X11Release 6.1

By Jeanne Sparlin and Dennis Heideman

The graphics subsystem of AIX 4.3 will contain the X11R6.1 release of the X Window System®. This article discusses the new functionality included in the X Window System for X11R6. It also discusses portability concerns and migration issues from older versions of the X Window System and AIXwindows.

The X Window System was originally developed by Project Athena at Massachusetts Institute of Technology® (MIT®). The X Consortium, originally housed at MIT, has been integrated into The Open Group. Today, the X Window System is part of The Open Group (TOG) technology, which is supported by all the major UNIX® vendors. It is also upgradable.

X11R6 was originally delivered from the X Consortium in May 1994. Some additional functionality for X11R6 (Record Extension and Session Management protocol and library) was completed and delivered in March 1996. The X Consortium delivered defect fixes in the form of patches: Patch 1 for X11R6.1 was delivered shortly after its release. X11R6.2 was a software release of functionality needed for CDEnext, the next version of Common Desktop Environment (CDE). X11R6.3, also known as Broadway, was delivered from the X Consortium in December 1996. Broadway consists of five distinct technologies:

- Audio server—Represents a standard for network transparent audio services
- Remote execution—Specifies a MIME format for invoking applications remotely
- Embedding—Views an X application embedded in a Web browser
- Security—Distinguishes and processes applications that originate inside or outside a company's firewall
- Low Bandwidth X (LBX)—Sends compressed X protocol across a serial line

What's New?

AIXwindows® 4.3 is the IBM implementation of the X Window System. X11R6.1 patch 1 contains new functionality in the following areas: X Server, Libraries, and Utilities/Clients.

X Server

In AIXwindows 4.3, the X Server uses the X11R6.1 sample source as its base. Several benefits of this for the X server include:

- Improved maintenance. Maintenance is improved because bug fixes from The Open Group can be easily applied.
- Brandability. The Open Group brands software by using a validation suite of tests called VSW5. The X Consortium

Jeanne Sparlin

Dennis Heideman

sample source contains many bug fixes and functionality that improve the ability to brand the X Server.

- Font functionality. X11R6 contains enhanced font functionality for scalable fonts and glyph rendering on demand.

Changes to the X Server internals do not require changes to X Client applications. Changes to the X Server internals only affect graphics hardware vendors and X Extension vendors. Hardware vendors that ship X device-dependent load modules (`loadddx`) must port their code to use the new screen and font structures. These changes facilitate the use of overlay planes and the new font functionality. In most cases, vendors that ship non-IBM X Extensions must recompile their code. If the extension software uses the Screen or Font structure in the X Server, then the code must also be ported.

X Extensions

The X11R6.1 X Server supports the following new extensions:

- Double Buffer extension
- RECORD extension
- XTest extension
- Keyboard extension
- BIGREQUESTS extension
- XCMISC extension

Double Buffer Extension

The Double Buffer Extension (DBE) provides a method to do flicker-free animation. DBE has a relative method of buffer swapping; that is, the application always draws to the hidden or back buffer and shows the display or front buffer. When the buffers are swapped, the same identifier is used for the new back and new front buffers. This differs from DBE's predecessor, the MultiBuffer Extension (MBX), which uses an absolute form of buffering. In MBX, an application had to keep track of which buffer was being used as the draw buffer and which buffer was being used as the display buffer, and explicitly display the correct buffer.

DBE provides functionality to query, create, destroy, and swap buffers. It is integrated with IBM's 3-D software and Ancillary Buffer Extension. Although the DBE was an X11R6 standard, it was shipped in the AIX X11R5 distribution.

Keyboard Extension

The Keyboard Extension (XKB) provides keyboard functionality that is not present in the core X protocol. It has functionality for people with physical disabilities as well as configurable indicators, named bells, and detailed keyboard descriptions.

