifies the size of the inner border (the distance between the outer edge of the characters and the window border) in pixels. The default is 2.

−cb Set the *vt100* resource **cutToBeginningOfLine** to FALSE.

+cb Set the *vt100* resource **cutToBeginningOfLine** to TRUE.

−cc *characterclassrange*:*value*[,...]
This sets classes indicated by the given ranges for using in selecting by words. See the section specifying character classes.

−cn This option indicates that newlines should not be cut in line-mode selections.

+cn This option indicates that newlines should be cut in line-mode selections.

−cr *color*
This option specifies the color to use for text cursor. The default is to use the same foreground color that is used for text.

−cu This option indicates that *xterm* should work around a bug in the *more*(1) program that causes it to incorrectly display lines that are exactly the width of the window and are followed by a line beginning with a tab (the leading tabs are not displayed). This option is so named because it was originally thought to be a bug in the *curses*(3x) cursor motion package.

+cu This option indicates that *xterm* should not work around the *more*(1) bug mentioned above.

−e *program* **[** *arguments* **...]**
This option specifies the program (and its command line arguments) to be run in the *xterm* window. It also sets the window title and icon name to be the basename of the program being executed if neither *−T* nor *−n* are given on the command line. **This must be the last option on the command line.**

−fb *font* This option specifies a font to be used when displaying bold text. This font must be the same height and width as the normal font. If only one of the normal or bold fonts is specified, it will be used as the normal font and the bold font will be produced by overstriking this font. The default is to do overstriking of the normal font.

−fi This option sets the font for active icons if that feature was compiled in to xterm.

−im Turn on the **useInsertMode** resource.

+im Turn off the **useInsertMode** resource.

−j This option indicates that *xterm* should do jump scrolling. Normally, text is scrolled one line at a time; this option allows *xterm* to move multiple lines at a time so that it doesn't fall as far behind. Its use is strongly recommended since it make *xterm* much faster when scanning through large amounts of text. The VT100 escape sequences for enabling and disabling smooth scroll as well as the "VT Options" menu can be used to turn this feature on or off.

+j This option indicates that *xterm* should not do jump scrolling.

−ls This option indicates that the shell that is started in the *xterm* window will be a login shell (i.e., the first character of argv[0] will be a dash, indicating to the shell that it should read the user's .login or .profile).

+ls This option indicates that the shell that is started should not be a login shell (i.e. it will be a normal "subshell").

−mb This option indicates that *xterm* should ring a margin bell when the user types near the right end of a line. This option can be turned on and off from the "VT Options" menu.

+mb This option indicates that margin bell should not be rung.

−mc milliseconds
This option specifies the maximum time between multi-click selections.

−ms *color*
This option specifies the color to be used for the pointer cursor. The default is to use the foreground color.

−nb *number*
This option specifies the number of characters from the right end of a line at which the margin bell, if enabled, will ring. The default is 10.

−rw This option indicates that reverse-wraparound should be allowed. This allows the cursor to back up from the leftmost column of one line to the rightmost column of the previous line. This is very useful for editing long shell command lines and is encouraged. This option can be turned on and off from the "VT Options" menu.

+rw This option indicates that reverse-wraparound should not be allowed.

−aw This option indicates that auto-wraparound should be allowed. This allows the cursor to automatically wrap to the beginning of the next line when when it is at the rightmost position of a line and text is output.

+aw This option indicates that auto-wraparound should not be allowed.

−s This option indicates that *xterm* may scroll asynchronously, meaning that the screen does not have to be kept completely up to date while scrolling. This allows *xterm* to run faster when network latencies are very high and is typically useful when running across a very large internet or many gateways.

+s This option indicates that *xterm* should scroll synchronously.

−sb This option indicates that some number of lines that are scrolled off the top of the window should be saved and that a scrollbar should be displayed so that those lines can be viewed. This option may be turned on and off from the ‘‘VT Options’’ menu.

+sb This option indicates that a scrollbar should not be displayed.

−sf This option indicates that Sun Function Key escape codes should be generated for function keys.

+sf This option indicates that the standard escape codes should be generated for function keys.

−si This option indicates that output to a window should not automatically reposition the screen to the bottom of the scrolling region. This option can be turned on and off from the ‘‘VT Options’’ menu.

+si This option indicates that output to a window should cause it to scroll to the bottom.

−sk This option indicates that pressing a key while using the scrollbar to review previous lines of text should cause the window to be repositioned automatically in the normal position at the bottom of the scroll region.

+sk This option indicates that pressing a key while using the scrollbar should not cause the window to be repositioned.

−sl *number*
This option specifies the number of lines to save that have been scrolled off the top of the screen. The default is 64.

−t This option indicates that *xterm* should start in Tektronix mode, rather than in VT102 mode. Switching between the two windows is done using the ‘‘Options’’ menus.

+t This option indicates that *xterm* should start in VT102 mode.

−tm *string*
This option specifies a series of terminal setting keywords followed by the characters that should be bound to those functions, similar to the *stty* program. Allowable keywords include: intr, quit, erase, kill, eof, eol, swtch, start, stop, brk, susp, dsusp, rprnt, flush, weras, and lnext. Control characters may be specified as ˆchar (e.g. ˆc or ˆu) and ˆ? may be used to indicate delete.

−tn *name*
This option specifies the name of the terminal type to be set in the TERM environment variable. This terminal type must exist in the *termcap(5)* database and should have *li#* and *co#* entries.

−ut This option indicates that *xterm* shouldn’t write a record into the the system log file */etc/utmp*.

+ut This option indicates that *xterm* should write a record into the system log file */etc/utmp*.

−vb This option indicates that a visual bell is preferred over an audible one. Instead of ringing the terminal bell whenever a Control-G is received, the window will be flashed.

+vb This option indicates that a visual bell should not be used.

−wf This option indicates that *xterm* should wait for the window to be mapped the first time before starting the subprocess so that the initial terminal size settings and environment variables are correct. It is the application's responsibility to catch subsequent terminal size changes.

+wf This option indicates that *xterm* show not wait before starting the subprocess.

−C This option indicates that this window should receive console output. This is not supported on all systems. To obtain console output, you must be the owner of the console device, and you must have read and write permission for it. If you are running X under *xdm* on the console screen you may need to have the session startup and reset programs explicitly change the ownership of the console device in order to get this option to work.

−S*ccn* This option specifies the last two letters of the name of a pseudoterminal to use in slave mode, plus the number of the inherited file descriptor. The option is parsed "%c%c%d". This allows *xterm* to be used as an input and output channel for an existing program and is sometimes used in specialized applications.

The following command line arguments are provided for compatibility with older versions. They may not be supported in the next release as the X Toolkit provides standard options that accomplish the same task.

%geom This option specifies the preferred size and position of the Tektronix window. It is shorthand for specifying the "**tekGeometry*" resource.

This option specifies the preferred position
It is shorthand for specifying the "**iconGeometry*" resource.

−T *string*
This option specifies the title for *xterm*'s windows. It is equivalent to **−title**.

−n *string* This option specifies the icon name for *xterm*'s windows. It is shorthand for specifying the "**iconName*" resource. Note that this is not the same as the toolkit option **−name** (see below). The default icon name is the application name.

−r This option indicates that reverse video should be simulated by swapping the foreground and background colors. It is equivalent to **−rv**.

−w *number*
This option specifies the width in pixels of the border surrounding the window. It is equivalent to **−borderwidth** or **−bw**.

