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 the blinking character attribute nor the double-wide and double-size character sets. **Terminfo** entries that work with **xterm** include ''xterm,'' ''vt102,'' ''vt100'' and ''ansi,'' and **xterm** automatically searches the **/usr/share/lib/terminfo** directory in this order for these entries and then sets the ''TERM'' environment variable (see **terminfo**(4)).

Many of the special **xterm** features (like logging) 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. Four different font sizes and five different lines types are supported. 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 **terminfo**(4) 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 and to specify a new log file name. See *Xterm Control Sequences* for details. Enabling the escape sequence to change the log file name is a compile-time option; by default this escape sequence is ignored for security reasons.

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.

–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.

–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 **curses**(3X) cursor motion package that causes the **more**(1) program to display lines that are exactly the width of the window and are followed by a line beginning with a tab to be displayed incorrectly (the leading tabs are not displayed).

+cu This option indicates that that **xterm** should not work around the **curses**(3X) 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.

–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.

–l This option indicates that B xterm should send all terminal output to a log file as well as to the screen. This option can be enabled or disabled using the ''VT Options'' menu.

+l This option indicates that **xterm** should not do logging.

–lf *filename*
This option specifies the name of the file to which the output log described above is written. If *file* begins with a pipe symbol (|), the rest of the string is assumed to be a command to be used as the endpoint of a pipe. The ability to log to a pipe is a compile-time option which is disabled by default for security reasons. The default filename is ''**XtermLog.***XXXXX*'' (where *XXXXX* is the process id of **xterm**) and is created in the directory from which **xterm** was started (or the user's home directory in the case of a login window).

–ls This option indicates that the shell that is started in the **xterm** window 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.

#*geom* This option specifies the preferred position of the icon window. It is shorthand for specifying the ''**iconGeometry*'' resource.

−T *string*

This option specifies the title for **xterm** It is equivalent to –**title**.

–**n** *string*
This option specifies the icon name for **xterm** 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 –**reversevideo** or –**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**(7)

–**display** *display*
This option specifies the X server to contact; see **X**(7)

–**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.

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.

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*):

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.''

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 *curses*(3x) should be worked around. 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.

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.''

logFile (class **Logfile)**
Specifies the name of the file to which a terminal session is logged. The default is ''**XtermLog.***XXXXX*'' **(where** *XXXXX* **is the process id of xterm**).

logging (class **Logging)**
Specifies whether or not a terminal session should be logged. The default is ''false.''

logInhibit (class **LogInhibit)**
Specifies whether or not terminal session logging should be inhibited. The default is ''false.''

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 Tektronix mode should be disallowed. 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* terminfo entries (used to switch between alternate screens on startup of many screen-oriented programs) from the TERM 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.

logging (class **SmeBSB)**
This entry invokes the **set-logging(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.

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**(1) and **tty**(7) 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. MIT 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 simply a list of *range*:*value* pairs where 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 very 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 cutbuffers 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*.

set-logging(*on/off/toggle***)**
This action toggles the **logging** resource and is also invoked by the **logging** 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 **rever-***sevideo* 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 **rever-***sewrap* 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 Bappcursor 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 **margin-***bell* 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\
~Meta <Btn1Down>:              select-start() \n\
~Meta <Btn1Motion>:            select-extend() \n\
!Ctrl <Btn2Down>:              popup-menu(vtMenu) \n\
!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\
~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\
!Ctrl <Btn2Down>:	popup-menu(tekMenu) \ n\
!Lock Ctrl <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 variable ‘‘TERM’’ 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), **X11**(7), **tty**(1)

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** 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.

The Tek widget does not support key/button re-binding.

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 log file name and the COPY file name.

COPYRIGHT

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)