XKB provides a method to create detailed keyboard descriptions to display a keyboard on the screen. This displayed keyboard can then be used to create keyboard events without actually pressing the keys. Other functionality available for people with physical disabilities includes:

The Keyboard Extension (XKB) provides keyboard functionality that is not present in the core X protocol.

- Sticky keys. Keys that logically stay pressed while the user presses another key. For example, assume your desktop is configured so that the <Ctrl> <F9> key sequence iconifies a window. It is difficult for a user with one arm to press the <Ctrl> and <F9> keys at the same time. When sticky keys are invoked, the <Ctrl> key stays logically pressed while the <F9> key is pressed and the window becomes an icon.
- Mouse keys. These configurable keys can be used to move the core pointer.
- Repeat keys. The keyboard extension allows users to set the timeout rate

between the first keypress and the generated repeat events.

- Bounce keys. The Bounce keys control temporarily disables a key after it has been pressed, effectively "debouncing" the keyboard.

Configurable Indicators. XKB provides for configurable indicators (for example, LEDs). Although the core X implementation supports up to 32 LEDs on an input device, it does not provide any linkage between the state of the LEDs and the logical state of the input device. For example, most keyboards have a Caps Lock LED, but X does not provide a mechanism to make the LED automatically follow the logical state of the Caps Lock key.

The core X implementation does not allow clients to determine what bits in the `led_mask` field of the `XKeyboardState` can map to the particular LEDs on the keyboard. For example, there is no method for a client to determine what bit to set in the `led_mask` field to turn on the Scroll Lock LED, or if the keyboard even has a Scroll Lock LED.

XKB provides indicator names and programmable indicators to help solve these problems. Using XKB, clients can determine the following:

- Names of the various indicators
- The way that individual indicators should be updated to reflect keyboard changes
- Which keyboard indicators (out of 32) reported by the protocol are actually present on the keyboard

Clients can also request immediate notification of changes to the state of any subset of the keyboard indicators. This makes it straightforward to provide an onscreen "virtual" LED panel.

Named Bells. The core X protocol only allows applications to explicitly sound the system bell with a given duration, pitch, and volume. XKB extends this capability by allowing clients to attach symbolic names to bells, disable audible bells, and receive an event whenever the keyboard bell is rung.

Several new sample applications that use the Keyboard Extension are shipped with X11R6.1:

- The `xkbcomp` keymap compiler converts a description of an XKB keymap into one of several output formats:

  —Format 1. The `xkm` (compiled keymap file) can be read directly by XKB-capable X servers or utilities.

  —Format 2. C header files can be included by X servers or utilities that need a built-in default keymap.

  —Format 3. XKB source files can verify that the files which typically comprise an XKB keymap are merged correctly. They can also be used to create a single file that contains a complete description of the keymap.

- The `xkbevd` application listens for specified XKB events and executes requested commands if they occur.
- The `xkbprint` application generates a printable or encapsulated PostScript® description of the XKB keyboard description specified by source.

The Keyboard Extension provides compatibility with non-Keyboard Extension X Window System client applications.

The Record Extension and the X Test Extension used together can provide test and/or replay scenarios that drive X client applications.

Record/XTest Extensions

The Record Extension and the X Test Extension used together can provide test and/or replay scenarios that drive X client applications. The X Test Extension was shipped in the AIXwindows X11R5 distribution; the Record Extension is new in AIXwindows 4.3. The Record Extension captures both user actions and consequences. For example, a test case might consist of pressing the right

mouse button to display a pop-up menu or selecting a button in the pop-up menu.

The pop-up must be displayed before pressing an included button. The `MappingNotify` event of the pop-up is the result of pressing the button. The Record Extension can capture a subset of core X protocol and arbitrary extensions. Requests are timed for synchronization. Clients are registered so that one or more clients can be recorded by a single record application.

Note: The Record Extension and X Test Extension record and playback scenarios have different semantics than the xrecorder tool shipped in the AIXwindows samples. The Record/XTest Extensions actually capture events from and send events to applications. The xrecorder tool plays a protocol stream to the X Server; it does not actually communicate with an application.