The following standard X Toolkit command line arguments are commonly used with *xterm*:

−bg *color*
This option specifies the color to use for the background of the window. The default is "white."

−bd *color*
This option specifies the color to use for the border of the window. The default is "black."

–bw *number*
This option specifies the width in pixels of the border surrounding the window.

–fg *color*
This option specifies the color to use for displaying text. The default is "black."

–fn *font* This option specifies the font to be used for displaying normal text. The default is *fixed*.

–name *name*
This option specifies the application name under which resources are to be obtained, rather than the default executable file name. *Name* should not contain "." or "*" characters.

–title *string*
This option specifies the window title string, which may be displayed by window managers if the user so chooses. The default title is the command line specified after the **–e** option, if any, otherwise the application name.

–rv This option indicates that reverse video should be simulated by swapping the foreground and background colors.

–geometry *geometry*
This option specifies the preferred size and position of the VT102 window; see *X(1).*

–display *display*
This option specifies the X server to contact; see *X(1).*

–xrm *resourcestring*
This option specifies a resource string to be used. This is especially useful for setting resources that do not have separate command line options.

–iconic This option indicates that *xterm* should ask the window manager to start it as an icon rather than as the normal window.

RESOURCES

The program understands all of the core X Toolkit resource names and classes as well as:

iconGeometry (class **IconGeometry)**
Specifies the preferred size and position of the application when iconified. It is not necessarily obeyed by all window managers.

iconName (class **IconName)**
Specifies the icon name. The default is the application name.

termName (class **TermName)**
Specifies the terminal type name to be set in the TERM environment variable.

title (class **Title)**
Specifies a string that may be used by the window manager when displaying this application.

ttyModes (class **TtyModes)**
Specifies a string containing terminal setting keywords and the characters to which they may be bound. Allowable keywords include: intr, quit, erase, kill, eof, eol, swtch, start, stop, brk, susp, dsusp, rprnt, flush, weras, and lnext. Control characters may be specified as ^char (e.g. ^c or ^u) and ^? may be used to indicate Delete. This is very useful for overriding the default terminal settings

without having to do an *stty* every time an *xterm* is started.

useInsertMode (class **UseInsertMode**)
Force use of insert mode by adding appropriate entries to the TERMCAP environment variable. This is useful if the system termcap is broken. The default is "false."

utmpInhibit (class **UtmpInhibit**)
Specifies whether or not *xterm* should try to record the user's terminal in */etc/utmp*.

sunFunctionKeys (class **SunFunctionKeys**)
Specifies whether or not Sun Function Key escape codes should be generated for function keys instead of standard escape sequences.

waitForMap (class **WaitForMap**)
Specifies whether or not *xterm* should wait for the initial window map before starting the subprocess. The default is "false."

The following resources are specified as part of the *vt100* widget (class *VT100*):

activeIcon (class **ActiveIcon**)
Specifies whether or not active icon windows are to be used when the *xterm* window is iconified, if this feature is compiled into xterm. The active icon is a miniature representation of the content of the window and will update as the content changes. Not all window managers necessarily support application icon windows. Some window managers will allow you to enter keystrokes into the active icon window. The default is "false."

allowSendEvents (class **AllowSendEvents**)
Specifies whether or not synthetic key and button events (generated using the X protocol SendEvent request) should be interpreted or discarded. The default is "false" meaning they are discarded. Note that allowing such events creates a very large security hole.

alwaysHighlight (class **AlwaysHighlight**)
Specifies whether or not *xterm* should always display a highlighted text cursor. By default, a hollow text cursor is displayed whenever the pointer moves out of the window or the window loses the input focus.

appcursorDefault (class **AppcursorDefault**)
If "true," the cursor keys are initially in application mode. The default is "false."

appkeypadDefault (class **AppkeypadDefault**)
If "true," the keypad keys are initially in application mode. The default is "false."

autoWrap (class **AutoWrap**)
Specifies whether or not auto-wraparound should be enabled. The default is "true."

bellSuppressTime (class **BellSuppressTime**)
Number of milliseconds after a bell command is sent during which additional bells will be suppressed. Default is 200. If set non-zero, additional bells will also be suppressed until the server reports that processing of the first bell has

been completed; this feature is most useful with the visible bell.

boldFont (class **BoldFont)**
Specifies the name of the bold font to use instead of overstriking.

c132 (class **C132)**
Specifies whether or not the VT102 DECCOLM escape sequence should be honored. The default is ‘‘false.’’

cutNewline (class **CutNewline)**
If false, triple clicking to select a line does not include the Newline at the end of the line. If true, the Newline is selected. The default is ‘‘true.’’

cutToBeginningOfLine (class **CutToBeginningOfLine)**
If false, triple clicking to select a line selects only from the current word forward. If true, the entire line is selected. The default is ‘‘true.’’

charClass (class **CharClass)**
Specifies comma-separated lists of character class bindings of the form [*low*−]*high*:*value*. These are used in determining which sets of characters should be treated the same when doing cut and paste. See the section on specifying character classes.

curses (class **Curses)**
Specifies whether or not the last column bug in *more*(1) should be worked around. See the **−cu** option for details. The default is ‘‘false.’’

background (class **Background)**
Specifies the color to use for the background of the window. The default is ‘‘white.’’

foreground (class **Foreground)**
Specifies the color to use for displaying text in the window. Setting the class name instead of the instance name is an easy way to have everything that would normally appear in the text color change color. The default is ‘‘black.’’

cursorColor (class **Foreground)**
Specifies the color to use for the text cursor. The default is ‘‘black.’’

eightBitInput (class **EightBitInput**)
If true, Meta characters input from the keyboard are presented as a single character with the eighth bit turned on. If false, Meta characters are converted into a two-character sequence with the character itself preceded by ESC. The default is ‘‘true.’’

eightBitOutput (class **EightBitOutput**)
Specifies whether or not eight-bit characters sent from the host should be accepted as is or stripped when printed. The default is ‘‘true.’’

font (class **Font)**
Specifies the name of the normal font. The default is ‘‘fixed.’’

font1 (class **Font1)**
Specifies the name of the first alternative font.

font2 (class **Font2)**
Specifies the name of the second alternative font.

font3 (class **Font3)**
Specifies the name of the third alternative font.

font4 (class **Font4)**
Specifies the name of the fourth alternative font.

font5 (class **Font5)**
Specifies the name of the fifth alternative font.

font6 (class **Font6)**
Specifies the name of the sixth alternative font.

geometry (class **Geometry)**
Specifies the preferred size and position of the VT102 window.

hpLowerleftBugCompat (class **HpLowerleftBugCompat)**
Specifies whether to work around a bug in HP's *xdb*, which ignores termcap and always sends ESC F to move to the lower left corner. "true" causes xterm to interpret ESC F as a request to move to the lower left corner of the screen. The default is "false."

iconBorderWidth (class **BorderWidth)**
Specifies the border width for the active icon window if this feature is compiled into xterm. The default is 0 (no border). Not all window managers will make the border visible.

iconBorderColor (class **BorderColor)**
Specifies the border color for the active icon window if this feature is compiled into xterm. Not all window managers will make the icon border visible.

iconFont (class **IconFont)**
Specifies the font for the miniature active icon window, if this feature is compiled into xterm. The default is "nil2".

internalBorder (class **BorderWidth)**
Specifies the number of pixels between the characters and the window border. The default is 2.

jumpScroll (class **JumpScroll)**
Specifies whether or not jump scroll should be used. The default is "true."