Other New Extensions

Two additional extensions—BIGREQUEST and XCMISC—are used internally by Xlib to process requests larger than the current maximum request length and to reuse X resource IDs when many resources are allocated.

What's New—Libraries

In AIXwindows 4.3, the client libraries use the X11R6.1 and Motif® 2.1 source as their base. This provides the ability to create threaded applications and 64-bit applications.

Changes to the AIXwindows 2-D libraries include the addition of thread-aware libraries, 64-bit support, and new functionality using the Inter-Client Exchange (ICE) library, X Session Management (SM) library, and the Input Method Protocol.

Note: Thread-aware means that the library can be used in threaded applications only if the application enables the threading capability. For AIXwindows 2-D libraries, the application must call one of the following subroutines:

- `XInitThreads`
- `XtToolkitThreadInitialize`

Library	Source
`libXm.a`	Motif 2.1
`libMrm.a`	Motif Resource Manager
`libX11.a`	Xlib
`libXt.a`	Toolkit Intrinsics
`libSM.a`	Session Management Library
`libICE.a`	Inter-Client Exchange Library
`libXau.a`	X Authorization
`libXext.a`	X Extensions Library
`libXtst.a`	XTest Library

Figure 1. Thread-aware libraries in AIXwindows 4.3

X Libraries

- `libX11.a`
- `libXt.a`
- `libXext.a`
- `libXtst.a`
- `libXau.a`
- `libXdmcp.a` (X Display Manager Control Protocol)
- `libSM.a`
- `libICE.a`
- `libXi.a`

Motif Libraries

- `libXm.a`
- `libMrm.a`

Figure 2. AIXwindows libraries for building 64-bit applications

Figure 1 shows the thread-aware libraries in AIXwindows 4.3.

Note: CDE libraries are not thread safe or thread aware because they require Motif 1.2, which is not thread aware.

One outstanding feature of AIX 4.3 is support for a 64-bit operating system. Figure 2 shows AIXwindows 2-D libraries that can be used to build 64-bit client applications.

Only the X11R6 and Motif 2.1 libraries are 64-bit enabled. The following libraries are *not* 64-bit enabled:

- `libUil.a`—Code generation library delivered from Open Software Foundation® that is used to build sample widgets
- Desktop Libraries (Common Desktop Environment)—CDE Desktop ships several libraries that Independent Software Vendors (ISVs) could use to create applications; none are 64-bit enabled
- `libXmu.a`—Xmu is the Miscellaneous Utilities library; it is a sample and has not been upgraded to 64-bit
- `libfont.a`—Used by the font server and X server to open the fonts. Very few client applications use this library to directly open fonts. Most X applications call the `XLoadFont` or `XLoadQueryFont` subroutines to ask the X Server to open the font on its behalf

ICE

The Inter-Client Exchange protocol provides a common framework to build protocols on which clients can communicate with each other and not communicate through the X protocol. Communicating through the X protocol requires a round trip through the X server. ICE provides authentication, byte order negotiation, version negotiation, and error-reporting conventions.

Common functionality does not need to be reinvented each time it is used. ICE supports multiplexing of multiple protocols over a single transport connection. The `iceauth` (similar to the `xauth` program) client is shipped as a sample client application. It was written to manipulate the ICE authority file.

Both Xlib and Session Management library use ICE.

Session Management

A *session* is a group of clients, each having a particular state (for example, geometry). Knowing the client application state allows a client to be restarted at a later time as if it were never terminated.

The Session Management protocol and library provide a mechanism to save and restore a user's session. The X Session Management Protocol (XSMP), built on ICE, provides a uniform mechanism for users to save and restore their sessions using the services of a network-based session manager.

SMlib (`libSM.a`) is the C interface to the protocol. Xt also supports XSMP. SMlib contains subroutines for both X client applications and a session manager. The session manager software determines when a client is terminating and saves the client's session or state. A client application registers with session manager and stays attached to session manager during its execution. Multiple instances of clients are managed independently. The `xsm` session manager is shipped as a sample. AIXwindows uses `dtsession` provided by the CDE desktop; `dtsession` does not use `libSM.a`.

Input Method Protocol

The Input Method Protocol can create portable internationalized client applications and adapt native language, local customs, and character string encodings without changes to the application. The X11R5 distribution from the X Consortium contained a standard way to do internationalized input; however X11R5 did not address the protocol used between the Input Method (XIM) client and XIM server.

The standard Input Method Server Protocol was introduced in X11R6. There is no upgrade to AIX Input Method Server; however, the Omron Input Method Server ships as part of the Japanese kit. The Omron Input Method uses the standard Input Method Protocol.

Motif 2.1

Motif 2.1 has many upgrades from Motif 1.2. Some changes that resulted from work done for CDE were already present in the AIXwindows X11R5 implementation. Some changes discussed in this article are new to Motif 2.1, but not new to AIXwindows.

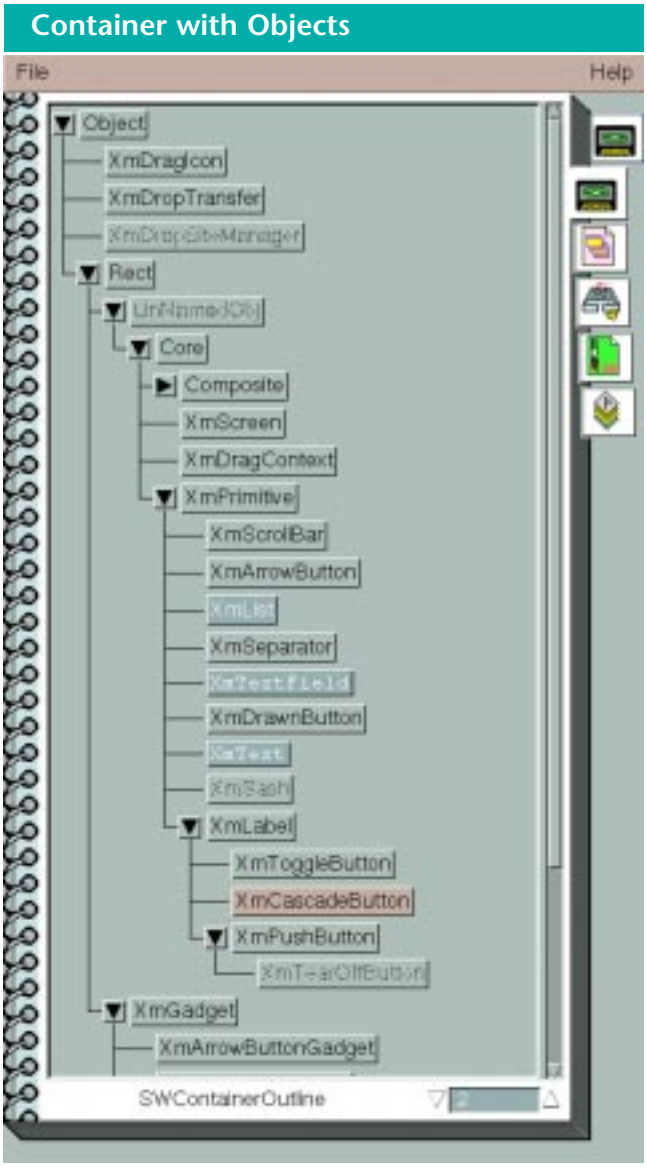

Figure 3. Container with objects

Motif 2.1 contains all of the CDE visual attributes and behavior that were added through the CDE project. CDE color model and CDE menu behavior are part of Motif 2.1, which is not new from Motif 1.2 in AIX 4.1 and AIX 4.2.

Container Widget

A Container widget presents information about application-defined objects on a two-dimensional background for selection and/or manipulation. Each object is represented by an icon, which can display a text label and either a large or small pixmap.