loginShell (class **LoginShell)**
Specifies whether or not the shell to be run in the window should be started as a login shell. The default is "false."

marginBell (class **MarginBell)**
Specifies whether or not the bell should be run when the user types near the right margin. The default is "false."

multiClickTime (class **MultiClickTime)**
Specifies the maximum time in milliseconds between multi-click select events. The default is 250 milliseconds.

multiScroll (class **MultiScroll)**
Specifies whether or not scrolling should be done asynchronously. The default is "false."

nMarginBell (class **Column)**
Specifies the number of characters from the right margin at which the margin bell should be rung, when enabled.

pointerColor (class **Foreground)**
Specifies the foreground color of the pointer. The default is

"XtDefaultForeground."

pointerColorBackground (class **Background)**
Specifies the background color of the pointer. The default is "XtDefaultBackground."

pointerShape (class **Cursor)**
Specifies the name of the shape of the pointer. The default is "xterm."

resizeGravity (class **ResizeGravity)**
Affects the behavior when the window is resized to be taller or shorter. **NorthWest** specifies that the top line of text on the screen stay fixed. If the window is made shorter, lines are dropped from the bottom; if the window is made taller, blank lines are added at the bottom. This is compatible with the behavior in R4. **SouthWest** (the default) specifies that the bottom line of text on the screen stay fixed. If the window is made taller, additional saved lines will be scrolled down onto the screen; if the window is made shorter, lines will be scrolled off the top of the screen, and the top saved lines will be dropped.

reverseVideo (class **ReverseVideo)**
Specifies whether or not reverse video should be simulated. The default is "false."

reverseWrap (class **ReverseWrap)**
Specifies whether or not reverse-wraparound should be enabled. The default is "false."

saveLines (class **SaveLines)**
Specifies the number of lines to save beyond the top of the screen when a scrollbar is turned on. The default is 64.

scrollBar (class **ScrollBar)**
Specifies whether or not the scrollbar should be displayed. The default is "false."

scrollTtyOutput (class **ScrollCond)**
Specifies whether or not output to the terminal should automatically cause the scrollbar to go to the bottom of the scrolling region. The default is "true."

scrollKey (class **ScrollCond)**
Specifies whether or not pressing a key should automatically cause the scrollbar to go to the bottom of the scrolling region. The default is "false."

scrollLines (class **ScrollLines)**
Specifies the number of lines that the *scroll-back* and *scroll-forw* actions should use as a default. The default value is 1.

signalInhibit (class **SignalInhibit)**
Specifies whether or not the entries in the "Main Options" menu for sending signals to *xterm* should be disallowed. The default is "false."

tekGeometry (class **Geometry)**
Specifies the preferred size and position of the Tektronix window.

tekInhibit (class **TekInhibit)**
Specifies whether or not the escape sequence to enter Tektronix mode should be ignored. The default is "false."

tekSmall (class **TekSmall)**
Specifies whether or not the Tektronix mode window should start in its smallest

size if no explicit geometry is given. This is useful when running *xterm* on displays with small screens. The default is ‘‘false.’’

tekStartup (class **TekStartup)**
Specifies whether or not *xterm* should start up in Tektronix mode. The default is ‘‘false.’’

titeInhibit (class **TiteInhibit)**
Specifies whether or not *xterm* should remove remove *ti* and *te* termcap entries (used to switch between alternate screens on startup of many screen-oriented programs) from the TERMCAP string. If set, *xterm* also ignores the escape sequence to switch to the alternate screen.

translations (class **Translations)**
Specifies the key and button bindings for menus, selections, ‘‘programmed strings,’’ etc. See **ACTIONS** below.

visualBell (class **VisualBell)**
Specifies whether or not a visible bell (i.e. flashing) should be used instead of an audible bell when Control-G is received. The default is ‘‘false.’’

The following resources are specified as part of the *tek4014* widget (class *Tek4014*):

width (class **Width)**
Specifies the width of the Tektronix window in pixels.

height (class **Height)**
Specifies the height of the Tektronix window in pixels.

fontLarge (class **Font)**
Specifies the large font to use in the Tektronix window.

font2 (class **Font)**
Specifies font number 2 to use in the Tektronix window.

font3 (class **Font)**
Specifies font number 3 to use in the Tektronix window.

fontSmall (class **Font)**
Specifies the small font to use in the Tektronix window.

initialFont (class **InitialFont)**
Specifies which of the four Tektronix fonts to use initially. Values are the same as for the *set-tek-text* action. The default is ‘‘large.’’

ginTerminator (class **GinTerminator)**
Specifies what character(s) should follow a GIN report or status report. The possibilities are ‘‘none,’’ which sends no terminating characters, ‘‘CRonly,’’ which sends CR, and ‘‘CR&EOT,’’ which sends both CR and EOT. The default is ‘‘none.’’

The resources that may be specified for the various menus are described in the documentation for the Athena **SimpleMenu** widget. The name and classes of the entries in each of the menus are listed below.

The *mainMenu* has the following entries:

securekbd (class **SmeBSB)**
This entry invokes the **secure()** action.

allowsends (class **SmeBSB)**
This entry invokes the **allow-send-events(toggle)** action.

redraw (class **SmeBSB)**
This entry invokes the **redraw()** action.

line1 (class **SmeLine)**
This is a separator.

suspend (class **SmeBSB)**
This entry invokes the **send-signal(tstp)** action on systems that support job control.

continue (class **SmeBSB)**
This entry invokes the **send-signal(cont)** action on systems that support job control.

interrupt (class **SmeBSB)**
This entry invokes the **send-signal(int)** action.

hangup (class **SmeBSB)**
This entry invokes the **send-signal(hup)** action.

terminate (class **SmeBSB)**
This entry invokes the **send-signal(term)** action.

kill (class **SmeBSB)**
This entry invokes the **send-signal(kill)** action.

line2 (class **SmeLine)**
This is a separator.

quit (class **SmeBSB)**
This entry invokes the **quit()** action.

The *vtMenu* has the following entries:

scrollbar (class **SmeBSB)**
This entry invokes the **set-scrollbar(toggle)** action.

jumpscroll (class **SmeBSB)**
This entry invokes the **set-jumpscroll(toggle)** action.

reversevideo (class **SmeBSB)**
This entry invokes the **set-reverse-video(toggle)** action.

autowrap (class **SmeBSB)**
This entry invokes the **set-autowrap(toggle)** action.

reversewrap (class **SmeBSB)**
This entry invokes the **set-reversewrap(toggle)** action.

autolinefeed (class **SmeBSB)**
This entry invokes the **set-autolinefeed(toggle)** action.

appcursor (class **SmeBSB)**
This entry invokes the **set-appcursor(toggle)** action.

appkeypad (class **SmeBSB)**
This entry invokes the **set-appkeypad(toggle)** action.

scrollkey (class **SmeBSB)**
This entry invokes the **set-scroll-on-key(toggle)** action.

scrollttyoutput (class **SmeBSB)**
This entry invokes the **set-scroll-on-tty-output(toggle)** action.

allow132 (class **SmeBSB)**
This entry invokes the **set-allow132(toggle)** action.

cursesemul (class **SmeBSB)**
This entry invokes the **set-cursesemul(toggle)** action.

visualbell (class **SmeBSB)**
This entry invokes the **set-visualbell(toggle)** action.

marginbell (class **SmeBSB)**
This entry invokes the **set-marginbell(toggle)** action.

altscreen (class **SmeBSB)**
This entry is currently disabled.

activeicon (class **SMeBSB)**
This entry toggles active icons on and off if this feature was compiled into *xterm*. It is enabled only if *xterm* was started with the command line option +ai or the **activeIcon** resource set to ‘‘True.’’

line1 (class **SmeLine)**
This is a separator.

softreset (class **SmeBSB)**
This entry invokes the **soft-reset()** action.

hardreset (class **SmeBSB)**
This entry invokes the **hard-reset()** action.

clearsavedlines (class **SmeBSB)"**
This entry invokes the **clear-saved-lines()** action.

line2 (class **SmeLine)**
This is a separator.

tekshow (class **SmeBSB)**
This entry invokes the **set-visibility(tek,toggle)** action.

tekmode (class **SmeBSB)**
This entry invokes the **set-terminal-type(tek)** action.

vthide (class **SmeBSB)**
This entry invokes the **set-visibility(vt,off)** action.

The *fontMenu* has the following entries:

fontdefault (class **SmeBSB)**
This entry invokes the **set-vt-font(d)** action.

font1 (class **SmeBSB)**
This entry invokes the **set-vt-font(1)** action.

font2 (class **SmeBSB)**
This entry invokes the **set-vt-font(2)** action.

font3 (class **SmeBSB)**
This entry invokes the **set-vt-font(3)** action.

font4 (class **SmeBSB)**
This entry invokes the **set-vt-font(4)** action.

font5 (class **SmeBSB)**
This entry invokes the **set-vt-font(5)** action.

font6 (class **SmeBSB)**
This entry invokes the **set-vt-font(6)** action.

fontescape (class **SmeBSB)**
This entry invokes the **set-vt-font(e)** action.

fontsel (class **SmeBSB)**
This entry invokes the **set-vt-font(s)** action.

The *tekMenu* has the following entries:

tektextlarge (class **SmeBSB)**
This entry invokes the **set-tek-text(l)** action.

tektext2 (class **SmeBSB)**
This entry invokes the **set-tek-text(2)** action.

tektext3 (class **SmeBSB)**
This entry invokes the **set-tek-text(3)** action.

tektextsmall (class **SmeBSB)**
This entry invokes the **set-tek-text(s)** action.

line1 (class **SmeLine)**
This is a separator.

tekpage (class **SmeBSB)**
This entry invokes the **tek-page()** action.

tekreset (class **SmeBSB)**
This entry invokes the **tek-reset()** action.

tekcopy (class **SmeBSB)**
This entry invokes the **tek-copy()** action.

line2 (class **SmeLine)**
This is a separator.

vtshow (class **SmeBSB)**
This entry invokes the **set-visibility(vt,toggle)** action.

vtmode (class **SmeBSB)**
This entry invokes the **set-terminal-type(vt)** action.

tekhide (class **SmeBSB)**
This entry invokes the **set-visibility(tek,toggle)** action.

The following resources are useful when specified for the Athena Scrollbar widget:

thickness (class **Thickness)**
Specifies the width in pixels of the scrollbar.

background (class **Background)**
Specifies the color to use for the background of the scrollbar.

foreground (class **Foreground)**
Specifies the color to use for the foreground of the scrollbar. The ‘‘thumb’’ of the scrollbar is a simple checkerboard pattern alternating pixels for foreground and background color.

POINTER USAGE

Once the VT102 window is created, *xterm* allows you to select text and copy it within the same or other windows.

The selection functions are invoked when the pointer buttons are used with no modifiers, and when they are used with the ‘‘shift’’ key. The assignment of the functions described below to keys and buttons may be changed through the resource database; see **ACTIONS** below.

Pointer button one (usually left) is used to save text into the cut buffer. Move the cursor to beginning of the text, and then hold the button down while moving the cursor to the end of the region and releasing the button. The selected text is highlighted and is saved in the global cut buffer and made the PRIMARY selection when the button is released. Double-clicking selects by words. Triple-clicking selects by lines. Quadruple-clicking goes back to characters, etc. Multiple-click is determined by the time from button up to button down, so you can change the selection unit in the middle of a selection. If the key/button bindings specify that an X selection is to be made, *xterm* will leave the selected text highlighted for as long as it is the selection owner.

Pointer button two (usually middle) ‘types’ (pastes) the text from the PRIMARY selection, if any, otherwise from the cut buffer, inserting it as keyboard input.

Pointer button three (usually right) extends the current selection. (Without loss of generality, you can swap ‘‘right’’ and ‘‘left’’ everywhere in the rest of this paragraph.) If pressed while closer to the right edge of the selection than the left, it extends/contracts the right edge of the selection. If you contract the selection past the left edge of the selection, *xterm* assumes you really meant the left edge, restores the original selection, then extends/contracts the left edge of the selection. Extension starts in the selection unit mode that the last selection or extension was performed in; you can multiple-click to cycle through them.

By cutting and pasting pieces of text without trailing new lines, you can take text from several places in different windows and form a command to the shell, for example, or take output from a program and insert it into your favorite editor. Since the cut buffer is globally shared among different applications, you should regard it as a ‘file’ whose contents you know. The terminal emulator and other text programs should be treating it as if it were a text file, i.e., the text is delimited by new lines.

The scroll region displays the position and amount of text currently showing in the window (highlighted) relative to the amount of text actually saved. As more text is saved (up to the maximum), the size of the highlighted area decreases.

Clicking button one with the pointer in the scroll region moves the adjacent line to the top of the display window.

Clicking button three moves the top line of the display window down to the pointer position.

Clicking button two moves the display to a position in the saved text that corresponds to the pointer’s position in the scrollbar.

Unlike the VT102 window, the Tektronix window dows not allow the copying of text. It does allow Tektronix GIN mode, and in this mode the cursor will change from an arrow to a cross. Pressing any key will send that key and the current coordinate of the cross cursor. Pressing button one, two, or three will return the letters 'l', 'm', and 'r', respectively. If the 'shift' key is pressed when a pointer button is pressed, the corresponding upper case letter is sent. To distinguish a pointer button from a key, the high bit of the character is set (but this is bit is normally stripped unless the terminal mode is RAW; see *tty*(4) for details).

MENUS

Xterm has four menus, named *mainMenu*, *vtMenu*, *fontMenu*, and *tekMenu*. Each menu pops up under the correct combinations of key and button presses. Most menus are divided into two section, separated by a horizontal line. The top portion contains various modes that can be altered. A check mark appears next to a mode that is currently active. Selecting one of these modes toggles its state. The bottom portion of the menu are command entries; selecting one of these performs the indicated function.

The **xterm** menu pops up when the ''control'' key and pointer button one are pressed in a window. The *mainMenu* contains items that apply to both the VT102 and Tektronix windows. The **Secure Keyboard** mode is be used when typing in passwords or other sensitive data in an unsecure environment; see **SECURITY** below. Notable entries in the command section of the menu are the **Continue**, **Suspend**, **Interrupt**, **Hangup**, **Terminate** and **Kill** which sends the SIGCONT, SIGTSTP, SIGINT, SIGHUP, SIGTERM and SIGKILL signals, respectively, to the process group of the process running under *xterm* (usually the shell). The **Continue** function is especially useful if the user has accidentally typed CTRL-Z, suspending the process.