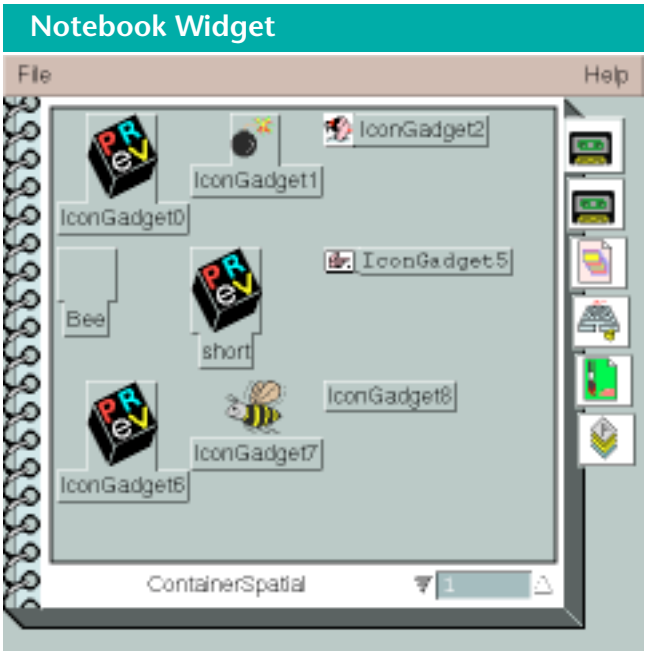

Figure 4. Notebook widget

Figure 3 presents objects in a container. Each of these objects displays a text label.

The application determines the location of each icon, but the user can change the icon location within the Container by dragging the icon and dropping it onto a new location within the Container.

Notebook Widget

A Notebook widget simulates a real notebook by grouping components into separate pages. It also contains components for accessing and describing pages. Only one page can be seen at a time. Notebooks can contain five fields: the page itself, Page Scroller, Major Tabs, Minor Tabs, and Status area. The Tabs and Status area are not required fields; their use depends on the needs of the particular application.

The page shown in Figure 4 is a Container component that displays dialog icons; however, since the Notebook can be used to contain any component, it can be used for many different applications.

The Page Scroller component (shown on the lower right corner of Figure 4) can be used to display different pages of the Notebook, much as a reader would flip pages back and forth in a book. By default, a SpinBox component is used as the Page

Scroller, but the application can specify a different component. For example, an application could specify a ScrollBar component to be used as the Page Major Tabs are used by applications to divide the pages of the Notebook into major sections. In Figure 5, the Major Tabs (on the right-hand side of the page) are used to divide the pages into dialogs and widgets. Minor Tabs, used by applications to subdivide the major sections, are not used in Figure 5.

The application uses the Status area (not used in Figure 5) to display additional information about a page. The application supplies the information, which can be different for each page shown.

PanedWindow Widget

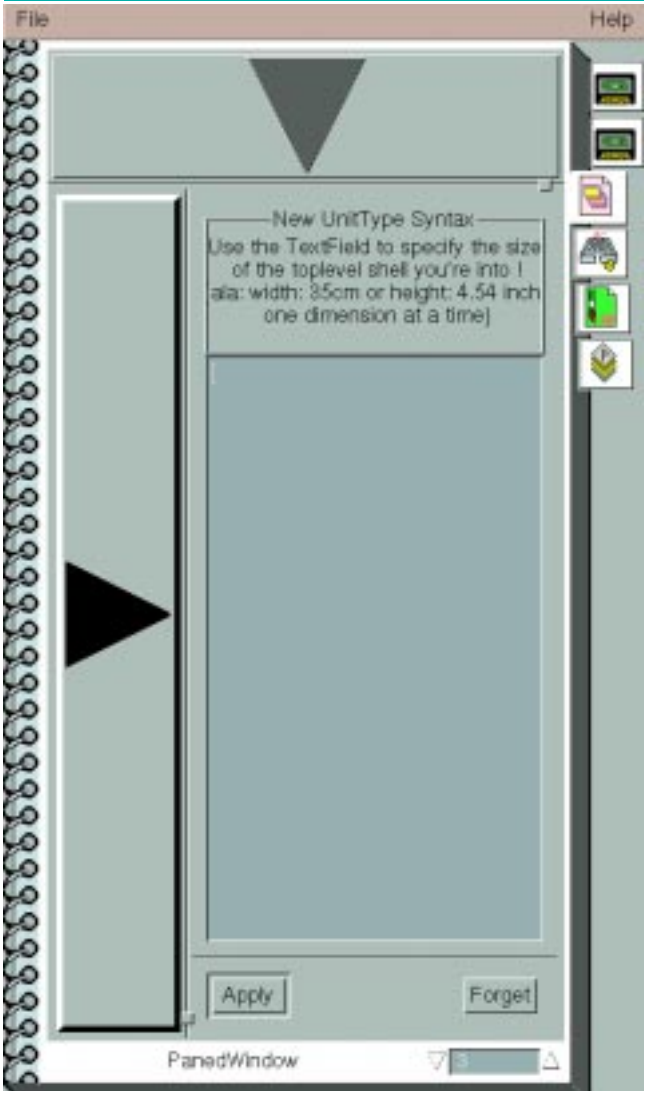

Figure 5. PanedWindow widget

ComboBox

A ComboBox combines the capabilities of the TextField and List widgets. Applications use ComboBoxes to allow users to either type in information as their selection or to select information from a list of choices provided by the application. If the List contains more items than are displayed, a vertical ScrollBar appears next to the List.

A calendar program provides an example. Suppose you want to give users the option of entering the month name or selecting it from a List widget. The ComboBox displays the month name as a text field, but has an arrow button to the side. By selecting that arrow, you get a List widget that contains all the month names from which to choose.

SpinBox

A SpinBox allows a ring of choices to be displayed, in sequence. The SpinBox consists of a non-editable text area where a single

Thermometer Widget

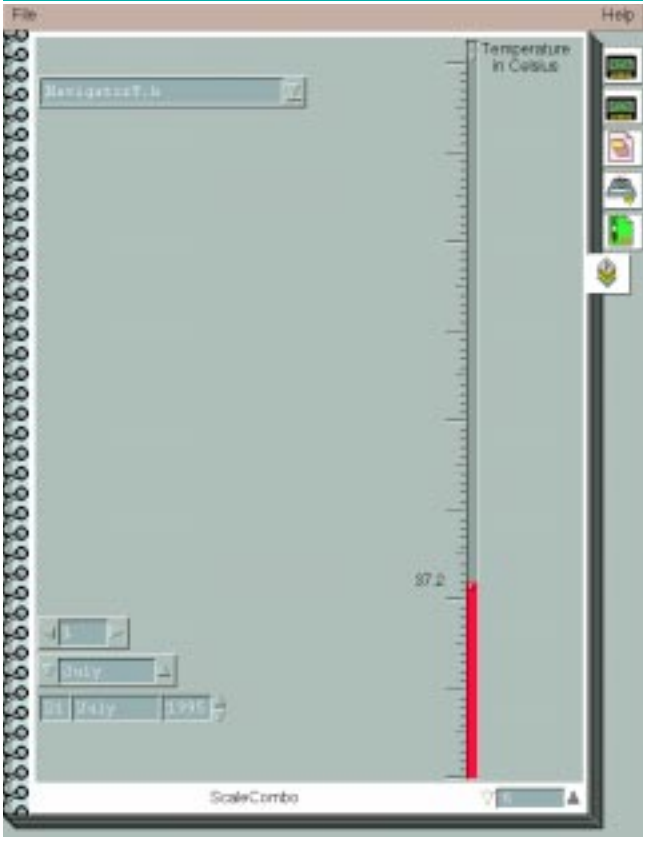

Figure 6. Thermometer widget

choice is displayed; there are two arrow buttons next to the text area, one on each side, for cycling through the choices.