The *vtMenu* sets various modes in the VT102 emulation, and is popped up when the ''control'' key and pointer button two are pressed in the VT102 window. In the command section of this menu, the soft reset entry will reset scroll regions. This can be convenient when some program has left the scroll regions set incorrectly (often a problem when using VMS or TOPS-20). The full reset entry will clear the screen, reset tabs to every eight columns, and reset the terminal modes (such as wrap and smooth scroll) to their initial states just after *xterm* has finished processing the command line options.

The *fontMenu* sets the font used in the VT102 window. In addition to the default font and a number of alternatives that are set with resources, the menu offers the font last specified by the Set Font escape sequence (see the document *Xterm Control Sequences*) and the current selection as a font name (if the PRIMARY selection is owned).

The *tekMenu* sets various modes in the Tektronix emulation, and is popped up when the ''control'' key and pointer button two are pressed in the Tektronix window. The current font size is checked in the modes section of the menu. The **PAGE** entry in the command section clears the Tektronix window.

SECURITY

X environments differ in their security consciousness. Most servers, run under *xdm*, are capable of using a ''magic cookie'' authorization scheme that can provide a reasonable level of security for many people. If your server is only using a host-based mechanism to control access to the server (see *xhost(1)*), then if you enable access for a host and other users are also permitted to run clients on that same host, there is every possibility that someone can run an application that will use the basic services of the X protocol to snoop on your activities, potentially capturing a transcript of everything you type at the keyboard. This is of particular concern when you want to type in a password or other sensitive data. The best solution to this problem is to use a better authorization mechanism that host-based control, but a simple mechanism exists for protecting keyboard input in *xterm*.

The **xterm** menu (see **MENUS** above) contains a **Secure Keyboard** entry which, when enabled, ensures that all keyboard input is directed *only* to *xterm* (using the GrabKeyboard protocol request). When an application prompts you for a password (or other sensitive data), you can enable **Secure Keyboard** using the menu, type in the data, and then disable **Secure Keyboard** using the menu again. Only one X client at a time can secure the keyboard, so when you attempt to enable **Secure Keyboard** it may fail. In this case, the bell will sound. If the **Secure Keyboard** succeeds, the foreground and background colors will be exchanged (as if you selected the **Reverse Video** entry in the **Modes** menu); they will be exchanged again when you exit secure mode. If the colors do *not* switch, then you should be *very* suspicious that you are being spoofed. If the application you are running displays a prompt before asking for the password, it is safest to enter secure mode *before* the prompt gets displayed, and to make sure that the prompt gets displayed correctly (in the new colors), to minimize the probability of spoofing. You can also bring up the menu again and make sure that a check mark appears next to the entry.

Secure Keyboard mode will be disabled automatically if your xterm window becomes iconified (or otherwise unmapped), or if you start up a reparenting window manager (that places a title bar or other decoration around the window) while in **Secure Keyboard** mode. (This is a feature of the X protocol not easily overcome.) When this happens, the foreground and background colors will be switched back and the bell will sound in warning.

CHARACTER CLASSES

Clicking the middle mouse button twice in rapid succession will cause all characters of the same class (e.g. letters, white space, punctuation) to be selected. Since different people have different preferences for what should be selected (for example, should filenames be selected as a whole or only the separate subnames), the default mapping can be overridden through the use of the *charClass* (class *CharClass*) resource.

This resource is a series of comma-separated of *range*:*value* pairs. The *range* is either a single number or *low*-*high* in the range of 0 to 127, corresponding to the ASCII code for the character or characters to be set. The *value* is arbitrary, although the default table uses the character number of the first character occurring in the set.

The default table is

```
static int charClass[128] = {
/* NUL  SOH  STX  ETX  EOT  ENQ  ACK  BEL */
    32,   1,   1,   1,   1,   1,   1,   1,
/*  BS   HT   NL   VT   NP   CR   SO   SI */
     1,  32,   1,   1,   1,   1,   1,   1,
/* DLE  DC1  DC2  DC3  DC4  NAK  SYN  ETB */
     1,   1,   1,   1,   1,   1,   1,   1,
/* CAN   EM  SUB  ESC   FS   GS   RS   US */
     1,   1,   1,   1,   1,   1,   1,   1,
/*  SP    !    "    #    $    %    &    ' */
    32,  33,  34,  35,  36,  37,  38,  39,
/*   (    )    *    +    ,    -    .    / */
    40,  41,  42,  43,  44,  45,  46,  47,
/*   0    1    2    3    4    5    6    7 */
    48,  48,  48,  48,  48,  48,  48,  48,
```

```
/*   8     9     :     ;     <     =     >     ? */
    48,   48,   58,   59,   60,   61,   62,   63,
/*   @     A     B     C     D     E     F     G */
    64,   48,   48,   48,   48,   48,   48,   48,
/*   H     I     J     K     L     M     N     O */
    48,   48,   48,   48,   48,   48,   48,   48,
/*   P     Q     R     S     T     U     V     W */
    48,   48,   48,   48,   48,   48,   48,   48,
/*   X     Y     Z     [     \     ]     ^     _ */
    48,   48,   48,   91,   92,   93,   94,   48,
/*   `     a     b     c     d     e     f     g */
    96,   48,   48,   48,   48,   48,   48,   48,
/*   h     i     j     k     l     m     n     o */
    48,   48,   48,   48,   48,   48,   48,   48,
/*   p     q     r     s     t     u     v     w */
    48,   48,   48,   48,   48,   48,   48,   48,
/*   x     y     z     {     |     }     ~   DEL */
    48,   48,   48,  123,  124,  125,  126,    1};
```

For example, the string "33:48,37:48,45-47:48,64:48" indicates that the exclamation mark, percent sign, dash, period, slash, and ampersand characters should be treated the same way as characters and numbers. This is useful for cutting and pasting electronic mailing addresses and filenames.

ACTIONS

It is possible to rebind keys (or sequences of keys) to arbitrary strings for input, by changing the translations for the vt100 or tek4014 widgets. Changing the translations for events other than key and button events is not expected, and will cause unpredictable behavior. The following actions are provided for using within the *vt100* or *tek4014* **translations** resources:

bell([*percent***])**
: This action rings the keyboard bell at the specified percentage above or below the base volume.

ignore() This action ignores the event but checks for special pointer position escape sequences.

insert() This action inserts the character or string associated with the key that was pressed.

insert-seven-bit()
: This action is a synonym for **insert()**

insert-eight-bit()
: This action inserts an eight-bit (Meta) version of the character or string associated with the key that was pressed. The exact action depends on the value of the **eightBitInput** resource.

insert-selection(*sourcename* **[, ...])**
: This action inserts the string found in the selection or cutbuffer indicated by *sourcename*. Sources are checked in the order given (case is significant) until one is found. Commonly-used selections include: *PRIMARY*, *SECONDARY*, and *CLIPBOARD*. Cut buffers are typically named *CUT_BUFFER0* through *CUT_BUFFER7*.

keymap(*name***)**
This action dynamically defines a new translation table whose resource name is *name* with the suffix *Keymap* (case is significant). The name *None* restores the original translation table.

popup-menu(*menuname***)**
This action displays the specified popup menu. Valid names (case is significant) include: *mainMenu*, *vtMenu*, *fontMenu*, and *tekMenu*.

secure() This action toggles the *Secure Keyboard* mode described in the section named **SECURITY**, and is invoked from the **securekbd** entry in *mainMenu*.

select-start()
This action begins text selection at the current pointer location. See the section on **POINTER USAGE** for information on making selections.

select-extend()
This action tracks the pointer and extends the selection. It should only be bound to Motion events.

select-end(*destname* **[, ...])**
This action puts the currently selected text into all of the selections or cut-buffers specified by *destname*.

select-cursor-start()
This action is similar to **select-start** except that it begins the selection at the current text cursor position.

select-cursor-end(*destname* **[, ...])**
This action is similar to **select-end** except that it should be used with **select-cursor-start**.

set-vt-font(*d/1/2/3/4/5/6/e/s* **[,***normalfont* **[,** *boldfont***]])**
This action sets the font or fonts currently being used in the VT102 window. The first argument is a single character that specifies the font to be used: *d* or *D* indicate the default font (the font initially used when *xterm* was started), *1* through *6* indicate the fonts specified by the *font1* through *font6* resources, *e* or *E* indicate the normal and bold fonts that have been set through escape codes (or specified as the second and third action arguments, respectively), and *s* or *S* indicate the font selection (as made by programs such as *xfontsel(1)*) indicated by the second action argument.

start-extend()
This action is similar to **select-start** except that the selection is extended to the current pointer location.

start-cursor-extend()
This action is similar to **select-extend** except that the selection is extended to the current text cursor position.

string(*string***)**
This action inserts the specified text string as if it had been typed. Quotation is necessary if the string contains whitespace or non-alphanumeric characters. If the string argument begins with the characters ‘‘0x’’, it is interpreted as a hex character constant.

scroll-back(*count* **[,***units***])**
This action scrolls the text window backward so that text that had previously

scrolled off the top of the screen is now visible. The *count* argument indicates the number of *units* (which may be *page*, *halfpage*, *pixel*, or *line*) by which to scroll.

scroll-forw(*count* **[,***units***])**
This action scrolls is similar to **scroll-back** except that it scrolls the other direction.

allow-send-events(*on/off/toggle***)**
This action set or toggles the **allowSendEvents** resource and is also invoked by the **allowsends** entry in *mainMenu*.

redraw()
This action redraws the window and is also invoked by the *redraw* entry in *mainMenu*.

send-signal(*signame***)**
This action sends the signal named by *signame* to the *xterm* subprocess (the shell or program specified with the *−e* command line option) and is also invoked by the **suspend**, **continue**, **interrupt**, **hangup**, **terminate**, and **kill** entries in *mainMenu*. Allowable signal names are (case is not significant): *tstp* (if supported by the operating system), *suspend* (same as *tstp*), *cont* (if supported by the operating system), *int*, *hup*, *term*, *quit*, *alrm*, *alarm* (same as *alrm*) and *kill*.

quit() This action sends a SIGHUP to the subprogram and exits. It is also invoked by the **quit** entry in *mainMenu*.

set-scrollbar(*on/off/toggle***)**
This action toggles the **scrollbar** resource and is also invoked by the **scrollbar** entry in *vtMenu*.

set-jumpscroll(*on/off/toggle***)**
This action toggles the **jumpscroll** resource and is also invoked by the **jumpscroll** entry in *vtMenu*.

set-reverse-video(*on/off/toggle***)**
This action toggles the *reverseVideo* resource and is also invoked by the **reversevideo** entry in *vtMenu*.

set-autowrap(*on/off/toggle***)**
This action toggles automatic wrapping of long lines and is also invoked by the **autowrap** entry in *vtMenu*.

set-reversewrap(*on/off/toggle***)**
This action toggles the **reverseWrap** resource and is also invoked by the **reversewrap** entry in *vtMenu*.

set-autolinefeed(*on/off/toggle***)**
This action toggles automatic insertion of linefeeds and is also invoked by the **autolinefeed** entry in *vtMenu*.

set-appcursor(*on/off/toggle***)**
This action toggles the handling Application Cursor Key mode and is also invoked by the **appcursor** entry in *vtMenu*.

set-appkeypad(*on/off/toggle***)**
This action toggles the handling of Application Keypad mode and is also invoked by the **appkeypad** entry in *vtMenu*.

set-scroll-on-key(*on/off/toggle***)**
This action toggles the **scrollKey** resource and is also invoked from the **scrollkey** entry in *vtMenu*.

set-scroll-on-tty-output(*on/off/toggle***)**
This action toggles the **scrollTtyOutput** resource and is also invoked from the **scrollttyoutput** entry in *vtMenu*.

set-allow132(*on/off/toggle***)**
This action toggles the **c132** resource and is also invoked from the **allow132** entry in *vtMenu*.

set-cursesemul(*on/off/toggle***)**
This action toggles the **curses** resource and is also invoked from the **cursesemul** entry in *vtMenu*.

set-visual-bell(*on/off/toggle***)**
This action toggles the **visualBell** resource and is also invoked by the **visualbell** entry in *vtMenu*.

set-marginbell(*on/off/toggle***)**
This action toggles the **marginBell** resource and is also invoked from the **marginbell** entry in *vtMenu*.

set-altscreen(*on/off/toggle***)**
This action toggles between the alternate and current screens.

soft-reset()
This action resets the scrolling region and is also invoked from the **softreset** entry in *vtMenu*.

hard-reset()
This action resets the scrolling region, tabs, window size, and cursor keys and clears the screen. It is also invoked from the **hardreset** entry in *vtMenu*.

clear-saved-lines()
This action does **hard-reset()** (see above) and also clears the history of lines saved off the top of the screen. It is also invoked from the **clearsavedlines** entry in *vtMenu*.

set-terminal-type(*type***)**
This action directs output to either the *vt* or *tek* windows, according to the *type* string. It is also invoked by the **tekmode** entry in *vtMenu* and the **vtmode** entry in *tekMenu*.

set-visibility(*vt/tek***,***on/off/toggle***)**
This action controls whether or not the *vt* or *tek* windows are visible. It is also invoked from the **tekshow** and **vthide** entries in *vtMenu* and the **vtshow** and **tekhide** entries in *tekMenu*.

set-tek-text(*large/2/3/small***)**
This action sets font used in the Tektronix window to the value of the resources **tektextlarge**, **tektext2**, **tektext3**, and **tektextsmall** according to the argument. It is also by the entries of the same names as the resources in *tekMenu*.

tek-page()
This action clears the Tektronix window and is also invoked by the **tekpage** entry in *tekMenu*.

tek-reset()
This action resets the Tektronix window and is also invoked by the *tekreset* entry in *tekMenu*.

tek-copy()
This action copies the escape codes used to generate the current window contents to a file in the current directory beginning with the name COPY. It is also invoked from the *tekcopy* entry in *tekMenu*.

visual-bell()
This action flashes the window quickly.

The Tektronix window also has the following action:

gin-press(*l/L/m/M/r/R*)
This action sends the indicated graphics input code.