Thermometer Scale and Tic Marks

The Thermometer widget is an enhancement to the ScrollBar widget; it uses tics to create a thermometer appearance, as shown in Figure 6.

The ScrollBar widget allows the user to view data that is too large to be displayed all at once. ScrollBars are usually located inside a ScrolledWindow and adjacent to the widget that contains the data to be viewed. When the user interacts with the ScrollBar, the data within the other widget scrolls.

A ScrollBar consists of two arrows placed at each end of a rectangle called the *scroll region*. A smaller rectangle, called the *slider*, is placed within the scroll region. The data is scrolled by clicking either arrow, selecting on the scroll region, or dragging the slider. When an arrow is selected, the slider within the scroll region is moved in the direction of the arrow by an amount supplied by the application. If the mouse button is held down, the slider continues to move at a constant rate. The ratio of the slider size to the scroll region size typically corresponds to the relationship between the size of the visible data and the total size of the data. For example, if 10 percent of the data is visible, the slider typically occupies 10 percent of the scroll region. This provides the user with a visual clue to the size of the invisible data.

XmIm API for I18N

The `XmNlayoutDirection` resource allows the user to specify display direction of the text. Choices include left to right or right to left. IBM supports bidirectional text in a previous release; the official Motif 2.1 software now supports the same.

General Rendition Attributes in XmString

General Rendition attributes enable different fonts, colors, or other attributes to be specified within one string or within a word. Motif represents much textual data using a data type called a *compound string*, also called `XmString`. This is a stream of components representing text, a display direction, and rendition tags. These rendition tags specify how the text will appear when rendered, including parameters such as the following:

- Font
- Color
- Tab stops (if any)
- Underlining
- Other text features

A compound string can have multiple text segments, possibly with different directions and rendition tags.

Complex Text Language (CTL)

Complex Text Language (CTL) is supported for composed, context sensitive, and intermixed bidirectional characters. CTL is required because bidirectional languages have different display characteristics. For example, in Hebrew, letters are displayed right to left while numbers are displayed left to right. In Arabic, letters are displayed right to left and numbers are displayed right to left. In Arabic, letters can also be displayed differently if they are at the beginning, the middle, or the end of the word.

CTL, which conforms to X/Open® Portable Layout Services specification, is part of a pluggable layout engine. The pluggable layout engine is passed the raw text and returns the rearranged and correct text. The `dtterm` function supports CTL as well as the Hebrew, Arabic, Thai, and Vietnamese languages.

What's New—Clients and Utilities

In AIXwindows 4.3, `dtterm` is the terminal emulator of choice; however, `aixterm` and `xterm` are still supported. Samples include the X11R6.1 versions of sample client applications: `xrdb` supports new symbols and `xdpyinfo` has added extension information for MultiBuffer Extension (MBX), Shape, Sync, X Image Extension (XIE), XTest, Double Buffer Extension, and Record.

```
cc o motifdemo motifdemo.c /usr/lpp/X11/lib/R6/Motif1.2/Xm12.imp
     -I/usr/include/Motif1.2  -lXt   -lX11
```

Figure 7. Code to compile a Motif 1.2 program

```
cc -o motifdemo motifdemo.c  -lXm -Xt -lX11
```

Figure 8. Code to compile a Motif 2.1 library

What's New—Fonts

X Logical Font Description Enhancements (XLFD) added software to perform general 2-D linear transformations. This enables the user or application writer to curve and scale text. The transformation matrix is specified as part of the XLFD name.

The font library contains an option to rasterize only a subset of glyphs. The X Server can then rasterize and cache glyphs on an as-needed basis. This is useful when rasterizing fonts, such as Chinese, that contain a large number of glyphs. The fontserver supports a TrueType® pluggable rasterizer, which was previously supported only in the Japanese kit.