The default bindings in the VT102 window are:

```
          Shift <KeyPress> Prior:      scroll-back(1,halfpage) \n\
           Shift <KeyPress> Next:      scroll-forw(1,halfpage) \n\
         Shift <KeyPress> Select:      select-cursor-start() \
                            select-cursor-end(PRIMARY, CUT_BUFFER0) \n\
         Shift <KeyPress> Insert:      insert-selection(PRIMARY, CUT_BUFFER0) \n\
                ~Meta<KeyPress>:       insert-seven-bit() \n\
                 Meta<KeyPress>:       insert-eight-bit() \n\
               !Ctrl <Btn1Down>:       popup-menu(mainMenu) \n\
          !Lock Ctrl <Btn1Down>:       popup-menu(mainMenu) \n\
!Lock Ctrl @Num_Lock <Btn1Down>:popup-menu(mainMenu) \n\
   ! @Num_Lock Ctrl <Btn1Down>:popup-menu(mainMenu) \n\
              ~Meta <Btn1Down>:        select-start() \n\
            ~Meta <Btn1Motion>:        select-extend() \n\
               !Ctrl <Btn2Down>:       popup-menu(vtMenu) \n\
          !Lock Ctrl <Btn2Down>:       popup-menu(vtMenu) \n\
!Lock Ctrl @Num_Lock <Btn2Down>:popup-menu(vtMenu) \n\
   ! @Num_Lock Ctrl <Btn2Down>:popup-menu(vtMenu) \n\
        ~Ctrl ~Meta <Btn2Down>:        ignore() \n\
          ~Ctrl ~Meta <Btn2Up>:        insert-selection(PRIMARY, CUT_BUFFER0) \n\
               !Ctrl <Btn3Down>:       popup-menu(fontMenu) \n\
          !Lock Ctrl <Btn3Down>:       popup-menu(fontMenu) \n\
!Lock Ctrl @Num_Lock <Btn3Down>:popup-menu(fontMenu) \n\
   ! @Num_Lock Ctrl <Btn3Down>:popup-menu(fontMenu) \n\
        ~Ctrl ~Meta <Btn3Down>:        start-extend() \n\
            ~Meta <Btn3Motion>:        select-extend() \n\
                      <BtnUp>:         select-end(PRIMARY, CUT_BUFFER0) \n\
                    <BtnDown>:         bell(0)
```

The default bindings in the Tektronix window are:

```
                ~Meta<KeyPress>:       insert-seven-bit() \n\
                 Meta<KeyPress>:       insert-eight-bit() \n\
               !Ctrl <Btn1Down>:       popup-menu(mainMenu) \n\
          !Lock Ctrl <Btn1Down>:       popup-menu(mainMenu) \n\
!Lock Ctrl @Num_Lock <Btn1Down>:popup-menu(mainMenu) \n\
```

```
!Ctrl @Num_Lock <Btn1Down>: popup-menu(mainMenu) \n\
!Ctrl <Btn2Down>:           popup-menu(tekMenu) \n\
!Lock Ctrl <Btn2Down>:      popup-menu(tekMenu) \n\
!Lock Ctrl @Num_Lock <Btn2Down>:popup-menu(tekMenu) \n\
!Ctrl @Num_Lock <Btn2Down>: popup-menu(tekMenu) \n\
Shift ~Meta<Btn1Down>:      gin-press(L) \n\
~Meta<Btn1Down>:            gin-press(l) \n\
Shift ~Meta<Btn2Down>:      gin-press(M) \n\
~Meta<Btn2Down>:            gin-press(m) \n\
Shift ~Meta<Btn3Down>:      gin-press(R) \n\
~Meta<Btn3Down>:            gin-press(r)
```

Below is a sample how of the **keymap()** action is used to add special keys for entering commonly-typed works:

```
*VT100.Translations: #override <Key>F13: keymap(dbx)
*VT100.dbxKeymap.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)
```

ENVIRONMENT

Xterm sets the environment variables ‘‘TERM’’ and ‘‘TERMCAP’’ properly for the size window you have created. It also uses and sets the environment variable ‘‘DISPLAY’’ to specify which bit map display terminal to use. The environment variable ‘‘WINDOWID’’ is set to the X window id number of the *xterm* window.

SEE ALSO

resize(1), X(1), pty(4), tty(4)
Xterm Control Sequences

BUGS

Large pastes do not work on some systems. This is not a bug in *xterm*; it is a bug in the pseudo terminal driver of those systems. *xterm* feeds large pastes to the pty only as fast as the pty will accept data, but some pty drivers do not return enough information to know if the write has succeeded.

Many of the options are not resettable after *xterm* starts.

Only fixed-width, character-cell fonts are supported.

This program still needs to be rewritten. It should be split into very modular sections, with the various emulators being completely separate widgets that don't know about each other. Ideally, you'd like to be able to pick and choose emulator widgets and stick them into a single control widget.

There needs to be a dialog box to allow entry of the Tek COPY file name.

AUTHORS

Far too many people, including:

Loretta Guarino Reid (DEC-UEG-WSL), Joel McCormack (DEC-UEG-WSL), Terry Weissman (DEC-UEG-WSL), Edward Moy (Berkeley), Ralph R. Swick (MIT-Athena), Mark Vandevoorde (MIT-Athena), Bob McNamara (DEC-MAD), Jim Gettys (MIT-Athena), Bob Scheifler (MIT X Consortium), Doug Mink (SAO), Steve Pitschke (Stellar), Ron Newman (MIT-Athena), Jim Fulton (MIT X Consortium), Dave Serisky (HP), Jonathan Kamens (MIT-Athena)

NAME

xwd - dump an image of an X window

SYNOPSIS

xwd [-debug] [-help] [-nobdrs] [-out *file*] [-xy] [-frame] [-add *value*] [-root | -id *id* | -name *name*] [-icmap] [-screen] [-display *display*]

DESCRIPTION

Xwd is an X Window System window dumping utility. *Xwd* allows X users to store window images in a specially formatted dump file. This file can then be read by various other X utilities for redisplay, printing, editing, formatting, archiving, image processing, etc. The target window is selected by clicking the pointer in the desired window. The keyboard bell is rung once at the beginning of the dump and twice when the dump is completed.

OPTIONS

-display *display*
This argument allows you to specify the server to connect to; see *X(1)*.

-help
Print out the 'Usage:' command syntax summary.

-nobdrs
This argument specifies that the window dump should not include the pixels that compose the X window border. This is useful in situations where you may wish to include the window contents in a document as an illustration.

-out *file*
This argument allows the user to explicitly specify the output file on the command line. The default is to output to standard out.

-xy
This option applies to color displays only. It selects 'XY' format dumping instead of the default 'Z' format.

-add *value*
This option specifies an signed value to be added to every pixel.

-frame
This option indicates that the window manager frame should be included when manually selecting a window.

-root
This option indicates that the root window should be selected for the window dump, without requiring the user to select a window with the pointer.

-id *id*
This option indicates that the window with the specified resource id should be selected for the window dump, without requiring the user to select a window with the pointer.

-name *name*
This option indicates that the window with the specified WM_NAME property should be selected for the window dump, without requiring the user to select a window with the pointer.

-icmap
Normally the colormap of the chosen window is used to obtain RGB values. This option forces the first installed colormap of the screen to be used instead.

-screen
This option indicates that the GetImage request used to obtain the image should be done on the root window, rather than directly on the specified window. In this way, you can obtain pieces of other windows that overlap the specified window, and more importantly, you can capture menus or other popups that are independent windows but appear over the specified window.

-silent
Operate silently, i.e. don't ring any bells before and after dumping the window.

ENVIRONMENT

DISPLAY
To get default host and display number.

FILES

XWDFile.h
X Window Dump File format definition file.

SEE ALSO

xwud(1), xpr(1), X(1)

AUTHORS

Tony Della Fera, Digital Equipment Corp., MIT Project Athena
William F. Wyatt, Smithsonian Astrophysical Observatory

NAME

xwininfo − window information utility for X

SYNOPSIS

xwininfo [−help] [−id *id*] [−root] [−name *name*] [−int] [−children] [−tree] [−stats] [−bits] [−events] [−size] [−wm] [−shape] [−frame] [−all] [−english] [−metric] [−display *display*]

DESCRIPTION

Xwininfo is a utility for displaying information about windows. Various information is displayed depending on which options are selected. If no options are chosen, **−stats** is assumed.

The user has the option of selecting the target window with the mouse (by clicking any mouse button in the desired window) or by specifying its window id on the command line with the **−id** option. Or instead of specifying the window by its id number, the **−name** option may be used to specify which window is desired by name. There is also a special **−root** option to quickly obtain information on the screen's root window.

OPTIONS

−help Print out the 'Usage:' command syntax summary.

−id *id* This option allows the user to specify a target window *id* on the command line rather than using the mouse to select the target window. This is very useful in debugging X applications where the target window is not mapped to the screen or where the use of the mouse might be impossible or interfere with the application.

−name *name*
This option allows the user to specify that the window named *name* is the target window on the command line rather than using the mouse to select the target window.

−root This option specifies that X's root window is the target window. This is useful in situations where the root window is completely obscured.

−int This option specifies that all X window ids should be displayed as integer values. The default is to display them as hexadecimal values.

−children
This option causes the root, parent, and children windows' ids and names of the selected window to be displayed.

−tree This option is like **−children** but displays all children recursively.

−stats This option causes the display of various attributes pertaining to the location and appearance of the selected window. Information displayed includes the location of the window, its width and height, its depth, border width, class, colormap id if any, map state, backing-store hint, and location of the corners.

−bits This option causes the display of various attributes pertaining to the selected window's raw bits and how the selected window is to be stored. Displayed information includes the selected window's bit gravity, window gravity, backing-store hint, backing-planes value, backing pixel, and whether or not the window has save-under set.

−events This option causes the selected window's event masks to be displayed. Both the event mask of events wanted by some client and the event mask of events not to propagate are displayed.

−size This option causes the selected window's sizing hints to be displayed. Displayed information includes: for both the normal size hints and the zoom size hints, the user supplied location if any; the program supplied location if any; the user supplied size if any; the program supplied size if any; the minimum size if any; the maximum size if any; the resize increments if any; and the minimum and maximum aspect ratios if any.

−wm This option causes the selected window's window manager hints to be displayed. Information displayed may include whether or not the application accepts input, what the window's icon window # and name is, where the window's icon should go, and what the window's initial state should be.

−shape This option causes the selected window's window and border shape extents to be displayed.

−frame This option causes window manager frames to be considered when manually selecting windows.

−metric This option causes all individual height, width, and x and y positions to be displayed in millimeters as well as number of pixels, based on what the server thinks the resolution is. Geometry specifications that are in **+x+y** form are not changed.

−english This option causes all individual height, width, and x and y positions to be displayed in inches (and feet, yards, and miles if necessary) as well as number of pixels. **−metric** and **−english** may both be enabled at the same time.

−all This option is a quick way to ask for all information possible.

−display *display*
This option allows you to specify the server to connect to; see *X(1)*.

EXAMPLE

The following is a sample summary taken with no options specified:

```
xwininfo: Window id: 0x60000f "xterm"

  Absolute upper-left X: 2
  Absolute upper-left Y: 85
  Relative upper-left X:  0
  Relative upper-left Y:  25
  Width: 579
  Height: 316
  Depth: 8
  Visual Class: PseudoColor
  Border width: 0
  Class: InputOutput
  Colormap: 0x27 (installed)
  Bit Gravity State: NorthWestGravity
  Window Gravity State: NorthWestGravity
  Backing Store State: NotUseful
  Save Under State: no
  Map State: IsViewable
  Override Redirect State: no
  Corners:  +2+85  -699+85  -699-6.4  +2-623
  -geometry 80x24+0+58
```

ENVIRONMENT

DISPLAY
To get the default host and display number.

SEE ALSO

X(1), xprop(1)

BUGS

Using **−stats −bits** shows some redundant information.

The -geometry string displayed must make assumptions about the window's border width and the behavior of the application and the window manager. As a result, the location given is not always correct.

AUTHOR

Mark Lillibridge, MIT Project Athena

NAME

xwud - image displayer for X

SYNOPSIS

xwud [-in *file*] [-noclick] [-geometry *geom*] [-display *display*] [-new] [-std <maptype>] [-raw] [-vis <vis-type-or-id>] [-scale] [-help] [-rv] [-plane *number*] [-fg *color*] [-bg *color*]

DESCRIPTION

Xwud is an X Window System image undumping utility. *Xwud* allows X users to display in a window an image saved in a specially formatted dump file, such as produced by *xwd(1)*.

OPTIONS

-bg *color*
: If a bitmap image (or a single plane of an image) is displayed, this option can be used to specify the color to display for the "0" bits in the image.

-display *display*
: This option allows you to specify the server to connect to; see *X(1)*.

-fg *color*
: If a bitmap image (or a single plane of an image) is displayed, this option can be used to specify the color to display for the "1" bits in the image.

-geometry *geom*
: This option allows you to specify the size and position of the window. Typically you will only want to specify the position, and let the size default to the actual size of the image.

-help
: Print out a short description of the allowable options.

-in *file*
: This option allows the user to explicitly specify the input file on the command line. If no input file is given, the standard input is assumed.

-new
: This option forces creation of a new colormap for displaying the image. If the image characteristics happen to match those of the display, this can get the image on the screen faster, but at the cost of using a new colormap (which on most displays will cause other windows to go technicolor).

-noclick
: Clicking any button in the window will terminate the application, unless this option is specified. Termination can always be achieved by typing 'q', 'Q', or ctrl-c.

-plane *number*
: You can select a single bit plane of the image to display with this option. Planes are numbered with zero being the least significant bit. This option can be used to figure out which plane to pass to *xpr(1)* for printing.

-raw
: This option forces the image to be displayed with whatever color values happen to currently exist on the screen. This option is mostly useful when undumping an image back onto the same screen that the image originally came from, while the original windows are still on the screen, and results in getting the image on the screen faster.

-rv
: If a bitmap image (or a single plane of an image) is displayed, this option forces the foreground and background colors to be swapped. This may be needed when displaying a bitmap image which has the color sense of pixel values "0" and "1" reversed from what they are on your display.

-scale Allow the window to be resized, and scale the image to the size of the window.

-std *maptype*
This option causes the image to be displayed using the specified Standard Colormap. The property name is obtained by converting the type to upper case, prepending "RGB_", and appending "_MAP". Typical types are "best", "default", and "gray". See *xstdcmap(1)* for one way of creating Standard Colormaps.

-vis *vis-type-or-id*
This option allows you to specify a particular visual or visual class. The default is to pick the "best" one. A particular class can be specified: "StaticGray", "GrayScale", "StaticColor", "PseudoColor", "DirectColor", or "TrueColor". Or "Match" can be specified, meaning use the same class as the source image. Alternatively, an exact visual id (specific to the server) can be specified, either as a hexadecimal number (prefixed with "0x") or as a decimal number. Finally, "default" can be specified, meaning to use the same class as the colormap of the root window. Case is not significant in any of these strings.

ENVIRONMENT

DISPLAY
To get default display.

FILES

XWDFile.h
X Window Dump File format definition file.

BUGS

xwud doesn't handle big/deep images very well on servers that don't have the BIG-REQUESTS extension.

SEE ALSO

xwd(1), xpr(1), xstdcmap(1), X(1)

AUTHOR

Bob Scheifler, MIT X Consortium