TrueType is a PC font format; many TrueType fonts are available through the Internet and other common sources. The fontserver has been renamed to xfs, which avoids a name collision with the `afs` command. The fontserver now uses the official port 7100 that has been registered with the Internet Assigned Numbers Authority. The `fsconf` utility has been renamed to `xfsconf`.

What's Gone

AIXwindows 4.3 no longer supports the following:

- Display PostScript Extension (DPS)
- X Consortium 3-D Extension (PEX)
- IBM RT PC® AIXwindows samples `aixwm`, `aixinit`, and `aixclock`
- The `X11.compat.fnt.oldX` fileset, which contained fonts from the X11R1 vintage
- X Image Extension was never supported
- X Sync Extension was never supported

The default AIXwindows 4.3 2-D libraries are the X11R6 versions. The X11R3, X11R4, and X11R5 versions of these libraries are shipped as compat. The X11R3 version is archived into normal libraries. To use the X11R4 or X11R5 versions, users must set the `LIBPATH` environment variable.

Motif 2.1 is now the default Motif library; Motif 1.1 and Motif 1.2 have versions shipped as compat.

Motif

The CDE desktop is built with Motif 1.2. If application vendors want to build with Motif 1.2, they must use the `X11.compat.adt.motif12 fileset`, which contains the headers and libraries. Only Motif 2.1 samples are shipped. CDE 1.0 and UIM/X are Motif 1.2 programs and will not execute with Motif 2.1; however, they will execute without changing the `LIBPATH`.

When porting Motif 1.x applications to Motif 2.1, be aware of the fact that `XmString` is now a union data type. Because of this, any applications that referenced `XmString` data as a character pointer must be ported. It is no longer possible to do straight character pointer manipulation on `XmString` data.

To compile a Motif 1.2 program, use the line shown in Figure 7. Figure 8 shows how to compile a Motif application using the Motif 2.1 library. The differences are shown by comparing the figures.

Note that in the Motif 1.2 compilation, `-lXm` is not specified. Instead, the compiler is pointed to the `Xm12.imp` file that has all the needed symbols. The Motif 1.2 header files are specified by using `-I/usr/include/Motif1.2`.

Migration and Portability

The following sections discuss migration and portability.

Libraries

The Display and Graphics Context (GC) structures in Xlib are now considered opaque structures. These structures are only accessed through macros or by specifying `-DXLIB _ILLEGAL_ACCESS` during compile. GC is commonly used in application programming.

Clients

Many new symbols are defined in `xrdb` to specify the extensions and visual classes that are available. `xset` supports two new options that immediately activate or deactivate the screen saver:

- `imake` supports the [C filename] command-line option
- `threads.tmpl` is added for multi-threaded rules

`xterm` supports a few escape sequences from HP™ terminals. The `xterm` logging feature was removed and `xterm` is minimally internationalized to use the Xlib built-in input method with 8-bit character sets.

Reasons to Move Up

AIXwindows 4.3 and X11R6.1 provides improved functionality for both users and application writers. This includes new extensions (XKB, Record, DBE, and XTEST), upgraded libraries for threaded applications and 64-bit applications, and upgraded samples and X Client applications.

Jeanne Sparlin, IBM Corporation, 11400 Burnet Road, Austin, TX 78758. Ms. Sparlin joined IBM in 1983 after completing her computer science training at the University of Texas, Austin. She has previously worked in the Data Management, SNA, Dialog Design Aid, and the X Window System development departments on the RT PC. She joined the Graphics Architecture department in October 1988. Ms. Sparlin previously represented IBM on the Advisory Committee of the X Window System Consortium. She has a BS in Education from Northeast Missouri State University.

Dennis Heideman, IBM Corporation, 11400 Burnet Road, Austin, TX 78758. Mr. Heideman is a staff software engineer and the technical lead for Motif programming on AIX. He is responsible for Motif/X Client side libraries on AIX. He has a BA in Computer Information Systems from Southwest Texas State University.